

实验八—RISCV 汇编程序设计

姓名：张云策 学号：3200105787 学院：云峰学园

课程名称：计算机系统 1 同组学生姓名：/

实验时间：2021.5.24 实验地点：紫金港东 4-509 指导老师：吴磊

一、实验目的和要求

- 1.成功搭建 RISC v 的执行环境。
- 2.理解 ELF 和汇编的关系。
- 3.理解 overflow。

二、实验内容和原理

2.1 实验内容

阅读 key.s，理解内容并得出密码。
攻击 bof，并且分析原因。

2.2 设计模块

无

三、主要仪器设备

Ubuntu 虚拟机，spike 工具。

四、 操作方法与实验步骤

4.1 操作方法

无

4.2 实验步骤

5.1

1. 理解 ELF 文件和汇编文件。
2. 输入学号和 key
3. 尝试破解自己学号对应的 key。

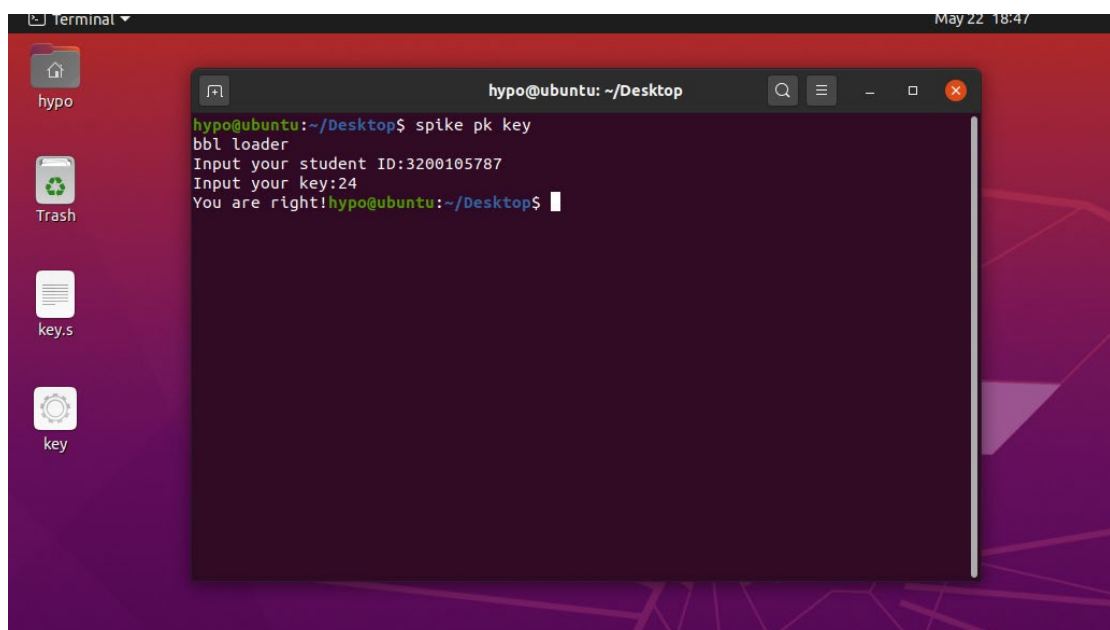
5.2

1. 理解 bof 文件。
2. 攻击 bof。

五、 实验结果与分析

一.key

key 函数实现的功能是利用三个循环，首先，通过 read_student 函数以及其他附属函数，实现判断输入字符是否大于 57，是的话显示输入输入错误，字符在<47,57>时，输入进数组。之后进入 read_key，进行十次循环，前五位不变，后五位分别取与九的差值，之后差值和前五位实现加和操作，取结果，3200105787 得结果 24。



```
hypo@ubuntu: ~/Desktop
hypo@ubuntu:~/Desktop$ spike pk key
bbl loader
Input your student ID:3200105787
Input your key:24
You are right!hypo@ubuntu:~/Desktop$
```

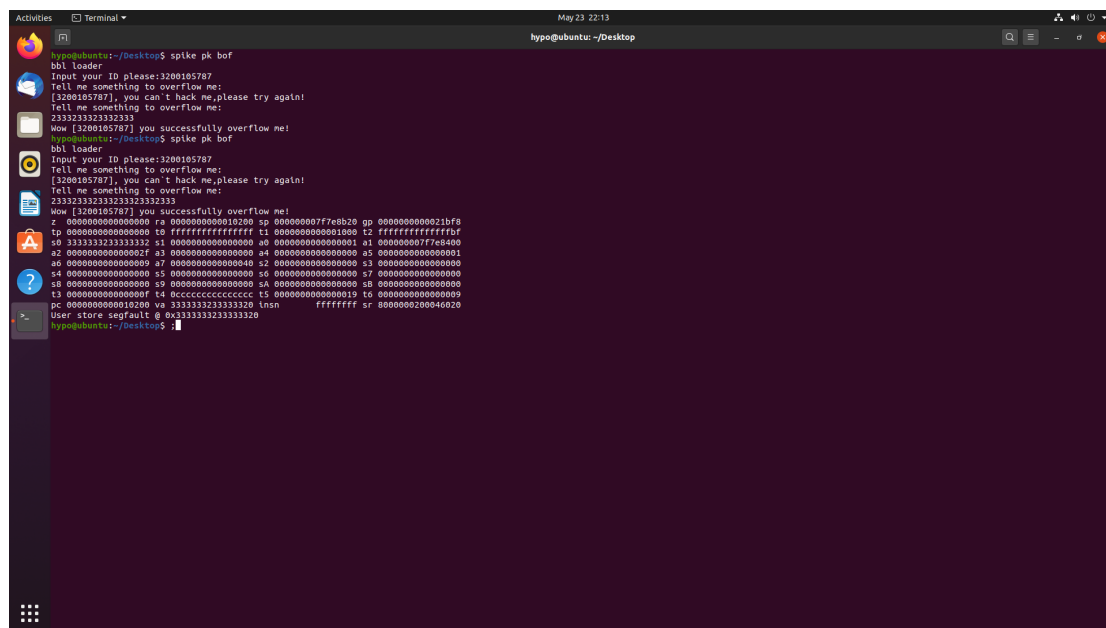
二.bof

bof 目标为实现缓冲区溢出，即输入一串字符串超出输入限制，实现缓存溢出攻击效果。

hear 分析：

首先，前六行代码实现设置函数，先实现“挖坑”。

其次，利用“N”“Y”的 ascii 码，将两者输入。之后进行输入用户输入字符串，如果用户输入字符串过长，当缓冲区边界限制不严格时，由于变量传入畸形数据或程序运行错误，导致 s0 寄存器被“撑爆”，从而覆盖了相邻内存区域的数据，就会实现缓存溢出，替代了 a4, a5 原有的值，然后与 11111111 ‘b’ 取同操作，判断是否相等，如若相等，则可以输出“you’re right!”



```
hypo@ubuntu: ~/Desktop$ spike pk bof
bbl loader
Input your ID please:3200105787
Tell me something to overflow me:
[3200105787], you can't hack me, please try again!
Tell me something to overflow me:
Z333233323332333
Now [3200105787] you successfully overflow me!
hypo@ubuntu: ~/Desktop$ spike pk bof
bbl loader
Input your ID please:3200105787
Tell me something to overflow me:
[3200105787], you can't hack me, please try again!
Tell me something to overflow me:
Z3332333233323333333
Now [3200105787] you successfully overflow me!
z 0000000000000000 ra 0000000000000000 sp 00000000777e8b20 gp 0000000000021bf8
tp 0000000000000000 tb ffffffff ffffffff t1 0000000000000000 t2 ffffffff ffffffff
a0 3333333233333332 s1 0000000000000000 a0 0000000000000001 a1 00000000777e8b20
a2 000000000000002f a3 0000000000000000 a4 0000000000000000 a5 0000000000000001
a6 0000000000000009 a7 0000000000000040 i2 0000000000000000 i3 0000000000000000
s4 0000000000000000 s5 0000000000000000 s6 0000000000000000 s7 0000000000000000
s8 0000000000000000 s9 0000000000000000 sA 0000000000000000 sB 0000000000000000
t3 000000000000000f t4 0000000000000000 t5 0000000000000000 t6 0000000000000009
pc 0000000000000000 va 3333333233333320 insn ffffffff sr 0000000020084020
User store segfault @ 0x3333333233333320
hypo@ubuntu: ~/Desktop$
```

出现如下图的情况，是因为在输入了过多数据，导致 va 溢出，从一开始函数起点的占据字节数都被输入值所覆盖，所以函数的返回地址变成了 va 现存地址，这超出了程序的地址空间，所以出现段错误

```
hypo@ubuntu:~/Desktop$ spike pk bof
bbl loader
Input your ID please:3200105787
Tell me something to overflow me:
[3200105787], you can't hack me,please try again!
Tell me something to overflow me:
2333233323323323332333
Wow [3200105787] you successfully overflow me!
z 0000000000000000 ra 000000000010200 sp 000000007f7e8b20 gp 0000000000021bf0
tp 0000000000000000 t0 ffffffff t1 0000000000001000 t2 ffffffff t3 ffffffff
s0 333333233333332 s1 0000000000000000 a0 0000000000000001 a1 000000007f7e8400
a2 000000000000002f a3 0000000000000000 a4 0000000000000000 a5 0000000000000001
a6 0000000000000009 a7 0000000000000040 s2 0000000000000000 s3 0000000000000000
s4 0000000000000000 s5 0000000000000000 s6 0000000000000000 s7 0000000000000000
s8 0000000000000000 s9 0000000000000000 sA 0000000000000000 sB 0000000000000000
t3 000000000000000f t4 0cccccccccccccc t5 0000000000000019 t6 0000000000000009
pc 00000000000010200 va 3333332333333320 insn ffffffff sr 8000000200046020
User store segfault @ 0x3333332333333320
hypo@ubuntu:~/Desktop$
```

六、 讨论、心得

新奇，困难。

个人感觉，或许可以通过输入某些类似宏定义的东西来对目标计算机进行一些破坏，在学习过程中，也了解到了譬如“gets”“strcpy”等函数的不完备性，也切实体会到了它们的缺陷，明白了下一步写代码时应该如何写，如何做，如何保证自己的程序尽力不被攻击（至少不会被这种攻击击溃）。