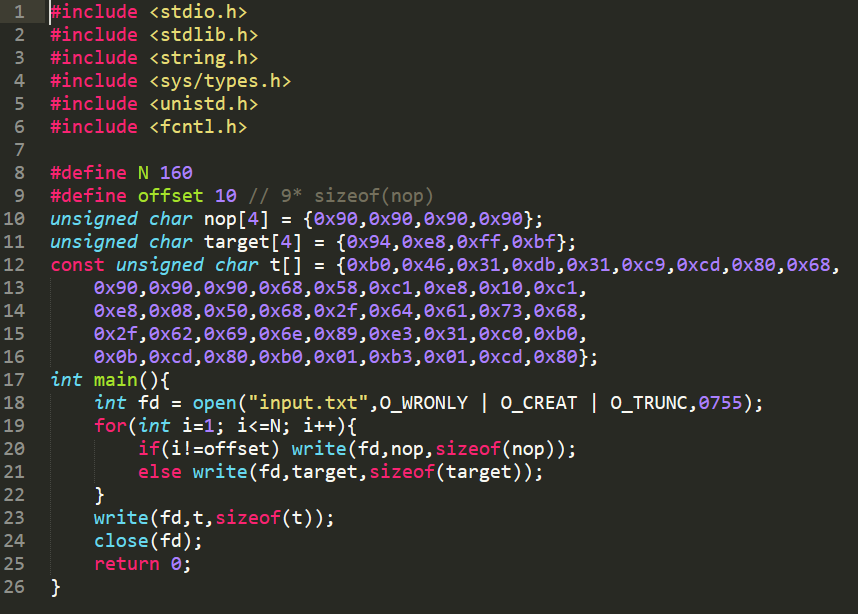
Assignment3

2016310932 배현웅

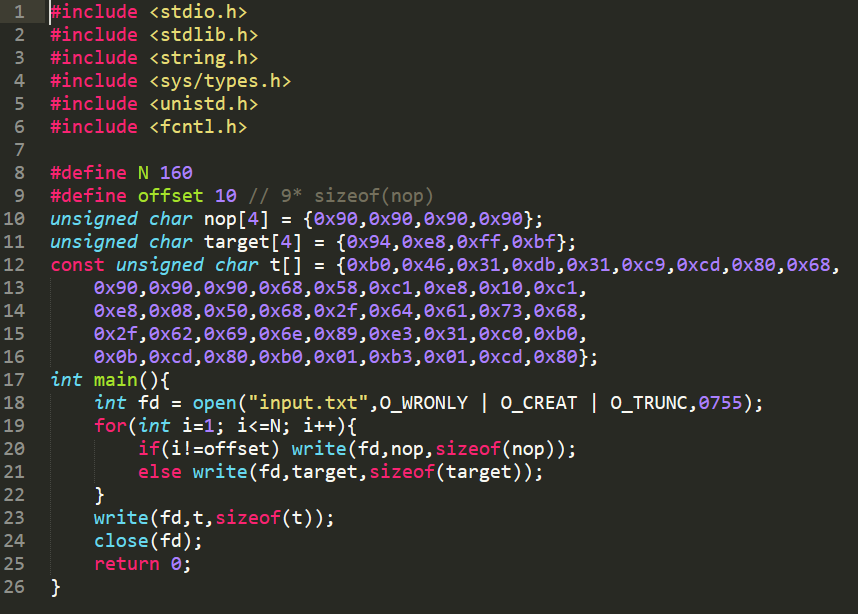
1. Overview
2. First of all, I found the relative address of RET from strcpy by gdb.
3. Make shell code for attack which allow root privilege.
4. Between shell code and RET, add no operation code(=0x90) for nop sled.
5. Code



Line 9 : Prepare relative offset of RET compared to strcpy function, cat\_file

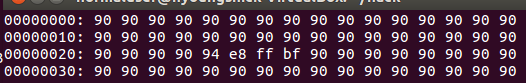
Line 11 : target address for code which we want to exploit. (in this case, root shell code)

Line 12 : machine code of shell code which we want to execute

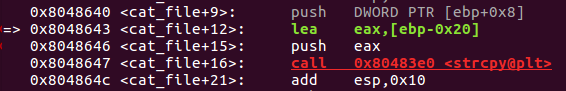


The input.txt struct is as below

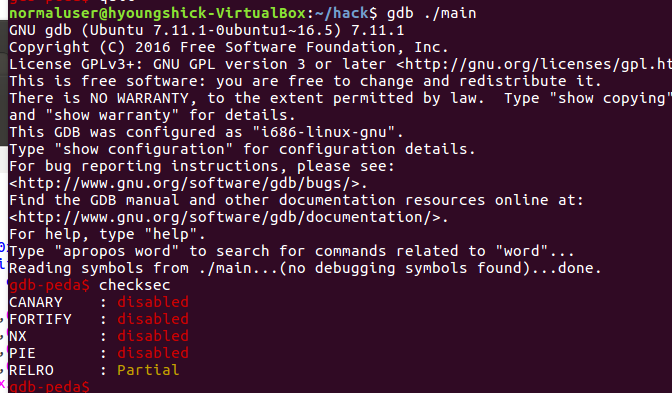
Nop \* 9 + target(=which return address written here) + nop\* 150 + shell code



In input.txt, the target address is placed at 0x24 which is relative address of strcpy in cat\_file function.

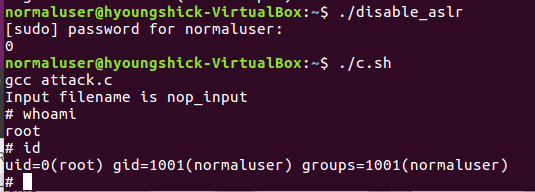


* By disas cat\_file in gdb, I found relative address.



To check memory protection which applied to this exec file, checksec in gdb

* Execution picture (=result)



* Extension Goal

Fail - I tried ROP(Return Oriented Programming) which is known to bypass ASLR. However, it need leakage of address such as printf, then calculate address of shell code function by relative address. because the address of execve, system, exit function are determined in runtime. It needs pwntools which allow to find runtime address of dynamic address affected by ASLR. (동적으로 stack주소가 할당되어서 실행되었을 때, libc.so.6의 주소, 정확히말하면 execve 라이브러리의 주소를 상대적인 주소로 absolute 주소를 알아내는 방식이었는데, 런타임동안에 가능한 공격이었습니다. 어떤 방식으로 ASLR을 우회할 수 있는지 궁금합니다.)