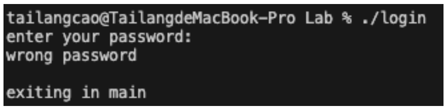
Buffer Overflow Exploit Lab

Tailang Cao u1480633

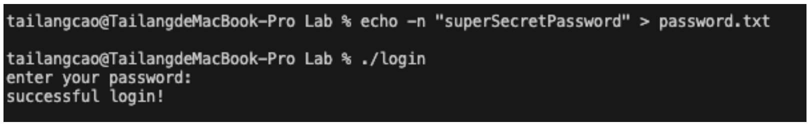
1. Test the code with random password, gives the wrong password message correctly



```
tailangcao@TailangdeMacBook-Pro Lab % ./login
enter your password:
wrong password

exiting in main
```

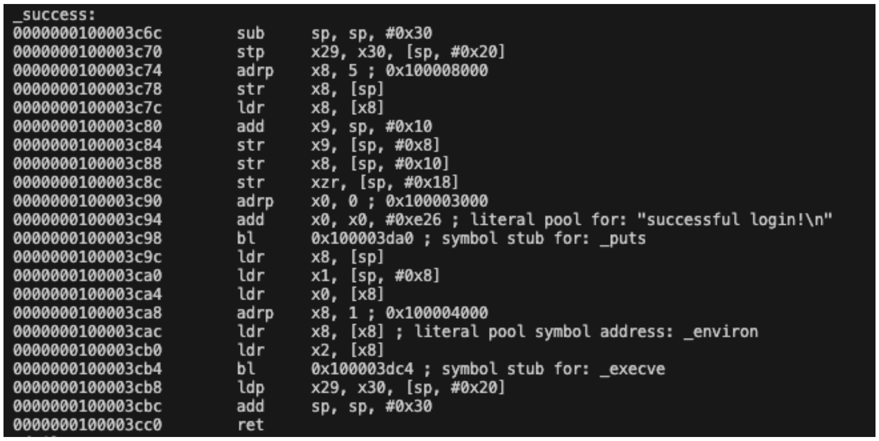
2. Test the code with the correct password, logged in successfully



```
tailangcao@TailangdeMacBook-Pro Lab % echo -n "superSecretPassword" > password.txt

tailangcao@TailangdeMacBook-Pro Lab % ./login
enter your password:
successful login!
```

3. View the assembly code with otool -tV login



```
_success:
0000000100003c6c        sub     sp, sp, #0x30
0000000100003c70        stp     x29, x30, [sp, #0x20]
0000000100003c74        adrp    x8, 5 ; 0x100008000
0000000100003c78        str     x8, [sp]
0000000100003c7c        ldr     x8, [x8]
0000000100003c80        add     x9, sp, #0x10
0000000100003c84        str     x9, [sp, #0x8]
0000000100003c88        str     x8, [sp, #0x10]
0000000100003c8c        str     xzr, [sp, #0x18]
0000000100003c90        adrp    x0, 0 ; 0x100003000
0000000100003c94        add     x0, x0, #0xe26 ; literal pool for: "successful login!\n"
0000000100003c98        bl      0x100003da0 ; symbol stub for: _puts
0000000100003c9c        ldr     x8, [sp]
0000000100003ca0        ldr     x1, [sp, #0x8]
0000000100003ca4        ldr     x0, [x8]
0000000100003ca8        adrp    x8, 1 ; 0x100004000
0000000100003cac        ldr     x8, [x8] ; literal pool symbol address: _environ
0000000100003cb0        ldr     x2, [x8]
0000000100003cb4        bl      0x100003dc4 ; symbol stub for: _execve
0000000100003cb8        ldp     x29, x30, [sp, #0x20]
0000000100003cbc        add     sp, sp, #0x30
0000000100003cc0        ret
```

4. The success function starts from address 0000000100003c6c

5. To make the attack successful, we should be able to make the function return to the success function instead of the main function by making the password the success address.

6. Write the password in little indian way hence backwards:

   % python3 -c 'import sys; sys.stdout.buffer.write(b"a"*24 + b"\x6c\x3c\x00\x00\x01\x00\x00\x00")' > password.txt

7. After multiple attempts the attack still wouldn’t work and the login function would still return in main. My guess is the address is incorrect or the way I write it in the password.txt cant direct it to success function.