* **实验说明：拆炸弹游戏。**

执行./bomb文件，提示有6个关卡，每一个关卡都需要用户输入一些东西（不需要改代码），如果正确，则进入下一关卡；否则炸弹引爆，程序结束。用户需要获得正确答案解除炸弹，就需要通过反汇编指令执行bomb文件得到汇编代码（可以将其保存到bomb.s文件来具体查看），通过分析调试汇编代码来得到每一关卡的正确答案。此实验在于帮助大家学习理解汇编指令，并了解如何用汇编指令编写程序。

但是需要注意的是，objdump命令得到的代码是运行前的代码，里面有些地址可能是错误的（因为加在器会把可执行文件加载到内存变成进程运行时动态修改地址，下学期的编译那一章会讲相应内容），因此如果发现某些地址无法解释，要在gdb中看运行时地址。

为了方便同学输入数据，不至于每次都全部重新输入一遍，这个bomb可执行文件支持用文件作为输入。因此，可以把已经成功排雷的字符串放在一个文件中，然后 ./bomb abc ，程序即可从abc文件读入字符串并自动排雷。

提示，gdb中两个常用的操作：1）使用disassemble命令查看函数汇编代码；2）使用x命令查看内存中的值。比如x/s 0x400232 即可查看400232地址处存储的字符串。

**注意：**该实验需要使用objdump和gdb来实现反汇编和调试，关于它们的使用补充如下，同学们可以参考。

* **反编译工具Objdump：**

objdump -d ./bomb>bomb.s #反汇编bomb程序，并且输出到bomb.s。

objdump -t ./bomb  #输出符号表

* **调试工具gdb：**

1. **开始：**

gdb

gdb <file>

1. **运行和停止：**

quit或者Ctrl-d #退出gdb调试

run #运行程序 （r）

run 1 2 3 #传入命令行参数1 2 3运行程序

ctrl+c #运行中暂停程序

kill #杀死程序

1. **断点：简写b**

b #在当前行设置断电

break sum #在程序入口处设置断点

break \*0x80483c3 #在0x80483c3地址处设置断点

break 5 #在第5行设置断点

info b #查看所有断点

delete 1 #删除断点1

delete #删除所有断点

clear sum #清除函数sum入口的所有断点

1. **执行：**

stepi #执行一条指令 （s），与n的不同在于可进入函数

stepi 4 #行四条指令

nexti  #类似于stepi，但是继续执行函数调用而不停止 （n）

continue #继续执行，直到下一个断点 （c）

until 3 #继续执行，直到程序到达断点3

finish #继续执行，直到当前函数返回

call sum(1, 2) #调用sum(1,2)并打印返回值

set var 变量名=变量的值 #可以修改变量的值

1. **检查数据：**

x/w 0xbffff890 #检查从地址0xbffff890开始的字符

x/<n/f/u> #从addr内存地址开始，以u对应的字节数为一个单位，以f格式显示n个单位的内存数据。

所有用法请见网页查询 <http://visualgdb.com/gdbreference/commands/x>

1. **检查代码：**

disas 函数名 #查看函数的汇编代码

print/display 变量名 #打印该变量的值

info b #查看所有断点信息（加断点编号可以查看某一个断点信息）

 b 行数 #对当前行数设置断点，程序运行时运行到该行停止，并输入下面将要执行的一行信息

* **Vi常用命令：**

<https://blog.csdn.net/zhouguangcai8/article/details/79553775>

<https://blog.csdn.net/cyl101816/article/details/82026678>