

**实验报告**



**题目： 拆解二进制炸弹**

**班 级： 2023211321**

**学 号： 2023212872**

**姓 名： 计子毅**

**学 院： 计算机学院**

**2023 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。

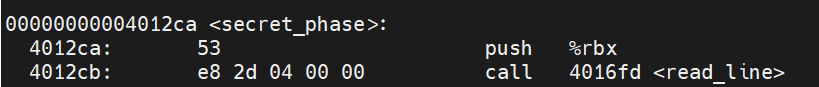
本实验通过要求使用课程所学知识拆除一个“binary bombs”来增强对程序的机器级表示、汇编语言、调试器和逆向工程等方面原理与技能的掌握。 “binary bombs”是一个Linux可执行程序，包含了5个阶段（或关卡）。炸弹运行的每个阶段要求你输入一个特定字符串，你的输入符合程序预期的输入，该阶段的炸弹就被拆除引信；否则炸弹“爆炸”，打印输出 “BOOM!!!”。炸弹的每个阶段考察了机器级程序语言的一个不同方面，难度逐级递增。

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

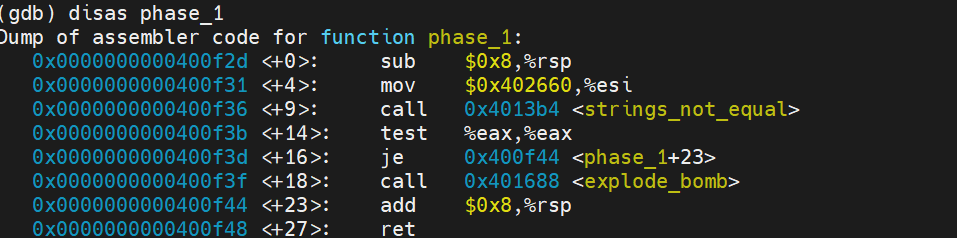
**准备工作：**解压bomb541.rar压缩包，得到bomb541文件夹，包含bomb.c,bomb,README三个文件，编辑一个save文件用来保存已经通过的答案。

**阶段1：**用objdump -d bomb查看整个程序的汇编代码，发现有phase\_1, phase\_2,phase\_3 ,phase\_4 ,phase\_5, phase\_6共6个炸弹，并且在phase\_defused函数中发现了secre\_phase这个隐藏炸弹，且在前6个炸弹被拆除后自动调用这两个函数。

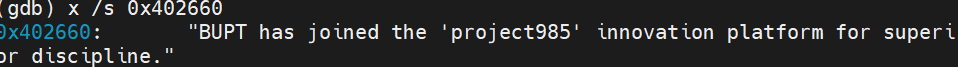


**阶段2：**拆解phase\_1

使用gdb bomb进入gdb调试模式，并使用disas phase\_1查看炸弹1的汇编代码



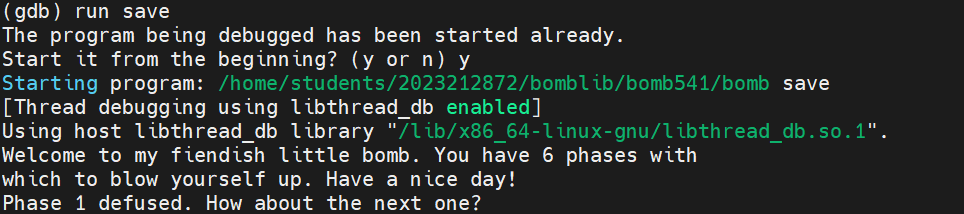
发现这个函数的作用是判断我们输入的字符串是否与要求的字符串相等，如果不相等则炸弹爆炸，我们可以看到<+4>行存在一个地址，我们用x /s 0x402660来查看该地址中存放的值



发现是一串字符串，则可以知道该字符串即我们需要输入的字符串。

我们将程序run运行，输入这串字符串，发现正确，记录到文件save中，并重新进入gdb模式，并run save，可以自动输入save中的答案，正确，则第一个炸弹通过。





**阶段3：**拆解phase\_2

先用disas phase\_2查看炸弹2汇编代码，由下面这行代码猜测我们要输入6个数



用disas 查看这个函数的汇编代码，并发现地址0x402971，查看该地址内容，发现确实是需要我们输入6个整数

图形用户界面, 文本

描述已自动生成

由下面这两行代码我们知道输入的第一个数是1



文本

描述已自动生成

由这里我们可以知道每一次将前一个数乘2，为一个等差数列，则根据第一个数是1，则输入的6个数为，1 2 4 8 16 32，输入发现正确，记录到save中。

**阶段4：**拆解phase\_3

用disas查看phase\_3的汇编代码





由该地址中内容可以知道我们需要输入两个整数



由这两行可以知道输入的第一个数必须小于7

图形用户界面, 文本, 应用程序

描述已自动生成

观察这一部分的代码，发现由8次向<phase\_3+130>的跳转，以及<57>的地址可以知道主要依赖于switch函数，并每次输入的数对应每一次跳转，用x /12xg 0x4026e0查看switch的分支

图形用户界面, 文本

描述已自动生成

则从上到下，从左到右分别对应0-7的跳转地址，如果输入的第二个数与跳转地址的数相等则成功拆除炸弹

则输入0，跳转到0x40102a，对应的数是0xe2，是十进制207，则输入的两个数可以是0 207（其余答案同理）。

**阶段5：**拆解phase\_4

文本

描述已自动生成



可知我们需要输入两个整数，且由<+43>行可以知道输入的第一个数不能大于14.



观察到调用了一个函数func4，使用disas查看func4的汇编代码

图形用户界面, 文本

描述已自动生成

可以看出，这是一个递归流程，并且可以由汇编代码写出对应的c语言代码

int fun(int edi, int esi, int edx)

{

int eax = edx;

eax -= esi;

int ecx = eax;

ecx = 0;

eax += ecx;

eax /= 2;

ecx = eax + esi;

if (ecx > edi)

{

eax = fun(edi, esi, ecx - 1);

eax \*= 2;

}

else

{

eax = 0;

}

if (ecx < edi)

{

eax = fun(edi, ecx + 1, edx);

eax = eax \* 2 + 1;

}

return eax;

}

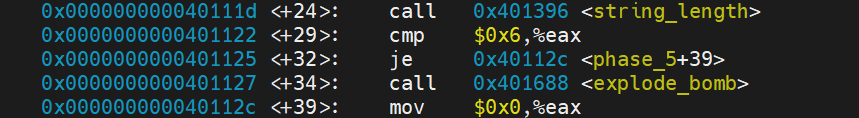
则由phase\_4汇编代码可以知道，edi=第一个输入数，esi=0，edx=14

文本

中度可信度描述已自动生成

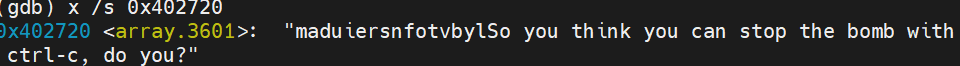
由此可知，函数的返回值要等于5，且输入的第二个数也等于5，通过调试上述代码，发现当输入的第一个数为10时，返回值为5,则输入的两个数为10 5

**阶段6：**拆解phase\_5



可以知道需要输入长度为6的字符串





查看该地址得到一段字符串，(%rdx)在这里起到了索引的作用，比如(%rdx)=0x1，就是将a字符传给%edx

文本

描述已自动生成

循环6次，并每次取出输入字符的2进制表示的低4位





查看该地址中的内容发现字符串，则最终要判断输入的字符串对应取出的字符串是否相等。

接下来将oilers对应到长字符串的位置中，分别是11，5，15，6，7，8位，对应二进制数是1011，0101，1111，0110，0111，1000，则需输入后4位是对应这串排序的字符，输入答案jdouvw，正确。

**阶段7：**拆解phase\_6



由此可知，我们需要输入6个数字

图形用户界面, 文本

描述已自动生成

其中，r13d用来当作数组索引，相当于i=0，实现两层循环使输入的数字不大于6，且互不相同



使用x /24dw 0x6042f0查看地址中的内容

图片包含 图表

描述已自动生成

是一个结构体，内容是数据，节点位数，指向下一个节点的指针

电脑屏幕的照片

中度可信度描述已自动生成

这一部分是将原链表的数据放到地址为0x20的新链表中

图形用户界面, 文本, 应用程序

描述已自动生成

将新链表中的结构体按照数据进行升序排序，并检查输入的6个数字，对应的节点数是否满足升序排序，再根据之前查看的结构体各个节点的保存的内容，得到升序排序后的节点数是4 1 2 5 6 3。

**阶段8：**拆解隐藏炸弹

用disas查看phase\_defused的汇编代码

电脑屏幕的照片

中度可信度描述已自动生成



发现了我们需要在某个炸弹中输入一个“DrEvil”字符串，使得密钥正确，才能进入隐藏关卡

文本

描述已自动生成

并且查看这个地址后发现，输入的格式为两个数字，再加上这个字符串，可以判断处罚隐藏关卡的输入在炸弹4，我们在炸弹4的答案中加入“DrEvil”字符串，系统进入了secret\_phase函数

文本

描述已自动生成

随后查看secret\_phase的汇编代码

图形用户界面, 文本, 应用程序

描述已自动生成

调用strtol函数，将字符串转换为整型数据num，存在%rax中，num-1>1000(0x3e8)，则会爆炸，所以输入的数字必须小于等于1001，查看604110中的内容

文本

低可信度描述已自动生成

发现是一个二叉搜索树，往左子树走地址加0x8，往右子树走地址加0x10，并且看到调用了fun7函数，且传入的参数是该树状结构，我们查看该函数的汇编代码

文本

描述已自动生成

结合secret-phase的汇编代码，要求fun7返回的值要等于3

且fun7对应c语言代码为：

int func7(Type \*p, int input)

{

if(p == NULL)

return -1;

if(&p <= input)

{

if(&p == input)

return 0;

else

{

p = p + 0x10;

int n = func7(p, input);

return 2 \* n + 1;

}

}

else

{

p = p + 0x8;

int n = func7(p, input);

return 2 \* n;

}

}

则判断，我们需要从根节点36，连续往右走2次，地址加了32，则取出数据107，则输入的数字应为107

五、总结体会

遇见的最大问题即为对汇编代码的不熟悉，但从第一个炸弹开始以来，不断地积累汇编代码的阅读技巧，虽然使得炸弹爆炸了8次，但也从其中学会了关于汇编代码的许多。在炸弹的拆解过程中，每一次引爆都使长时间理解汇编代码的努力化为泡影，心情不免产生抵触情绪，但最终是坚持了下来，让我坚持下来的不只是炸弹拆解的有趣性，还有成功拆解和学习到新知识的成就感。在其中一次引爆中，我自以为的认为switch函数的穿透顺序是按照汇编代码的顺序排列的，后来根据switch函数的汇编代码仔细分析才发现地址跳转与语句的储存顺序没有联系；还有在隐藏关卡中炸弹爆炸了2次，主要是对树状结构地址寻址的不认识与不熟练，经过对该数据结构的学习后成功的拆解了炸弹。

该实验我投入了3天的时间进行，花费了大量的精力，但我认为这都是值得的，从这次实验中我真的学到了很多关于汇编代码的知识。

我建议在接下来的实验中可以增加更多函数的汇编代码，以便于对各种基础函数的汇编代码进行熟悉。

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、与张泽珅同学讨论过隐藏关卡的触发条件

2、

此外，我还参考了以下资料：

1. csdn中有关汇编代码关键字作用的文章
2. ASCII码表

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）图片包含 图示

描述已自动生成