


LinkedIn: [Kelvin Kimotho](#)

unpackme.py 

Medium

Reverse Engineering

picoCTF 2022

packing

AUTHOR: LT 'SYREAL' JONES

Hints ?

Description

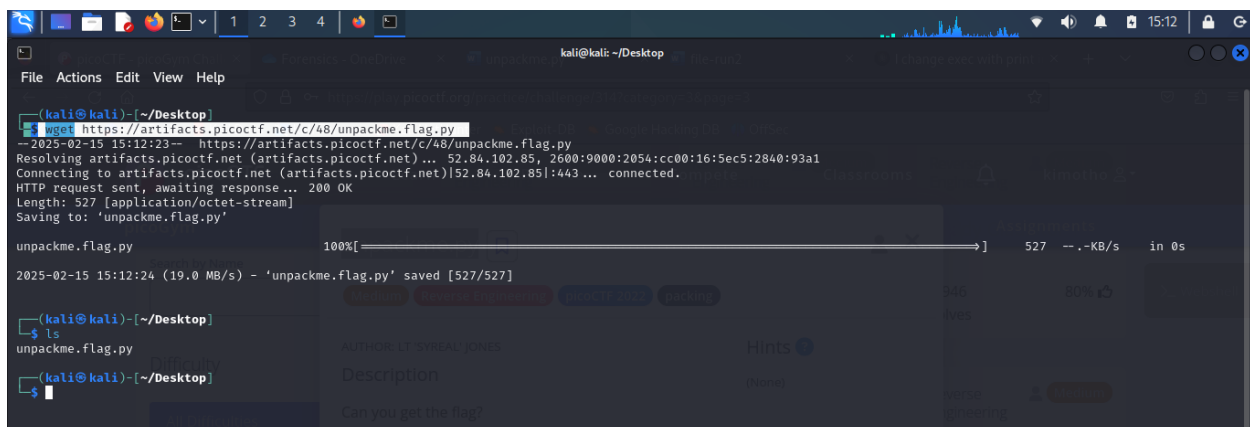
(None)

Can you get the flag?

Reverse engineer this [Python program](#).

## Solution

I downloaded the python script named **unpackme.py** using **wget** a command line tool.



```
(kali@kali)-[~/Desktop]
└─$ wget https://artifacts.picoctf.net/c/48/unpackme.flag.py
--2025-02-15 15:12:23-- https://artifacts.picoctf.net/c/48/unpackme.flag.py
Resolving artifacts.picoctf.net (artifacts.picoctf.net) ... 52.84.102.85, 2600:9000:2054:cc00:16:5ec5:2840:93a1
Connecting to artifacts.picoctf.net (artifacts.picoctf.net)|52.84.102.85|:443 ... connected.
HTTP request sent, awaiting response... 200 OK
Length: 527 [application/octet-stream]
Saving to: 'unpackme.flag.py'

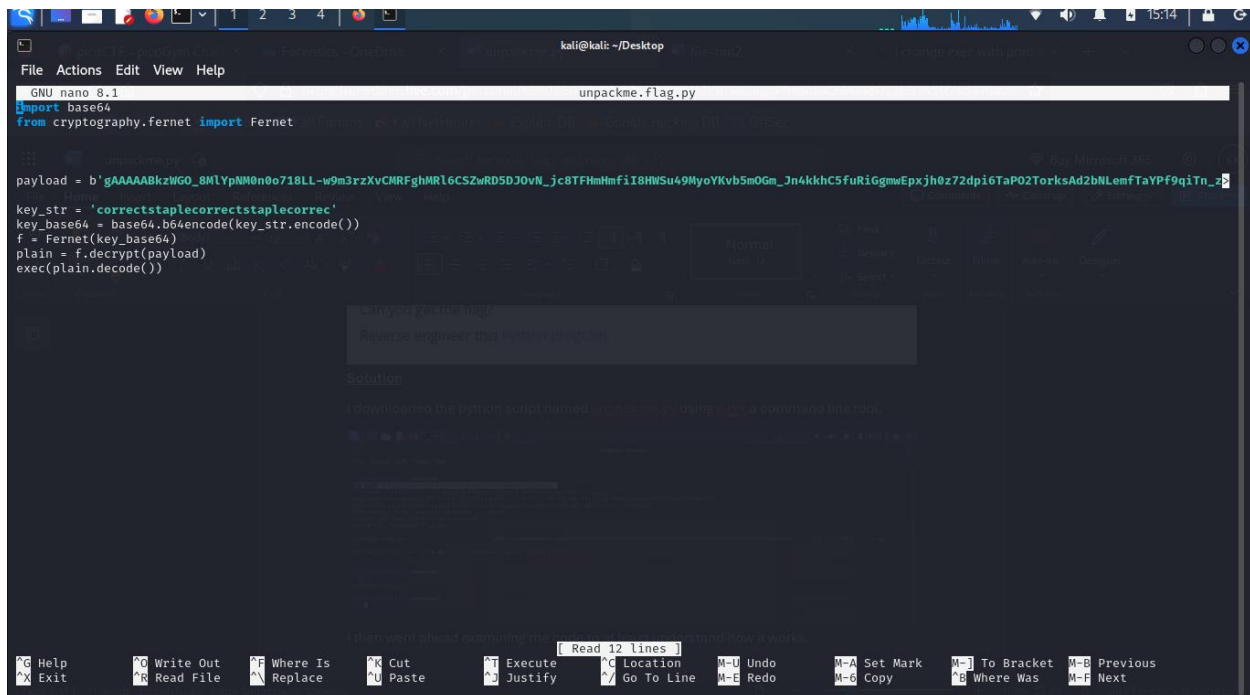
unpackme.flag.py      100%[=====] 527 --.-KB/s  in 0s

2025-02-15 15:12:24 (19.0 MB/s) - 'unpackme.flag.py' saved [527/527]

(kali@kali)-[~/Desktop]
└─$ ls
unpackme.flag.py

(kali@kali)-[~/Desktop]
└─$
```

I then went ahead examining the code to at least understand how it works.



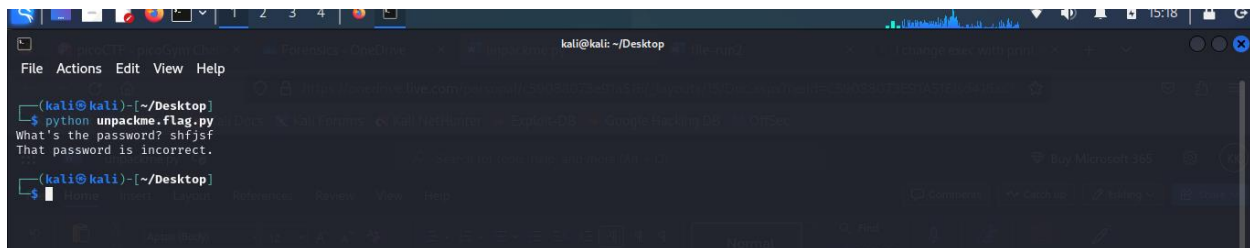
```
GNU nano 8.1 unpackme.flag.py
import base64
from cryptography.fernet import Fernet

payload = b'gAAAAABkzWGO_8MLYpNM0n0o718LL-w9m3rzXvCMRFghMRl6CSZwRD5D30vN_jc8TFHmHmfI8HWSu49MyoYKvb5mOGm_Jn4kkhC5fuRiGgmwEpxjh0z72dpi6TaP02TorksAd2bLemfTaVPf9qiTn_z'

key_str = 'correctstaplecorrectstaplecorrec'
key_base64 = base64.b64encode(key_str.encode())
f = Fernet(key_base64)
plain = f.decrypt(payload)
exec(plain.decode())
```

This script is designed to decrypt a payload using the cryptography library's **Fernet** symmetric encryption. The payload is a **base64-encoded** string that has been encrypted with a specific key. The goal of the script is to decrypt this payload and reveal the underlying plaintext.

I realized that the script executed the decrypted plaintext instead of printing out the decrypted text. This explained the reason as to why the program prompted me for a password yet the script itself has no password check.



```
(kali@kali)-[~/Desktop]
$ python unpackme.flag.py
What's the password? shfjsf
That password is incorrect.
```

I went ahead and replaced the **exec** method with the **print** method in python.

```
File Actions Edit View Help
GNU nano 8.1 unpackme.flag.py *
import base64
from cryptography.fernet import Fernet

payload = b'gAAAAABkzWGO_8MLYpNM0n0e718LL-w9m3rzXvCMRFghMRl6CSZwRD5D30vN_jc8TFHmHmfiI8HWSu49MyoYKvb5m0Gm_3n4kkhC5fuRiGgmwEpxjh0z72dpi6TaP02TorksAd2bNLemfTaYPf9qiTn_2'

key_str = 'correctstaplecorrectstaplecorrec'
key_base64 = base64.b64encode(key_str.encode())
f = Fernet(key_base64)
plain = f.decrypt(payload)
print(plain.decode())
```

This would print out the decrypted plain text instead of executing it. I then ran the program. The decrypted plaintext was printed out and the flag was within the text.

```
kali@kali: ~/Desktop
File Actions Edit View Help

(kali@kali)-[~/Desktop]
$ python unpackme.flag.py

pw = input('What\'s the password? ')

if pw == 'batteryhorse':
    print('p1coGTF{175_chr157m45_5274ff21}')
else:
    print('That password is incorrect.')

(kali@kali)-[~/Desktop]
$
```

The encrypted was a python script for password checking which was being executed after decryption instead of being printed out.