《计算机组成原理实验》

实验报告

（实验一）

学 院 名 称 ：	数据科学与计算机学院
专业（班级）：	18 计教学 3 班
学 生 姓 名 ：	张天祎
学　　号	18340215
时　　间 ：	2019　年　10　月　9　日

成 绩 ：

实验一 ： X86汇编基础二进制炸弹

一. 实验目的

⑴初步认识X86汇编语言；

⑵掌握阅读程序反汇编代码的方法，了解程序在机器上运行的实质；

⑶熟悉Linux环境、掌握调试器gdb和反汇编工具objdump的使用。

二. 实验内容

使用课程知识拆除一个“Binary Bomb”（，简称炸弹）来增强对程序的机器级表示、汇编语言、调试器和逆向工程等理解。二进制炸弹是一个Linux可执行C程序，包含phase_1~phase_6共6个阶段和一个隐藏阶段secret_phase。你将获得一个唯一且每位同学差异化的炸弹程序。炸弹运行各阶段要求输入一个字符串，若输入符合程序预期，该阶段炸弹被“拆除”，否则“爆炸”。实验目标是你需要拆除尽可能多的炸弹。

每个炸弹阶段考察机器级语言程序不同方面，难度递增。

阶段1：字符串比较阶段2：循环

阶段3：条件/分支：含switch语句阶段4：递归调用和栈

阶段5：指针

阶段6：链表/指针/结构

隐藏阶段，第4阶段的之后附加一特定字符串后才会出现

必要的拆弹技术：

为了完成二进制炸弹拆除任务，需要

1.使用gdb调试器和objdump反汇编工具；

2.单步跟踪调试每一阶段的机器代码

3.理解汇编语言代码的行为或作用

4.进而设法“推断”出拆除炸弹所需的目标字符串。

5.需要在每一阶段的开始代码前和引爆炸弹的函数前设置断点，便于调试。

三. 实验器材

PC机一台，装有Linux操作系统的虚拟机一套。

四. 实验过程与结果

首先获得 bomb 等文件，再用 objdump 指令生成反汇编文件。拆弹操作指导中有实现上述两步的具体细节，这里不赘述。很多 gdb 指令用法和汇编代码的含义通过查资料、和同学交流边学边用，个中曲折在下文中也不赘述。

gdb ./bomb 进入 gdb 调试。每次开始时先在<explode_bomb>处打断点，即使输入错误，也能够避免炸弹爆炸（但偶尔会忘记导致炸弹爆炸）：

```
(gdb) b explode_bomb
Breakpoint 1 at 0x8049257
(gdb)
```

如果操作失误/输入错误答案，会在炸弹前停住：

```
(gdb) r
Starting program: /home/zty/Desktop/bomb71/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
What a great TA!

Breakpoint 1, 0x08049257 in explode_bomb ()
(gdb)
```

4.1 第一关

首先根据拆弹操作指导一顿操作破了第一关。

然后回头细看操作和汇编代码的细节。<phase_1>开始的堆栈操作选择性忽略（现在也没搞太明白），然后推入了一个陌生的地址。接着调用<strings_not_equal>函数（直觉告诉我它就是比较两个字符串，相等返回1，不相等返回0），下方test操作数均为寄存器eax，即将eax按位与，若eax为0，则ZF置0，否则置1。然后是jne（jump if not equal）指令，若eax为0，ZF为0，离开函数，否则跳转至炸弹<explode_bomb>。

分析到这里，<phase_1>就是要输入陌生地址处的字符串。

先在push陌生地址之后的0x8048b65打断点：

```
(gdb) b *0x8048b65
Breakpoint 2 at 0x8048b65
(gdb)
```

然后运行，随意输入一个答案，程序在该断点停住，打印出陌生地址的字符串：

```
(gdb) r
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/zty/Desktop/bomb71/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
What a great TA!

Breakpoint 2, 0x08048b65 in phase_1 ()
(gdb) x/s 0x804a204
0x804a204:      "There are rumors on the internets."
(gdb)
```

我们获得了第一关的答案，重新运行，并输入该值，在断点停住，继续运行：

```
(gdb) r
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/zty/Desktop/bomb71/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
There are rumors on the internets.

Breakpoint 2, 0x08048b65 in phase_1 ()
(gdb) c
Continuing.
Phase 1 defused. How about the next one?
```

将本关答案“There are rumors on the internets.”保存在solution.txt中。

4.2 第二关

<phase_2>一开始先堆栈及字符串操作，然后调用<read_six_numbers>函数将字符串转为6个数字。接着比较5和第一个输入数，若无符号大于则调转至炸弹，故我们知道第一个数要小于等于5。然后将循环变量ebx赋值为1，跳至循环节。进入循环，将xi（i=ebx）赋值给eax，将ebx赋值给ecx，再将eax左移ecx的低16位，即左移ecx位，也即将eax乘上2^ebx。最后将eax和xi+1比较，若相等则继续循环，否则爆炸。

故有xi+1=xi*(2^i)，x1<=5。我们不妨取x1为1，经计算得六数为1、2、8、64、1024、32768。

直接常规gdb运行并输入答案：

```
(gdb) r
Starting program: /home/zty/Desktop/bomb71/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
There are rumors on the internets.
Phase 1 defused. How about the next one?
1 2 8 64 1024 32768

That's number 2.  Keep going!
```

顺利过关，将本关答案“1 2 8 64 1024 32768”保存在 solution.txt 中。

4.3 第三关

<phase_3>一开始堆栈及字符串操作，然后调用< __isoc99_sscanf@plt>函数将字符串转换为整数，接着将输入整数个数与 1 比较，若小于等于 1 则爆炸，再结合偏移寻址猜测输入应该为两个整数。再将第一个数和 7 比较，无符号大于则跳转至炸弹，即第一个数要小于等于 7。又将第一个数赋值给 eax，然后根据地址表偏移跳转，跳转的地址为0x804a260+4*($eax)。在跳转指令之前设置断点，先输入一个随意的答案，通过查该地址的跳转表获得下一步的地址（因为第一个输入小于 7，只打 8 个连续地址对应的 16 进制地址）：

```
(gdb) b *0x8048c23
Breakpoint 2 at 0x8048c23
```

```
(gdb) x/8x 0x804a260
0x804a260:      0x08048c31      0x08048c38      0x08048c7e      0x08048c85
0x804a270:      0x08048c8c      0x08048c93      0x08048c9a      0x08048ca1
(gdb)
```

理论上这几个跳转地址都是可行的，但在后文中还添加了一个限制第一个输入必须要小于 5，所以第一个输入可以为 0、1、2、3、4。这里不妨取第一个输入为 0，后续沿汇编代码对 eax 进行赋值 0x197，再-0x12a+0x1a4-0x88+0x88-0x88+0x88-0x88，最后 eax 为 393，故第二个输入为 393。

常规运行并输入答案：

```
Starting program: /home/zty/Desktop/bomb71/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
There are rumors on the internets.
1 2 8 64 1024 32768
0 393
Phase 1 defused. How about the next one?
That's number 2.  Keep going!

Breakpoint 2, 0x08048c23 in phase_3 ()
(gdb) c
Continuing.
Halfway there!
```

顺利过关，将本关答案“0 393”保存在 solution.txt 中。

4.4 第四关

和上一关类似，<phase_4>一开始堆栈及字符串操作，然后调用< __isoc99_sscanf@plt>函数将字符串转换为整数，将输入整数个数和2比较，不相等则跳转至炸弹。但与上一题不同的是，将堆栈地址推入栈的操作顺序刚好反了一下，所以接下来其实是将第二个输入的值赋给eax。（其实这个点卡了我很久，起初分析递归函数，后面将函数当黑箱，但不管怎么做，程序都会爆炸，后来一行一行gdb调试，终于发现是自己的输入数反了）然后eax减2，再与2比较，无符号小于等于则跳转，否则运行至炸弹，即第二个输入要大于等于2小于等于4。接着将第二个输入和6推入栈，即传参，再调用递归函数

<func4>，调用完成后比较eax和第一个输入，相等则跳出，否则运行至炸弹。起初一直在分析递归函数内部调用传参，搞的很复杂，后来突然想明白了，直接把函数当成一个黑箱，出函数后直接查找eax的值就好了，我们不妨取第二个输入为3，先随便输入第一个值，在比较前设置断点，查询eax的值：

```
(gdb) b *0x8048d52
Breakpoint 2 at 0x8048d52
(gdb)
```

```
(gdb) r
Starting program: /home/zty/Desktop/bomb71/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
There are rumors on the internets.
1 2 8 64 1024 32768
0 393
Phase 1 defused. How about the next one?
That's number 2.  Keep going!
Halfway there!
12 3

Breakpoint 2, 0x08048d52 in phase_4 ()
(gdb) p $eax
$1 = 60
(gdb)
```

这样反而不费吹灰之力获得了此题的答案，实际上，经过测试，40 2、60 3、80 4的答案都是可以的，估计递归调用返回20倍的第二个输入。

常规运行并输入答案：

```
(gdb) r
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/zty/Desktop/bomb71/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
There are rumors on the internets.
1 2 8 64 1024 32768
0 393
60 3
Phase 1 defused. How about the next one?
That's number 2.  Keep going!
Halfway there!

Breakpoint 2, 0x08048d52 in phase_4 ()
(gdb) c
Continuing.
So you got that one.  Try this one.
```

顺利过关，将本关答案“60 3”保存在 solution.txt 中。

4.5 第五关

<phase_5>一开始堆栈操作，然后将指向字符串的指针赋值给ebx。函数<string_length>获取字符串长度，存入eax中，并和6比较，不相等则跳转至炸弹。所以我

们应该输入一个长度为6的字符串。接着将ebx的值（指向字符串的指针）赋给eax，将ebx加6，即字符串末尾后一个位置，将ecx赋值为0。再将根据eax所存地址寻址得到的字符赋值给edx，然后将edx和1111（2）与运算，即取edx低半字节。然后是关键的一步：将0x804a280+4*edx寻址后加给ecx。这里的0x804a280是一个陌生地址。不妨先在这条语句前设置断点，随意输入一个答案然后打印出这段连续地址存的值（由加操作猜测应该是一个整数）：

```
(gdb) b *0x8048d97
Breakpoint 2 at 0x8048d97
```

```
So you got that one.  Try this one.
123456

Breakpoint 2, 0x08048d97 in phase_5 ()
(gdb) x/16x 0x804a280
0x804a280 <array.3033>: 0x00000002      0x0000000a      0x00000006      0x00000
001
0x804a290 <array.3033+16>:      0x0000000c      0x00000010      0x00000009    0
x00000003
0x804a2a0 <array.3033+32>:      0x00000004      0x00000007      0x0000000e    0
x00000005
0x804a2b0 <array.3033+48>:      0x0000000b      0x00000008      0x0000000f    0
x0000000d
(gdb) x/32x 0x804a280
0x804a280 <array.3033>: 0x00000002      0x0000000a      0x00000006      0x00000
001
0x804a290 <array.3033+16>:      0x0000000c      0x00000010      0x00000009    0
x00000003
0x804a2a0 <array.3033+32>:      0x00000004      0x00000007      0x0000000e    0
x00000005
0x804a2b0 <array.3033+48>:      0x0000000b      0x00000008      0x0000000f    0
x0000000d
0x804a2c0:      0x79206f53      0x7420756f      0x6b6e6968      0x756f7920
0x804a2d0:      0x6e616320      0x6f747320      0x68742070      0x6f622065
0x804a2e0:      0x7720626d      0x20687469      0x6c727463      0x202c632d
0x804a2f0:      0x79206f64      0x003f756f      0x4f525245      0x49203a52
(gdb)
```

由图片可以看到从该陌生地址开始连续存了16个数，根据不同的偏移值可以寻址到不同的值，再加到ecx上。接下来eax++，然后和ebx比较，不等于则继续循环，等于则将ecx与42比较。若继续循环则再根据eax所存地址的寻址结果低半字进行偏移寻址。6次循环后，获得6个数的和，再和42比较，等于则跳出，不等于则运行炸弹。综合上面的分析，首先要在表中找到6个数的和为42。然后根据他们的地址获得偏移量，这也就获得了我们应该要输入的字符串的每个字符的低半字节。答案有很多，这里不妨取6*7=42。7的偏移量为9，然后在ASII码表中寻找低半字节为1001的字符，发现’,’是一个。所以一个答案是“,,,,,,”。

常规运行并输入答案：

```
(gdb) r
Starting program: /home/zty/Desktop/bomb71/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
There are rumors on the internets.
1 2 8 64 1024 32768
0 393
60 3
Phase 1 defused. How about the next one?
That's number 2.  Keep going!
Halfway there!
So you got that one.  Try this one.
,,,,,,
Good work!  On to the next...
```

顺利过关，将本关答案“,,,,,,”保存在 solution.txt 中。

4.6 第六关

<phase_6>一开始堆栈操作，然后同前将字符串转为6个数字，开始进入循环，先将循环变量esi赋值为0，然后将xi(i=1:6)的值赋给eax，接着eax--，将eax和5比较，无符号大于则跳转至炸弹，即每一个输入都要小于等于6，大于等于1。

再esi++，将esi与6比较，等于则跳转出嵌套循环，否则将esi（循环变量）赋值给ebx，再进行跳转进入内循环。进入内循环先将xj(j=2:6)的值赋给eax，再将eax与xi比较，即将xj与xi比较，不相等则继续内循环，相等则运行至炸弹，即xi不能和它后面的任何数相同。由此知道输入六数各不相同，结合前面的每一个输入都要小于等于6，大于等于1，输入六数为1、2、3、4、5、6，但顺序还未知。

跳出嵌套循环后将第一个数的地址赋给eax，将第六个数后面一个的地址赋给ebx，将ecx赋值为7，再将ecx赋值给edx，即7。用edx减去eax寻址后的值，接着将eax所存地址指向的值改为edx的值，即xi=7-xi。eax加4，即改为指向下一个数，将eax和ebx比较，不相等则继续循环。这样的循环过后所以xi都变为7-xi然后。将ebx赋为0，跳转至新循环。先将ebx赋值给esi，将xi（i=1:6）赋值给ecx，将eax赋为1，将一个陌生地址赋值给edx，比较ecx和1，有符号大于则跳转，进行内循环，否则继续外循环。和上面几个例子一样，先来查看一下这个地址存放的东西，根据实验指导知道这个地址很有可能是链表头。

先在这个地址操作前打断点，然后运行，先随意输入一个答案:

```
(gdb) b *0x8048e5a
Breakpoint 2 at 0x8048e5a
(gdb)
```

```
(gdb) r
Starting program: /home/zty/Desktop/bomb71/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
There are rumors on the internets.
1 2 8 64 1024 32768
0 393
60 3
,,,,,,
Phase 1 defused. How about the next one?
That's number 2.  Keep going!
Halfway there!
So you got that one.  Try this one.
Good work!  On to the next...
1 2 3 4 5 6

Breakpoint 2, 0x08048e5a in phase_6 ()
(gdb) p *0x804c154@30
$1 = {514, 1, 134529376, 367, 2, 134529388, 206, 3, 134529400, 237, 4,
  134529412, 608, 5, 134529424, 454, 6, 0, 0, 875771953, 892416560, 0, 0, 0,
  0, 0, 0, 0, 0, 0}
(gdb)
```

（P * 0x804c154@30这个操作是同学告诉我的，真的是非常好用，我自己做的时候是根据循环一个一个gdb指令查询。）

可以看到这个链表只有6个节点，每个节点第一个数放的是值，后面一个放的的节点的序列号，第三个放的是指向下一个节点的指针。

继续分析汇编代码，进入内循环，先将edx所存地址偏移寻址后赋回edx，即将edx存的指针改为指向的节点存的下一个地址，然后eax++，比较eax和ecx（xi），不相等则继续内循环，否则将edx此时所存的地址值压入x6后面的第esi（1~6）个位置，接着ebx++，比较6和ebx，相等则跳出嵌套循环，否则继续外循环。综合嵌套循环的分析，压入栈的新的六个值实际上是指向第xi个节点（即原输入的7-xi）的地址。

跳出循环后，将x6后面第一个地址存放的地址值（链表头）赋给ebx，将x6后面第二个地址存放的地址值赋给ebx，将eax存入ebx所指节点的地址域，然后是重复上面的操作，总的说是构建链表，最后一个节点的指针域放0（NULL）。再一次把循环变量esi赋为5，进入新循环，将ebx所指节点的地址域存入eax，eax自寻址，将eax与ebx所指节点的值比较，有符号大于等于则跳转继续循环，否则运行炸弹。若继续循环esi--，若等于0则跳出循环。所以新构建的链表的前一个节点值要大于后一个节点值。至此，此题已经明晰了，就是一个链表排序。回过头看gdb查看到的链表值，由大到小是608、514、454、367、237、206，对应序号分别是5、1、6、2、4、3。所以最后的答案为2、6、1、5、3、4。

常规运行并输入答案：

```
(gdb) r
Starting program: /home/zty/Desktop/bomb71/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
There are rumors on the internets.
1 2 8 64 1024 32768
0 393
60 3
,,,,,,
Phase 1 defused. How about the next one?
That's number 2.  Keep going!
Halfway there!
So you got that one.  Try this one.
Good work!  On to the next...
2 6 1 5 3 4
Congratulations! You've defused the bomb!
Your instructor has been notified and will verify your solution.
[Inferior 1 (process 33088) exited normally]
(gdb)
```

顺利过关，将本关答案“2 6 1 5 3 4”保存在 solution.txt 中。

4.7 隐藏关

说实话，这一关开始自己看了半天都没什么头绪，是经过同学提醒得到的。

在每一个<phase_i>函数结束后，都会调用一次< phase_defused>函数，但里面堆栈、传参及调用外部函数的操作很频繁，没有看出个所以然。里面也有很多陌生的地址，不知道它存了什么东西。于是开始疯狂设断点，一步一步地用十进制，十六进制，字符串形式打印陌生地址及它连续的地址存放的值，终于在一个地方看到了曙光：

```
Breakpoint 2 at 0x8049421
(gdb) r
```

```
Starting program: /home/zty/Desktop/bomb71/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
There are rumors on the internets.
1 2 8 64 1024 32768
0 393
60 3
,,,,,,
2 6 1 5 3 4
Phase 1 defused. How about the next one?
That's number 2.  Keep going!
Halfway there!
So you got that one.  Try this one.
Good work!  On to the next...

Breakpoint 2, 0x08049421 in phase_defused ()
(gdb) x/s 0x804a4eb
0x804a4eb:      "%d %d %s"
(gdb) x/s 0x804c8f0
0x804c8f0 <input_strings+240>:  "60 3"
(gdb)
```

由这里可以看到，< phase_defused>函数只有在完成6关以后才能够正常运行，前面两个陌生地址值，第一个表示扫入两个整数和一个字符串，后面一个居然是第四关的答案。

由此不免产生想法，可能是在第四关的答案后面加一个字符串才能居然隐藏关卡。但这个字符串是什么还不知道，继续查询陌生地址值：

```
(gdb) x/s 0x804a3a0
0x804a3a0:      "Congratulations! You've defused the bomb!"
(gdb) x/s 0x804a4f4
0x804a4f4:      "SecretSYSU"
(gdb) x/s 0x804a340
0x804a340:      "Curses, you've found the secret phase!"
(gdb)
```

前后两个字符串都是很正常的恭喜，中间这个怎么看怎么想是密码。于是尝试在第四关的后面加上字符串“SecretSYSU”：

```
(gdb) r
Starting program: /home/zty/Desktop/bomb71/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
There are rumors on the internets.
1 2 8 64 1024 32768
0 393
60 3 SecretSYSU
,,,,,,
2 6 1 5 3 4
Phase 1 defused. How about the next one?
That's number 2.  Keep going!
Halfway there!
So you got that one.  Try this one.
Good work!  On to the next...
Curses, you've found the secret phase!
But finding it and solving it are quite different...
```

果然成功进入隐藏关卡，现在开始分析< secret _phase >。先还是堆栈操作，然后调用了一个奇怪的函数，开始第一次关键判断，将eax和0x3e8比较，大于则爆炸。但eax在何处被怎么修改没弄清楚，不过试了两个值后发现这个判断并不会那么容易爆炸，就先选择性忽略了。然后遇到了一个奇怪的地址，无脑查询：

```
Breakpoint 2, 0x08048f45 in secret_phase ()
(gdb) p *0x804c0a0
$2 = 36
```

这个值即使改变输入也不发生变化，目测是个定值，然后它被推入栈中，接着调用函数<fun7>。进入函数一步一步设短点看寄存器值：

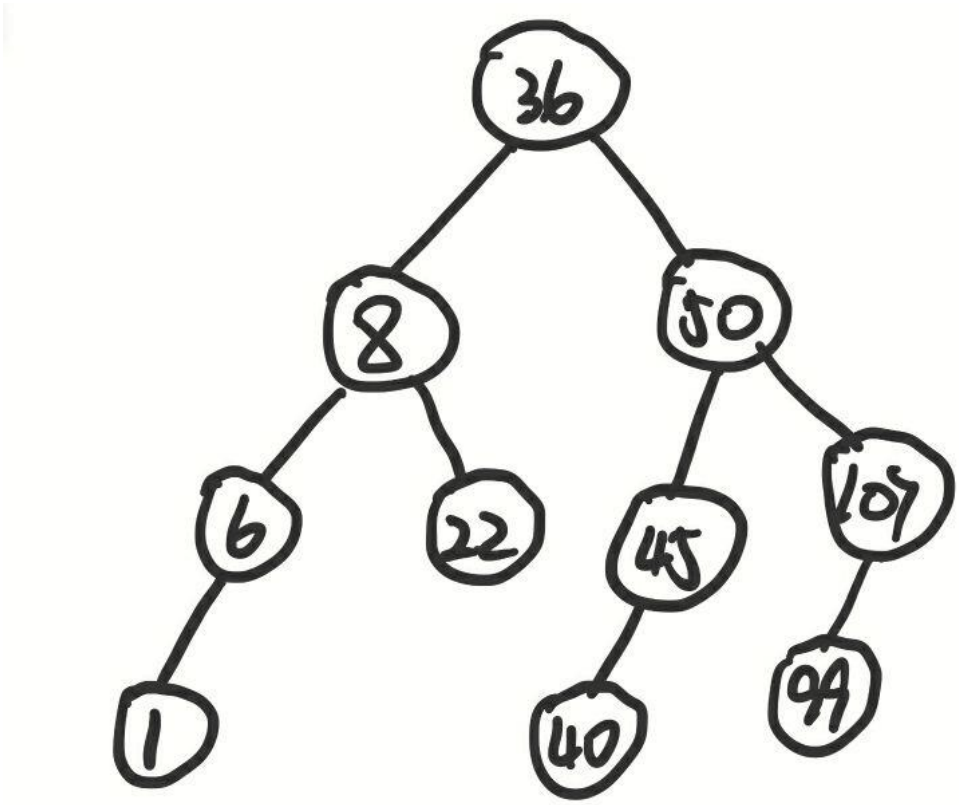
```
(gdb) b *0x8048ed2
Breakpoint 3 at 0x8048ed2
(gdb) c
Continuing.

Breakpoint 3, 0x08048ed2 in fun7 ()
(gdb) p $edx
$5 = 134529184
(gdb) p/x $edx
$6 = 0x804c0a0
(gdb) p/x $ecx
$7 = 0x14
(gdb) p $ecx
$8 = 20
(gdb)
```

发现edx存的是36的地址，ecx存的是我们的输入（这里为随便输入的一个值20）。然后edx不为0则不跳转继续运行，若为0则跳转将eax赋为0xffffffff，然后再跳转离开（目测因为eax不为1爆炸）。edx因为存的是0x804c0a0，不跳转，然后ebx赋为36，将ecx和ebx比较，有符号大于则跳转递归调用，若跳转则传入两个参数递归调用，否则eax赋为0，将ecx和ebx比较，同样递归调用。在传参压栈时，发现edx有偏移寻址，于是查找edx的后面连续地址：

```
(gdb) p *0x804c0a0@30
$13 = {36, 134529196, 134529208, 8, 134529244, 134529220, 50, 134529232,
  134529256, 22, 134529328, 134529304, 45, 134529268, 134529340, 6,
  134529280, 134529316, 107, 134529292, 134529352, 40, 0, 0, 1, 0, 0, 99, 0,
  0}
(gdb)
```

发现edx是树的根地址，每个节点第一个数存的是节点数值，第二个数为左子树地址，第三个数为右子树地址。得到树如下：



继续分析代码，若输入值和各个节点值比较，从下往上，若沿左向上则eax赋为0再递归再让eax=eax*2+1，直到节点值和输入相等。在< secret _phase >函数中我们知道成功的条件是eax = 1。综合分析树和递归，得到输入40、45、50时eax都返回1。（其实得到树后就可以开始遍历了233）

常规运行并输入答案：

```
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/zty/Desktop/bomb71/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
There are rumors on the internets.
1 2 8 64 1024 32768
0 393
60 3 SecretSYSU
,,,,,,
2 6 1 5 3 4
Phase 1 defused. How about the next one?
That's number 2.  Keep going!
Halfway there!
So you got that one.  Try this one.
Good work!  On to the next...
Curses, you've found the secret phase!
But finding it and solving it are quite different...
45
Wow! You've defused the secret stage!
Congratulations! You've defused the bomb!
Your instructor has been notified and will verify your solution.
[Inferior 1 (process 34632) exited normally]
(gdb)
```

顺利过关，将本关答案第四关附加字符串“SecretSYSU”“45”保存在 solution.txt 中。

至此，实验完美结束。

五. 实验心得

对于这个实验刚开始我是拒绝的，一直觉得比较底层，会不会太难，开始还准备好好看汇编语言的书再来进行。但国庆一趟6天的外出让计划泡汤，回学校后已没有时间进行知识储备，只能硬着头皮上，开始几道题还是简单的，半蒙半猜还是得到了答案，从第5关开始遇到比较大的困难，后来和同学交流，得到了很多很好的gdb指令，让拆炸弹的速度大大提升。正所谓：“工欲善其事，必先利其器。”真应该一开始就好好学一下gdb的各种指令。只拿大一储备的gdb指令来做题目实在是比较慢。

我觉得整个实验真的非常不错，每一关其实都是教我们如何将C语言变成汇编语言，程序底层的实现究竟是什么样的，从这个角度能够更透彻的理解程序。比如switch的实现原来是靠一个内部的表来跳转，真的是印象深刻。

但我拆弹的时间比较仓促，有些汇编代码的理解我都选择性略过了，尤其是一些外部函数的具体细节和堆栈的操作，但由于这样囫囵吞枣的方式，导致我对函数传参，函数调用的方式还是有些不清楚，递归的分析也不是太清楚。虽然实验结束了，但带着这些疑问，我觉得还是有必要再去多了解一下这些内容。以后也可以和这个实验的代码相对照。

【程序代码】

./bomb: file format elf32-i386

Disassembly of section .init:

080486f4 <_init>:

80486f4: 53 push %ebx

80486f5: 83 ec 08 sub $0x8,%esp

80486f8: e8 33 02 00 00 call 8048930 <__x86.get_pc_thunk.bx>

80486fd: 81 c3 03 39 00 00 add $0x3903,%ebx

8048703: 8b 83 fc ff ff ff mov -0x4(%ebx),%eax

8048709: 85 c0 test %eax,%eax

804870b: 74 05 je 8048712 <_init+0x1e>

804870d: e8 be 01 00 00 call 80488d0 <__gmon_start__@plt>

8048712: 83 c4 08 add $0x8,%esp

8048715: 5b pop %ebx

8048716: c3 ret

Disassembly of section .plt:

08048720 <.plt>:

8048720: ff 35 04 c0 04 08 pushl 0x804c004

8048726: ff 25 08 c0 04 08 jmp *0x804c008

804872c: 00 00 add %al,(%eax)

…

08048730 <read@plt>:

```
 8048730:  ff 25 0c c0 04 08          jmp     *0x804c00c
 8048736:  68 00 00 00 00             push    $0x0
 804873b:  e9 e0 ff ff ff             jmp     8048720 <.plt>

08048740 <fflush@plt>:
 8048740:  ff 25 10 c0 04 08          jmp     *0x804c010
 8048746:  68 08 00 00 00             push    $0x8
 804874b:  e9 d0 ff ff ff             jmp     8048720 <.plt>

08048750 <fgets@plt>:
 8048750:  ff 25 14 c0 04 08          jmp     *0x804c014
 8048756:  68 10 00 00 00             push    $0x10
 804875b:  e9 c0 ff ff ff             jmp     8048720 <.plt>

08048760 <signal@plt>:
 8048760:  ff 25 18 c0 04 08          jmp     *0x804c018
 8048766:  68 18 00 00 00             push    $0x18
 804876b:  e9 b0 ff ff ff             jmp     8048720 <.plt>

08048770 <sleep@plt>:
 8048770:  ff 25 1c c0 04 08          jmp     *0x804c01c
 8048776:  68 20 00 00 00             push    $0x20
 804877b:  e9 a0 ff ff ff             jmp     8048720 <.plt>

08048780 <alarm@plt>:
 8048780:  ff 25 20 c0 04 08          jmp     *0x804c020
 8048786:  68 28 00 00 00             push    $0x28
 804878b:  e9 90 ff ff ff             jmp     8048720 <.plt>
```

```
08048790 <__stack_chk_fail@plt>:
 8048790:  ff 25 24 c0 04 08          jmp     *0x804c024
 8048796:  68 30 00 00 00             push    $0x30
 804879b:  e9 80 ff ff ff             jmp     8048720 <.plt>

080487a0 <strcpy@plt>:
 80487a0:  ff 25 28 c0 04 08          jmp     *0x804c028
 80487a6:  68 38 00 00 00             push    $0x38
 80487ab:  e9 70 ff ff ff             jmp     8048720 <.plt>

080487b0 <getenv@plt>:
 80487b0:  ff 25 2c c0 04 08          jmp     *0x804c02c
 80487b6:  68 40 00 00 00             push    $0x40
 80487bb:  e9 60 ff ff ff             jmp     8048720 <.plt>

080487c0 <puts@plt>:
 80487c0:  ff 25 30 c0 04 08          jmp     *0x804c030
 80487c6:  68 48 00 00 00             push    $0x48
 80487cb:  e9 50 ff ff ff             jmp     8048720 <.plt>

080487d0 <__memmove_chk@plt>:
 80487d0:  ff 25 34 c0 04 08          jmp     *0x804c034
 80487d6:  68 50 00 00 00             push    $0x50
 80487db:  e9 40 ff ff ff             jmp     8048720 <.plt>

080487e0 <exit@plt>:
 80487e0:  ff 25 38 c0 04 08          jmp     *0x804c038
 80487e6:  68 58 00 00 00             push    $0x58
 80487eb:  e9 30 ff ff ff             jmp     8048720 <.plt>
```

```
080487f0 <__libc_start_main@plt>:
 80487f0:   ff 25 3c c0 04 08          jmp     *0x804c03c
 80487f6:   68 60 00 00 00             push    $0x60
 80487fb:   e9 20 ff ff ff             jmp     8048720 <.plt>

08048800 <write@plt>:
 8048800:   ff 25 40 c0 04 08          jmp     *0x804c040
 8048806:   68 68 00 00 00             push    $0x68
 804880b:   e9 10 ff ff ff             jmp     8048720 <.plt>

08048810 <__isoc99_sscanf@plt>:
 8048810:   ff 25 44 c0 04 08          jmp     *0x804c044
 8048816:   68 70 00 00 00             push    $0x70
 804881b:   e9 00 ff ff ff             jmp     8048720 <.plt>

08048820 <fopen@plt>:
 8048820:   ff 25 48 c0 04 08          jmp     *0x804c048
 8048826:   68 78 00 00 00             push    $0x78
 804882b:   e9 f0 fe ff ff             jmp     8048720 <.plt>

08048830 <__errno_location@plt>:
 8048830:   ff 25 4c c0 04 08          jmp     *0x804c04c
 8048836:   68 80 00 00 00             push    $0x80
 804883b:   e9 e0 fe ff ff             jmp     8048720 <.plt>

08048840 <__printf_chk@plt>:
 8048840:   ff 25 50 c0 04 08          jmp     *0x804c050
 8048846:   68 88 00 00 00             push    $0x88
```

```
 804884b:   e9 d0 fe ff ff          jmp     8048720 <.plt>

08048850 <socket@plt>:
 8048850:   ff 25 54 c0 04 08       jmp     *0x804c054
 8048856:   68 90 00 00 00          push    $0x90
 804885b:   e9 c0 fe ff ff          jmp     8048720 <.plt>

08048860 <__fprintf_chk@plt>:
 8048860:   ff 25 58 c0 04 08       jmp     *0x804c058
 8048866:   68 98 00 00 00          push    $0x98
 804886b:   e9 b0 fe ff ff          jmp     8048720 <.plt>

08048870 <gethostbyname@plt>:
 8048870:   ff 25 5c c0 04 08       jmp     *0x804c05c
 8048876:   68 a0 00 00 00          push    $0xa0
 804887b:   e9 a0 fe ff ff          jmp     8048720 <.plt>

08048880 <strtol@plt>:
 8048880:   ff 25 60 c0 04 08       jmp     *0x804c060
 8048886:   68 a8 00 00 00          push    $0xa8
 804888b:   e9 90 fe ff ff          jmp     8048720 <.plt>

08048890 <connect@plt>:
 8048890:   ff 25 64 c0 04 08       jmp     *0x804c064
 8048896:   68 b0 00 00 00          push    $0xb0
 804889b:   e9 80 fe ff ff          jmp     8048720 <.plt>

080488a0 <close@plt>:
 80488a0:   ff 25 68 c0 04 08       jmp     *0x804c068
```

```
 80488a6:   68 b8 00 00 00          push   $0xb8
 80488ab:   e9 70 fe ff ff          jmp    8048720 <.plt>

080488b0 <__ctype_b_loc@plt>:
 80488b0:   ff 25 6c c0 04 08       jmp    *0x804c06c
 80488b6:   68 c0 00 00 00          push   $0xc0
 80488bb:   e9 60 fe ff ff          jmp    8048720 <.plt>

080488c0 <__sprintf_chk@plt>:
 80488c0:   ff 25 70 c0 04 08       jmp    *0x804c070
 80488c6:   68 c8 00 00 00          push   $0xc8
 80488cb:   e9 50 fe ff ff          jmp    8048720 <.plt>

Disassembly of section .plt.got:

080488d0 <__gmon_start__@plt>:
 80488d0:   ff 25 fc bf 04 08       jmp    *0x804bffc
 80488d6:   66 90                   xchg   %ax,%ax

Disassembly of section .text:

080488e0 <_start>:
 80488e0:   31 ed                   xor    %ebp,%ebp
 80488e2:   5e                      pop    %esi
 80488e3:   89 e1                   mov    %esp,%ecx
 80488e5:   83 e4 f0                and    $0xfffffff0,%esp
 80488e8:   50                      push   %eax
 80488e9:   54                      push   %esp
 80488ea:   52                      push   %edx
```

```
80488eb:   e8 23 00 00 00          call    8048913 <_start+0x33>
80488f0:   81 c3 10 37 00 00       add     $0x3710,%ebx
80488f6:   8d 83 90 e0 ff ff       lea     -0x1f70(%ebx),%eax
80488fc:   50                      push    %eax
80488fd:   8d 83 30 e0 ff ff       lea     -0x1fd0(%ebx),%eax
8048903:   50                      push    %eax
8048904:   51                      push    %ecx
8048905:   56                      push    %esi
8048906:   c7 c0 f6 89 04 08       mov     $0x80489f6,%eax
804890c:   50                      push    %eax
804890d:   e8 de fe ff ff          call    80487f0 <__libc_start_main@plt>
8048912:   f4                      hlt
8048913:   8b 1c 24                mov     (%esp),%ebx
8048916:   c3                      ret
8048917:   66 90                   xchg    %ax,%ax
8048919:   66 90                   xchg    %ax,%ax
804891b:   66 90                   xchg    %ax,%ax
804891d:   66 90                   xchg    %ax,%ax
804891f:   90                      nop

08048920 <_dl_relocate_static_pie>:
8048920:   f3 c3                   repz ret
8048922:   66 90                   xchg    %ax,%ax
8048924:   66 90                   xchg    %ax,%ax
8048926:   66 90                   xchg    %ax,%ax
8048928:   66 90                   xchg    %ax,%ax
804892a:   66 90                   xchg    %ax,%ax
804892c:   66 90                   xchg    %ax,%ax
804892e:   66 90                   xchg    %ax,%ax
```

```
08048930 <__x86.get_pc_thunk.bx>:
 8048930:  8b 1c 24          mov    (%esp),%ebx
 8048933:  c3                ret
 8048934:  66 90             xchg   %ax,%ax
 8048936:  66 90             xchg   %ax,%ax
 8048938:  66 90             xchg   %ax,%ax
 804893a:  66 90             xchg   %ax,%ax
 804893c:  66 90             xchg   %ax,%ax
 804893e:  66 90             xchg   %ax,%ax

08048940 <deregister_tm_clones>:
 8048940:  b8 c0 c7 04 08    mov    $0x804c7c0,%eax
 8048945:  3d c0 c7 04 08    cmp    $0x804c7c0,%eax
 804894a:  74 24             je     8048970
<deregister_tm_clones+0x30>
 804894c:  b8 00 00 00 00    mov    $0x0,%eax
 8048951:  85 c0             test   %eax,%eax
 8048953:  74 1b             je     8048970
<deregister_tm_clones+0x30>
 8048955:  55                push   %ebp
 8048956:  89 e5             mov    %esp,%ebp
 8048958:  83 ec 14          sub    $0x14,%esp
 804895b:  68 c0 c7 04 08    push   $0x804c7c0
 8048960:  ff d0             call   *%eax
 8048962:  83 c4 10          add    $0x10,%esp
 8048965:  c9                leave
 8048966:  c3                ret
 8048967:  89 f6             mov    %esi,%esi
```

```
 8048969:  8d bc 27 00 00 00 00    lea    0x0(%edi,%eiz,1),%edi
 8048970:  f3 c3                   repz ret
 8048972:  8d b4 26 00 00 00 00    lea    0x0(%esi,%eiz,1),%esi
 8048979:  8d bc 27 00 00 00 00    lea    0x0(%edi,%eiz,1),%edi

08048980 <register_tm_clones>:
 8048980:  b8 c0 c7 04 08          mov    $0x804c7c0,%eax
 8048985:  2d c0 c7 04 08          sub    $0x804c7c0,%eax
 804898a:  c1 f8 02                sar    $0x2,%eax
 804898d:  89 c2                   mov    %eax,%edx
 804898f:  c1 ea 1f                shr    $0x1f,%edx
 8048992:  01 d0                   add    %edx,%eax
 8048994:  d1 f8                   sar    %eax
 8048996:  74 20                   je     80489b8
<register_tm_clones+0x38>
 8048998:  ba 00 00 00 00          mov    $0x0,%edx
 804899d:  85 d2                   test   %edx,%edx
 804899f:  74 17                   je     80489b8
<register_tm_clones+0x38>
 80489a1:  55                      push   %ebp
 80489a2:  89 e5                   mov    %esp,%ebp
 80489a4:  83 ec 10                sub    $0x10,%esp
 80489a7:  50                      push   %eax
 80489a8:  68 c0 c7 04 08          push   $0x804c7c0
 80489ad:  ff d2                   call   *%edx
 80489af:  83 c4 10                add    $0x10,%esp
 80489b2:  c9                      leave
 80489b3:  c3                      ret
 80489b4:  8d 74 26 00             lea    0x0(%esi,%eiz,1),%esi
```

```
80489b8:  f3 c3                    repz ret
80489ba:  8d b6 00 00 00 00        lea    0x0(%esi),%esi

080489c0 <__do_global_dtors_aux>:
80489c0:  80 3d e8 c7 04 08 00     cmpb   $0x0,0x804c7e8
80489c7:  75 17                    jne                              80489e0
<__do_global_dtors_aux+0x20>
80489c9:  55                       push   %ebp
80489ca:  89 e5                    mov    %esp,%ebp
80489cc:  83 ec 08                 sub    $0x8,%esp
80489cf:  e8 6c ff ff ff           call   8048940 <deregister_tm_clones>
80489d4:  c6 05 e8 c7 04 08 01     movb   $0x1,0x804c7e8
80489db:  c9                       leave
80489dc:  c3                       ret
80489dd:  8d 76 00                 lea    0x0(%esi),%esi
80489e0:  f3 c3                    repz ret
80489e2:  8d b4 26 00 00 00 00     lea    0x0(%esi,%eiz,1),%esi
80489e9:  8d bc 27 00 00 00 00     lea    0x0(%edi,%eiz,1),%edi

080489f0 <frame_dummy>:
80489f0:  55                       push   %ebp
80489f1:  89 e5                    mov    %esp,%ebp
80489f3:  5d                       pop    %ebp
80489f4:  eb 8a                    jmp    8048980 <register_tm_clones>

080489f6 <main>:
80489f6:  8d 4c 24 04              lea    0x4(%esp),%ecx
80489fa:  83 e4 f0                 and    $0xfffffff0,%esp
80489fd:  ff 71 fc                 pushl  -0x4(%ecx)
```

```
8048a00:  55                    push   %ebp
8048a01:  89 e5                 mov    %esp,%ebp
8048a03:  53                    push   %ebx
8048a04:  51                    push   %ecx
8048a05:  8b 01                 mov    (%ecx),%eax
8048a07:  8b 59 04              mov    0x4(%ecx),%ebx
8048a0a:  83 f8 01              cmp    $0x1,%eax
8048a0d:  0f 84 fe 00 00 00     je     8048b11 <main+0x11b>
8048a13:  83 f8 02              cmp    $0x2,%eax
8048a16:  0f 85 21 01 00 00     jne    8048b3d <main+0x147>
8048a1c:  83 ec 08              sub    $0x8,%esp
8048a1f:  68 c8 a0 04 08        push   $0x804a0c8
8048a24:  ff 73 04              pushl  0x4(%ebx)
8048a27:  e8 f4 fd ff ff        call   8048820 <fopen@plt>
8048a2c:  a3 f0 c7 04 08        mov    %eax,0x804c7f0
8048a31:  83 c4 10              add    $0x10,%esp
8048a34:  85 c0                 test   %eax,%eax
8048a36:  0f 84 e4 00 00 00     je     8048b20 <main+0x12a>
8048a3c:  e8 46 06 00 00        call   8049087 <initialize_bomb>
8048a41:  83 ec 0c              sub    $0xc,%esp
8048a44:  68 4c a1 04 08        push   $0x804a14c
8048a49:  e8 72 fd ff ff        call   80487c0 <puts@plt>
8048a4e:  c7 04 24 88 a1 04 08  movl   $0x804a188,(%esp)
8048a55:  e8 66 fd ff ff        call   80487c0 <puts@plt>
8048a5a:  e8 6c 08 00 00        call   80492cb <read_line>;每次关卡开始都会读入一行字符串，并传入phase_n函数
8048a5f:  89 04 24              mov    %eax,(%esp)
8048a62:  e8 f3 00 00 00        call   8048b5a <phase_1>;调用phase_1函数
```

8048a67: e8 70 09 00 00 call 80493dc <phase_defused>;每次关卡结束都会调用这个函数，和隐藏关卡有关

8048a6c: c7 04 24 b4 a1 04 08 movl $0x804a1b4,(%esp)

8048a73: e8 48 fd ff ff call 80487c0 <puts@plt>

8048a78: e8 4e 08 00 00 call 80492cb <read_line>

8048a7d: 89 04 24 mov %eax,(%esp)

8048a80: e8 f8 00 00 00 call 8048b7d <phase_2>

8048a85: e8 52 09 00 00 call 80493dc <phase_defused>

8048a8a: c7 04 24 01 a1 04 08 movl $0x804a101,(%esp)

8048a91: e8 2a fd ff ff call 80487c0 <puts@plt>

8048a96: e8 30 08 00 00 call 80492cb <read_line>

8048a9b: 89 04 24 mov %eax,(%esp)

8048a9e: e8 45 01 00 00 call 8048be8 <phase_3>

8048aa3: e8 34 09 00 00 call 80493dc <phase_defused>

8048aa8: c7 04 24 1f a1 04 08 movl $0x804a11f,(%esp)

8048aaf: e8 0c fd ff ff call 80487c0 <puts@plt>

8048ab4: e8 12 08 00 00 call 80492cb <read_line>

8048ab9: 89 04 24 mov %eax,(%esp)

8048abc: e8 43 02 00 00 call 8048d04 <phase_4>

8048ac1: e8 16 09 00 00 call 80493dc <phase_defused>

8048ac6: c7 04 24 e0 a1 04 08 movl $0x804a1e0,(%esp)

8048acd: e8 ee fc ff ff call 80487c0 <puts@plt>

8048ad2: e8 f4 07 00 00 call 80492cb <read_line>

8048ad7: 89 04 24 mov %eax,(%esp)

8048ada: e8 90 02 00 00 call 8048d6f <phase_5>

8048adf: e8 f8 08 00 00 call 80493dc <phase_defused>

8048ae4: c7 04 24 2e a1 04 08 movl $0x804a12e,(%esp)

8048aeb: e8 d0 fc ff ff call 80487c0 <puts@plt>

8048af0: e8 d6 07 00 00 call 80492cb <read_line>

```
8048af5:   89 04 24                mov     %eax,(%esp)
8048af8:   e8 be 02 00 00          call    8048dbb <phase_6>
8048afd:   e8 da 08 00 00          call    80493dc <phase_defused>
8048b02:   b8 00 00 00 00          mov     $0x0,%eax
8048b07:   8d 65 f8                lea     -0x8(%ebp),%esp
8048b0a:   59                      pop     %ecx
8048b0b:   5b                      pop     %ebx
8048b0c:   5d                      pop     %ebp
8048b0d:   8d 61 fc                lea     -0x4(%ecx),%esp
8048b10:   c3                      ret
8048b11:   a1 e0 c7 04 08          mov     0x804c7e0,%eax
8048b16:   a3 f0 c7 04 08          mov     %eax,0x804c7f0
8048b1b:   e9 1c ff ff ff          jmp     8048a3c <main+0x46>
8048b20:   ff 73 04                pushl   0x4(%ebx)
8048b23:   ff 33                   pushl   (%ebx)
8048b25:   68 ca a0 04 08          push    $0x804a0ca
8048b2a:   6a 01                   push    $0x1
8048b2c:   e8 0f fd ff ff          call    8048840 <__printf_chk@plt>
8048b31:   c7 04 24 08 00 00 00    movl    $0x8,(%esp)
8048b38:   e8 a3 fc ff ff          call    80487e0 <exit@plt>
8048b3d:   83 ec 04                sub     $0x4,%esp
8048b40:   ff 33                   pushl   (%ebx)
8048b42:   68 e7 a0 04 08          push    $0x804a0e7
8048b47:   6a 01                   push    $0x1
8048b49:   e8 f2 fc ff ff          call    8048840 <__printf_chk@plt>
8048b4e:   c7 04 24 08 00 00 00    movl    $0x8,(%esp)
8048b55:   e8 86 fc ff ff          call    80487e0 <exit@plt>
```

08048b5a <phase_1>:

8048b5a: 55 push %ebp

8048b5b: 89 e5 mov %esp,%ebp

8048b5d: 83 ec 10 sub $0x10,%esp;以上皆为堆栈操作，没有搞太明白

8048b60: 68 04 a2 04 08 push $0x804a204;推入一个陌生的地址，经查询为字符串"There are rumors on the internets."

8048b65: ff 75 08 pushl 0x8(%ebp)

8048b68: e8 b0 04 00 00 call 804901d <strings_not_equal>;没有仔细分析内部代码，但顾名思义猜测字符串相等返回0，否则返回1

8048b6d: 83 c4 10 add $0x10,%esp

8048b70: 85 c0 test %eax,%eax;test为按位与操作，当eax寄存器为0时ZF为1，否则为0

8048b72: 75 02 jne 8048b76 <phase_1+0x1c>;不等于则跳转掉用炸弹函数，至此知道应该输入预设内部的上方字符串

8048b74: c9 leave

8048b75: c3 ret

8048b76: e8 d6 06 00 00 call 8049251 <explode_bomb>

8048b7b: eb f7 jmp 8048b74 <phase_1+0x1a>

08048b7d <phase_2>:

8048b7d: 55 push %ebp

8048b7e: 89 e5 mov %esp,%ebp

8048b80: 53 push %ebx

8048b81: 83 ec 3c sub $0x3c,%esp

8048b84: 65 a1 14 00 00 00 mov %gs:0x14,%eax

8048b8a: 89 45 f4 mov %eax,-0xc(%ebp)

8048b8d: 31 c0 xor %eax,%eax

8048b8f: 8d 45 dc lea -0x24(%ebp),%eax

8048b92: 50 push %eax

8048b93: ff 75 08 pushl 0x8(%ebp);以上堆栈及字符串操作

8048b96: e8 f6 06 00 00 call 8049291 <read_six_numbers>;将字符串转为6个数字

8048b9b: 83 c4 10 add $0x10,%esp

8048b9e: 83 7d dc 05 cmpl $0x5,-0x24(%ebp);比较5和第一个输入数

8048ba2: 77 07 ja 8048bab <phase_2+0x2e>;若无符号大于则调转至炸弹

8048ba4: bb 01 00 00 00 mov $0x1,%ebx;ebx赋值为1，循环变量

8048ba9: eb 0f jmp 8048bba <phase_2+0x3d>;跳至循环节

8048bab: e8 a1 06 00 00 call 8049251 <explode_bomb>

8048bb0: eb f2 jmp 8048ba4 <phase_2+0x27>

8048bb2: 83 c3 01 add $0x1,%ebx;ebx++

8048bb5: 83 fb 06 cmp $0x6,%ebx;ebx和6比较

8048bb8: 74 18 je 8048bd2 <phase_2+0x55>;等于则跳出函数，否则继续循环

8048bba: 8b 44 9d d8 mov -0x28(%ebp,%ebx,4),%eax;eax=xi

8048bbe: 89 45 d4 mov %eax,-0x2c(%ebp)

8048bc1: 89 d9 mov %ebx,%ecx;ecx=ebx

8048bc3: d3 e0 shl %cl,%eax;eax左移ecx的低16位，即左移ecx位

8048bc5: 39 44 9d dc cmp %eax,-0x24(%ebp,%ebx,4);eax与xi+1比较

8048bc9: 74 e7 je 8048bb2 <phase_2+0x35>;等于则重新循环

8048bcb: e8 81 06 00 00 call 8049251 <explode_bomb>;否则爆炸

```
8048bd0:  eb e0                  jmp     8048bb2 <phase_2+0x35>
8048bd2:  8b 45 f4               mov     -0xc(%ebp),%eax
8048bd5:  65 33 05 14 00 00 00   xor     %gs:0x14,%eax
8048bdc:  75 05                  jne     8048be3 <phase_2+0x66>
8048bde:  8b 5d fc               mov     -0x4(%ebp),%ebx
8048be1:  c9                     leave
8048be2:  c3                     ret
8048be3:  e8 a8 fb ff ff         call    8048790 <__stack_chk_fail@plt>

08048be8 <phase_3>:
8048be8:  55                     push    %ebp
8048be9:  89 e5                  mov     %esp,%ebp
8048beb:  83 ec 18               sub     $0x18,%esp
8048bee:  65 a1 14 00 00 00      mov     %gs:0x14,%eax
8048bf4:  89 45 f4               mov     %eax,-0xc(%ebp)
8048bf7:  31 c0                  xor     %eax,%eax
8048bf9:  8d 45 f0               lea     -0x10(%ebp),%eax;将堆栈地址推
入栈
8048bfc:  50                     push    %eax
8048bfd:  8d 45 ec               lea     -0x14(%ebp),%eax;将堆栈地址推
入栈
8048c00:  50                     push    %eax
8048c01:  68 91 a4 04 08         push    $0x804a491
8048c06:  ff 75 08               pushlr   0x8(%ebp);以上堆栈及字符串操
作
8048c09:  e8 02 fc ff ff         call    8048810 <__isoc99_sscanf@plt>;将字
符串转换为整数
8048c0e:  83 c4 10               add     $0x10,%esp
8048c11:  83 f8 01               cmp     $0x1,%eax;此处eax好像是输入的
```

整数个数，和1比较

8048c14: 7e 14 jle 8048c2a <phase_3+0x42>;若有符号小于等于则跳转，即输入数要为2个

8048c16: 83 7d ec 07 cmpl $0x7,-0x14(%ebp);将第一个数和7比较

8048c1a: 0f 87 88 00 00 00 ja 8048ca8 <phase_3+0xc0>;无符号大于则跳转至炸弹，即第一个数要小于等于7

8048c20: 8b 45 ec mov -0x14(%ebp),%eax;将第一个数赋值给eax

8048c23: ff 24 85 60 a2 04 08 jmp *0x804a260(,%eax,4);根据地址表偏移跳转

8048c2a: e8 22 06 00 00 call 8049251 <explode_bomb>

8048c2f: eb e5 jmp 8048c16 <phase_3+0x2e>

8048c31: b8 97 01 00 00 mov $0x197,%eax;以下对eax进行一系列加减操作

8048c36: eb 05 jmp 8048c3d <phase_3+0x55>

8048c38: b8 00 00 00 00 mov $0x0,%eax

8048c3d: 2d 2a 01 00 00 sub $0x12a,%eax

8048c42: 05 a4 01 00 00 add $0x1a4,%eax

8048c47: 2d 88 00 00 00 sub $0x88,%eax

8048c4c: 05 88 00 00 00 add $0x88,%eax

8048c51: 2d 88 00 00 00 sub $0x88,%eax

8048c56: 05 88 00 00 00 add $0x88,%eax

8048c5b: 2d 88 00 00 00 sub $0x88,%eax

8048c60: 83 7d ec 05 cmpl $0x5,-0x14(%ebp)

8048c64: 7f 05 jg 8048c6b <phase_3+0x83>

8048c66: 39 45 f0 cmp %eax,-0x10(%ebp);比较eax和第二个输入

8048c69: 74 05 je 8048c70 <phase_3+0x88>

```
8048c6b:   e8 e1 05 00 00          call    8049251 <8048c66>
8048c70:   8b 45 f4                mov     -0xc(%ebp),%eax
8048c73:   65 33 05 14 00 00 00    xor     %gs:0x14,%eax
8048c7a:   75 38                   jne     8048cb4 <phase_3+0xcc>
8048c7c:   c9                      leave
8048c7d:   c3                      ret
8048c7e:   b8 00 00 00 00          mov     $0x0,%eax;下列多个地址值为跳
转的地址
8048c83:   eb bd                   jmp     8048c42 <phase_3+0x5a>
8048c85:   b8 00 00 00 00          mov     $0x0,%eax
8048c8a:   eb bb                   jmp     8048c47 <phase_3+0x5f>
8048c8c:   b8 00 00 00 00          mov     $0x0,%eax
8048c91:   eb b9                   jmp     8048c4c <phase_3+0x64>
8048c93:   b8 00 00 00 00          mov     $0x0,%eax
8048c98:   eb b7                   jmp     8048c51 <phase_3+0x69>
8048c9a:   b8 00 00 00 00          mov     $0x0,%eax
8048c9f:   eb b5                   jmp     8048c56 <phase_3+0x6e>
8048ca1:   b8 00 00 00 00          mov     $0x0,%eax
8048ca6:   eb b3                   jmp     8048c5b <phase_3+0x73>
8048ca8:   e8 a4 05 00 00          call    8049251 <explode_bomb>
8048cad:   b8 00 00 00 00          mov     $0x0,%eax
8048cb2:   eb ac                   jmp     8048c60 <phase_3+0x78>
8048cb4:   e8 d7 fa ff ff          call    8048790 <__stack_chk_fail@plt>

08048cb9 <func4>:
8048cb9:   55                      push    %ebp
8048cba:   89 e5                   mov     %esp,%ebp
8048cbc:   57                      push    %edi
8048cbd:   56                      push    %esi
```

```
8048cbe:  53                    push   %ebx
8048cbf:  83 ec 0c              sub    $0xc,%esp
8048cc2:  8b 75 08              mov    0x8(%ebp),%esi
8048cc5:  8b 7d 0c              mov    0xc(%ebp),%edi
8048cc8:  b8 00 00 00 00        mov    $0x0,%eax
8048ccd:  85 f6                 test   %esi,%esi
8048ccf:  7e 07                 jle    8048cd8 <func4+0x1f>
8048cd1:  89 f8                 mov    %edi,%eax
8048cd3:  83 fe 01              cmp    $0x1,%esi
8048cd6:  75 08                 jne    8048ce0 <func4+0x27>
8048cd8:  8d 65 f4              lea    -0xc(%ebp),%esp
8048cdb:  5b                    pop    %ebx
8048cdc:  5e                    pop    %esi
8048cdd:  5f                    pop    %edi
8048cde:  5d                    pop    %ebp
8048cdf:  c3                    ret
8048ce0:  83 ec 08              sub    $0x8,%esp
8048ce3:  57                    push   %edi
8048ce4:  8d 46 ff              lea    -0x1(%esi),%eax
8048ce7:  50                    push   %eax
8048ce8:  e8 cc ff ff ff        call   8048cb9 <func4>;递归调用
8048ced:  83 c4 08              add    $0x8,%esp
8048cf0:  8d 1c 38              lea    (%eax,%edi,1),%ebx
8048cf3:  57                    push   %edi
8048cf4:  83 ee 02              sub    $0x2,%esi
8048cf7:  56                    push   %esi
8048cf8:  e8 bc ff ff ff        call   8048cb9 <func4>;递归调用
8048cfd:  83 c4 10              add    $0x10,%esp
8048d00:  01 d8                 add    %ebx,%eax
```

8048d02: eb d4 jmp 8048cd8 <func4+0x1f>

08048d04 <phase_4>:

8048d04: 55 push %ebp

8048d05: 89 e5 mov %esp,%ebp

8048d07: 83 ec 18 sub $0x18,%esp

8048d0a: 65 a1 14 00 00 00 mov %gs:0x14,%eax

8048d10: 89 45 f4 mov %eax,-0xc(%ebp)

8048d13: 31 c0 xor %eax,%eax

8048d15: 8d 45 ec lea -0x14(%ebp),%eax;将堆栈地址推入栈

8048d18: 50 push %eax

8048d19: 8d 45 f0 lea -0x10(%ebp),%eax; ;将堆栈地址推入栈

8048d1c: 50 push %eax

8048d1d: 68 91 a4 04 08 push $0x804a491

8048d22: ff 75 08 pushl 0x8(%ebp);以上堆栈及字符串操作

8048d25: e8 e6 fa ff ff call 8048810 <__isoc99_sscanf@plt>;将字符串转换为整数

8048d2a: 83 c4 10 add $0x10,%esp

8048d2d: 83 f8 02 cmp $0x2,%eax;eax是输入的整数个数，和2比较

8048d30: 75 0b jne 8048d3d <phase_4+0x39>;不相等则跳转至炸弹

8048d32: 8b 45 ec mov -0x14(%ebp),%eax;和上一题不同的是，这里是将第二个输入赋值给eax，因为前面堆栈的操作顺序刚好反了一下

8048d35: 83 e8 02 sub $0x2,%eax;eax减2

8048d38: 83 f8 02 cmp $0x2,%eax;eax与2比较

8048d3b: 76 05 jbe 8048d42 <phase_4+0x3e>;无符号

小于等于则跳转，否则运行至炸弹，即第二个输入要大于等于2小于等于4

8048d3d: e8 0f 05 00 00 call 8049251 <explode_bomb>

8048d42: 83 ec 08 sub $0x8,%esp

8048d45: ff 75 ec pushl -0x14(%ebp);将第二个输入和6推入栈，即传参

8048d48: 6a 06 push $0x6

8048d4a: e8 6a ff ff ff call 8048cb9 <func4>;调用递归函数

8048d4f: 83 c4 10 add $0x10,%esp

8048d52: 39 45 f0 cmp %eax,-0x10(%ebp);比较eax和第一个输入

8048d55: 74 05 je 8048d5c <phase_4+0x58>;相等则跳出，否则运行至炸弹

8048d57: e8 f5 04 00 00 call 8049251 <explode_bomb>

8048d5c: 8b 45 f4 mov -0xc(%ebp),%eax

8048d5f: 65 33 05 14 00 00 00 xor %gs:0x14,%eax

8048d66: 75 02 jne 8048d6a <phase_4+0x66>

8048d68: c9 leave

8048d69: c3 ret

8048d6a: e8 21 fa ff ff call 8048790 <__stack_chk_fail@plt>

08048d6f <phase_5>:

8048d6f: 55 push %ebp

8048d70: 89 e5 mov %esp,%ebp

8048d72: 53 push %ebx

8048d73: 83 ec 10 sub $0x10,%esp;以上堆栈操作

8048d76: 8b 5d 08 mov 0x8(%ebp),%ebx;将指向字符串的指针赋值给ebx

8048d79: 53 push %ebx

8048d7a: e8 7c 02 00 00 call 8048ffb <string_length>;获取字符

串长度

8048d7f: 83 c4 10 add $0x10,%esp

8048d82: 83 f8 06 cmp $0x6,%eax;eax是输入的字符串，和6比较

8048d85: 75 2d jne 8048db4 <phase_5+0x45>;不相等则跳转至炸弹

8048d87: 89 d8 mov %ebx,%eax;将ebx的值（指向字符串的指针）赋给eax

8048d89: 83 c3 06 add $0x6,%ebx;将ebx加6，即字符串末尾后一个位置

8048d8c: b9 00 00 00 00 mov $0x0,%ecx;将ecx赋值为0

8048d91: 0f b6 10 movzbl (%eax),%edx;将根据eax所存地址寻址得到的字符赋值给edx

8048d94: 83 e2 0f and $0xf,%edx;将edx和1111 (2) 与运算，即取edx低半字节

8048d97: 03 0c 95 80 a2 04 08 add 0x804a280(,%edx,4),%ecx;这里将0x804a280+4*edx寻址后加给ecx

8048d9e: 83 c0 01 add $0x1,%eax;eax++

8048da1: 39 d8 cmp %ebx,%eax;比较eax和ebx

8048da3: 75 ec jne 8048d91 <phase_5+0x22>;不等于则继续循环

8048da5: 83 f9 42 cmp $0x42,%ecx;等于则将ecx与42比较

8048da8: 74 05 je 8048daf <phase_5+0x40>;等于则跳出，不等于则运行炸弹

8048daa: e8 a2 04 00 00 call 8049251 <explode_bomb>

8048daf: 8b 5d fc mov -0x4(%ebp),%ebx

8048db2: c9 leave

8048db3: c3 ret

8048db4: e8 98 04 00 00 call 8049251 <explode_bomb>

8048db9: eb cc jmp 8048d87 <phase_5+0x18>

08048dbb <phase_6>:

8048dbb: 55 push %ebp

8048dbc: 89 e5 mov %esp,%ebp

8048dbe: 56 push %esi

8048dbf: 53 push %ebx

8048dc0: 83 ec 48 sub $0x48,%esp

8048dc3: 65 a1 14 00 00 00 mov %gs:0x14,%eax

8048dc9: 89 45 f4 mov %eax,-0xc(%ebp)

8048dcc: 31 c0 xor %eax,%eax

8048dce: 8d 45 c4 lea -0x3c(%ebp),%eax

8048dd1: 50 push %eax

8048dd2: ff 75 08 pushl 0x8(%ebp);以上堆栈操作

8048dd5: e8 b7 04 00 00 call 8049291 <read_six_numbers>;同前将字符串转为6个数字

8048dda: 83 c4 10 add $0x10,%esp

8048ddd: be 00 00 00 00 mov $0x0,%esi;将循环变量esi赋值为0

8048de2: 8b 44 b5 c4 mov -0x3c(%ebp,%esi,4),%eax; 将xi(i=1:6)的值赋给eax

8048de6: 83 e8 01 sub $0x1,%eax;eax--

8048de9: 83 f8 05 cmp $0x5,%eax;将eax和5比较

8048dec: 77 0c ja 8048dfa <phase_6+0x3f>;无符号大于则跳转至炸弹，即每一个输入都要小于等于6，大于等于1

8048dee: 83 c6 01 add $0x1,%esi;esi++

8048df1: 83 fe 06 cmp $0x6,%esi;将esi与6比较

8048df4: 74 24 je 8048e1a <phase_6+0x5f>;等于则跳转出嵌套循环

8048df6: 89 f3 mov %esi,%ebx;否则将esi（循环变量）赋值给ebx，再进行跳转进入内循环

8048df8: eb 0f jmp 8048e09 <phase_6+0x4e>

8048dfa: e8 52 04 00 00 call 8049251 <explode_bomb>

8048dff: eb ed jmp 8048dee <phase_6+0x33>

8048e01: 83 c3 01 add $0x1,%ebx;ebx++

8048e04: 83 fb 05 cmp $0x5,%ebx;将ebx和5比较

8048e07: 7f d9 jg 8048de2 <phase_6+0x27>;有符号大于则跳转继续外循环

8048e09: 8b 44 9d c4 mov -0x3c(%ebp,%ebx,4),%eax; 将xj(j=2:6)的值赋给eax

8048e0d: 39 44 b5 c0 cmp %eax,-0x40(%ebp,%esi,4);将eax与xi比较，即将xj与xi比较

8048e11: 75 ee jne 8048e01 <phase_6+0x46>;不相等则继续内循环，相等则运行至炸弹，即xi不能和它后面的任何数相同

8048e13: e8 39 04 00 00 call 8049251 <explode_bomb>

8048e18: eb e7 jmp 8048e01 <phase_6+0x46>

8048e1a: 8d 45 c4 lea -0x3c(%ebp),%eax;将第一个数的地址赋给eax

8048e1d: 8d 5d dc lea -0x24(%ebp),%ebx;将第六个数后面一个的地址赋给ebx

8048e20: b9 07 00 00 00 mov $0x7,%ecx;将ecx赋值为7

8048e25: 89 ca mov %ecx,%edx;将ecx赋值给edx，即7

8048e27: 2b 10 sub (%eax),%edx;edx减去eax寻址后的值

8048e29: 89 10 mov %edx,(%eax);将eax所存地址指向的值改为edx的值，即xi=7-xi

8048e2b: 83 c0 04 add $0x4,%eax;eax加4，即改为指向下一个数

8048e2e: 39 c3 cmp %eax,%ebx;将eax和ebx比较

8048e30: 75 f3 jne 8048e25 <phase_6+0x6a>;不相等则继续循环

8048e32: bb 00 00 00 00 mov $0x0,%ebx;将ebx赋为0

8048e37: eb 16 jmp 8048e4f <phase_6+0x94>;跳转至新循环

8048e39: 8b 52 08 mov 0x8(%edx),%edx;edx所存地址偏移寻址后赋回edx

8048e3c: 83 c0 01 add $0x1,%eax;eax++

8048e3f: 39 c8 cmp %ecx,%eax;比较eax和ecx（xi）

8048e41: 75 f6 jne 8048e39 <phase_6+0x7e>;不相等则继续内循环

8048e43: 89 54 b5 dc mov %edx,-0x24(%ebp,%esi,4);将edx此时所存的地址值压入x6后面的第esi个位置

8048e47: 83 c3 01 add $0x1,%ebx;ebx++

8048e4a: 83 fb 06 cmp $0x6,%ebx;比较6和ebx

8048e4d: 74 17 je 8048e66 <phase_6+0xab>;相等则跳出嵌套循环，否则继续外循环

8048e4f: 89 de mov %ebx,%esi;将ebx赋值给esi

8048e51: 8b 4c 9d c4 mov -0x3c(%ebp,%ebx,4),%ecx;将xi（i=1:6）赋值给ecx

8048e55: b8 01 00 00 00 mov $0x1,%eax;将eax赋为1

8048e5a: ba 54 c1 04 08 mov $0x804c154,%edx;将一个陌生地址赋值给edx

8048e5f: 83 f9 01 cmp $0x1,%ecx;比较ecx和1

8048e62: 7f d5 jg 8048e39 <phase_6+0x7e>;有符号大于则跳转，进行内循环的链表操作

8048e64: eb dd jmp 8048e43 <phase_6+0x88>;否则继续外循环

8048e66: 8b 5d dc mov -0x24(%ebp),%ebx;将x6后面第一个地址存放的地址值（链表头）赋给ebx

8048e69: 8b 45 e0 mov -0x20(%ebp),%eax;将x6后面第二个地址存放的地址值赋给ebx

8048e6c: 89 43 08 mov %eax,0x8(%ebx);将eax存入ebx所指节点的地址域

8048e6f: 8b 55 e4 mov -0x1c(%ebp),%edx;接下来是重复上面的操作，总的说是构建链表

8048e72: 89 50 08 mov %edx,0x8(%eax)

8048e75: 8b 45 e8 mov -0x18(%ebp),%eax

8048e78: 89 42 08 mov %eax,0x8(%edx)

8048e7b: 8b 55 ec mov -0x14(%ebp),%edx

8048e7e: 89 50 08 mov %edx,0x8(%eax)

8048e81: 8b 45 f0 mov -0x10(%ebp),%eax

8048e84: 89 42 08 mov %eax,0x8(%edx)

8048e87: c7 40 08 00 00 00 00 movl $0x0,0x8(%eax);最后一个节点的指针域放0（NULL）

8048e8e: be 05 00 00 00 mov $0x5,%esi;把esi赋为5

8048e93: eb 08 jmp 8048e9d <phase_6+0xe2>;跳入循环

8048e95: 8b 5b 08 mov 0x8(%ebx),%ebx;ebx指向下一个节点

8048e98: 83 ee 01 sub $0x1,%esi;esi--

8048e9b: 74 10 je 8048ead <phase_6+0xf2>;若等于则跳出循环

8048e9d: 8b 43 08 mov 0x8(%ebx),%eax;将ebx所指节点的地址域存入eax

8048ea0: 8b 00 mov (%eax),%eax;eax自寻址

8048ea2: 39 03 cmp %eax,(%ebx);将eax与ebx所指节

点的值比较

8048ea4: 7d ef jge 8048e95 <phase_6+0xda>;有符号大于等于则跳转继续循环，否则运行炸弹

8048ea6: e8 a6 03 00 00 call 8049251 <explode_bomb>

8048eab: eb e8 jmp 8048e95 <phase_6+0xda>

8048ead: 8b 45 f4 mov -0xc(%ebp),%eax

8048eb0: 65 33 05 14 00 00 00 xor %gs:0x14,%eax

8048eb7: 75 07 jne 8048ec0 <phase_6+0x105>

8048eb9: 8d 65 f8 lea -0x8(%ebp),%esp

8048ebc: 5b pop %ebx

8048ebd: 5e pop %esi

8048ebe: 5d pop %ebp

8048ebf: c3 ret

8048ec0: e8 cb f8 ff ff call 8048790 <__stack_chk_fail@plt>

08048ec5 <fun7>:

8048ec5: 55 push %ebp

8048ec6: 89 e5 mov %esp,%ebp

8048ec8: 53 push %ebx

8048ec9: 83 ec 04 sub $0x4,%esp

8048ecc: 8b 55 08 mov 0x8(%ebp),%edx;edx为传入的地址

8048ecf: 8b 4d 0c mov 0xc(%ebp),%ecx;ecx为输入的答案

8048ed2: 85 d2 test %edx,%edx;edx不为0则不跳转继续运行

8048ed4: 74 3c je 8048f12 <fun7+0x4d>

8048ed6: 8b 1a mov (%edx),%ebx;ebx赋为节点值

8048ed8: 39 cb cmp %ecx,%ebx;将ecx和ebx比较

```
8048eda:   7f 0e                    jg      8048eea <fun7+0x25>;有符号大于则跳转递归调用
8048edc:   b8 00 00 00 00           mov     $0x0,%eax;eax赋为0
8048ee1:   39 cb                    cmp     %ecx,%ebx;将ecx和ebx比较
8048ee3:   75 18                    jne     8048efd <fun7+0x38>;不相等则跳转递归调用
8048ee5:   8b 5d fc                 mov     -0x4(%ebp),%ebx
8048ee8:   c9                       leave
8048ee9:   c3                       ret
8048eea:   83 ec 08                 sub     $0x8,%esp
8048eed:   51                       push    %ecx;传入两个参数
8048eee:   ff 72 04                 pushl   0x4(%edx)
8048ef1:   e8 cf ff ff ff           call    8048ec5 <fun7>
8048ef6:   83 c4 10                 add     $0x10,%esp
8048ef9:   01 c0                    add     %eax,%eax
8048efb:   eb e8                    jmp     8048ee5 <fun7+0x20>
8048efd:   83 ec 08                 sub     $0x8,%esp
8048f00:   51                       push    %ecx;传入两个参数
8048f01:   ff 72 08                 pushl   0x8(%edx)
8048f04:   e8 bc ff ff ff           call    8048ec5 <fun7>
8048f09:   83 c4 10                 add     $0x10,%esp
8048f0c:   8d 44 00 01              lea     0x1(%eax,%eax,1),%eax;eax = eax*2+1
8048f10:   eb d3                    jmp     8048ee5 <fun7+0x20>
8048f12:   b8 ff ff ff ff           mov     $0xffffffff,%eax
8048f17:   eb cc                    jmp     8048ee5 <fun7+0x20>

08048f19 <secret_phase>:
8048f19:   55                       push    %ebp
```

```
8048f1a:  89 e5                 mov    %esp,%ebp
8048f1c:  53                    push   %ebx
8048f1d:  83 ec 04              sub    $0x4,%esp;堆栈操作
8048f20:  e8 a6 03 00 00        call   80492cb <read_line>;读入一行
8048f25:  83 ec 04              sub    $0x4,%esp
8048f28:  6a 0a                 push   $0xa
8048f2a:  6a 00                 push   $0x0
8048f2c:  50                    push   %eax
8048f2d:  e8 4e f9 ff ff        call   8048880 <strtol@plt>
8048f32:  89 c3                 mov    %eax,%ebx
8048f34:  8d 40 ff              lea    -0x1(%eax),%eax
8048f37:  83 c4 10              add    $0x10,%esp
8048f3a:  3d e8 03 00 00        cmp    $0x3e8,%eax
8048f3f:  77 35                 ja     8048f76 <secret_phase+0x5d>
8048f41:  83 ec 08              sub    $0x8,%esp
8048f44:  53                    push   %ebx
8048f45:  68 a0 c0 04 08        push   $0x804c0a0
8048f4a:  e8 76 ff ff ff        call   8048ec5 <fun7>
8048f4f:  83 c4 10              add    $0x10,%esp
8048f52:  83 f8 01              cmp    $0x1,%eax
8048f55:  74 05                 je     8048f5c <secret_phase+0x43>
8048f57:  e8 f5 02 00 00        call   8049251 <explode_bomb>
8048f5c:  83 ec 0c              sub    $0xc,%esp
8048f5f:  68 28 a2 04 08        push   $0x804a228
8048f64:  e8 57 f8 ff ff        call   80487c0 <puts@plt>
8048f69:  e8 6e 04 00 00        call   80493dc <phase_defused>
8048f6e:  83 c4 10              add    $0x10,%esp
8048f71:  8b 5d fc              mov    -0x4(%ebp),%ebx
8048f74:  c9                    leave
```

```
 8048f75:   c3                      ret
 8048f76:   e8 d6 02 00 00          call   8049251 <explode_bomb>
 8048f7b:   eb c4                   jmp    8048f41 <secret_phase+0x28>

08048f7d <sig_handler>:
 8048f7d:   55                      push   %ebp
 8048f7e:   89 e5                   mov    %esp,%ebp
 8048f80:   83 ec 14                sub    $0x14,%esp
 8048f83:   68 c0 a2 04 08          push   $0x804a2c0
 8048f88:   e8 33 f8 ff ff          call   80487c0 <puts@plt>
 8048f8d:   c7 04 24 03 00 00 00    movl   $0x3,(%esp)
 8048f94:   e8 d7 f7 ff ff          call   8048770 <sleep@plt>
 8048f99:   83 c4 08                add    $0x8,%esp
 8048f9c:   68 0d a4 04 08          push   $0x804a40d
 8048fa1:   6a 01                   push   $0x1
 8048fa3:   e8 98 f8 ff ff          call   8048840 <__printf_chk@plt>
 8048fa8:   83 c4 04                add    $0x4,%esp
 8048fab:   ff 35 e4 c7 04 08       pushl  0x804c7e4
 8048fb1:   e8 8a f7 ff ff          call   8048740 <fflush@plt>
 8048fb6:   c7 04 24 01 00 00 00    movl   $0x1,(%esp)
 8048fbd:   e8 ae f7 ff ff          call   8048770 <sleep@plt>
 8048fc2:   c7 04 24 15 a4 04 08    movl   $0x804a415,(%esp)
 8048fc9:   e8 f2 f7 ff ff          call   80487c0 <puts@plt>
 8048fce:   c7 04 24 10 00 00 00    movl   $0x10,(%esp)
 8048fd5:   e8 06 f8 ff ff          call   80487e0 <exit@plt>

08048fda <invalid_phase>:
 8048fda:   55                      push   %ebp
 8048fdb:   89 e5                   mov    %esp,%ebp
```

```
 8048fdd:   83 ec 0c                 sub    $0xc,%esp
 8048fe0:   ff 75 08                 pushl  0x8(%ebp)
 8048fe3:   68 1d a4 04 08           push   $0x804a41d
 8048fe8:   6a 01                    push   $0x1
 8048fea:   e8 51 f8 ff ff           call   8048840 <__printf_chk@plt>
 8048fef:   c7 04 24 08 00 00 00     movl   $0x8,(%esp)
 8048ff6:   e8 e5 f7 ff ff           call   80487e0 <exit@plt>

08048ffb <string_length>:
 8048ffb:   55                       push   %ebp
 8048ffc:   89 e5                    mov    %esp,%ebp
 8048ffe:   8b 55 08                 mov    0x8(%ebp),%edx
 8049001:   80 3a 00                 cmpb   $0x0,(%edx)
 8049004:   74 10                    je     8049016 <string_length+0x1b>
 8049006:   b8 00 00 00 00           mov    $0x0,%eax
 804900b:   83 c0 01                 add    $0x1,%eax
 804900e:   80 3c 02 00              cmpb   $0x0,(%edx,%eax,1)
 8049012:   75 f7                    jne    804900b <string_length+0x10>
 8049014:   5d                       pop    %ebp
 8049015:   c3                       ret
 8049016:   b8 00 00 00 00           mov    $0x0,%eax
 804901b:   eb f7                    jmp    8049014 <string_length+0x19>

0804901d <strings_not_equal>:
 804901d:   55                       push   %ebp
 804901e:   89 e5                    mov    %esp,%ebp
 8049020:   57                       push   %edi
 8049021:   56                       push   %esi
 8049022:   53                       push   %ebx
```

```
8049023:  8b 5d 08          mov    0x8(%ebp),%ebx
8049026:  8b 75 0c          mov    0xc(%ebp),%esi
8049029:  53                push   %ebx
804902a:  e8 cc ff ff ff    call   8048ffb <string_length>
804902f:  89 c7             mov    %eax,%edi
8049031:  89 34 24          mov    %esi,(%esp)
8049034:  e8 c2 ff ff ff    call   8048ffb <string_length>
8049039:  83 c4 04          add    $0x4,%esp
804903c:  ba 01 00 00 00    mov    $0x1,%edx
8049041:  39 c7             cmp    %eax,%edi
8049043:  74 0a             je     804904f
<strings_not_equal+0x32>
8049045:  89 d0             mov    %edx,%eax
8049047:  8d 65 f4          lea    -0xc(%ebp),%esp
804904a:  5b                pop    %ebx
804904b:  5e                pop    %esi
804904c:  5f                pop    %edi
804904d:  5d                pop    %ebp
804904e:  c3                ret
804904f:  0f b6 03          movzbl (%ebx),%eax
8049052:  84 c0             test   %al,%al
8049054:  74 23             je     8049079
<strings_not_equal+0x5c>
8049056:  3a 06             cmp    (%esi),%al
8049058:  75 26             jne    8049080
<strings_not_equal+0x63>
804905a:  83 c3 01          add    $0x1,%ebx
804905d:  83 c6 01          add    $0x1,%esi
8049060:  0f b6 03          movzbl (%ebx),%eax
```

```
8049063:  84 c0                 test   %al,%al
8049065:  74 0b                 je     8049072
<strings_not_equal+0x55>
8049067:  38 06                 cmp    %al,(%esi)
8049069:  74 ef                 je     804905a
<strings_not_equal+0x3d>
804906b:  ba 01 00 00 00        mov    $0x1,%edx
8049070:  eb d3                 jmp    8049045
<strings_not_equal+0x28>
8049072:  ba 00 00 00 00        mov    $0x0,%edx
8049077:  eb cc                 jmp    8049045
<strings_not_equal+0x28>
8049079:  ba 00 00 00 00        mov    $0x0,%edx
804907e:  eb c5                 jmp    8049045
<strings_not_equal+0x28>
8049080:  ba 01 00 00 00        mov    $0x1,%edx
8049085:  eb be                 jmp    8049045
<strings_not_equal+0x28>

08049087 <initialize_bomb>:
8049087:  55                    push   %ebp
8049088:  89 e5                 mov    %esp,%ebp
804908a:  81 ec 20 20 00 00     sub    $0x2020,%esp
8049090:  65 a1 14 00 00 00     mov    %gs:0x14,%eax
8049096:  89 45 f4              mov    %eax,-0xc(%ebp)
8049099:  31 c0                 xor    %eax,%eax
804909b:  68 7d 8f 04 08        push   $0x8048f7d
80490a0:  6a 02                 push   $0x2
80490a2:  e8 b9 f6 ff ff        call   8048760 <signal@plt>
```

```
 80490a7:   8d 85 f4 df ff ff       lea      -0x200c(%ebp),%eax
 80490ad:   89 04 24                mov      %eax,(%esp)
 80490b0:   e8 2b 0d 00 00          call     8049de0 <init_driver>
 80490b5:   83 c4 10                add      $0x10,%esp
 80490b8:   85 c0                   test     %eax,%eax
 80490ba:   78 0e                   js       80490ca <initialize_bomb+0x43>
 80490bc:   8b 45 f4                mov      -0xc(%ebp),%eax
 80490bf:   65 33 05 14 00 00 00    xor      %gs:0x14,%eax
 80490c6:   75 24                   jne      80490ec <initialize_bomb+0x65>
 80490c8:   c9                      leave
 80490c9:   c3                      ret
 80490ca:   83 ec 04                sub      $0x4,%esp
 80490cd:   8d 85 f4 df ff ff       lea      -0x200c(%ebp),%eax
 80490d3:   50                      push     %eax
 80490d4:   68 2e a4 04 08          push     $0x804a42e
 80490d9:   6a 01                   push     $0x1
 80490db:   e8 60 f7 ff ff          call     8048840 <__printf_chk@plt>
 80490e0:   c7 04 24 08 00 00 00    movl     $0x8,(%esp)
 80490e7:   e8 f4 f6 ff ff          call     80487e0 <exit@plt>
 80490ec:   e8 9f f6 ff ff          call     8048790 <__stack_chk_fail@plt>

080490f1 <initialize_bomb_solve>:
 80490f1:   55                      push     %ebp
 80490f2:   89 e5                   mov      %esp,%ebp
 80490f4:   5d                      pop      %ebp
 80490f5:   c3                      ret

080490f6 <blank_line>:
 80490f6:   55                      push     %ebp
```

```
80490f7:   89 e5                    mov     %esp,%ebp
80490f9:   56                       push    %esi
80490fa:   53                       push    %ebx
80490fb:   8b 75 08                 mov     0x8(%ebp),%esi
80490fe:   0f b6 1e                 movzbl (%esi),%ebx
8049101:   84 db                    test    %bl,%bl
8049103:   74 1b                    je      8049120 <blank_line+0x2a>
8049105:   e8 a6 f7 ff ff           call    80488b0 <__ctype_b_loc@plt>
804910a:   83 c6 01                 add     $0x1,%esi
804910d:   0f be db                 movsbl %bl,%ebx
8049110:   8b 00                    mov     (%eax),%eax
8049112:   f6 44 58 01 20           testb   $0x20,0x1(%eax,%ebx,2)
8049117:   75 e5                    jne     80490fe <blank_line+0x8>
8049119:   b8 00 00 00 00           mov     $0x0,%eax
804911e:   eb 05                    jmp     8049125 <blank_line+0x2f>
8049120:   b8 01 00 00 00           mov     $0x1,%eax
8049125:   5b                       pop     %ebx
8049126:   5e                       pop     %esi
8049127:   5d                       pop     %ebp
8049128:   c3                       ret

08049129 <skip>:
8049129:   55                       push    %ebp
804912a:   89 e5                    mov     %esp,%ebp
804912c:   53                       push    %ebx
804912d:   83 ec 04                 sub     $0x4,%esp
8049130:   83 ec 04                 sub     $0x4,%esp
8049133:   ff 35 f0 c7 04 08        pushl   0x804c7f0
8049139:   6a 50                    push    $0x50
```

```
804913b:  a1 ec c7 04 08          mov    0x804c7ec,%eax
8049140:  8d 04 80                lea    (%eax,%eax,4),%eax
8049143:  c1 e0 04                shl    $0x4,%eax
8049146:  05 00 c8 04 08          add    $0x804c800,%eax
804914b:  50                      push   %eax
804914c:  e8 ff f5 ff ff          call   8048750 <fgets@plt>
8049151:  89 c3                   mov    %eax,%ebx
8049153:  83 c4 10                add    $0x10,%esp
8049156:  85 c0                   test   %eax,%eax
8049158:  74 10                   je     804916a <skip+0x41>
804915a:  83 ec 0c                sub    $0xc,%esp
804915d:  50                      push   %eax
804915e:  e8 93 ff ff ff          call   80490f6 <blank_line>
8049163:  83 c4 10                add    $0x10,%esp
8049166:  85 c0                   test   %eax,%eax
8049168:  75 c6                   jne    8049130 <skip+0x7>
804916a:  89 d8                   mov    %ebx,%eax
804916c:  8b 5d fc                mov    -0x4(%ebp),%ebx
804916f:  c9                      leave
8049170:  c3                      ret

08049171 <send_msg>:
8049171:  55                      push   %ebp
8049172:  89 e5                   mov    %esp,%ebp
8049174:  57                      push   %edi
8049175:  56                      push   %esi
8049176:  53                      push   %ebx
8049177:  81 ec 1c 40 00 00       sub    $0x401c,%esp
804917d:  65 a1 14 00 00 00       mov    %gs:0x14,%eax
```

```
8049183:  89 45 e4              mov    %eax,-0x1c(%ebp)
8049186:  31 c0                 xor    %eax,%eax
8049188:  8b 1d ec c7 04 08     mov    0x804c7ec,%ebx
804918e:  8d 54 9b fb           lea    -0x5(%ebx,%ebx,4),%edx
8049192:  c1 e2 04              shl    $0x4,%edx
8049195:  81 c2 00 c8 04 08     add    $0x804c800,%edx
804919b:  b9 ff ff ff ff        mov    $0xffffffff,%ecx
80491a0:  89 d7                 mov    %edx,%edi
80491a2:  f2 ae                 repnz scas %es:(%edi),%al
80491a4:  89 c8                 mov    %ecx,%eax
80491a6:  f7 d0                 not    %eax
80491a8:  83 c0 63              add    $0x63,%eax
80491ab:  3d 00 20 00 00        cmp    $0x2000,%eax
80491b0:  77 64                 ja     8049216 <send_msg+0xa5>
80491b2:  83 7d 08 00           cmpl   $0x0,0x8(%ebp)
80491b6:  b8 48 a4 04 08        mov    $0x804a448,%eax
80491bb:  b9 50 a4 04 08        mov    $0x804a450,%ecx
80491c0:  0f 44 c1              cmove  %ecx,%eax
80491c3:  52                    push   %edx
80491c4:  53                    push   %ebx
80491c5:  50                    push   %eax
80491c6:  ff 35 a0 c5 04 08     pushl  0x804c5a0
80491cc:  68 59 a4 04 08        push   $0x804a459
80491d1:  68 00 20 00 00        push   $0x2000
80491d6:  6a 01                 push   $0x1
80491d8:  8d 9d e4 bf ff ff     lea    -0x401c(%ebp),%ebx
80491de:  53                    push   %ebx
80491df:  e8 dc f6 ff ff        call   80488c0 <__sprintf_chk@plt>
80491e4:  83 c4 20              add    $0x20,%esp
```

```
80491e7:  8d 85 e4 df ff ff     lea     -0x201c(%ebp),%eax
80491ed:  50                    push    %eax
80491ee:  6a 00                 push    $0x0
80491f0:  53                    push    %ebx
80491f1:  68 a0 c1 04 08        push    $0x804c1a0
80491f6:  e8 bf 0d 00 00        call    8049fba <driver_post>
80491fb:  83 c4 10              add     $0x10,%esp
80491fe:  85 c0                 test    %eax,%eax
8049200:  78 2f                 js      8049231 <send_msg+0xc0>
8049202:  8b 45 e4              mov     -0x1c(%ebp),%eax
8049205:  65 33 05 14 00 00 00  xor     %gs:0x14,%eax
804920c:  75 3e                 jne     804924c <send_msg+0xdb>
804920e:  8d 65 f4              lea     -0xc(%ebp),%esp
8049211:  5b                    pop     %ebx
8049212:  5e                    pop     %esi
8049213:  5f                    pop     %edi
8049214:  5d                    pop     %ebp
8049215:  c3                    ret
8049216:  83 ec 08              sub     $0x8,%esp
8049219:  68 f8 a2 04 08        push    $0x804a2f8
804921e:  6a 01                 push    $0x1
8049220:  e8 1b f6 ff ff        call    8048840 <__printf_chk@plt>
8049225:  c7 04 24 08 00 00 00  movl    $0x8,(%esp)
804922c:  e8 af f5 ff ff        call    80487e0 <exit@plt>
8049231:  83 ec 0c              sub     $0xc,%esp
8049234:  8d 85 e4 df ff ff     lea     -0x201c(%ebp),%eax
804923a:  50                    push    %eax
804923b:  e8 80 f5 ff ff        call    80487c0 <puts@plt>
8049240:  c7 04 24 00 00 00 00  movl    $0x0,(%esp)
```

```
 8049247:  e8 94 f5 ff ff          call   80487e0 <exit@plt>
 804924c:  e8 3f f5 ff ff          call   8048790 <__stack_chk_fail@plt>

08049251 <explode_bomb>:
 8049251:  55                      push   %ebp
 8049252:  89 e5                   mov    %esp,%ebp
 8049254:  83 ec 14                sub    $0x14,%esp
 8049257:  68 65 a4 04 08          push   $0x804a465
 804925c:  e8 5f f5 ff ff          call   80487c0 <puts@plt>
 8049261:  c7 04 24 6e a4 04 08    movl   $0x804a46e,(%esp)
 8049268:  e8 53 f5 ff ff          call   80487c0 <puts@plt>
 804926d:  c7 04 24 00 00 00 00    movl   $0x0,(%esp)
 8049274:  e8 f8 fe ff ff          call   8049171 <send_msg>
 8049279:  c7 04 24 1c a3 04 08    movl   $0x804a31c,(%esp)
 8049280:  e8 3b f5 ff ff          call   80487c0 <puts@plt>
 8049285:  c7 04 24 08 00 00 00    movl   $0x8,(%esp)
 804928c:  e8 4f f5 ff ff          call   80487e0 <exit@plt>

08049291 <read_six_numbers>:
 8049291:  55                      push   %ebp
 8049292:  89 e5                   mov    %esp,%ebp
 8049294:  83 ec 08                sub    $0x8,%esp
 8049297:  8b 45 0c                mov    0xc(%ebp),%eax
 804929a:  8d 50 14                lea    0x14(%eax),%edx
 804929d:  52                      push   %edx
 804929e:  8d 50 10                lea    0x10(%eax),%edx
 80492a1:  52                      push   %edx
 80492a2:  8d 50 0c                lea    0xc(%eax),%edx
 80492a5:  52                      push   %edx
```

```
80492a6:  8d 50 08          lea    0x8(%eax),%edx
80492a9:  52                push   %edx
80492aa:  8d 50 04          lea    0x4(%eax),%edx
80492ad:  52                push   %edx
80492ae:  50                push   %eax
80492af:  68 85 a4 04 08    push   $0x804a485
80492b4:  ff 75 08          pushl  0x8(%ebp)
80492b7:  e8 54 f5 ff ff    call   8048810 <__isoc99_sscanf@plt>
80492bc:  83 c4 20          add    $0x20,%esp
80492bf:  83 f8 05          cmp    $0x5,%eax
80492c2:  7e 02             jle    80492c6
<read_six_numbers+0x35>
80492c4:  c9                leave
80492c5:  c3                ret
80492c6:  e8 86 ff ff ff    call   8049251 <explode_bomb>

080492cb <read_line>:
80492cb:  55                push   %ebp
80492cc:  89 e5             mov    %esp,%ebp
80492ce:  57                push   %edi
80492cf:  56                push   %esi
80492d0:  53                push   %ebx
80492d1:  83 ec 0c          sub    $0xc,%esp
80492d4:  e8 50 fe ff ff    call   8049129 <skip>
80492d9:  85 c0             test   %eax,%eax
80492db:  74 53             je     8049330 <read_line+0x65>
80492dd:  8b 15 ec c7 04 08 mov    0x804c7ec,%edx
80492e3:  8d 1c 92          lea    (%edx,%edx,4),%ebx
80492e6:  c1 e3 04          shl    $0x4,%ebx
```

```
80492e9:  81 c3 00 c8 04 08        add     $0x804c800,%ebx
80492ef:  b9 ff ff ff ff           mov     $0xffffffff,%ecx
80492f4:  b8 00 00 00 00           mov     $0x0,%eax
80492f9:  89 df                    mov     %ebx,%edi
80492fb:  f2 ae                    repnz scas %es:(%edi),%al
80492fd:  89 ce                    mov     %ecx,%esi
80492ff:  f7 d6                    not     %esi
8049301:  89 f1                    mov     %esi,%ecx
8049303:  83 e9 01                 sub     $0x1,%ecx
8049306:  83 f9 4e                 cmp     $0x4e,%ecx
8049309:  0f 8f 95 00 00 00        jg      80493a4 <read_line+0xd9>
804930f:  8d 04 92                 lea     (%edx,%edx,4),%eax
8049312:  c1 e0 04                 shl     $0x4,%eax
8049315:  c6 84 01 ff c7 04 08     movb    $0x0,0x804c7ff(%ecx,%eax,1)
804931c:  00
804931d:  83 c2 01                 add     $0x1,%edx
8049320:  89 15 ec c7 04 08        mov     %edx,0x804c7ec
8049326:  89 d8                    mov     %ebx,%eax
8049328:  8d 65 f4                 lea     -0xc(%ebp),%esp
804932b:  5b                       pop     %ebx
804932c:  5e                       pop     %esi
804932d:  5f                       pop     %edi
804932e:  5d                       pop     %ebp
804932f:  c3                       ret
8049330:  a1 e0 c7 04 08           mov     0x804c7e0,%eax
8049335:  39 05 f0 c7 04 08        cmp     %eax,0x804c7f0
804933b:  74 1e                    je      804935b <read_line+0x90>
804933d:  83 ec 0c                 sub     $0xc,%esp
8049340:  68 b5 a4 04 08           push    $0x804a4b5
```

```
8049345:  e8 66 f4 ff ff          call   80487b0 <getenv@plt>
804934a:  83 c4 10                add    $0x10,%esp
804934d:  85 c0                   test   %eax,%eax
804934f:  74 23                   je     8049374 <read_line+0xa9>
8049351:  83 ec 0c                sub    $0xc,%esp
8049354:  6a 00                   push   $0x0
8049356:  e8 85 f4 ff ff          call   80487e0 <exit@plt>
804935b:  83 ec 0c                sub    $0xc,%esp
804935e:  68 97 a4 04 08          push   $0x804a497
8049363:  e8 58 f4 ff ff          call   80487c0 <puts@plt>
8049368:  c7 04 24 08 00 00 00    movl   $0x8,(%esp)
804936f:  e8 6c f4 ff ff          call   80487e0 <exit@plt>
8049374:  a1 e0 c7 04 08          mov    0x804c7e0,%eax
8049379:  a3 f0 c7 04 08          mov    %eax,0x804c7f0
804937e:  e8 a6 fd ff ff          call   8049129 <skip>
8049383:  85 c0                   test   %eax,%eax
8049385:  0f 85 52 ff ff ff       jne    80492dd <read_line+0x12>
804938b:  83 ec 0c                sub    $0xc,%esp
804938e:  68 97 a4 04 08          push   $0x804a497
8049393:  e8 28 f4 ff ff          call   80487c0 <puts@plt>
8049398:  c7 04 24 00 00 00 00    movl   $0x0,(%esp)
804939f:  e8 3c f4 ff ff          call   80487e0 <exit@plt>
80493a4:  83 ec 0c                sub    $0xc,%esp
80493a7:  68 c0 a4 04 08          push   $0x804a4c0
80493ac:  e8 0f f4 ff ff          call   80487c0 <puts@plt>
80493b1:  a1 ec c7 04 08          mov    0x804c7ec,%eax
80493b6:  8d 50 01                lea    0x1(%eax),%edx
80493b9:  89 15 ec c7 04 08       mov    %edx,0x804c7ec
80493bf:  6b c0 50                imul   $0x50,%eax,%eax
```

```
80493c2:  05 00 c8 04 08          add    $0x804c800,%eax
80493c7:  ba db a4 04 08          mov    $0x804a4db,%edx
80493cc:  b9 04 00 00 00          mov    $0x4,%ecx
80493d1:  89 c7                   mov    %eax,%edi
80493d3:  89 d6                   mov    %edx,%esi
80493d5:  f3 a5                   rep movsl %ds:(%esi),%es:(%edi)
80493d7:  e8 75 fe ff ff          call   8049251 <explode_bomb>

080493dc <phase_defused>:
80493dc:  55                      push   %ebp
80493dd:  89 e5                   mov    %esp,%ebp
80493df:  83 ec 74                sub    $0x74,%esp
80493e2:  65 a1 14 00 00 00       mov    %gs:0x14,%eax
80493e8:  89 45 f4                mov    %eax,-0xc(%ebp)
80493eb:  31 c0                   xor    %eax,%eax
80493ed:  6a 01                   push   $0x1
80493ef:  e8 7d fd ff ff          call   8049171 <send_msg>
80493f4:  83 c4 10                add    $0x10,%esp
80493f7:  83 3d ec c7 04 08 06    cmpl   $0x6,0x804c7ec;只有完成6关后
才会出现
80493fe:  74 12                   je     8049412 <phase_defused+0x36>
8049400:  8b 45 f4                mov    -0xc(%ebp),%eax
8049403:  65 33 05 14 00 00 00    xor    %gs:0x14,%eax
804940a:  0f 85 81 00 00 00       jne    8049491 <phase_defused+0xb5>
8049410:  c9                      leave
8049411:  c3                      ret
8049412:  83 ec 0c                sub    $0xc,%esp
8049415:  8d 45 a4                lea    -0x5c(%ebp),%eax
8049418:  50                      push   %eax
```

```
8049419:  8d 45 a0              lea    -0x60(%ebp),%eax
804941c:  50                    push   %eax
804941d:  8d 45 9c              lea    -0x64(%ebp),%eax
8049420:  50                    push   %eax
8049421:  68 eb a4 04 08        push   $0x804a4eb
8049426:  68 f0 c8 04 08        push   $0x804c8f0
804942b:  e8 e0 f3 ff ff        call   8048810 <__isoc99_sscanf@plt>
8049430:  83 c4 20              add    $0x20,%esp
8049433:  83 f8 03              cmp    $0x3,%eax
8049436:  74 1e                 je     8049456 <phase_defused+0x7a>
8049438:  83 ec 0c              sub    $0xc,%esp
804943b:  68 a0 a3 04 08        push   $0x804a3a0
8049440:  e8 7b f3 ff ff        call   80487c0 <puts@plt>
8049445:  c7 04 24 cc a3 04 08  movl   $0x804a3cc,(%esp)
804944c:  e8 6f f3 ff ff        call   80487c0 <puts@plt>
8049451:  83 c4 10              add    $0x10,%esp
8049454:  eb aa                 jmp                          8049400
<phase_defused+0x24>
8049456:  83 ec 08              sub    $0x8,%esp
8049459:  68 f4 a4 04 08        push   $0x804a4f4
804945e:  8d 45 a4              lea    -0x5c(%ebp),%eax
8049461:  50                    push   %eax
8049462:  e8 b6 fb ff ff        call   804901d <strings_not_equal>
8049467:  83 c4 10              add    $0x10,%esp
804946a:  85 c0                 test   %eax,%eax
804946c:  75 ca                 jne    8049438 <phase_defused+0x5c>
804946e:  83 ec 0c              sub    $0xc,%esp
8049471:  68 40 a3 04 08        push   $0x804a340
8049476:  e8 45 f3 ff ff        call   80487c0 <puts@plt>
```

```
804947b:  c7 04 24 68 a3 04 08     movl   $0x804a368,(%esp)
8049482:  e8 39 f3 ff ff           call   80487c0 <puts@plt>
8049487:  e8 8d fa ff ff           call   8048f19 <secret_phase>
804948c:  83 c4 10                 add    $0x10,%esp
804948f:  eb a7                    jmp                          8049438
<phase_defused+0x5c>
8049491:  e8 fa f2 ff ff           call   8048790 <__stack_chk_fail@plt>

08049496 <sigalrm_handler>:
8049496:  55                       push   %ebp
8049497:  89 e5                    mov    %esp,%ebp
8049499:  83 ec 08                 sub    $0x8,%esp
804949c:  6a 00                    push   $0x0
804949e:  68 4c a5 04 08           push   $0x804a54c
80494a3:  6a 01                    push   $0x1
80494a5:  ff 35 c0 c7 04 08        pushl  0x804c7c0
80494ab:  e8 b0 f3 ff ff           call   8048860 <__fprintf_chk@plt>
80494b0:  c7 04 24 01 00 00 00     movl   $0x1,(%esp)
80494b7:  e8 24 f3 ff ff           call   80487e0 <exit@plt>

080494bc <rio_readlineb>:
80494bc:  55                       push   %ebp
80494bd:  89 e5                    mov    %esp,%ebp
80494bf:  57                       push   %edi
80494c0:  56                       push   %esi
80494c1:  53                       push   %ebx
80494c2:  83 ec 1c                 sub    $0x1c,%esp
80494c5:  89 d7                    mov    %edx,%edi
80494c7:  83 f9 01                 cmp    $0x1,%ecx
```

```
80494ca:   76 7d                    jbe     8049549 <rio_readlineb+0x8d>
80494cc:   89 c3                    mov     %eax,%ebx
80494ce:   8d 44 0a ff              lea     -0x1(%edx,%ecx,1),%eax
80494d2:   89 45 e0                 mov     %eax,-0x20(%ebp)
80494d5:   c7 45 e4 01 00 00 00     movl    $0x1,-0x1c(%ebp)
80494dc:   8d 73 0c                 lea     0xc(%ebx),%esi
80494df:   eb 0a                    jmp     80494eb <rio_readlineb+0x2f>
80494e1:   e8 4a f3 ff ff           call    8048830 <__errno_location@plt>
80494e6:   83 38 04                 cmpl    $0x4,(%eax)
80494e9:   75 67                    jne     8049552 <rio_readlineb+0x96>
80494eb:   8b 43 04                 mov     0x4(%ebx),%eax
80494ee:   85 c0                    test    %eax,%eax
80494f0:   7f 23                    jg      8049515 <rio_readlineb+0x59>
80494f2:   83 ec 04                 sub     $0x4,%esp
80494f5:   68 00 20 00 00           push    $0x2000
80494fa:   56                       push    %esi
80494fb:   ff 33                    pushl   (%ebx)
80494fd:   e8 2e f2 ff ff           call    8048730 <read@plt>
8049502:   89 43 04                 mov     %eax,0x4(%ebx)
8049505:   83 c4 10                 add     $0x10,%esp
8049508:   85 c0                    test    %eax,%eax
804950a:   78 d5                    js      80494e1 <rio_readlineb+0x25>
804950c:   85 c0                    test    %eax,%eax
804950e:   74 49                    je      8049559 <rio_readlineb+0x9d>
8049510:   89 73 08                 mov     %esi,0x8(%ebx)
8049513:   eb d6                    jmp     80494eb <rio_readlineb+0x2f>
8049515:   8b 4b 08                 mov     0x8(%ebx),%ecx
8049518:   0f b6 11                 movzbl (%ecx),%edx
804951b:   83 c1 01                 add     $0x1,%ecx
```

```
804951e:  89 4b 08                  mov     %ecx,0x8(%ebx)
8049521:  83 e8 01                  sub     $0x1,%eax
8049524:  89 43 04                  mov     %eax,0x4(%ebx)
8049527:  83 c7 01                  add     $0x1,%edi
804952a:  88 57 ff                  mov     %dl,-0x1(%edi)
804952d:  80 fa 0a                  cmp     $0xa,%dl
8049530:  74 09                     je      804953b <rio_readlineb+0x7f>
8049532:  83 45 e4 01               addl    $0x1,-0x1c(%ebp)
8049536:  3b 7d e0                  cmp     -0x20(%ebp),%edi
8049539:  75 b0                     jne     80494eb <rio_readlineb+0x2f>
804953b:  c6 07 00                  movb    $0x0,(%edi)
804953e:  8b 45 e4                  mov     -0x1c(%ebp),%eax
8049541:  8d 65 f4                  lea     -0xc(%ebp),%esp
8049544:  5b                        pop     %ebx
8049545:  5e                        pop     %esi
8049546:  5f                        pop     %edi
8049547:  5d                        pop     %ebp
8049548:  c3                        ret
8049549:  c7 45 e4 01 00 00 00      movl    $0x1,-0x1c(%ebp)
8049550:  eb e9                     jmp     804953b <rio_readlineb+0x7f>
8049552:  b8 ff ff ff ff            mov     $0xffffffff,%eax
8049557:  eb 05                     jmp     804955e <rio_readlineb+0xa2>
8049559:  b8 00 00 00 00            mov     $0x0,%eax
804955e:  85 c0                     test    %eax,%eax
8049560:  75 0f                     jne     8049571 <rio_readlineb+0xb5>
8049562:  83 7d e4 01               cmpl    $0x1,-0x1c(%ebp)
8049566:  75 d3                     jne     804953b <rio_readlineb+0x7f>
8049568:  c7 45 e4 00 00 00 00      movl    $0x0,-0x1c(%ebp)
804956f:  eb cd                     jmp     804953e <rio_readlineb+0x82>
```

```
 8049571:  c7 45 e4 ff ff ff ff     movl    $0xffffffff,-0x1c(%ebp)
 8049578:  eb c4                    jmp     804953e <rio_readlineb+0x82>

0804957a <submitr>:
 804957a:  55                       push    %ebp
 804957b:  89 e5                    mov     %esp,%ebp
 804957d:  57                       push    %edi
 804957e:  56                       push    %esi
 804957f:  53                       push    %ebx
 8049580:  81 ec 60 a0 00 00        sub     $0xa060,%esp
 8049586:  8b 75 08                 mov     0x8(%ebp),%esi
 8049589:  8b 45 10                 mov     0x10(%ebp),%eax
 804958c:  89 85 ac 5f ff ff        mov     %eax,-0xa054(%ebp)
 8049592:  8b 45 14                 mov     0x14(%ebp),%eax
 8049595:  89 85 a8 5f ff ff        mov     %eax,-0xa058(%ebp)
 804959b:  8b 45 18                 mov     0x18(%ebp),%eax
 804959e:  89 85 a4 5f ff ff        mov     %eax,-0xa05c(%ebp)
 80495a4:  8b 5d 1c                 mov     0x1c(%ebp),%ebx
 80495a7:  8b 45 20                 mov     0x20(%ebp),%eax
 80495aa:  89 85 a0 5f ff ff        mov     %eax,-0xa060(%ebp)
 80495b0:  65 a1 14 00 00 00        mov     %gs:0x14,%eax
 80495b6:  89 45 e4                 mov     %eax,-0x1c(%ebp)
 80495b9:  31 c0                    xor     %eax,%eax
 80495bb:  c7 85 c4 5f ff ff 00     movl    $0x0,-0xa03c(%ebp)
 80495c2:  00 00 00
 80495c5:  6a 00                    push    $0x0
 80495c7:  6a 01                    push    $0x1
 80495c9:  6a 02                    push    $0x2
 80495cb:  e8 80 f2 ff ff           call    8048850 <socket@plt>
```

```
80495d0:  89 85 b0 5f ff ff     mov    %eax,-0xa050(%ebp)
80495d6:  83 c4 10                  add    $0x10,%esp
80495d9:  85 c0                     test   %eax,%eax
80495db:  0f 88 30 01 00 00         js     8049711 <submitr+0x197>
80495e1:  83 ec 0c                  sub    $0xc,%esp
80495e4:  56                        push   %esi
80495e5:  e8 86 f2 ff ff        call   8048870 <gethostbyname@plt>
80495ea:  83 c4 10                  add    $0x10,%esp
80495ed:  85 c0                     test   %eax,%eax
80495ef:  0f 84 70 01 00 00         je     8049765 <submitr+0x1eb>
80495f5:  8d b5 c8 5f ff ff     lea    -0xa038(%ebp),%esi
80495fb:  c7 85 ca 5f ff ff 00  movl   $0x0,-0xa036(%ebp)
8049602:  00 00 00
8049605:  c7 85 ce 5f ff ff 00  movl   $0x0,-0xa032(%ebp)
804960c:  00 00 00
804960f:  c7 85 d2 5f ff ff 00  movl   $0x0,-0xa02e(%ebp)
8049616:  00 00 00
8049619:  66 c7 85 d6 5f ff ff  movw   $0x0,-0xa02a(%ebp)
8049620:  00 00
8049622:  66 c7 85 c8 5f ff ff  movw   $0x2,-0xa038(%ebp)
8049629:  02 00
804962b:  6a 0c                     push   $0xc
804962d:  ff 70 0c                  pushl  0xc(%eax)
8049630:  8b 40 10                  mov    0x10(%eax),%eax
8049633:  ff 30                     pushl  (%eax)
8049635:  8d 85 cc 5f ff ff     lea    -0xa034(%ebp),%eax
804963b:  50                        push   %eax
804963c:  e8 8f f1 ff ff        call   80487d0 <__memmove_chk@plt>
8049641:  0f b7 45 0c               movzwl 0xc(%ebp),%eax
```

```
8049645:  66 c1 c8 08              ror     $0x8,%ax
8049649:  66 89 85 ca 5f ff ff     mov     %ax,-0xa036(%ebp)
8049650:  83 c4 0c                 add     $0xc,%esp
8049653:  6a 10                    push    $0x10
8049655:  56                       push    %esi
8049656:  ff b5 b0 5f ff ff        pushl   -0xa050(%ebp)
804965c:  e8 2f f2 ff ff           call    8048890 <connect@plt>
8049661:  83 c4 10                 add     $0x10,%esp
8049664:  85 c0                    test    %eax,%eax
8049666:  0f 88 70 01 00 00        js      80497dc <submitr+0x262>
804966c:  ba ff ff ff ff           mov     $0xffffffff,%edx
8049671:  b8 00 00 00 00           mov     $0x0,%eax
8049676:  89 d1                    mov     %edx,%ecx
8049678:  89 df                    mov     %ebx,%edi
804967a:  f2 ae                    repnz scas %es:(%edi),%al
804967c:  89 ce                    mov     %ecx,%esi
804967e:  f7 d6                    not     %esi
8049680:  89 d1                    mov     %edx,%ecx
8049682:  8b bd ac 5f ff ff        mov     -0xa054(%ebp),%edi
8049688:  f2 ae                    repnz scas %es:(%edi),%al
804968a:  89 8d b4 5f ff ff        mov     %ecx,-0xa04c(%ebp)
8049690:  89 d1                    mov     %edx,%ecx
8049692:  8b bd a8 5f ff ff        mov     -0xa058(%ebp),%edi
8049698:  f2 ae                    repnz scas %es:(%edi),%al
804969a:  89 cf                    mov     %ecx,%edi
804969c:  f7 d7                    not     %edi
804969e:  89 bd 9c 5f ff ff        mov     %edi,-0xa064(%ebp)
80496a4:  89 d1                    mov     %edx,%ecx
80496a6:  8b bd a4 5f ff ff        mov     -0xa05c(%ebp),%edi
```

```
80496ac:  f2 ae                    repnz scas %es:(%edi),%al
80496ae:  8b 95 9c 5f ff ff    mov     -0xa064(%ebp),%edx
80496b4:  2b 95 b4 5f ff ff    sub     -0xa04c(%ebp),%edx
80496ba:  29 ca                    sub     %ecx,%edx
80496bc:  8d 44 76 fd              lea     -0x3(%esi,%esi,2),%eax
80496c0:  8d 44 02 7b              lea     0x7b(%edx,%eax,1),%eax
80496c4:  3d 00 20 00 00           cmp     $0x2000,%eax
80496c9:  0f 87 76 01 00 00        ja      8049845 <submitr+0x2cb>
80496cf:  8d 95 e4 9f ff ff    lea     -0x601c(%ebp),%edx
80496d5:  b9 00 08 00 00           mov     $0x800,%ecx
80496da:  b8 00 00 00 00           mov     $0x0,%eax
80496df:  89 d7                    mov     %edx,%edi
80496e1:  f3 ab                    rep stos %eax,%es:(%edi)
80496e3:  b9 ff ff ff ff       mov     $0xffffffff,%ecx
80496e8:  89 df                    mov     %ebx,%edi
80496ea:  f2 ae                    repnz scas %es:(%edi),%al
80496ec:  89 ca                    mov     %ecx,%edx
80496ee:  f7 d2                    not     %edx
80496f0:  89 d1                    mov     %edx,%ecx
80496f2:  83 e9 01                 sub     $0x1,%ecx
80496f5:  89 8d b4 5f ff ff    mov     %ecx,-0xa04c(%ebp)
80496fb:  0f 84 3b 06 00 00        je      8049d3c <submitr+0x7c2>
8049701:  8d b5 e4 9f ff ff    lea     -0x601c(%ebp),%esi
8049707:  bf 01 00 00 00           mov     $0x1,%edi
804970c:  e9 cb 01 00 00           jmp     80498dc <submitr+0x362>
8049711:  8b 85 a0 5f ff ff    mov     -0xa060(%ebp),%eax
8049717:  c7 00 45 72 72 6f        movl    $0x6f727245,(%eax)
804971d:  c7 40 04 72 3a 20 43     movl    $0x43203a72,0x4(%eax)
8049724:  c7 40 08 6c 69 65 6e     movl    $0x6e65696c,0x8(%eax)
```

```
804972b:  c7 40 0c 74 20 75 6e      movl    $0x6e752074,0xc(%eax)
8049732:  c7 40 10 61 62 6c 65      movl    $0x656c6261,0x10(%eax)
8049739:  c7 40 14 20 74 6f 20      movl    $0x206f7420,0x14(%eax)
8049740:  c7 40 18 63 72 65 61      movl    $0x61657263,0x18(%eax)
8049747:  c7 40 1c 74 65 20 73      movl    $0x73206574,0x1c(%eax)
804974e:  c7 40 20 6f 63 6b 65      movl    $0x656b636f,0x20(%eax)
8049755:  66 c7 40 24 74 00         movw    $0x74,0x24(%eax)
804975b:  b8 ff ff ff ff            mov     $0xffffffff,%eax
8049760:  e9 f2 04 00 00            jmp     8049c57 <submitr+0x6dd>
8049765:  8b 85 a0 5f ff ff         mov     -0xa060(%ebp),%eax
804976b:  c7 00 45 72 72 6f         movl    $0x6f727245,(%eax)
8049771:  c7 40 04 72 3a 20 44      movl    $0x44203a72,0x4(%eax)
8049778:  c7 40 08 4e 53 20 69      movl    $0x6920534e,0x8(%eax)
804977f:  c7 40 0c 73 20 75 6e      movl    $0x6e752073,0xc(%eax)
8049786:  c7 40 10 61 62 6c 65      movl    $0x656c6261,0x10(%eax)
804978d:  c7 40 14 20 74 6f 20      movl    $0x206f7420,0x14(%eax)
8049794:  c7 40 18 72 65 73 6f      movl    $0x6f736572,0x18(%eax)
804979b:  c7 40 1c 6c 76 65 20      movl    $0x2065766c,0x1c(%eax)
80497a2:  c7 40 20 73 65 72 76      movl    $0x76726573,0x20(%eax)
80497a9:  c7 40 24 65 72 20 61      movl    $0x61207265,0x24(%eax)
80497b0:  c7 40 28 64 64 72 65      movl    $0x65726464,0x28(%eax)
80497b7:  66 c7 40 2c 73 73         movw    $0x7373,0x2c(%eax)
80497bd:  c6 40 2e 00               movb    $0x0,0x2e(%eax)
80497c1:  83 ec 0c                  sub     $0xc,%esp
80497c4:  ff b5 b0 5f ff ff         pushl   -0xa050(%ebp)
80497ca:  e8 d1 f0 ff ff            call    80488a0 <close@plt>
80497cf:  83 c4 10                  add     $0x10,%esp
80497d2:  b8 ff ff ff ff            mov     $0xffffffff,%eax
80497d7:  e9 7b 04 00 00            jmp     8049c57 <submitr+0x6dd>
```

```
80497dc:  8b 85 a0 5f ff ff      mov     -0xa060(%ebp),%eax
80497e2:  c7 00 45 72 72 6f      movl    $0x6f727245,(%eax)
80497e8:  c7 40 04 72 3a 20 55   movl    $0x55203a72,0x4(%eax)
80497ef:  c7 40 08 6e 61 62 6c   movl    $0x6c62616e,0x8(%eax)
80497f6:  c7 40 0c 65 20 74 6f   movl    $0x6f742065,0xc(%eax)
80497fd:  c7 40 10 20 63 6f 6e   movl    $0x6e6f6320,0x10(%eax)
8049804:  c7 40 14 6e 65 63 74   movl    $0x7463656e,0x14(%eax)
804980b:  c7 40 18 20 74 6f 20   movl    $0x206f7420,0x18(%eax)
8049812:  c7 40 1c 74 68 65 20   movl    $0x20656874,0x1c(%eax)
8049819:  c7 40 20 73 65 72 76   movl    $0x76726573,0x20(%eax)
8049820:  66 c7 40 24 65 72      movw    $0x7265,0x24(%eax)
8049826:  c6 40 26 00            movb    $0x0,0x26(%eax)
804982a:  83 ec 0c               sub     $0xc,%esp
804982d:  ff b5 b0 5f ff ff      pushl   -0xa050(%ebp)
8049833:  e8 68 f0 ff ff         call    80488a0 <close@plt>
8049838:  83 c4 10               add     $0x10,%esp
804983b:  b8 ff ff ff ff         mov     $0xffffffff,%eax
8049840:  e9 12 04 00 00         jmp     8049c57 <submitr+0x6dd>
8049845:  8b 85 a0 5f ff ff      mov     -0xa060(%ebp),%eax
804984b:  c7 00 45 72 72 6f      movl    $0x6f727245,(%eax)
8049851:  c7 40 04 72 3a 20 52   movl    $0x52203a72,0x4(%eax)
8049858:  c7 40 08 65 73 75 6c   movl    $0x6c757365,0x8(%eax)
804985f:  c7 40 0c 74 20 73 74   movl    $0x74732074,0xc(%eax)
8049866:  c7 40 10 72 69 6e 67   movl    $0x676e6972,0x10(%eax)
804986d:  c7 40 14 20 74 6f 6f   movl    $0x6f6f7420,0x14(%eax)
8049874:  c7 40 18 20 6c 61 72   movl    $0x72616c20,0x18(%eax)
804987b:  c7 40 1c 67 65 2e 20   movl    $0x202e6567,0x1c(%eax)
8049882:  c7 40 20 49 6e 63 72   movl    $0x72636e49,0x20(%eax)
8049889:  c7 40 24 65 61 73 65   movl    $0x65736165,0x24(%eax)
```

```
8049890:  c7 40 28 20 53 55 42     movl    $0x42555320,0x28(%eax)
8049897:  c7 40 2c 4d 49 54 52     movl    $0x5254494d,0x2c(%eax)
804989e:  c7 40 30 5f 4d 41 58     movl    $0x58414d5f,0x30(%eax)
80498a5:  c7 40 34 42 55 46 00     movl    $0x465542,0x34(%eax)
80498ac:  83 ec 0c                 sub     $0xc,%esp
80498af:  ff b5 b0 5f ff ff        pushl   -0xa050(%ebp)
80498b5:  e8 e6 ef ff ff           call    80488a0 <close@plt>
80498ba:  83 c4 10                 add     $0x10,%esp
80498bd:  b8 ff ff ff ff           mov     $0xffffffff,%eax
80498c2:  e9 90 03 00 00           jmp     8049c57 <submitr+0x6dd>
80498c7:  88 16                    mov     %dl,(%esi)
80498c9:  8d 76 01                 lea     0x1(%esi),%esi
80498cc:  83 c3 01                 add     $0x1,%ebx
80498cf:  83 ad b4 5f ff ff 01     subl    $0x1,-0xa04c(%ebp)
80498d6:  0f 84 60 04 00 00        je      8049d3c <submitr+0x7c2>
80498dc:  0f b6 13                 movzbl (%ebx),%edx
80498df:  8d 4a d6                 lea     -0x2a(%edx),%ecx
80498e2:  89 f8                    mov     %edi,%eax
80498e4:  80 f9 0f                 cmp     $0xf,%cl
80498e7:  77 0d                    ja      80498f6 <submitr+0x37c>
80498e9:  b8 d9 ff 00 00           mov     $0xffd9,%eax
80498ee:  d3 e8                    shr     %cl,%eax
80498f0:  83 f0 01                 xor     $0x1,%eax
80498f3:  83 e0 01                 and     $0x1,%eax
80498f6:  84 c0                    test    %al,%al
80498f8:  74 cd                    je      80498c7 <submitr+0x34d>
80498fa:  80 fa 5f                 cmp     $0x5f,%dl
80498fd:  74 c8                    je      80498c7 <submitr+0x34d>
80498ff:  89 d0                    mov     %edx,%eax
```

```
8049901:  83 e0 df              and    $0xffffffdf,%eax
8049904:  83 e8 41              sub    $0x41,%eax
8049907:  3c 19                 cmp    $0x19,%al
8049909:  76 bc                 jbe    80498c7 <submitr+0x34d>
804990b:  80 fa 20              cmp    $0x20,%dl
804990e:  74 54                 je     8049964 <submitr+0x3ea>
8049910:  8d 42 e0              lea    -0x20(%edx),%eax
8049913:  3c 5f                 cmp    $0x5f,%al
8049915:  76 09                 jbe    8049920 <submitr+0x3a6>
8049917:  80 fa 09              cmp    $0x9,%dl
804991a:  0f 85 d1 03 00 00     jne    8049cf1 <submitr+0x777>
8049920:  83 ec 0c              sub    $0xc,%esp
8049923:  0f b6 d2              movzbl %dl,%edx
8049926:  52                    push   %edx
8049927:  68 58 a6 04 08        push   $0x804a658
804992c:  6a 08                 push   $0x8
804992e:  6a 01                 push   $0x1
8049930:  8d 85 e4 df ff ff     lea    -0x201c(%ebp),%eax
8049936:  50                    push   %eax
8049937:  e8 84 ef ff ff        call   80488c0 <__sprintf_chk@plt>
804993c:  0f b6 85 e4 df ff ff  movzbl -0x201c(%ebp),%eax
8049943:  88 06                 mov    %al,(%esi)
8049945:  0f b6 85 e5 df ff ff  movzbl -0x201b(%ebp),%eax
804994c:  88 46 01              mov    %al,0x1(%esi)
804994f:  0f b6 85 e6 df ff ff  movzbl -0x201a(%ebp),%eax
8049956:  88 46 02              mov    %al,0x2(%esi)
8049959:  83 c4 20              add    $0x20,%esp
804995c:  8d 76 03              lea    0x3(%esi),%esi
804995f:  e9 68 ff ff ff        jmp    80498cc <submitr+0x352>
```

```
8049964:  c6 06 2b              movb   $0x2b,(%esi)
8049967:  8d 76 01              lea    0x1(%esi),%esi
804996a:  e9 5d ff ff ff        jmp    80498cc <submitr+0x352>
804996f:  01 c6                 add    %eax,%esi
8049971:  29 c3                 sub    %eax,%ebx
8049973:  74 27                 je     804999c <submitr+0x422>
8049975:  83 ec 04              sub    $0x4,%esp
8049978:  53                    push   %ebx
8049979:  56                    push   %esi
804997a:  57                    push   %edi
804997b:  e8 80 ee ff ff        call   8048800 <write@plt>
8049980:  83 c4 10              add    $0x10,%esp
8049983:  85 c0                 test   %eax,%eax
8049985:  7f e8                 jg     804996f <submitr+0x3f5>
8049987:  e8 a4 ee ff ff        call   8048830 <__errno_location@plt>
804998c:  83 38 04              cmpl   $0x4,(%eax)
804998f:  0f 85 41 01 00 00     jne    8049ad6 <submitr+0x55c>
8049995:  b8 00 00 00 00        mov    $0x0,%eax
804999a:  eb d3                 jmp    804996f <submitr+0x3f5>
804999c:  8b bd b4 5f ff ff     mov    -0xa04c(%ebp),%edi
80499a2:  85 ff                 test   %edi,%edi
80499a4:  0f 88 2c 01 00 00     js     8049ad6 <submitr+0x55c>
80499aa:  8b 85 b0 5f ff ff     mov    -0xa050(%ebp),%eax
80499b0:  89 85 d8 5f ff ff     mov    %eax,-0xa028(%ebp)
80499b6:  c7 85 dc 5f ff ff 00  movl   $0x0,-0xa024(%ebp)
80499bd:  00 00 00
80499c0:  8d 85 e4 5f ff ff     lea    -0xa01c(%ebp),%eax
80499c6:  89 85 e0 5f ff ff     mov    %eax,-0xa020(%ebp)
80499cc:  b9 00 20 00 00        mov    $0x2000,%ecx
```

```
80499d1:  8d 95 e4 7f ff ff     lea     -0x801c(%ebp),%edx
80499d7:  8d 85 d8 5f ff ff     lea     -0xa028(%ebp),%eax
80499dd:  e8 da fa ff ff        call    80494bc <rio_readlineb>
80499e2:  85 c0                 test    %eax,%eax
80499e4:  0f 8e 59 01 00 00     jle     8049b43 <submitr+0x5c9>
80499ea:  83 ec 0c              sub     $0xc,%esp
80499ed:  8d 85 e4 df ff ff     lea     -0x201c(%ebp),%eax
80499f3:  50                    push    %eax
80499f4:  8d 85 c4 5f ff ff     lea     -0xa03c(%ebp),%eax
80499fa:  50                    push    %eax
80499fb:  8d 85 e4 bf ff ff     lea     -0x401c(%ebp),%eax
8049a01:  50                    push    %eax
8049a02:  68 5f a6 04 08        push    $0x804a65f
8049a07:  8d 85 e4 7f ff ff     lea     -0x801c(%ebp),%eax
8049a0d:  50                    push    %eax
8049a0e:  e8 fd ed ff ff        call    8048810 <__isoc99_sscanf@plt>
8049a13:  8b 85 c4 5f ff ff     mov     -0xa03c(%ebp),%eax
8049a19:  83 c4 20              add     $0x20,%esp
8049a1c:  3d c8 00 00 00        cmp     $0xc8,%eax
8049a21:  0f 85 9d 01 00 00     jne     8049bc4 <submitr+0x64a>
8049a27:  8d 9d e4 7f ff ff     lea     -0x801c(%ebp),%ebx
8049a2d:  bf 70 a6 04 08        mov     $0x804a670,%edi
8049a32:  b9 03 00 00 00        mov     $0x3,%ecx
8049a37:  89 de                 mov     %ebx,%esi
8049a39:  f3 a6                 repz cmpsb %es:(%edi),%ds:(%esi)
8049a3b:  0f 97 c0              seta    %al
8049a3e:  1c 00                 sbb     $0x0,%al
8049a40:  84 c0                 test    %al,%al
8049a42:  0f 84 b3 01 00 00     je      8049bfb <submitr+0x681>
```

```
8049a48:  b9 00 20 00 00          mov    $0x2000,%ecx
8049a4d:  89 da                   mov    %ebx,%edx
8049a4f:  8d 85 d8 5f ff ff       lea    -0xa028(%ebp),%eax
8049a55:  e8 62 fa ff ff          call   80494bc <rio_readlineb>
8049a5a:  85 c0                   test   %eax,%eax
8049a5c:  7f cf                   jg     8049a2d <submitr+0x4b3>
8049a5e:  8b 85 a0 5f ff ff       mov    -0xa060(%ebp),%eax
8049a64:  c7 00 45 72 72 6f       movl   $0x6f727245,(%eax)
8049a6a:  c7 40 04 72 3a 20 43    movl   $0x43203a72,0x4(%eax)
8049a71:  c7 40 08 6c 69 65 6e    movl   $0x6e65696c,0x8(%eax)
8049a78:  c7 40 0c 74 20 75 6e    movl   $0x6e752074,0xc(%eax)
8049a7f:  c7 40 10 61 62 6c 65    movl   $0x656c6261,0x10(%eax)
8049a86:  c7 40 14 20 74 6f 20    movl   $0x206f7420,0x14(%eax)
8049a8d:  c7 40 18 72 65 61 64    movl   $0x64616572,0x18(%eax)
8049a94:  c7 40 1c 20 68 65 61    movl   $0x61656820,0x1c(%eax)
8049a9b:  c7 40 20 64 65 72 73    movl   $0x73726564,0x20(%eax)
8049aa2:  c7 40 24 20 66 72 6f    movl   $0x6f726620,0x24(%eax)
8049aa9:  c7 40 28 6d 20 73 65    movl   $0x6573206d,0x28(%eax)
8049ab0:  c7 40 2c 72 76 65 72    movl   $0x72657672,0x2c(%eax)
8049ab7:  c6 40 30 00             movb   $0x0,0x30(%eax)
8049abb:  83 ec 0c                sub    $0xc,%esp
8049abe:  ff b5 b0 5f ff ff       pushl  -0xa050(%ebp)
8049ac4:  e8 d7 ed ff ff          call   80488a0 <close@plt>
8049ac9:  83 c4 10                add    $0x10,%esp
8049acc:  b8 ff ff ff ff          mov    $0xffffffff,%eax
8049ad1:  e9 81 01 00 00          jmp    8049c57 <submitr+0x6dd>
8049ad6:  8b 85 a0 5f ff ff       mov    -0xa060(%ebp),%eax
8049adc:  c7 00 45 72 72 6f       movl   $0x6f727245,(%eax)
8049ae2:  c7 40 04 72 3a 20 43    movl   $0x43203a72,0x4(%eax)
```

```
8049ae9:  c7 40 08 6c 69 65 6e     movl   $0x6e65696c,0x8(%eax)
8049af0:  c7 40 0c 74 20 75 6e     movl   $0x6e752074,0xc(%eax)
8049af7:  c7 40 10 61 62 6c 65     movl   $0x656c6261,0x10(%eax)
8049afe:  c7 40 14 20 74 6f 20     movl   $0x206f7420,0x14(%eax)
8049b05:  c7 40 18 77 72 69 74     movl   $0x74697277,0x18(%eax)
8049b0c:  c7 40 1c 65 20 74 6f     movl   $0x6f742065,0x1c(%eax)
8049b13:  c7 40 20 20 74 68 65     movl   $0x65687420,0x20(%eax)
8049b1a:  c7 40 24 20 73 65 72     movl   $0x72657320,0x24(%eax)
8049b21:  c7 40 28 76 65 72 00     movl   $0x726576,0x28(%eax)
8049b28:  83 ec 0c                 sub    $0xc,%esp
8049b2b:  ff b5 b0 5f ff ff        pushl  -0xa050(%ebp)
8049b31:  e8 6a ed ff ff           call   80488a0 <close@plt>
8049b36:  83 c4 10                 add    $0x10,%esp
8049b39:  b8 ff ff ff ff           mov    $0xffffffff,%eax
8049b3e:  e9 14 01 00 00           jmp    8049c57 <submitr+0x6dd>
8049b43:  8b 85 a0 5f ff ff        mov    -0xa060(%ebp),%eax
8049b49:  c7 00 45 72 72 6f        movl   $0x6f727245,(%eax)
8049b4f:  c7 40 04 72 3a 20 43     movl   $0x43203a72,0x4(%eax)
8049b56:  c7 40 08 6c 69 65 6e     movl   $0x6e65696c,0x8(%eax)
8049b5d:  c7 40 0c 74 20 75 6e     movl   $0x6e752074,0xc(%eax)
8049b64:  c7 40 10 61 62 6c 65     movl   $0x656c6261,0x10(%eax)
8049b6b:  c7 40 14 20 74 6f 20     movl   $0x206f7420,0x14(%eax)
8049b72:  c7 40 18 72 65 61 64     movl   $0x64616572,0x18(%eax)
8049b79:  c7 40 1c 20 66 69 72     movl   $0x72696620,0x1c(%eax)
8049b80:  c7 40 20 73 74 20 68     movl   $0x68207473,0x20(%eax)
8049b87:  c7 40 24 65 61 64 65     movl   $0x65646165,0x24(%eax)
8049b8e:  c7 40 28 72 20 66 72     movl   $0x72662072,0x28(%eax)
8049b95:  c7 40 2c 6f 6d 20 73     movl   $0x73206d6f,0x2c(%eax)
8049b9c:  c7 40 30 65 72 76 65     movl   $0x65767265,0x30(%eax)
```

```
8049ba3:  66 c7 40 34 72 00      movw   $0x72,0x34(%eax)
8049ba9:  83 ec 0c               sub    $0xc,%esp
8049bac:  ff b5 b0 5f ff ff      pushl  -0xa050(%ebp)
8049bb2:  e8 e9 ec ff ff         call   80488a0 <close@plt>
8049bb7:  83 c4 10               add    $0x10,%esp
8049bba:  b8 ff ff ff ff         mov    $0xffffffff,%eax
8049bbf:  e9 93 00 00 00         jmp    8049c57 <submitr+0x6dd>
8049bc4:  83 ec 08               sub    $0x8,%esp
8049bc7:  8d 95 e4 df ff ff      lea    -0x201c(%ebp),%edx
8049bcd:  52                     push   %edx
8049bce:  50                     push   %eax
8049bcf:  68 70 a5 04 08         push   $0x804a570
8049bd4:  6a ff                  push   $0xffffffff
8049bd6:  6a 01                  push   $0x1
8049bd8:  ff b5 a0 5f ff ff      pushl  -0xa060(%ebp)
8049bde:  e8 dd ec ff ff         call   80488c0 <__sprintf_chk@plt>
8049be3:  83 c4 14               add    $0x14,%esp
8049be6:  ff b5 b0 5f ff ff      pushl  -0xa050(%ebp)
8049bec:  e8 af ec ff ff         call   80488a0 <close@plt>
8049bf1:  83 c4 10               add    $0x10,%esp
8049bf4:  b8 ff ff ff ff         mov    $0xffffffff,%eax
8049bf9:  eb 5c                  jmp    8049c57 <submitr+0x6dd>
8049bfb:  b9 00 20 00 00         mov    $0x2000,%ecx
8049c00:  8d 95 e4 7f ff ff      lea    -0x801c(%ebp),%edx
8049c06:  8d 85 d8 5f ff ff      lea    -0xa028(%ebp),%eax
8049c0c:  e8 ab f8 ff ff         call   80494bc <rio_readlineb>
8049c11:  85 c0                  test   %eax,%eax
8049c13:  7e 5a                  jle    8049c6f <submitr+0x6f5>
8049c15:  83 ec 08               sub    $0x8,%esp
```

```
8049c18:  8d 85 e4 7f ff ff     lea     -0x801c(%ebp),%eax
8049c1e:  50                    push    %eax
8049c1f:  8b b5 a0 5f ff ff     mov     -0xa060(%ebp),%esi
8049c25:  56                    push    %esi
8049c26:  e8 75 eb ff ff        call    80487a0 <strcpy@plt>
8049c2b:  83 c4 04              add     $0x4,%esp
8049c2e:  ff b5 b0 5f ff ff     pushl   -0xa050(%ebp)
8049c34:  e8 67 ec ff ff        call    80488a0 <close@plt>
8049c39:  bf 73 a6 04 08        mov     $0x804a673,%edi
8049c3e:  b9 03 00 00 00        mov     $0x3,%ecx
8049c43:  f3 a6                 repz cmpsb %es:(%edi),%ds:(%esi)
8049c45:  0f 97 c0              seta    %al
8049c48:  1c 00                 sbb     $0x0,%al
8049c4a:  83 c4 10              add     $0x10,%esp
8049c4d:  84 c0                 test    %al,%al
8049c4f:  0f 95 c0              setne   %al
8049c52:  0f b6 c0              movzbl %al,%eax
8049c55:  f7 d8                 neg     %eax
8049c57:  8b 7d e4              mov     -0x1c(%ebp),%edi
8049c5a:  65 33 3d 14 00 00 00  xor     %gs:0x14,%edi
8049c61:  0f 85 3d 01 00 00     jne     8049da4 <submitr+0x82a>
8049c67:  8d 65 f4              lea     -0xc(%ebp),%esp
8049c6a:  5b                    pop     %ebx
8049c6b:  5e                    pop     %esi
8049c6c:  5f                    pop     %edi
8049c6d:  5d                    pop     %ebp
8049c6e:  c3                    ret
8049c6f:  8b 85 a0 5f ff ff     mov     -0xa060(%ebp),%eax
8049c75:  c7 00 45 72 72 6f     movl    $0x6f727245,(%eax)
```

```
8049c7b:  c7 40 04 72 3a 20 43     movl   $0x43203a72,0x4(%eax)
8049c82:  c7 40 08 6c 69 65 6e     movl   $0x6e65696c,0x8(%eax)
8049c89:  c7 40 0c 74 20 75 6e     movl   $0x6e752074,0xc(%eax)
8049c90:  c7 40 10 61 62 6c 65     movl   $0x656c6261,0x10(%eax)
8049c97:  c7 40 14 20 74 6f 20     movl   $0x206f7420,0x14(%eax)
8049c9e:  c7 40 18 72 65 61 64     movl   $0x64616572,0x18(%eax)
8049ca5:  c7 40 1c 20 73 74 61     movl   $0x61747320,0x1c(%eax)
8049cac:  c7 40 20 74 75 73 20     movl   $0x20737574,0x20(%eax)
8049cb3:  c7 40 24 6d 65 73 73     movl   $0x7373656d,0x24(%eax)
8049cba:  c7 40 28 61 67 65 20     movl   $0x20656761,0x28(%eax)
8049cc1:  c7 40 2c 66 72 6f 6d     movl   $0x6d6f7266,0x2c(%eax)
8049cc8:  c7 40 30 20 73 65 72     movl   $0x72657320,0x30(%eax)
8049ccf:  c7 40 34 76 65 72 00     movl   $0x726576,0x34(%eax)
8049cd6:  83 ec 0c                 sub    $0xc,%esp
8049cd9:  ff b5 b0 5f ff ff        pushl  -0xa050(%ebp)
8049cdf:  e8 bc eb ff ff           call   80488a0 <close@plt>
8049ce4:  83 c4 10                 add    $0x10,%esp
8049ce7:  b8 ff ff ff ff           mov    $0xffffffff,%eax
8049cec:  e9 66 ff ff ff           jmp    8049c57 <submitr+0x6dd>
8049cf1:  a1 a0 a5 04 08           mov    0x804a5a0,%eax
8049cf6:  8b bd a0 5f ff ff        mov    -0xa060(%ebp),%edi
8049cfc:  89 07                    mov    %eax,(%edi)
8049cfe:  a1 df a5 04 08           mov    0x804a5df,%eax
8049d03:  89 47 3f                 mov    %eax,0x3f(%edi)
8049d06:  89 f8                    mov    %edi,%eax
8049d08:  8d 7f 04                 lea    0x4(%edi),%edi
8049d0b:  83 e7 fc                 and    $0xfffffffc,%edi
8049d0e:  29 f8                    sub    %edi,%eax
8049d10:  be a0 a5 04 08           mov    $0x804a5a0,%esi
```

```
8049d15:  29 c6                   sub    %eax,%esi
8049d17:  83 c0 43                add    $0x43,%eax
8049d1a:  c1 e8 02                shr    $0x2,%eax
8049d1d:  89 c1                   mov    %eax,%ecx
8049d1f:  f3 a5                   rep movsl %ds:(%esi),%es:(%edi)
8049d21:  83 ec 0c                sub    $0xc,%esp
8049d24:  ff b5 b0 5f ff ff       pushl  -0xa050(%ebp)
8049d2a:  e8 71 eb ff ff          call   80488a0 <close@plt>
8049d2f:  83 c4 10                add    $0x10,%esp
8049d32:  b8 ff ff ff ff          mov    $0xffffffff,%eax
8049d37:  e9 1b ff ff ff          jmp    8049c57 <submitr+0x6dd>
8049d3c:  8d 85 e4 9f ff ff       lea    -0x601c(%ebp),%eax
8049d42:  50                      push   %eax
8049d43:  ff b5 a4 5f ff ff       pushl  -0xa05c(%ebp)
8049d49:  ff b5 a8 5f ff ff       pushl  -0xa058(%ebp)
8049d4f:  ff b5 ac 5f ff ff       pushl  -0xa054(%ebp)
8049d55:  68 e4 a5 04 08          push   $0x804a5e4
8049d5a:  68 00 20 00 00          push   $0x2000
8049d5f:  6a 01                   push   $0x1
8049d61:  8d bd e4 7f ff ff       lea    -0x801c(%ebp),%edi
8049d67:  57                      push   %edi
8049d68:  e8 53 eb ff ff          call   80488c0 <__sprintf_chk@plt>
8049d6d:  b9 ff ff ff ff          mov    $0xffffffff,%ecx
8049d72:  b8 00 00 00 00          mov    $0x0,%eax
8049d77:  f2 ae                   repnz scas %es:(%edi),%al
8049d79:  89 cb                   mov    %ecx,%ebx
8049d7b:  f7 d3                   not    %ebx
8049d7d:  8d 7b ff                lea    -0x1(%ebx),%edi
8049d80:  83 c4 20                add    $0x20,%esp
```

```
 8049d83:  89 fb                   mov    %edi,%ebx
 8049d85:  8d b5 e4 7f ff ff       lea    -0x801c(%ebp),%esi
 8049d8b:  85 ff                   test   %edi,%edi
 8049d8d:  0f 84 17 fc ff ff       je     80499aa <submitr+0x430>
 8049d93:  89 bd b4 5f ff ff       mov    %edi,-0xa04c(%ebp)
 8049d99:  8b bd b0 5f ff ff       mov    -0xa050(%ebp),%edi
 8049d9f:  e9 d1 fb ff ff          jmp    8049975 <submitr+0x3fb>
 8049da4:  e8 e7 e9 ff ff          call   8048790 <__stack_chk_fail@plt>

08049da9 <init_timeout>:
 8049da9:  55                      push   %ebp
 8049daa:  89 e5                   mov    %esp,%ebp
 8049dac:  53                      push   %ebx
 8049dad:  83 ec 04                sub    $0x4,%esp
 8049db0:  8b 5d 08                mov    0x8(%ebp),%ebx
 8049db3:  85 db                   test   %ebx,%ebx
 8049db5:  74 24                   je     8049ddb <init_timeout+0x32>
 8049db7:  83 ec 08                sub    $0x8,%esp
 8049dba:  68 96 94 04 08          push   $0x8049496
 8049dbf:  6a 0e                   push   $0xe
 8049dc1:  e8 9a e9 ff ff          call   8048760 <signal@plt>
 8049dc6:  85 db                   test   %ebx,%ebx
 8049dc8:  b8 00 00 00 00          mov    $0x0,%eax
 8049dcd:  0f 48 d8                cmovs  %eax,%ebx
 8049dd0:  89 1c 24                mov    %ebx,(%esp)
 8049dd3:  e8 a8 e9 ff ff          call   8048780 <alarm@plt>
 8049dd8:  83 c4 10                add    $0x10,%esp
 8049ddb:  8b 5d fc                mov    -0x4(%ebp),%ebx
 8049dde:  c9                      leave
```

```
 8049ddf:   c3                      ret


08049de0 <init_driver>:
 8049de0:   55                      push   %ebp
 8049de1:   89 e5                   mov    %esp,%ebp
 8049de3:   57                      push   %edi
 8049de4:   56                      push   %esi
 8049de5:   53                      push   %ebx
 8049de6:   83 ec 34                sub    $0x34,%esp
 8049de9:   8b 75 08                mov    0x8(%ebp),%esi
 8049dec:   65 a1 14 00 00 00       mov    %gs:0x14,%eax
 8049df2:   89 45 e4                mov    %eax,-0x1c(%ebp)
 8049df5:   31 c0                   xor    %eax,%eax
 8049df7:   6a 01                   push   $0x1
 8049df9:   6a 0d                   push   $0xd
 8049dfb:   e8 60 e9 ff ff          call   8048760 <signal@plt>
 8049e00:   83 c4 08                add    $0x8,%esp
 8049e03:   6a 01                   push   $0x1
 8049e05:   6a 1d                   push   $0x1d
 8049e07:   e8 54 e9 ff ff          call   8048760 <signal@plt>
 8049e0c:   83 c4 08                add    $0x8,%esp
 8049e0f:   6a 01                   push   $0x1
 8049e11:   6a 1d                   push   $0x1d
 8049e13:   e8 48 e9 ff ff          call   8048760 <signal@plt>
 8049e18:   83 c4 0c                add    $0xc,%esp
 8049e1b:   6a 00                   push   $0x0
 8049e1d:   6a 01                   push   $0x1
 8049e1f:   6a 02                   push   $0x2
 8049e21:   e8 2a ea ff ff          call   8048850 <socket@plt>
```

```
8049e26:   83 c4 10                add    $0x10,%esp
8049e29:   85 c0                   test   %eax,%eax
8049e2b:   0f 88 a0 00 00 00       js     8049ed1 <init_driver+0xf1>
8049e31:   89 c3                   mov    %eax,%ebx
8049e33:   83 ec 0c                sub    $0xc,%esp
8049e36:   68 76 a6 04 08          push   $0x804a676
8049e3b:   e8 30 ea ff ff          call   8048870 <gethostbyname@plt>
8049e40:   83 c4 10                add    $0x10,%esp
8049e43:   85 c0                   test   %eax,%eax
8049e45:   0f 84 d1 00 00 00       je     8049f1c <init_driver+0x13c>
8049e4b:   8d 7d d4                lea    -0x2c(%ebp),%edi
8049e4e:   c7 45 d6 00 00 00 00    movl   $0x0,-0x2a(%ebp)
8049e55:   c7 45 da 00 00 00 00    movl   $0x0,-0x26(%ebp)
8049e5c:   c7 45 de 00 00 00 00    movl   $0x0,-0x22(%ebp)
8049e63:   66 c7 45 e2 00 00       movw   $0x0,-0x1e(%ebp)
8049e69:   66 c7 45 d4 02 00       movw   $0x2,-0x2c(%ebp)
8049e6f:   6a 0c                   push   $0xc
8049e71:   ff 70 0c                pushl  0xc(%eax)
8049e74:   8b 40 10                mov    0x10(%eax),%eax
8049e77:   ff 30                   pushl  (%eax)
8049e79:   8d 45 d8                lea    -0x28(%ebp),%eax
8049e7c:   50                      push   %eax
8049e7d:   e8 4e e9 ff ff          call   80487d0 <__memmove_chk@plt>
8049e82:   66 c7 45 d6 22 b9       movw   $0xb922,-0x2a(%ebp)
8049e88:   83 c4 0c                add    $0xc,%esp
8049e8b:   6a 10                   push   $0x10
8049e8d:   57                      push   %edi
8049e8e:   53                      push   %ebx
8049e8f:   e8 fc e9 ff ff          call   8048890 <connect@plt>
```

```
8049e94:   83 c4 10                 add    $0x10,%esp
8049e97:   85 c0                    test   %eax,%eax
8049e99:   0f 88 e9 00 00 00        js     8049f88 <init_driver+0x1a8>
8049e9f:   83 ec 0c                 sub    $0xc,%esp
8049ea2:   53                       push   %ebx
8049ea3:   e8 f8 e9 ff ff           call   80488a0 <close@plt>
8049ea8:   66 c7 06 4f 4b           movw   $0x4b4f,(%esi)
8049ead:   c6 46 02 00              movb   $0x0,0x2(%esi)
8049eb1:   83 c4 10                 add    $0x10,%esp
8049eb4:   b8 00 00 00 00           mov    $0x0,%eax
8049eb9:   8b 55 e4                 mov    -0x1c(%ebp),%edx
8049ebc:   65 33 15 14 00 00 00     xor    %gs:0x14,%edx
8049ec3:   0f 85 ec 00 00 00        jne    8049fb5 <init_driver+0x1d5>
8049ec9:   8d 65 f4                 lea    -0xc(%ebp),%esp
8049ecc:   5b                       pop    %ebx
8049ecd:   5e                       pop    %esi
8049ece:   5f                       pop    %edi
8049ecf:   5d                       pop    %ebp
8049ed0:   c3                       ret
8049ed1:   c7 06 45 72 72 6f        movl   $0x6f727245,(%esi)
8049ed7:   c7 46 04 72 3a 20 43     movl   $0x43203a72,0x4(%esi)
8049ede:   c7 46 08 6c 69 65 6e     movl   $0x6e65696c,0x8(%esi)
8049ee5:   c7 46 0c 74 20 75 6e     movl   $0x6e752074,0xc(%esi)
8049eec:   c7 46 10 61 62 6c 65     movl   $0x656c6261,0x10(%esi)
8049ef3:   c7 46 14 20 74 6f 20     movl   $0x206f7420,0x14(%esi)
8049efa:   c7 46 18 63 72 65 61     movl   $0x61657263,0x18(%esi)
8049f01:   c7 46 1c 74 65 20 73     movl   $0x73206574,0x1c(%esi)
8049f08:   c7 46 20 6f 63 6b 65     movl   $0x656b636f,0x20(%esi)
8049f0f:   66 c7 46 24 74 00        movw   $0x74,0x24(%esi)
```

```
8049f15:  b8 ff ff ff ff          mov    $0xffffffff,%eax
8049f1a:  eb 9d                   jmp    8049eb9 <init_driver+0xd9>
8049f1c:  c7 06 45 72 72 6f       movl   $0x6f727245,(%esi)
8049f22:  c7 46 04 72 3a 20 44    movl   $0x44203a72,0x4(%esi)
8049f29:  c7 46 08 4e 53 20 69    movl   $0x6920534e,0x8(%esi)
8049f30:  c7 46 0c 73 20 75 6e    movl   $0x6e752073,0xc(%esi)
8049f37:  c7 46 10 61 62 6c 65    movl   $0x656c6261,0x10(%esi)
8049f3e:  c7 46 14 20 74 6f 20    movl   $0x206f7420,0x14(%esi)
8049f45:  c7 46 18 72 65 73 6f    movl   $0x6f736572,0x18(%esi)
8049f4c:  c7 46 1c 6c 76 65 20    movl   $0x2065766c,0x1c(%esi)
8049f53:  c7 46 20 73 65 72 76    movl   $0x76726573,0x20(%esi)
8049f5a:  c7 46 24 65 72 20 61    movl   $0x61207265,0x24(%esi)
8049f61:  c7 46 28 64 64 72 65    movl   $0x65726464,0x28(%esi)
8049f68:  66 c7 46 2c 73 73       movw   $0x7373,0x2c(%esi)
8049f6e:  c6 46 2e 00             movb   $0x0,0x2e(%esi)
8049f72:  83 ec 0c                sub    $0xc,%esp
8049f75:  53                      push   %ebx
8049f76:  e8 25 e9 ff ff          call   80488a0 <close@plt>
8049f7b:  83 c4 10                add    $0x10,%esp
8049f7e:  b8 ff ff ff ff          mov    $0xffffffff,%eax
8049f83:  e9 31 ff ff ff          jmp    8049eb9 <init_driver+0xd9>
8049f88:  83 ec 0c                sub    $0xc,%esp
8049f8b:  68 76 a6 04 08          push   $0x804a676
8049f90:  68 30 a6 04 08          push   $0x804a630
8049f95:  6a ff                   push   $0xffffffff
8049f97:  6a 01                   push   $0x1
8049f99:  56                      push   %esi
8049f9a:  e8 21 e9 ff ff          call   80488c0 <__sprintf_chk@plt>
8049f9f:  83 c4 14                add    $0x14,%esp
```

```
8049fa2:   53                    push   %ebx
8049fa3:   e8 f8 e8 ff ff        call   80488a0 <close@plt>
8049fa8:   83 c4 10              add    $0x10,%esp
8049fab:   b8 ff ff ff ff        mov    $0xffffffff,%eax
8049fb0:   e9 04 ff ff ff        jmp    8049eb9 <init_driver+0xd9>
8049fb5:   e8 d6 e7 ff ff        call   8048790 <__stack_chk_fail@plt>

08049fba <driver_post>:
8049fba:   55                    push   %ebp
8049fbb:   89 e5                 mov    %esp,%ebp
8049fbd:   53                    push   %ebx
8049fbe:   83 ec 04              sub    $0x4,%esp
8049fc1:   8b 55 08              mov    0x8(%ebp),%edx
8049fc4:   8b 45 10              mov    0x10(%ebp),%eax
8049fc7:   8b 5d 14              mov    0x14(%ebp),%ebx
8049fca:   85 c0                 test   %eax,%eax
8049fcc:   75 17                 jne    8049fe5 <driver_post+0x2b>
8049fce:   85 d2                 test   %edx,%edx
8049fd0:   74 05                 je     8049fd7 <driver_post+0x1d>
8049fd2:   80 3a 00              cmpb   $0x0,(%edx)
8049fd5:   75 33                 jne    804a00a <driver_post+0x50>
8049fd7:   66 c7 03 4f 4b        movw   $0x4b4f,(%ebx)
8049fdc:   c6 43 02 00           movb   $0x0,0x2(%ebx)
8049fe0:   8b 5d fc              mov    -0x4(%ebp),%ebx
8049fe3:   c9                    leave
8049fe4:   c3                    ret
8049fe5:   83 ec 04              sub    $0x4,%esp
8049fe8:   ff 75 0c              pushl  0xc(%ebp)
8049feb:   68 85 a6 04 08        push   $0x804a685
```

```
8049ff0:   6a 01                    push   $0x1
8049ff2:   e8 49 e8 ff ff          call   8048840 <__printf_chk@plt>
8049ff7:   66 c7 03 4f 4b          movw   $0x4b4f,(%ebx)
8049ffc:   c6 43 02 00             movb   $0x0,0x2(%ebx)
804a000:   83 c4 10                add    $0x10,%esp
804a003:   b8 00 00 00 00          mov    $0x0,%eax
804a008:   eb d6                   jmp    8049fe0 <driver_post+0x26>
804a00a:   83 ec 04                sub    $0x4,%esp
804a00d:   53                      push   %ebx
804a00e:   ff 75 0c                pushl  0xc(%ebp)
804a011:   68 9c a6 04 08          push   $0x804a69c
804a016:   52                      push   %edx
804a017:   68 b3 a6 04 08          push   $0x804a6b3
804a01c:   68 b9 22 00 00          push   $0x22b9
804a021:   68 76 a6 04 08          push   $0x804a676
804a026:   e8 4f f5 ff ff          call   804957a <submitr>
804a02b:   83 c4 20                add    $0x20,%esp
804a02e:   eb b0                   jmp    8049fe0 <driver_post+0x26>

0804a030 <__libc_csu_init>:
804a030:   55                      push   %ebp
804a031:   57                      push   %edi
804a032:   56                      push   %esi
804a033:   53                      push   %ebx
804a034:   e8 f7 e8 ff ff          call   8048930 <__x86.get_pc_thunk.bx>
804a039:   81 c3 c7 1f 00 00       add    $0x1fc7,%ebx
804a03f:   83 ec 0c                sub    $0xc,%esp
804a042:   8b 6c 24 28             mov    0x28(%esp),%ebp
804a046:   8d b3 10 ff ff ff       lea    -0xf0(%ebx),%esi
```

```
 804a04c:   e8 a3 e6 ff ff          call   80486f4 <_init>
 804a051:   8d 83 0c ff ff ff       lea    -0xf4(%ebx),%eax
 804a057:   29 c6                   sub    %eax,%esi
 804a059:   c1 fe 02                sar    $0x2,%esi
 804a05c:   85 f6                   test   %esi,%esi
 804a05e:   74 25                   je     804a085 <__libc_csu_init+0x55>
 804a060:   31 ff                   xor    %edi,%edi
 804a062:   8d b6 00 00 00 00       lea    0x0(%esi),%esi
 804a068:   83 ec 04                sub    $0x4,%esp
 804a06b:   55                      push   %ebp
 804a06c:   ff 74 24 2c             pushl  0x2c(%esp)
 804a070:   ff 74 24 2c             pushl  0x2c(%esp)
 804a074:   ff 94 bb 0c ff ff ff    call   *-0xf4(%ebx,%edi,4)
 804a07b:   83 c7 01                add    $0x1,%edi
 804a07e:   83 c4 10                add    $0x10,%esp
 804a081:   39 fe                   cmp    %edi,%esi
 804a083:   75 e3                   jne    804a068 <__libc_csu_init+0x38>
 804a085:   83 c4 0c                add    $0xc,%esp
 804a088:   5b                      pop    %ebx
 804a089:   5e                      pop    %esi
 804a08a:   5f                      pop    %edi
 804a08b:   5d                      pop    %ebp
 804a08c:   c3                      ret
 804a08d:   8d 76 00                lea    0x0(%esi),%esi

0804a090 <__libc_csu_fini>:
 804a090:   f3 c3                   repz ret

Disassembly of section .fini:
```

```
0804a094 <_fini>:
 804a094:   53                      push   %ebx
 804a095:   83 ec 08                sub    $0x8,%esp
 804a098:   e8 93 e8 ff ff          call   8048930 <__x86.get_pc_thunk.bx>
 804a09d:   81 c3 63 1f 00 00       add    $0x1f63,%ebx
 804a0a3:   83 c4 08                add    $0x8,%esp
 804a0a6:   5b                      pop    %ebx
 804a0a7:   c3                      ret
```