

**实验报告**



**题目： 拆解二进制炸弹**

**班 级： 2023211311**

**学 号： 2023211198**

**姓 名： 杜昊阳**

**学 院： 计算机学院**

**2024 年 11月 12 日**

一、实验目的

1.理解C语言程序的机器级表示。

2.初步掌握GDB调试器的用法。

3.阅读C编译器生成的x86-64机器代码，理解不同控制结构生成的基本指令模式，过程的实现。

1. 实验环境
2. 服务器
   1. 服务器：10.120.11.12（X86版本）：**2024年11月14日23:59分关机（三周），提交作业截止时间2022年11月15日00：00，延迟扣分（每晚一天扣1.5分）**
   2. 服务器：10.99.0.230（华为arm版本，需要报名然后建立帐户）：**2024年11月20日23:59分关机（三周），提交作业截止时间2022年11月21日00：00，延迟扣分（每晚一天扣1.5分）**
   3. 服务器密码找回指南：<https://khbgo05wz2.feishu.cn/docx/doxcnioT927j2gDSTwHoAQLGVpg>
3. MobaXterm
4. Linux
5. Objdump命令反汇编
6. GDB调试工具
7. 积分榜

9-12班：[http://](http://10.120.11.13:19210/scoreboard)10.120.11.13:19230/scoreboard

13-21班：[http://](http://10.120.11.13:19210/scoreboard)10.120.11.13:19240/scoreboard

三、实验内容

登录bupt1服务器，在home目录下可以找到Evil博士专门为你量身定制的一个bomb，当运行时，它会要求你输入一个字符串，如果正确，则进入下一关，继续要求你输入下一个字符串；否则，炸弹就会爆炸，输出一行提示信息并向计分服务器提交扣分信息。因此，本实验要求你必须通过反汇编和逆向工程对bomb执行文件进行分析，找到正确的字符串来解除这个的炸弹。

本实验通过要求使用课程所学知识拆除一个“binary bombs”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方面原理与技能的掌握。 “binary bombs”是一个Linux可执行程序，包含了5个阶段（或关卡）。炸弹运行的每个阶段要求你输入一个特定字符串，你的输入符合程序预期的输入，该阶段的炸弹就被拆除引信；否则炸弹“爆炸”，打印输出 “BOOM!!!”。炸弹的每个阶段考察了机器级程序语言的一个不同方面，难度逐级递增。

为完成二进制炸弹拆除任务，需要使用gdb调试器和objdump来反汇编bomb文件，可以单步跟踪调试每一阶段的机器代码，也可以阅读反汇编代码，从中理解每一汇编语言代码的行为或作用，进而设法推断拆除炸弹所需的目标字符串。实验2的具体内容见实验2说明。

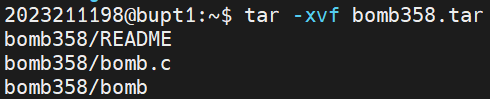
四、实验步骤及实验分析

建议按照：准备工作、阶段1、阶段2、…等来组织内容

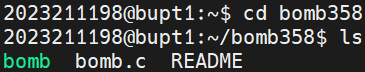
各阶段需要有操作步骤、运行截图、分析过程的内容

准备工作

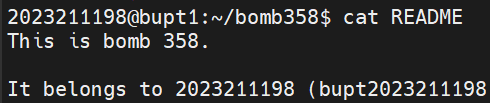
1.解压炸弹：使用tar -xvf bomb358.tar解压，得到炸弹目录



2.使用cd bomb358命令进入目录，使用ls命令，可看到三个文件bomb,bomb.c,README



3.输入cat README命令，阅读README文件内容，确认炸弹bomb358属于学生2023211198



4. 用vi编辑器，输入命令vi answer ，创建一个文件answer，用于存储答案

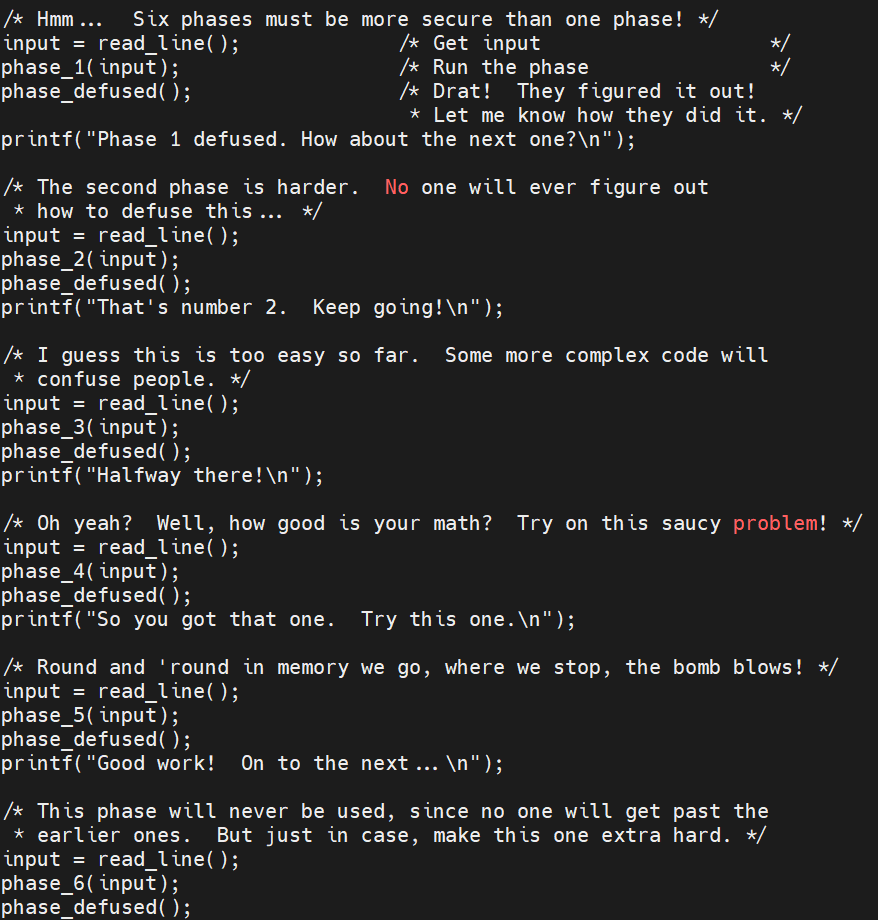


阶段一 总览

1.查看bomb.c文件内的代码



2. 发现实验分为六步，分别位于phase\_1,phase\_2,phase\_3,phase\_4,phase\_5,phase\_6六个函数中



3. 观察bomb.c文件最后的文字提示，推测还存在第七个隐藏关卡

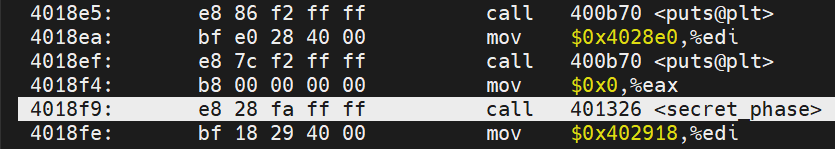


4. 寻找第七个炸弹

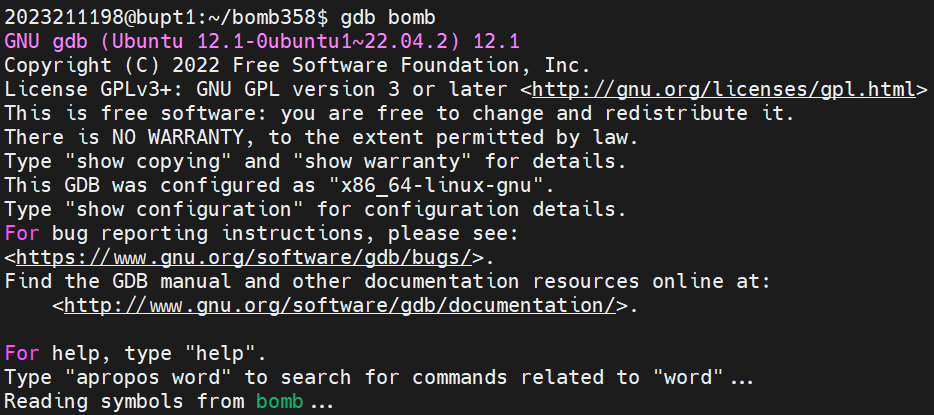
(1) 使用objdump工具objdump -d bomb > bomb.asm查看bomb文件的代码，



(2)在phase\_defused函数的代码中，我们可以发现隐藏的炸弹关卡secret\_phase

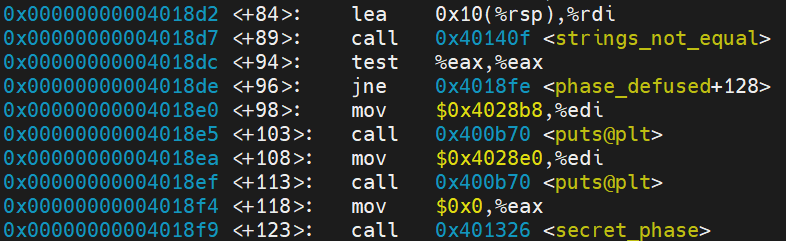


(3) 输入gdb bomb，进入gdb调试器



(4) 键入disas phase\_defused,获得phase\_defused函数的汇编代码  
观察汇编代码，进入隐藏关卡secret\_phase的条件包括通过前六个关卡（即输入d字符串数达到6），和两个整形参数，一个字符串参数





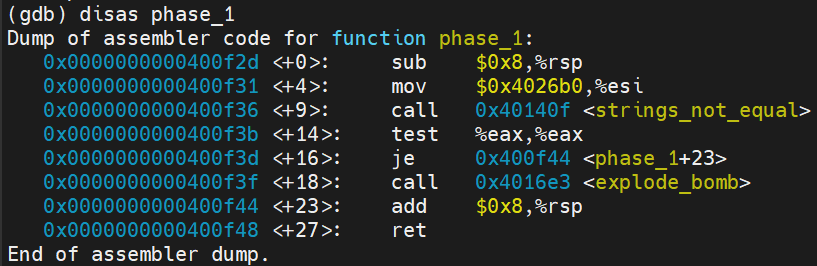
阶段二 拆解phase\_1

此阶段为字符串比较

1.输入b phase\_1，在phase\_1处设置断点



2.键入命令disas phase\_1,观察phase\_1的代码



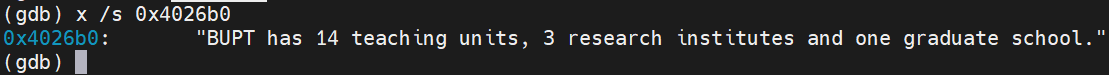
3. 根据<phase\_1+9>处的代码，发现该关卡应输入一字符串，并与对应的字符串相互比较，二者一致则拆除炸弹通过关。



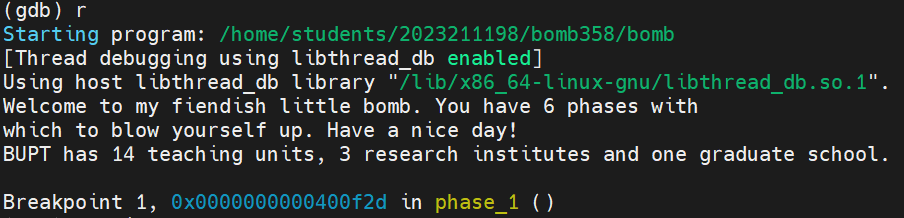
4. 根据<phasse\_1+4>处的代码，可发现从键盘输入的字符串应与存储在0x4026b0处的字符串相互比较。



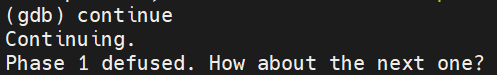
5. 故键入x /s 0x402670,以字符串形式显示该地址处的数据。得到一串字符串BUPT has 14 teaching units, 3 research institutes and one graduate school. 此即为第一关卡答案。



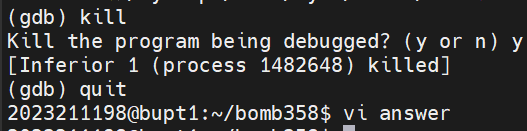
6. 键入命令r运行程序，输入字符串BUPT has 14 teaching units, 3 research institutes and one graduate school.



7.使用continue，观察到炸弹未爆炸，phase1成功拆除



8.kill退出进程，quit退出gdb，打开answer存储phase\_1答案



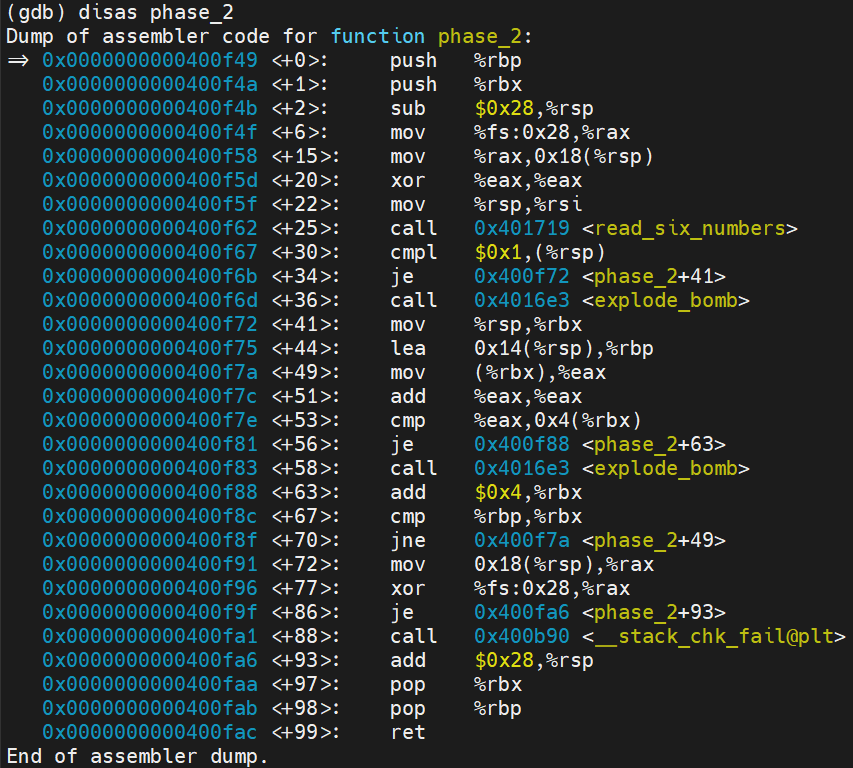
阶段三 拆解phase\_2

此阶段为数值计算

1.输入b phase\_2，在phase\_2处设置断点



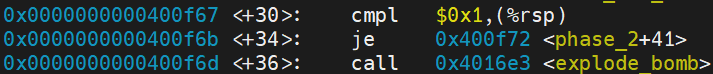
2. 键入命令disas phase\_2,观察phase\_2的代码



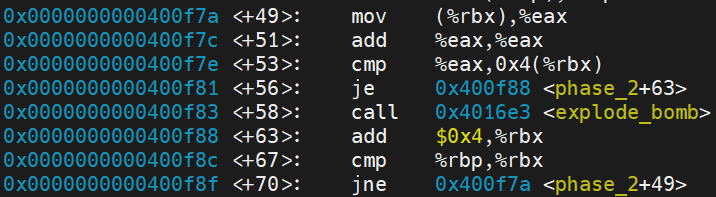
3.由<phase\_2+25>处的代码，发现该关卡应输入六个数字



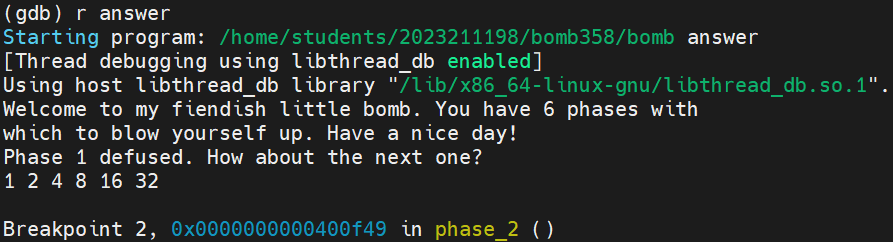
4. 由<phase\_2+30>到<phase\_2+36>处的代码，发现第一个数字应该等于1



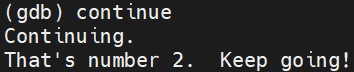
5. 由<phase\_2+49>到<phase\_2+70>处的代码，发现后面每一个数字是前一个的2倍，答案就是1 2 4 8 16 32



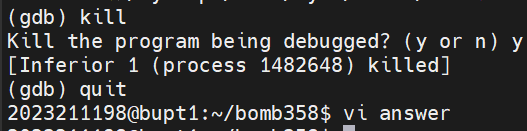
6. 键入命令r answer运行程序，第一题自动解答，输入1 2 4 8 16 32



7.使用continue，观察到炸弹未爆炸，phase2成功拆除



8.kill退出进程，quit退出gdb，打开answer存储phase\_2答案

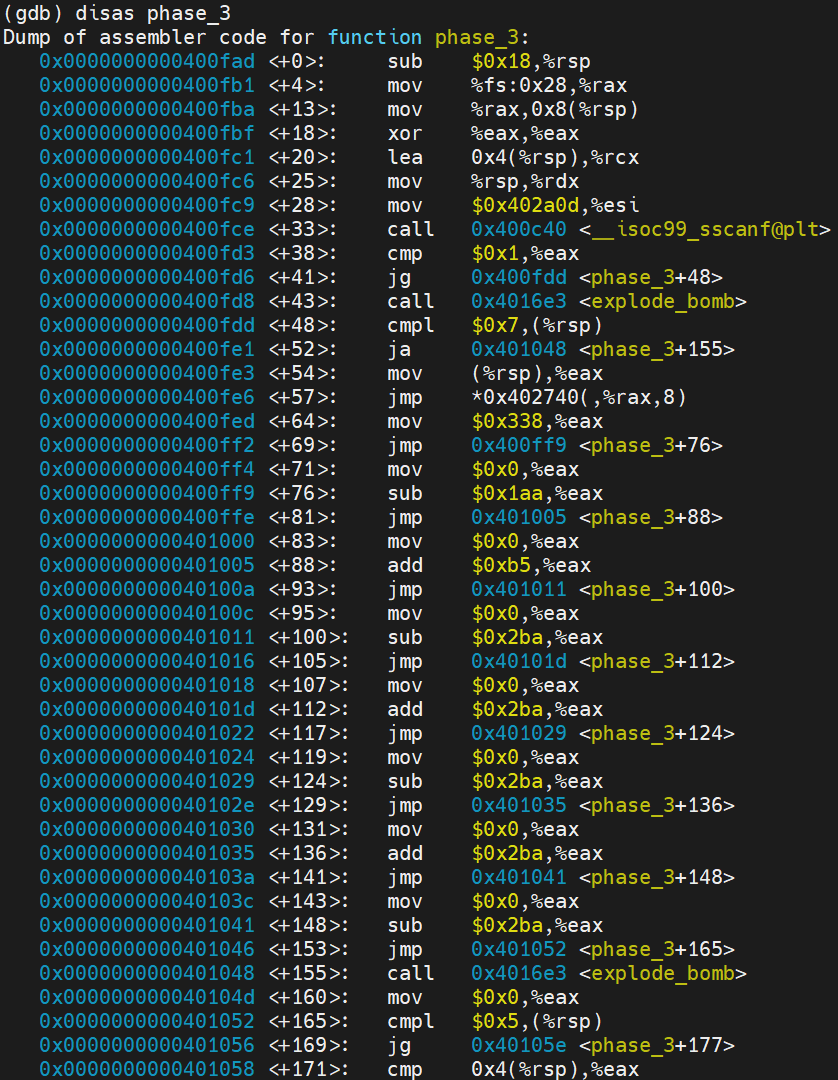


阶段四 拆解phase\_3

1.输入b phase\_3，在phase\_3处设置断点



2. 键入命令disas phase\_3,观察phase\_3的代码



3.由<phase\_3+33>处的代码，发现该关卡应输入两个数字



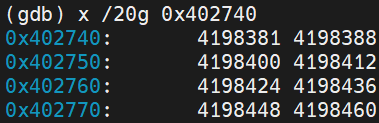
4.由<phase\_3+48>到<phase\_3+52>处的代码，发现第一个数字应小于等于7



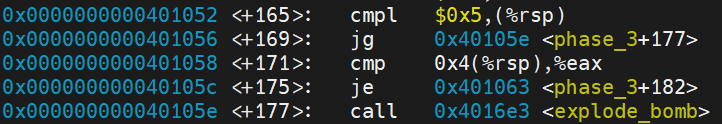
5.由<phase\_3+57>处的代码，发现一个switch语句，跳转位置与第一个输入有关



6.查看switch各分支的位置

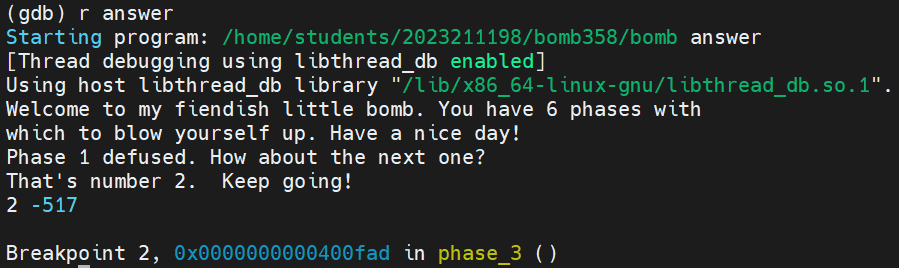


7.由<phase\_3+165>和<phase\_3+177>处的代码，发现第一个数字应小于等于7

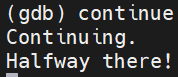


8.举例当第一个输入为2时，第二个输入的计算为0+181-698+698-698+698-698=-517

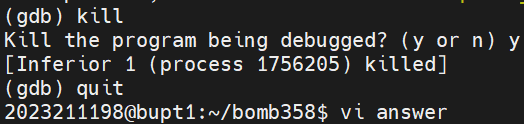
9. 键入命令r answer运行程序，前两题自动解答，输入2 -517



10.使用continue，观察到炸弹未爆炸，phase3成功拆除



11.kill退出进程，quit退出gdb，打开answer存储phase\_3答案

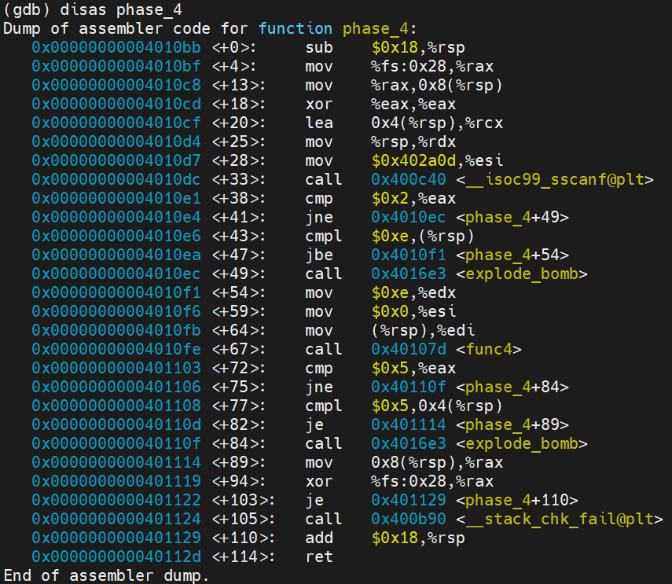


阶段五 拆解phase\_4

1.输入b phase\_4，在phase\_4处设置断点



2. 键入命令disas phase\_4,观察phase\_4的代码



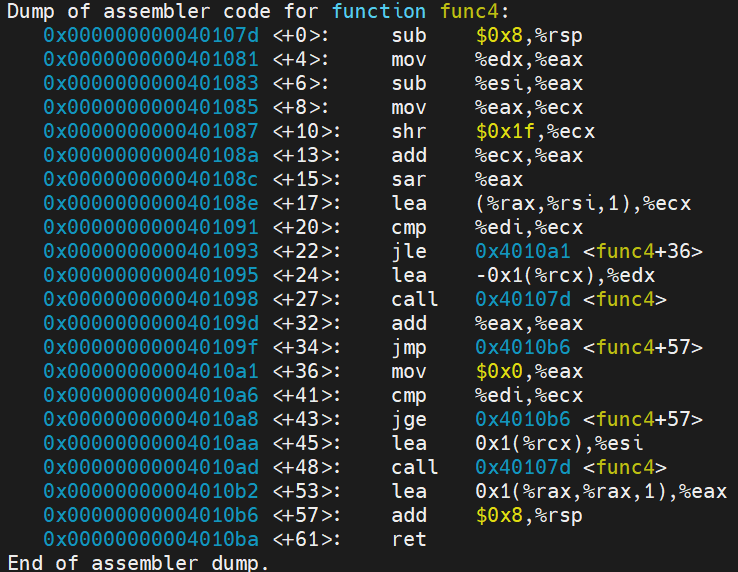
3.由<phase\_4+33>处的代码，发现该关卡应输入两个数字



4. 由<phase\_4+67>处的代码，发现调用func4函数



5. 键入命令disas func4,观察func4的代码

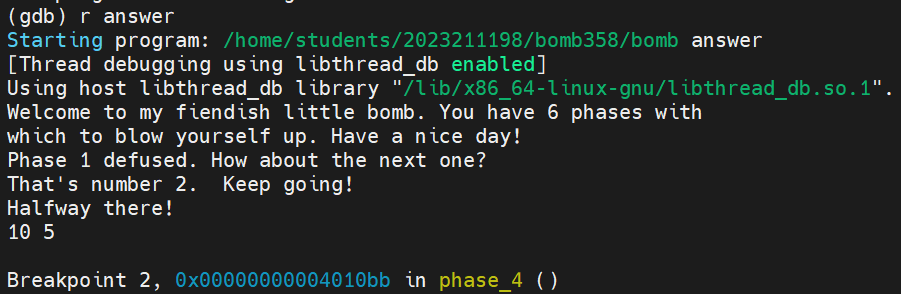


6.由<func4+27><func4+48>处的代码，发现调用func4函数，确定为递归调用

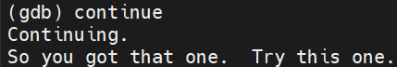


7.根据计算得出答案为10 5

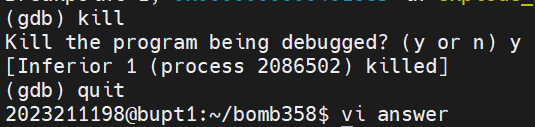
8. 键入命令r answer运行程序，前三题自动解答，输入10 5



9.使用continue，观察到炸弹未爆炸，phase4成功拆除



10.kill退出进程，quit退出gdb，打开answer存储phase\_4答案

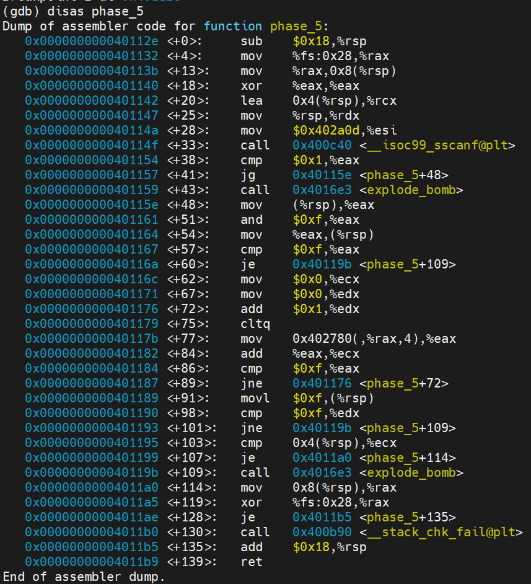


阶段六 拆解phase\_5

1.输入b phase\_5，在phase\_5处设置断点



2. 键入命令disas phase\_5,观察phase\_5的代码



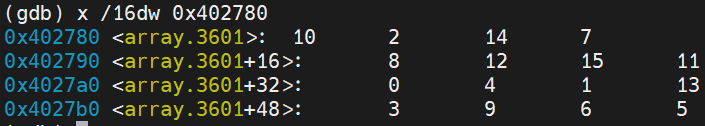
3.由<phase\_5+33>处的代码，发现该关卡应输入两个数字



4.由<phase\_5+77>得使用数组进行循环运算



5.使用x /16dw 0x402780指令查看数组数据

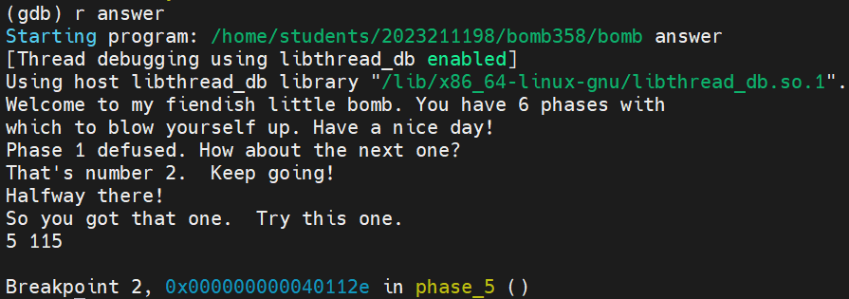


转化为c语言：

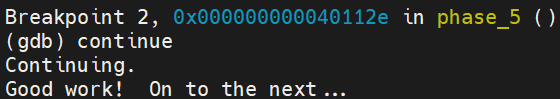
int array\_3601[16] = { 10, 2, 14, 7, 8, 12, 15, 11, 0, 4, 1, 13, 3, 9, 6, 5 };

6.由此推出结果为5 115

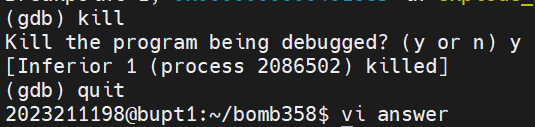
7. 键入命令r answer运行程序，前四题自动解答，输入5 115



9.使用continue，观察到炸弹未爆炸，phase5成功拆除



10.kill退出进程，quit退出gdb，打开answer存储phase\_5答案

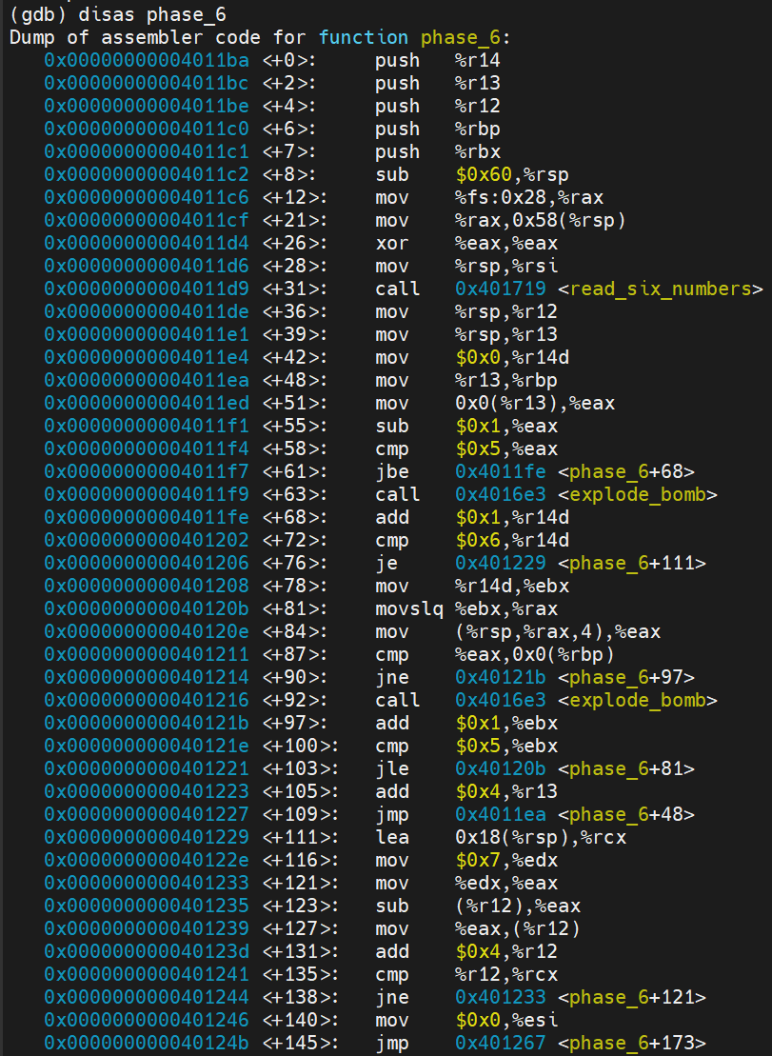


阶段七 拆解phase\_6

1.输入b phase\_6，在phase\_6处设置断点



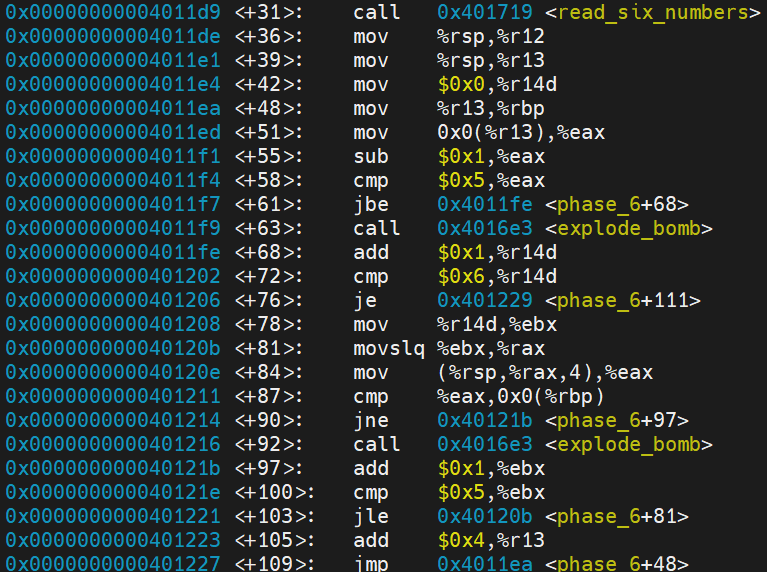
2. 键入命令disas phase\_6,观察phase\_6的代码



3.由<phase\_6+31>处的代码，发现该关卡应输入六个数字



4.由<phase\_6+31>和<phase\_6+109>处的代码可知，输入的六个数-1后都不大于五，即六个数都不大与6，且各数都不相同

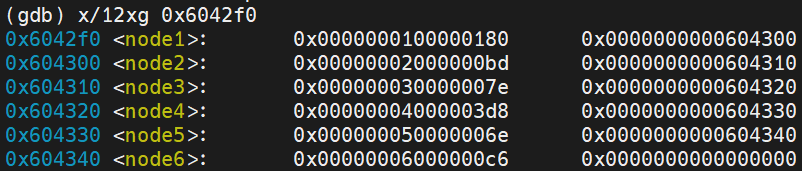


所以输入为1到6的组合

5.继续向下查看，由<phase\_6+181>处的代码，发现设计到对链表的操作

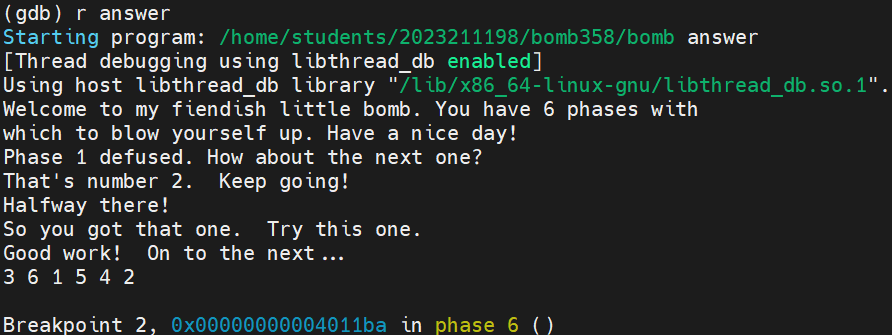


6.使用x /12xg 0x6042f0指令查看数组数据

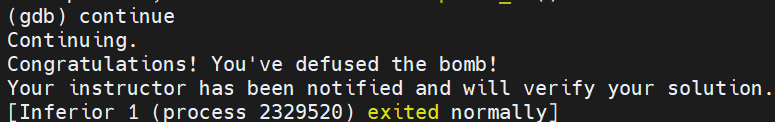


7.分析代码得出在对链表进行排序后倒序输出，计算的结果为3 6 1 5 4 2

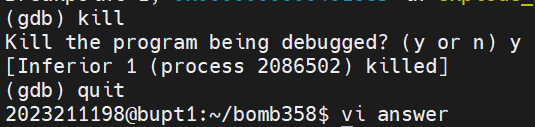
8. 键入命令r answer运行程序，前五题自动解答，输入3 6 1 5 4 2



9.使用continue，观察到炸弹未爆炸，phase6成功拆除



10.kill退出进程，quit退出gdb，打开answer存储phase\_6答案

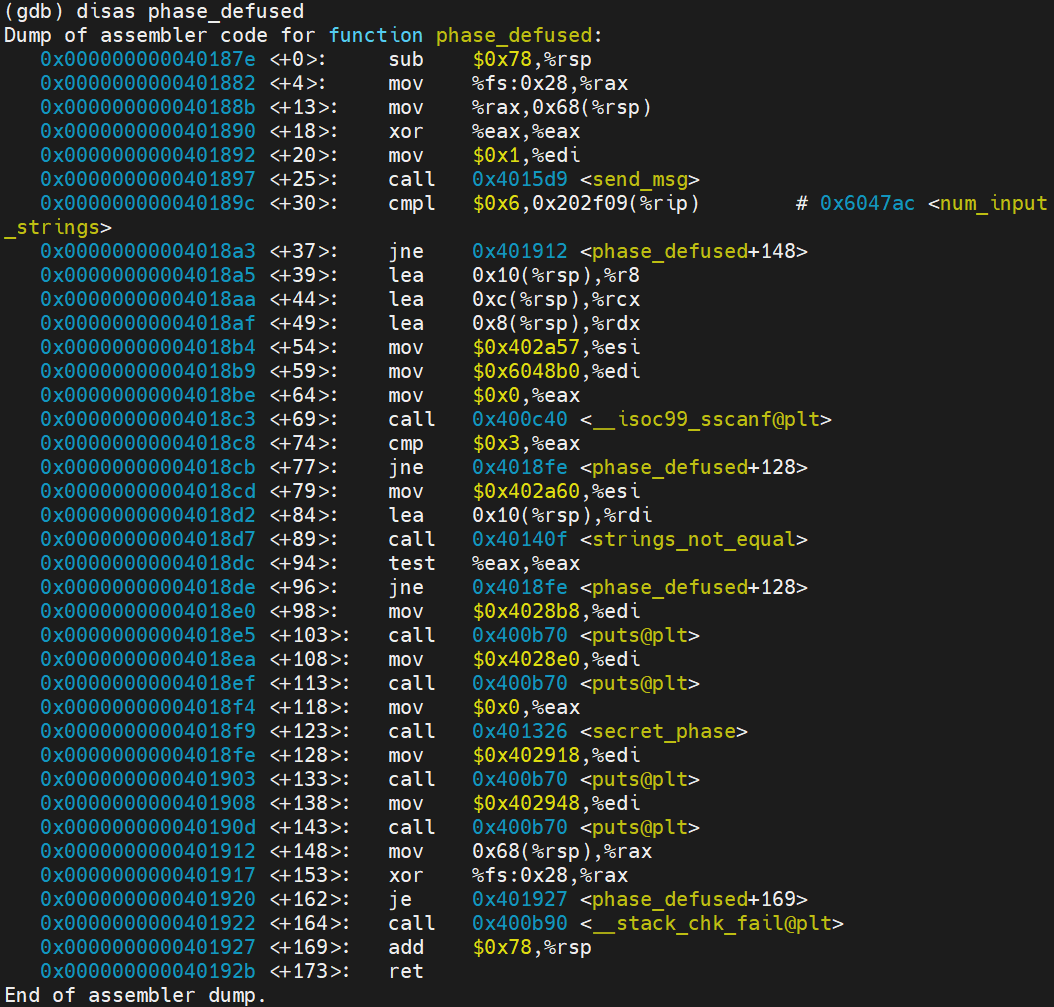


阶段八 拆解secret\_phase

1.输入b secret\_phase，在secret\_phase处设置断点



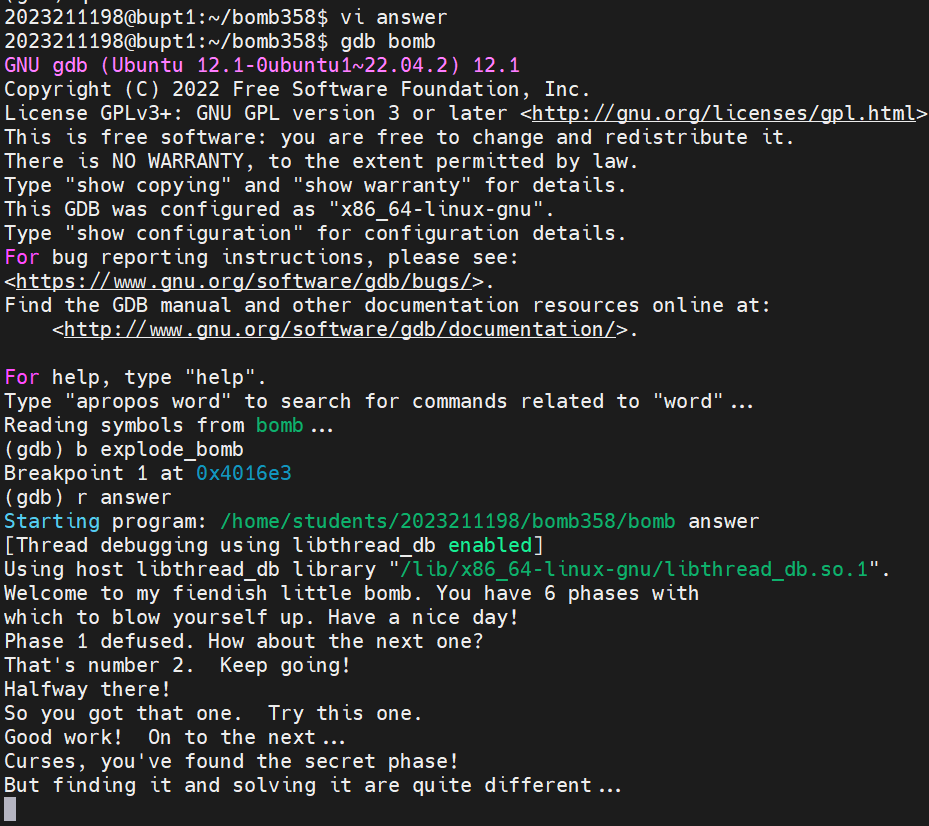
2. 因为阶段一中分析出secret\_phase的入口在phase\_defused中，所以键入disas phase\_defused,获得phase\_defused函数的汇编代码



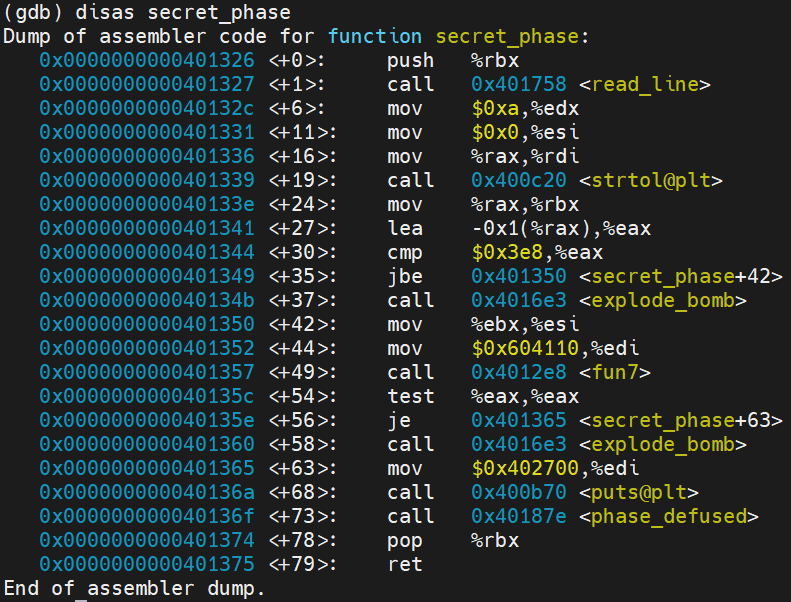
3.发现需要输入字符串与0x402a60作比较，使用x/s0x402a60查看字符串，为"DrEvil"



4. 发现输入的数据存储在0x6048b0中，此处为第四题的输入存储区，故可知，要进入隐藏关卡，需在第四题的答案后加一个字符串 DrEvil,所以在answer文件内第四题答案后添加DrEvil,r answer运行程序，进入隐藏关卡



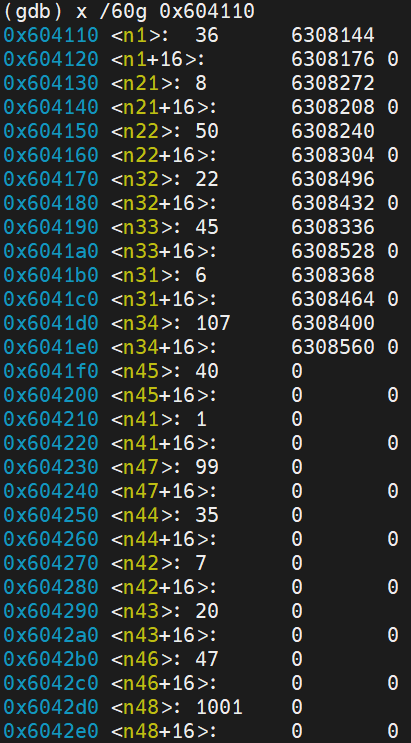
5.键入命令disas secret\_phase,观察secret\_phase的代码



6.由<secret\_phase+44>可知，输入与二叉树做比较



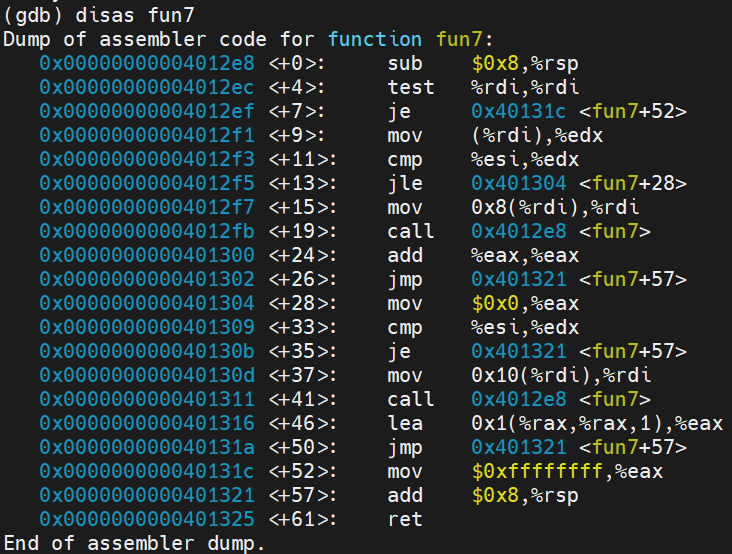
7. 使用x /60g 0x604110指令查看相应区域存储，确认为二叉树



7.由<secret\_phase+49>可知，调用了fun7函数

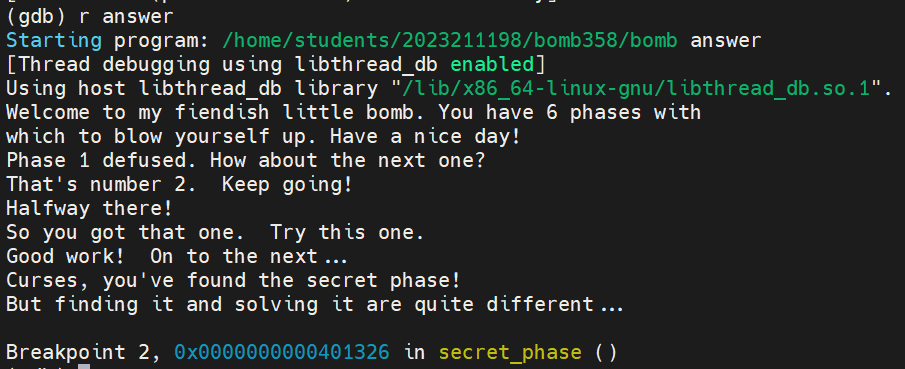


8. 键入命令fun7,观察fun7的代码

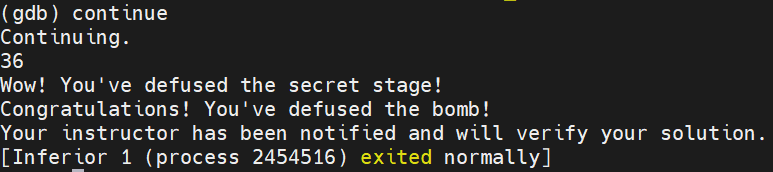


9.分析代码可知发生递归调用，功能为在二叉树中查找值，推算得结果为36

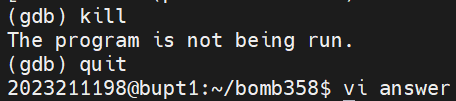
10. 键入命令r answer运行程序，前六题自动解答，到secret\_phase断点



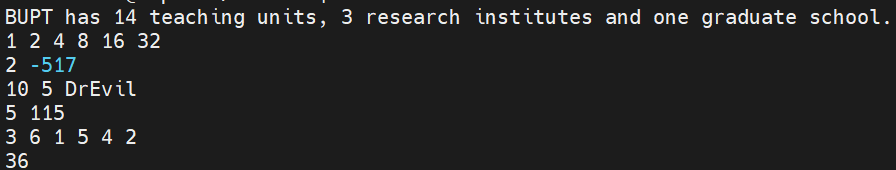
9.使用continue，输入36，观察到炸弹未爆炸，secret\_phase成功拆除



10.kill退出进程，quit退出gdb，打开answer存储secret\_phase答案



最终答案：



五、总结体会

总结心得（包括实验过程中遇到的问题、如何解决的、过关或挫败的感受、实验投入的时间和精力、意见和建议等）

1.不会将二进制文件转为汇编文件；通过上网查询学会。

2.不熟悉汇编代码，不会将汇编代码抽象出逻辑；通过上网查询汇编代码教程学会。

3.不会正确进行寄存器查询；通过指令查询学会

4.不会正确设置断点，无法达到预期效果；

六、诚信声明（不签扣10分）

需要填写如下声明，并在底部给出手写签名的电子版。

在完成本次实验过程中，我曾分别与以下各位同学就以下方面做过交流：

1、简单描述交流内容，例如：来自\*\*\*的建议，采用\*\*\*方式\*\*\*（无）

2、

此外，我还参考了以下资料：

1. CSDN(<https://blog.csdn.net/>)

(1) 【Linux C】GCC编译 && GDB调试 从入门到放弃 （gcc调试选项详解、gdb调试、条件断点、远程调试、脚本化调试）(<https://blog.csdn.net/weixin_43764974/article/details/130787032>)

(2) 汇编基础-----mov与常见寻址方式(<https://blog.csdn.net/a1309525802/article/details/137461445>)

用于研究汇编指令

(3) objdump命令详解 (<https://blog.csdn.net/qq_41683305/article/details/105375214>)

了解objdump生成汇编代码

1. Bilibili(<https://www.bilibili.com>)

速学150个Linux常用命令(<https://www.bilibili.com/video/BV12L411a7Ne/?spm_id_from=333.337.search-card.all.click>)

用于学习Linux常用指令



在我提交的程序中，还在对应的位置以注释形式记录了具体的参考内容。

我独立完成了本次实验除以上方面之外的所有工作，包括分析、设计、编码、调试与测试。

我清楚地知道，从以上方面获得的信息在一定程度上降低了实验的难度，可能影响起评分。

我从未使用他人代码，不管是原封不动地复制，还是经过某些等价转换。

我未曾也不会向同一课程（包括此后各届）的同学复制或公开我这份程序的代码，我有义务妥善保管好它们。

我编写这个程序无意于破坏或妨碍任何计算机系统的正常运行。

我清楚地知道，以上情况均为本课程纪律所禁止，若违反，对应的实验成绩将按照0分计。

（签名）