

# 哈尔滨工业大学

# 实验报告

## 实验（三）

题 目 Binary Bomb

二进制炸弹

专 业 计算机专业

学 号 1190200501

班 级 1903002

学 生 林燕燕

指 导 教 师 郑贵滨

实 验 地 点 G709

实 验 日 期 2021.04.23

## 计算机科学与技术学院

# 目 录

<b>第 1 章 实验基本信息</b> .....	<b>- 3 -</b>
1.1 实验目的.....	- 3 -
1.2 实验环境与工具.....	- 3 -
1.2.1 硬件环境.....	- 3 -
1.2.2 软件环境.....	- 3 -
1.2.3 开发工具.....	- 3 -
1.3 实验预习.....	- 3 -
<b>第 2 章 实验环境建立</b> .....	<b>- 4 -</b>
2.1 UBUNTU 下 CODEBLOCKS 反汇编（10 分） .....	- 4 -
2.2 UBUNTU 下 EDB 运行环境建立（10 分） .....	- 4 -
<b>第 3 章 各阶段炸弹破解与分析</b> .....	<b>- 5 -</b>
3.1 阶段 1 的破解与分析.....	- 5 -
3.2 阶段 2 的破解与分析.....	- 6 -
3.3 阶段 3 的破解与分析.....	- 7 -
3.4 阶段 4 的破解与分析.....	- 8 -
3.5 阶段 5 的破解与分析.....	- 9 -
3.6 阶段 6 的破解与分析.....	- 9 -
3.7 阶段 7 的破解与分析(隐藏阶段).....	- 9 -
<b>第 4 章 总结</b> .....	<b>- 10 -</b>
4.1 请总结本次实验的收获.....	- 10 -
4.2 请给出对本次实验内容的建议.....	- 10 -
<b>参考文献</b> .....	<b>- 11 -</b>

## 第 1 章 实验基本信息

### 1.1 实验目的

熟练掌握计算机系统的 ISA 指令系统与寻址方式

熟练掌握 Linux 下调试器的反汇编调试跟踪分析机器语言的方法

增强对程序机器级表示、汇编语言、调试器和逆向工程等的理解

### 1.2 实验环境与工具

#### 1.2.1 硬件环境

X64 CPU; 1.6GHz; 8G RAM; 256G SSD Disk; 1T HDD Disk

#### 1.2.2 软件环境

Windows10 64 位; Vmware 14pro; Ubuntu 20.04.2 LTS 64 位

#### 1.2.3 开发工具

Visual Studio Code 64 位; vim/gpedit+gcc

### 1.3 实验预习

- 请写出 C 语言下包含字符串比较、循环、分支（含 switch）、函数调用、递归、指针、结构、链表等的例子程序 sample.c。
- 生成执行程序 sample.out。
- 用 gcc -S 或 CodeBlocks 或 GDB 或 OBJDUMP 等，反汇编，比较。
- 列出每一部分的 C 语言对应的汇编语言。
- 修改编译选项-O (缺省 2)、O0、O1、O2、O3，-m32/m64。再次查看生成的汇编语言与原来的区别。
- 注意 O1 之后无栈帧，EBP 做别的用途。-fno-omit-frame-pointer 加上栈指针。
- GDB 命令详解 - tui 模式 ^XA 切换 layout 改变等等
- 有目的地学习：看 VS 的功能 GDB 命令用什么？

## 第 2 章 实验环境建立

### 2.1 Ubuntu 下 CodeBlocks 反汇编 (10 分)

CodeBlocks 运行 hellolinux.c。反汇编查看 printf 函数的实现。

要求：C、ASM、内存(显示 hello 等内容)、堆栈 (call printf 前)、寄存器同时在一个窗口。

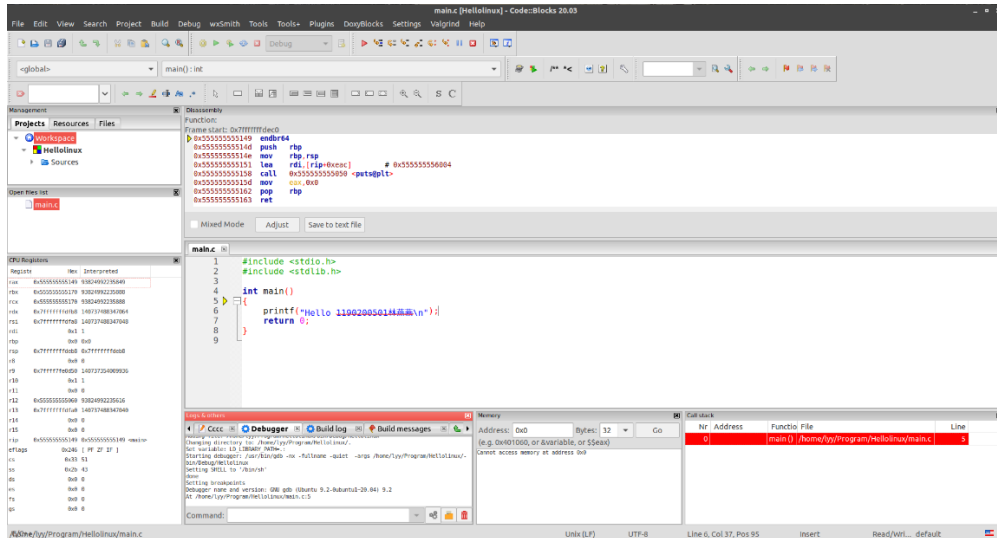


图 2-1 Ubuntu 下 CodeBlocks 反汇编截图

### 2.2 Ubuntu 下 EDB 运行环境建立 (10 分)

用 EDB 调试 hellolinux.c 的执行文件，截图，要求同 2.1

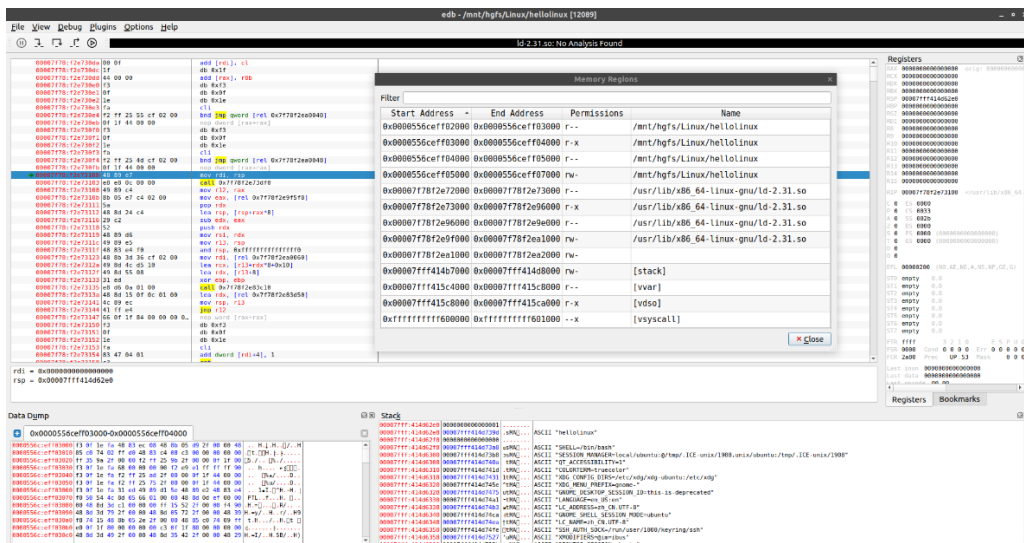


图 2-2 Ubuntu 下 EDB 截图

## 第 3 章 各阶段炸弹破解与分析

每阶段 15 分（密码 10 分，分析 5 分），总分不超过 80 分

### 3.1 阶段 1 的破解与分析

密码如下：Public speaking is very easy.

破解过程：

对 bomb 进行反汇编得到汇编代码，查看代码在 main 函数中找到 phase\_1 函数位置：

```

962    4012fb: e8 83 06 00 00    callq 401983 <read_line>
963    401300: 48 89 c7          mov    %rax,%rdi
964    401303: e8 f1 00 00 00    callq 4013f9 <phase_1>
965    401308: e8 a1 07 00 00    callq 401aae <phase_defused>
966    40130d: bf f8 30 40 00    mov    $0x4030f8,%edi
967    401312: e8 49 fd ff ff    callq 401060 <puts@plt>

```

继续查找 phase\_1 的位置：

```

1019  00000000004013f9 <phase_1>:
1020    4013f9: 55              push   %rbp
1021    4013fa: 48 89 e5        mov    %rsp,%rbp
1022    4013fd: be 4c 31 40 00  mov    $0x40314c,%esi
1023    401402: e8 22 04 00 00  callq 401829 <strings_not_equal>
1024    401407: 85 c0          test   %eax,%eax
1025    401409: 75 02          jne    40140d <phase_1+0x14>
1026    40140b: 5d            pop    %rbp
1027    40140c: c3            retq
1028    40140d: e8 13 05 00 00  callq 401925 <explode_bomb>
1029    401412: eb f7          jmp    40140b <phase_1+0x12>

```

从键盘输入或文件读取的字符串存入 rdi，esi 存放用于比对的字符串，地址在 0x40314c：

```

1022    4013fd: be 4c 31 40 00    mov    $0x40314c,%esi

```

```

2591 403149: 65 2e 00 50 75    gs add %dl,%cs:0x75(%rax)
2592 40314e: 62                (bad)
2593 40314f: 6c                insb    (%dx),%es:(%rdi)
2594 403150: 69 63 20 73 70 65 61 imul    $0x61657073,0x20(%rbx),%esp
2595 403157: 6b 69 6e 67       imul    $0x67,0x6e(%rcx),%ebp
2596 40315b: 20 69 73          and     %ch,0x73(%rcx)
2597 40315e: 20 76 65          and     %dh,0x65(%rsi)
2598 403161: 72 79             jb      4031dc <array.3403+0x1c>
2599 403163: 20 65 61          and     %ah,0x61(%rbp)
2600 403166: 73 79             jae     4031e1 <array.3403+0x21>
2601 403168: 2e 00 00          add     %al,%cs:(%rax)

```

由 16 进制机器码可得 esi 存放字符串十六进制数为 50 75 62 6c 69 63 20 73 70 65 61 6b 69 6e 67 20 69 73 20 76 65 72 79 20 65 61 73 79 2e 00 00, 转换为 ASCII 码得到 Public speaking is very easy. , 即为密码。

### 3.2 阶段 2 的破解与分析

密码如下: 1 2 4 8 16 32

破解过程:

phase2 中要求输入 6 个数, 输入采用循环, 且存在 `addl %eax,%eax` 表示后一个数为前一个数的两倍, 又有 `cmpl $1,-0x30(%rbp)` 表示第一个数为 1, 则密码为: 1 2 4 8 16 32



### 3.4 阶段 4 的破解与分析

密码如下：4 2 或 5 2 (两组密码)

破解过程：

00000000:00401551	55	pushq %rbp	
00000000:00401552	48 89 e5	movq %rsp, %rbp	
00000000:00401555	48 83 ec 10	subq \$0x10, %rsp	
00000000:00401559	48 8d 4d f8	leaq -8(%rbp), %rcx	x <sub>2</sub>
00000000:0040155d	48 8d 55 fc	leaq -4(%rbp), %rdx	x <sub>1</sub>
00000000:00401561	be 37 33 40 00	movl \$0x403337, %esi	ASCII "%d %d"
00000000:00401566	b8 00 00 00 00	movl \$0, %eax	
00000000:0040156b	e8 a0 fb ff ff	callq bomb! isoc99_sscanf@plt	
00000000:00401570	83 f8 02	cmpl \$2, %eax	eax=2 (输入2个数)
00000000:00401573	75 0c	jne 0x401581	
00000000:00401575	8b 45 fc	movl -4(%rbp), %eax	eax=x <sub>1</sub>
00000000:00401578	85 c0	testl %eax, %eax	
00000000:0040157a	78 05	js 0x401581	x <sub>1</sub> <0
00000000:0040157c	83 f8 0e	cmpl \$0xe, %eax	
00000000:0040157f	7e 05	jle 0x401586	x <sub>1</sub> ≤14
00000000:00401581	e8 9f 03 00 00	callq bomb!explode_bomb	
00000000:00401586	ba 0e 00 00 00	movl \$0xe, %edx	edx=14
00000000:0040158b	be 00 00 00 00	movl \$0, %esi	esi=0
00000000:00401590	8b 7d fc	movl -4(%rbp), %edi	edi=x <sub>1</sub>
00000000:00401593	e8 7f ff ff ff	callq bomb!func4	
00000000:00401598	83 f8 02	cmpl \$2, %eax	eax=2
00000000:0040159b	75 06	jne 0x4015a3	
00000000:0040159d	83 7d f8 02	cmpl \$2, -8(%rbp)	
00000000:004015a1	74 05	jbe 0x4015a9	x <sub>2</sub> ≥2
00000000:004015a3	e8 7d 03 00 00	callq bomb!explode_bomb	
00000000:004015a8	c9	leave	
00000000:004015a9	c3	retq	

phase4 中输入两个数 x<sub>1</sub> x<sub>2</sub>, x<sub>1</sub> 要大于 0 且小于等于 14

接着对 edx esi edi 赋值, 传 func4, 而 func4 的返回值要是 2

func 4

edx=14, esi=0, edi=x<sub>1</sub>

00000000:00401517	55	pushq %rbp	
00000000:00401518	48 89 e5	movq %rsp, %rbp	
00000000:0040151b	89 d1	movl %edx, %ecx	ecx=edx
00000000:0040151d	29 f1	subl %esi, %ecx	ecx=edx-esi
00000000:0040151f	89 c8	movl %ecx, %eax	eax=ecx=edx-esi
00000000:00401521	c1 e8 1f	shrl \$0x1f, %eax	eax>>31
00000000:00401524	01 c8	addl %ecx, %eax	eax=edx-esi+(edx-esi)>>31
00000000:00401526	d1 f8	sarll \$1, %eax	eax=(edx-esi+(edx-esi)>>31)/2
00000000:00401528	01 f0	addl %esi, %eax	eax=edx-esi+(edx-esi)>>31+esi
00000000:0040152a	39 f8	cmpl %edi, %eax	(edx-esi+((edx-esi)>>31)/2+esi-x <sub>1</sub> )≥0
00000000:0040152c	7f 09	jg 0x401537	(edx-esi)+(edx-esi)>>31≤(x <sub>1</sub> -esi)x <sub>2</sub>
00000000:0040152e	7c 13	jle 0x401543	eax<0 bomb
00000000:00401530	b8 00 00 00 00	movl \$0, %eax	
00000000:00401535	5d	popq %rbp	
00000000:00401536	c3	retq	
00000000:00401537	8d 50 ff	leal -1(%rax), %edx	
00000000:0040153a	e8 d8 ff ff ff	callq bomb!func4	
00000000:0040153f	01 c0	addl %eax, %eax	
00000000:00401541	eb f2	jmp 0x401535	
00000000:00401543	8d 70 01	leal 1(%rax), %esi	
00000000:00401546	e8 cc ff ff ff	callq bomb!func4	
00000000:0040154b	8d 44 00 01	leal 1(%rax, %rax), %eax	
00000000:0040154f	eb e4	jmp 0x401535	

进入 func4, 计算得到  $eax = ((edx - esi + (((unsigned)(edx - esi)) >> 31)) >> 1) + esi$ ,

将 eax 与 edi 即 x<sub>1</sub> 进行比较, 按情况赋值进行递归, 测试代码如下

```
#include <stdio.h>
int x1;
int eax;
int func4(int edx, int esi){
    eax = ((edx - esi + (((unsigned)(edx - esi)) >> 31)) >> 1) + esi;
    int text = eax - x1;
    if(text > 0){
        edx = eax - 1;
        func4(edx, esi);
        eax = eax * 2;
    }
    else if(text < 0){
        esi = eax + 1;
        func4(edx, esi);
        eax = 1 + eax * 2;
    }
}
```



```

else{
    eax = 0;
}
return 0;
}
int main(){
    for(x1 = 1; x1 <= 14; x1++) {
        func4(14, 0);
        if(eax == 2){
            printf("x1 = %d\n", x1);
        }
    }
    return 0;
}

```

x1 = 4  
x1 = 5

```

cmpl $2, -8(%rbp)
je 0x4015a8, x2 > 2
callq bomb__explode_bomb
leave
retq

```

返回 eax 需为 2, 计算结果为 x1 = 4 或 5, 又即 x2 = 2, 则密码为 4 2, 5 2 两组。

### 3.5 阶段 5 的破解与分析

密码如下:

破解过程:

### 3.6 阶段 6 的破解与分析

密码如下:

破解过程:

### 3.7 阶段 7 的破解与分析(隐藏阶段)

密码如下:

破解过程:

## 第 4 章 总结

### 4.1 请总结本次实验的收获

本次实验使我熟悉了计算机系统的 ISA 指令系统与寻址方式，加深了对反汇编程序的理解。在拆弹的过程中更加深刻地学习了各种反汇编代码，拆弹过程也很有趣。

### 4.2 请给出对本次实验内容的建议

希望可以更有趣些。

注：本章为酌情加分项。

## 参考文献

- [1] 林来兴. 空间控制技术[M]. 北京: 中国宇航出版社, 1992: 25-42.
- [2] 辛希孟. 信息技术与信息服务国际研讨会论文集: A 集[C]. 北京: 中国科学出版社, 1999.
- [3] 赵耀东. 新时代的工业工程师[M/OL]. 台北: 天下文化出版社, 1998 [1998-09-26]. <http://www.ie.nthu.edu.tw/info/ie.newie.htm> (Big5) .
- [4] 谌颖. 空间交会控制理论与方法研究[D]. 哈尔滨: 哈尔滨工业大学, 1992: 8-13.
- [5] KANAMORI H. Shaking Without Quaking[J]. Science, 1998, 279 (5359): 2063-2064.
- [6] CHRISTINE M. Plant Physiology: Plant Biology in the Genome Era[J/OL]. Science, 1998 , 281 : 331-332[1998-09-23]. <http://www.sciencemag.org/cgi/collection/anatmorp>.