

课程编号 1502760001-01

题目类型 实验 3

得分	教师签名	批改日期
	冯禹洪	

深圳大学实验报告

课程名称: 计算机系统(2)

实验项目名称: 逆向工程实验

学院: 计算机与软件学院

专业: 计算机与软件学院所有专业

指导教师: 冯禹洪

报告人: 叶茂林 学号: 2021155015 班级: 腾班

实验时间: 2023 年 4 月 16 日至 4 月 23 日

实验报告提交时间: 2023 年 4 月 23 日

教务处制

- 注: 1、报告内的项目或内容设置, 可根据实际情况加以调整和补充。
2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。

一、实验目标与要求：

1. 理解程序（控制语句、函数、返回值、堆栈结构）是如何运行的。
2. 掌握 GDB 调试工具和 objdump 反汇编工具。

二、实验环境：

1. 计算机（Intel CPU）。
2. Linux64 位操作系统（Ubuntu 17）。
3. GDB 调试工具。
4. objdump 反汇编工具。

三、实验方法与步骤：

本实验设计为一个黑客拆解二进制炸弹的游戏。我们仅给黑客（同学）提供一个二进制可执行文件 `bomb_64` 和主函数所在的源程序 `bomb_64.c`，不提供每个关卡的源代码。程序运行中有 6 个关卡（6 个 phase），每个关卡需要用户输入正确的字符串或数字才能通关，否则会引爆炸弹（打印出一条错误信息，并导致评分下降）！

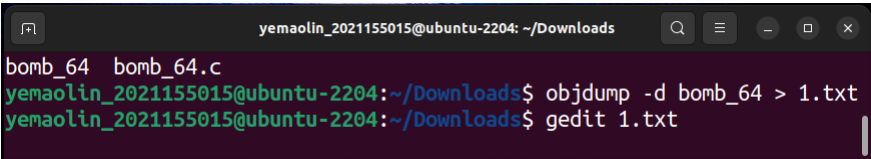
要求同学运用 **GDB 调试工具**和 **objdump 反汇编工具**，通过分析汇编代码，找到在每个 phase 程序段中，引导程序跳转到“`explode_bomb`”程序段的地方，并分析其成功跳转的条件，以此为突破口寻找应该在命令行输入何种字符串来通关。

本实验需解决 Phase_1(15 分)、Phase_2(15 分)、Phase_3(15 分)、Phase_4(15 分)、Phase_5(15 分)、Phase_6(10 分)。通过**截图+文字**的形式把实验过程写在实验报告上，最后并撰写**实验结论与心得**(15 分)。

四、实验过程及内容：

① 第一关（知识点：string，函数调用，栈）

将 `bomb` 文件用 `objdump -d bomb_64 > 1.txt` 命令进行反汇编，并将结果输出为 `1.txt`，如图 1 所示。

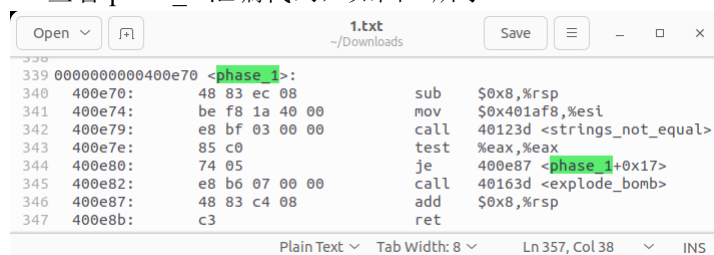


```
yemaolin_2021155015@ubuntu-2204: ~/Downloads
bomb_64  bomb_64.c
yemaolin_2021155015@ubuntu-2204:~/Downloads$ objdump -d bomb_64 > 1.txt
yemaolin_2021155015@ubuntu-2204:~/Downloads$ gedit 1.txt
```

图 1

- 注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。
- 2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。

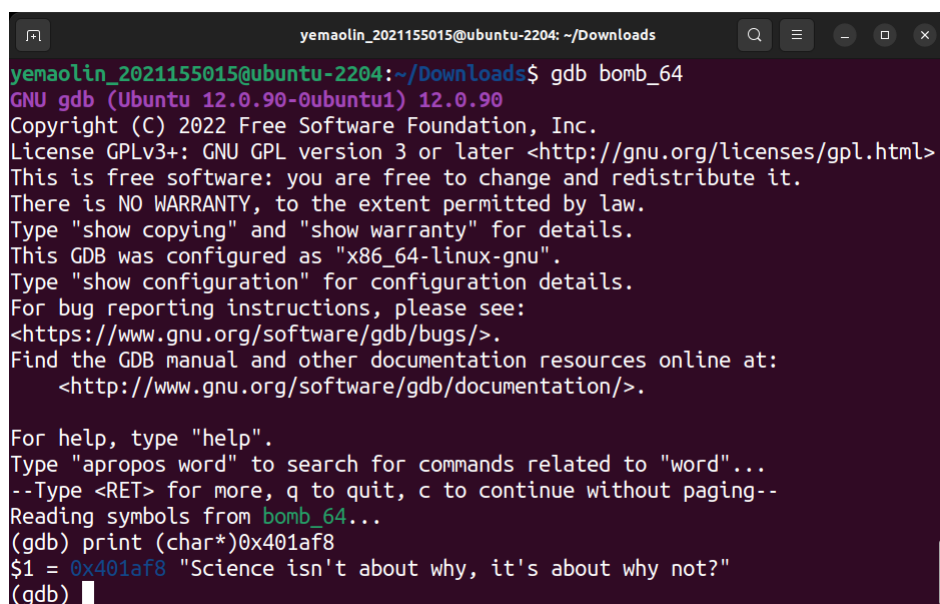
用 gedit 打开 1.txt 查看 phase_1 汇编代码，如图 2 所示。



```
339 000000000400e70: <phase_1>:
340 400e70: 48 83 ec 08      sub    $0x8,%rsp
341 400e74: be f8 1a 40 00   mov    $0x401af8,%esi
342 400e79: e8 bf 03 00 00   call   40123d <strings_not_equal>
343 400e7e: 85 c0           test   %eax,%eax
344 400e80: 74 05           je     400e87 <phase_1+0x17>
345 400e82: e8 b6 07 00 00   call   40163d <explode_bomb>
346 400e87: 48 83 c4 08      add    $0x8,%rsp
347 400e8b: c3             ret
```

图 2

通过观察汇编代码可知，程序先开栈，然后将 0x401af8 存进寄存器%esi，接着调用了一个字符串比较的函数，并且之后判断是否相等，不相等则引爆炸弹，推测 0x401af8 应该是一个字符串的首地址，用 gdb 调试，查看 0x401af8 该地址所对应的数据，如图 3 所示。

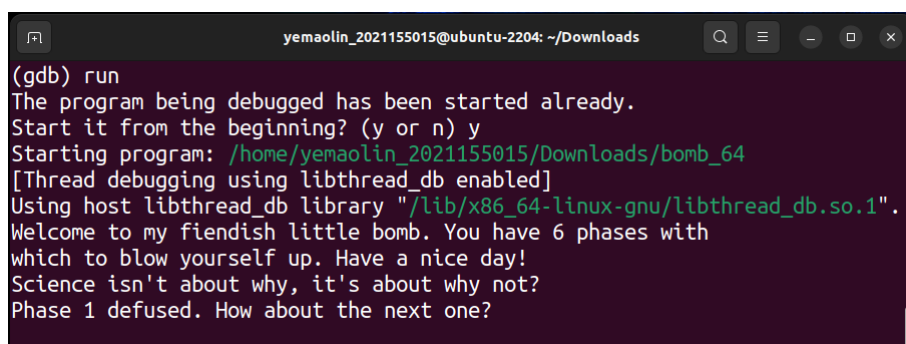


```
yemaolin_2021155015@ubuntu-2204: ~/Downloads
yemaolin_2021155015@ubuntu-2204:~/Downloads$ gdb bomb_64
GNU gdb (Ubuntu 12.0.90-0ubuntu1) 12.0.90
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
--Type <RET> for more, q to quit, c to continue without paging--
Reading symbols from bomb_64...
(gdb) print (char*)0x401af8
$1 = 0x401af8 "Science isn't about why, it's about why not?"
(gdb)
```

图 3

可知 0x401af8 该地址是字符串 Science isn't about why, it's about why not?的首地址，输入 run 进行运行程序验证答案是否正确，如图 4 所示，答案正确。



```
(gdb) run
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/yemaolin_2021155015/Downloads/bomb_64
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Science isn't about why, it's about why not?
Phase 1 defused. How about the next one?
```

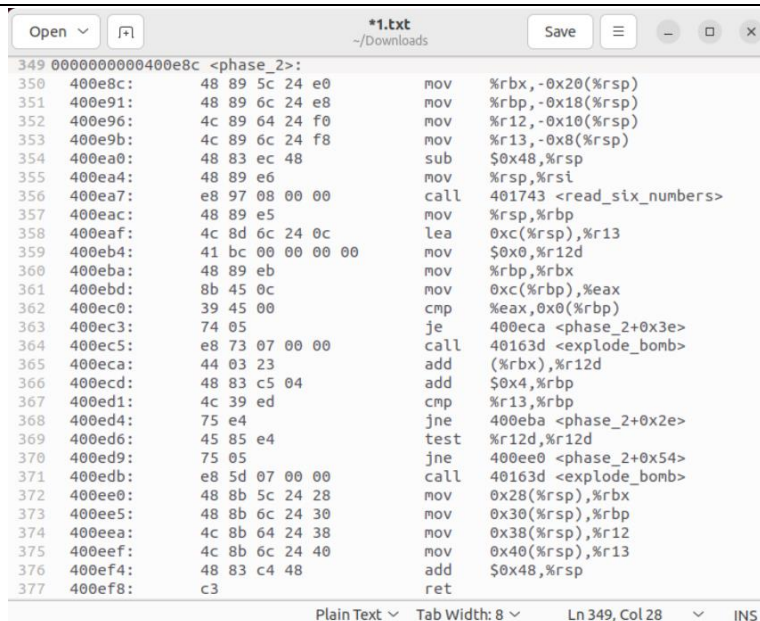
图 4

② 第二关（知识点：循环语句，数组）

查看 phase_2 汇编代码，如图 5 所示。

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

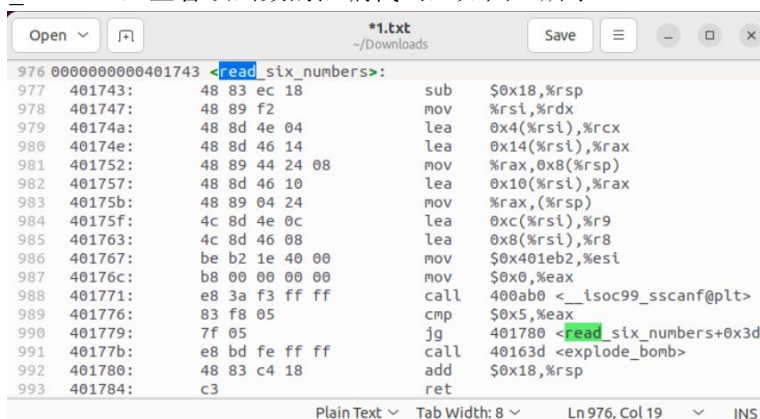
2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。



```
349 00000000400e8c <phase_2>:
350 400e8c: 48 89 5c 24 e0      mov    %rbx,-0x20(%rsp)
351 400e91: 48 89 6c 24 e8      mov    %rbp,-0x18(%rsp)
352 400e96: 4c 89 64 24 f0      mov    %r12,-0x10(%rsp)
353 400e9b: 4c 89 6c 24 f8      mov    %r13,-0x8(%rsp)
354 400ea0: 48 83 ec 48         sub    $0x48,%rsp
355 400ea4: 48 89 e6            mov    %rsp,%rsi
356 400ea7: e8 97 08 00 00      call   401743 <read_six_numbers>
357 400eac: 48 89 e5            mov    %rsp,%rbp
358 400eaf: 4c 8d 6c 24 0c      lea    0xc(%rsp),%r13
359 400eb4: 41 bc 00 00 00 00    mov    $0x0,%r12d
360 400eba: 48 89 eb            mov    %rbp,%rbx
361 400ebd: 8b 45 0c            mov    0xc(%rbp),%eax
362 400ec0: 39 45 00            cmp    %eax,0x0(%rbp)
363 400ec3: 74 05              je     400eca <phase_2+0x3e>
364 400ec5: e8 73 07 00 00      call   40163d <explode_bomb>
365 400eca: 44 03 23            add    (%rbx),%r12d
366 400ecd: 48 83 c5 04         add    $0x4,%rbp
367 400ed1: 4c 39 ed            cmp    %r13,%rbp
368 400ed4: 75 e4              jne    400eba <phase_2+0x2e>
369 400ed6: 45 85 e4            test   %r12d,%r12d
370 400ed9: 75 05              jne    400ee0 <phase_2+0x54>
371 400edb: e8 5d 07 00 00      call   40163d <explode_bomb>
372 400ee0: 48 8b 5c 24 28      mov    0x28(%rsp),%rbx
373 400ee5: 48 8b 6c 24 30      mov    0x30(%rsp),%rbp
374 400eea: 4c 8b 64 24 38      mov    0x38(%rsp),%r12
375 400eef: 4c 8b 6c 24 40      mov    0x40(%rsp),%r13
376 400ef4: 48 83 c4 48         add    $0x48,%rsp
377 400ef8: c3                 ret
```

图 5

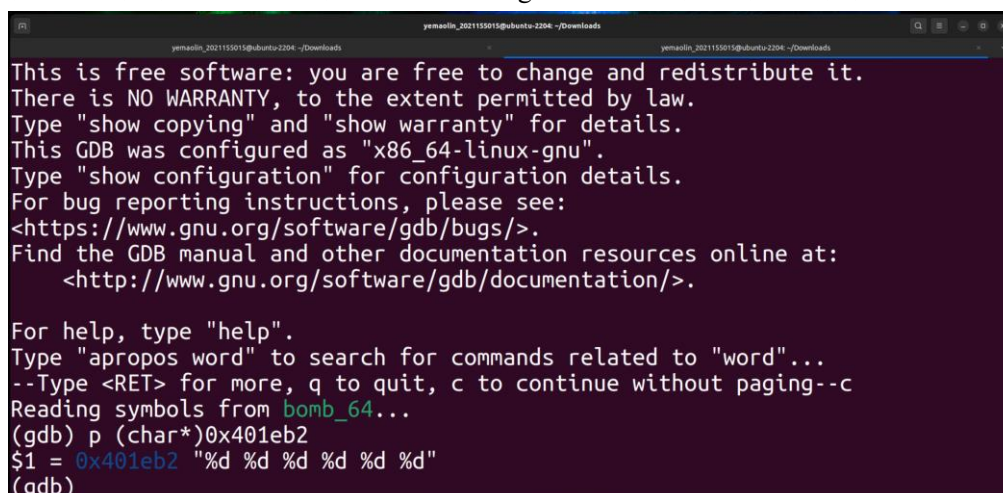
分析汇编代码可知，程序先保存现场，压栈了四个寄存器的值，然后保存栈顶指针，调用函数 `read_six_numbers`，查看该函数的汇编代码，如图 6 所示。



```
976 00000000401743 <read_six_numbers>:
977 401743: 48 83 ec 18         sub    $0x18,%rsp
978 401747: 48 89 f2            mov    %rsi,%rdx
979 40174a: 48 8d 4e 04         lea    0x4(%rsi),%rcx
980 40174e: 48 8d 46 14         lea    0x14(%rsi),%rax
981 401752: 48 89 44 24 08      mov    %rax,0x8(%rsp)
982 401757: 48 8d 46 10         lea    0x10(%rsi),%rax
983 40175b: 48 89 04 24         mov    %rax,(%rsp)
984 40175f: 4c 8d 4e 0c         lea    0xc(%rsi),%r9
985 401763: 4c 8d 46 08         lea    0x8(%rsi),%r8
986 401767: be b2 1e 40 00      mov    $0x401eb2,%esi
987 40176c: b8 00 00 00 00      mov    $0x0,%eax
988 401771: e8 3a f3 ff ff      call   400ab0 <__isoc99_sscanf@plt>
989 401776: 83 f8 05            cmp    $0x5,%eax
990 401779: 7f 05              jg     401780 <read_six_numbers+0x3d>
991 40177b: e8 bd fe ff ff      call   40163d <explode_bomb>
992 401780: 48 83 c4 18         add    $0x18,%rsp
993 401784: c3                 ret
```

图 6

观察到该函数将 `0x401eb2` 赋值给了 `%esi`，用 `gdb` 调试将该值打印出来看看，如图 7 所示。



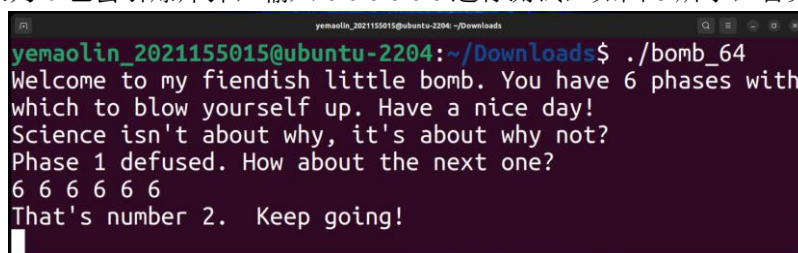
```
yemaojin_20211155015@ubuntu-2204: ~/Downloads
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
--Type <RET> for more, q to quit, c to continue without paging--
Reading symbols from bomb_64...
(gdb) p (char*)0x401eb2
$1 = 0x401eb2 "%d %d %d %d %d %d"
(gdb)
```

图 7

- 注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。
2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。

由此可知,程序读入了六个数据类型为int型的数字,然后开始比较地址为%rbp和%rbp+12的值,不相等则引爆炸弹,接着让%rbp增加4,循环比较了三次,%rbp一开始存储了数组的首地址,即比较了前3个元素和后3个元素是否相等,循环中还累加了前3个元素的值,如果为0也会引爆炸弹,输入666666进行测试,如图8所示,答案通过。

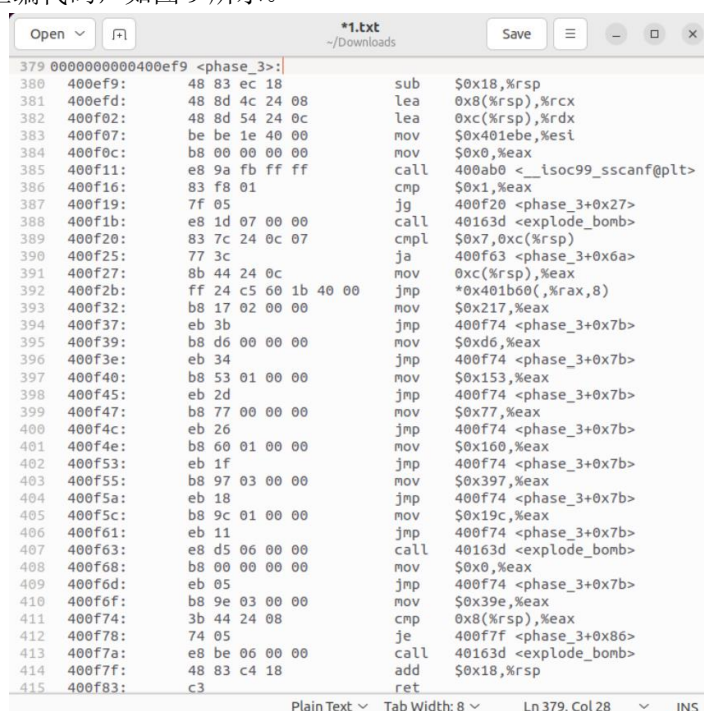


```
yemaolin_2021155015@ubuntu-2204: ~/Downloads
yemaolin_2021155015@ubuntu-2204:~/Downloads$ ./bomb_64
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Science isn't about why, it's about why not?
Phase 1 defused. How about the next one?
6 6 6 6 6 6
That's number 2. Keep going!
```

图 8

③ 第三关 (知识点: switch 语句)

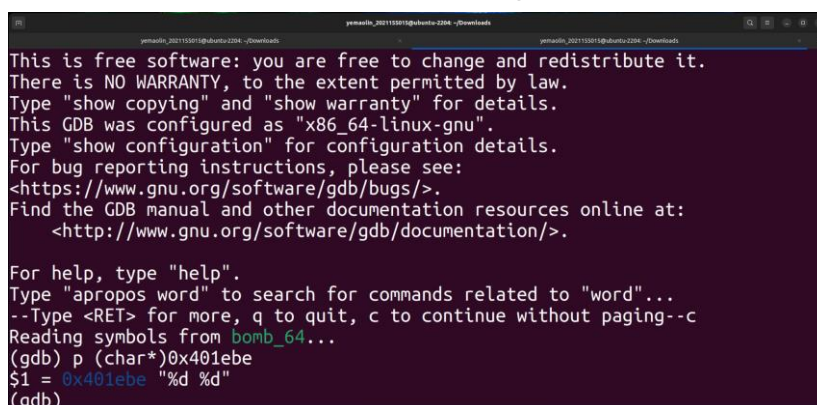
查看 phase_3 汇编代码,如图9所示。



```
379 000000000400ef9: <phase_3>:
380 400ef9: 48 83 ec 18      sub    $0x18,%rsp
381 400efd: 48 8d 4c 24 08    lea    0x8(%rsp),%rcx
382 400f02: 48 8d 54 24 0c    lea    0xc(%rsp),%rdx
383 400f07: be be 1e 40 00    mov    $0x401ebe,%esi
384 400f0c: b8 00 00 00 00    mov    $0x0,%eax
385 400f11: e8 9a fb ff ff    call   400ab0 <__isoc99_sscanf@plt>
386 400f16: 83 f8 01         cmp    $0x1,%eax
387 400f19: 7f 05           jg     400f20 <phase_3+0x27>
388 400f1b: e8 1d 07 00 00    call   40163d <explode_bomb>
389 400f20: 83 7c 24 0c 07    cmpl   $0x7,0xc(%rsp)
390 400f25: 77 3c           ja     400f63 <phase_3+0x6a>
391 400f27: 8b 44 24 0c      mov    0xc(%rsp),%eax
392 400f2b: ff 24 c5 60 1b 40 00 jmp     *0x401b60(,%rax,8)
393 400f32: b8 17 02 00 00    mov    $0x217,%eax
394 400f37: eb 3b           jmp     400f74 <phase_3+0x7b>
395 400f39: b8 d6 00 00 00    mov    $0xd6,%eax
396 400f3e: eb 34           jmp     400f74 <phase_3+0x7b>
397 400f40: b8 53 01 00 00    mov    $0x153,%eax
398 400f45: eb 2d           jmp     400f74 <phase_3+0x7b>
399 400f47: b8 77 00 00 00    mov    $0x77,%eax
400 400f4c: eb 26           jmp     400f74 <phase_3+0x7b>
401 400f4e: b8 60 01 00 00    mov    $0x160,%eax
402 400f53: eb 1f           jmp     400f74 <phase_3+0x7b>
403 400f55: b8 97 03 00 00    mov    $0x397,%eax
404 400f5a: eb 18           jmp     400f74 <phase_3+0x7b>
405 400f5c: b8 9c 01 00 00    mov    $0x19c,%eax
406 400f61: eb 11           jmp     400f74 <phase_3+0x7b>
407 400f63: e8 d5 06 00 00    call   40163d <explode_bomb>
408 400f68: b8 00 00 00 00    mov    $0x0,%eax
409 400f6d: eb 05           jmp     400f74 <phase_3+0x7b>
410 400f6f: b8 9e 03 00 00    mov    $0x39e,%eax
411 400f74: 3b 44 24 08      cmp    0x8(%rsp),%eax
412 400f78: 74 05           je     400f7f <phase_3+0x86>
413 400f7a: e8 be 06 00 00    call   40163d <explode_bomb>
414 400f7f: 48 83 c4 18      add    $0x18,%rsp
415 400f83: c3              ret
```

图 9

分析汇编代码,程序将 0x401ebe 赋值给了%esi,用 gdb 打印看看,如图10所示。



```
yemaolin_2021155015@ubuntu-2204: ~/Downloads
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
--Type <RET> for more, q to quit, c to continue without paging--
Reading symbols from bomb_64...
(gdb) p (char*)0x401ebe
$1 = 0x401ebe "%d %d"
(gdb)
```

图 10

- 注: 1、报告内的项目或内容设置,可根据实际情况加以调整和补充。
2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。

可知程序输入了两个 int 型的数据，存放在地址为 $\%rsp+8$ 和 $\%rsp+12$ 的内存中，如果输入数据的格式类型不对则引爆炸弹，然后判断第一个数是否大于 7，如果大于 7 则引爆炸弹，接下来是 switch 语句，将第一个数存进 $\%rax$ ，并根据 $\%rax$ 的值，跳转到地址为 $\ast(0x401b60+\%rax\ast 8)$ 的语句，通过 gdb 打印可以看出，分别跳转到哪里，如图 11 所示。

```
(gdb) p/x *(0x401b60+8*0)
$3 = 0x400f32
(gdb) p/x *(0x401b60+8*1)
$4 = 0x400f6f
(gdb) p/x *(0x401b60+8*2)
$5 = 0x400f39
(gdb) p/x *(0x401b60+8*3)
$6 = 0x400f40
(gdb) p/x *(0x401b60+8*4)
$7 = 0x400f47
(gdb) p/x *(0x401b60+8*5)
$8 = 0x400f4e
(gdb) p/x *(0x401b60+8*6)
$9 = 0x400f55
(gdb) p/x *(0x401b60+8*7)
$10 = 0x400f5c
(gdb)
```

图 11

跳转到相应位置之后，分析汇编代码可知，程序对于不同的第一个数准备了不同的数值与第二个数进行比较，若不相等则引爆炸弹，由汇编代码可以看出，0 对应 535，1 对应 926，2 对应 214，3 对应 339，4 对应 119，5 对应 352，6 对应 919，7 对应 412，输入 4 119 测试，如图 12 所示，测试通过。

```
yemaolin_2021155015@ubuntu-2204: ~/Downloads$ ./bomb_64
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Science isn't about why, it's about why not?
Phase 1 defused. How about the next one?
6 6 6 6 6 6
That's number 2. Keep going!
4 119
Halfway there!
```

图 12

④ 第四关（知识点：递归）

查看 phase_4 汇编代码，如图 13 所示。

```
436 000000000400fc1: <phase_4>:
437 400fc1: 48 83 ec 18      sub    $0x18,%rsp
438 400fc5: 48 8d 54 24 0c    lea    0xc(%rsp),%rdx
439 400fca: be c1 1e 40 00    mov    $0x401ec1,%esi
440 400fcf: b8 00 00 00 00    mov    $0x0,%eax
441 400fd4: e8 d7 fa ff ff    call   400ab0 <__isoc99_sscanf@plt>
442 400fd9: 83 f8 01         cmp    $0x1,%eax
443 400fdc: 75 07           jne    400fe5 <phase_4+0x24>
444 400fde: 83 7c 24 0c 00    cmpl   $0x0,0xc(%rsp)
445 400fe3: 7f 05           jg     400fea <phase_4+0x29>
446 400fe5: e8 53 06 00 00    call   40163d <explode_bomb>
447 400fea: 8b 7c 24 0c      mov    0xc(%rsp),%edi
448 400fee: e8 91 ff ff ff    call   400f84 <func4>
449 400ff3: 83 f8 37         cmp    $0x37,%eax
450 400ff6: 74 05           je     400ffd <phase_4+0x3c>
451 400ff8: e8 40 06 00 00    call   40163d <explode_bomb>
452 400ffd: 48 83 c4 18      add    $0x18,%rsp
453 401001: c3             ret
```

图 13

- 注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。
2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。

分析汇编代码，程序将 0x401ec1 赋值给了%esi，用 gdb 打印看看，如图 14 所示

```
yemaolin_2021155015@ubuntu-2204: ~/Downloads$ gdb bomb_64
GNU gdb (Ubuntu 12.0.90-0ubuntu1) 12.0.90
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
--Type <RET> for more, q to quit, c to continue without paging--c
Type "apropos word" to search for commands related to "word"...
Reading symbols from bomb_64...
(gdb) p (char*)0x401ec1
$1 = 0x401ec1 "0"
```

图 14

可知程序输入了一个 int 型数据，存储在地址为%rsp+12 处，如果输入数据的格式类型不对则引爆炸弹，然后把输入的数和 0 比较，如果小于等于 0 则引爆炸弹。接着保存输入数的值在%rdi，然后调用 func4 函数，判断函数返回值是否等于 0x37，不相等则引爆炸弹，那么我们需要分析 func4 的汇编代码，如图 15 所示。

```
000000000400f84 <func4>:
400f84: 48 89 5c 24 f0      mov     %rbx,-0x10(%rsp)
400f89: 48 89 6c 24 f8      mov     %rbp,-0x8(%rsp)
400f8e: 48 83 ec 18         sub     $0x18,%rsp
400f92: 89 fb             mov     %edi,%ebx
400f94: b8 01 00 00 00     mov     $0x1,%eax
400f99: 83 ff 01          cmp     $0x1,%edi
400f9c: 7e 14             jle     400fb2 <func4+0x2e>
400f9e: 8d 7b ff          lea     -0x1(%rbx),%edi
400fa1: e8 de ff ff ff     call    400f84 <func4>
400fa6: 89 c5             mov     %eax,%ebp
400fa8: 8d 7b fe          lea     -0x2(%rbx),%edi
400fab: e8 d4 ff ff ff     call    400f84 <func4>
400fb0: 01 e8             add     %ebp,%eax
400fb2: 48 8b 5c 24 08     mov     0x8(%rsp),%rbx
400fb7: 48 8b 6c 24 10     mov     0x10(%rsp),%rbp
400fbc: 48 83 c4 18         add     $0x18,%rsp
400fc0: c3               ret

422,13-19 37%
```

图 15

由汇编代码可知，如果%edi 小于等于 1 的话就会返回 1，但函数必须返回 0x37 才不会引爆炸弹，所以%edi 必须要大于 1，如果大于 1，接下来又让%edi 减一，然后递归调用 func4 函数，把返回值保存在%edx 中，然后又让%edi 减一，继续递归调用 func4，最后将返回值和%edx 加起来作为函数的返回值，由此可知，该函数的递归式为 func4(n)=func4(n-1)+func4(n-2)，并且 func4(1)=1，func4(2)=2。通过计算可知，func4(9)=0x37，输入 9 测试，如图 16 所示，测试通过。

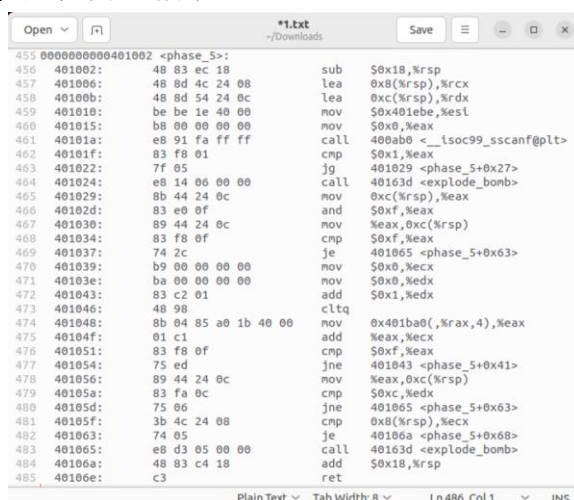
```
yemaolin_2021155015@ubuntu-2204: ~/Downloads$ ./bomb_64
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Science isn't about why, it's about why not?
Phase 1 defused. How about the next one?
6 6 6 6 6
That's number 2. Keep going!
4 119
Halfway there!
9
So you got that one. Try this one.
```

图 16

- 注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。
- 2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。

⑤ 第五关（知识点：字符串变换，ascii 转换，寻址）

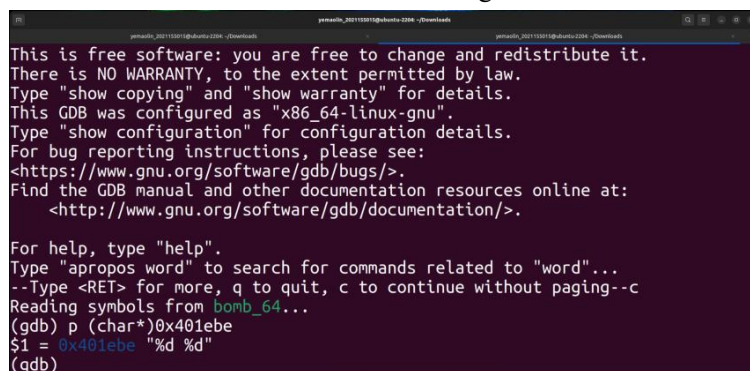
查看 phase_5 汇编代码，如图 17 所示。



```
455 000000000401002: <phase_5>:
456 401002: 48 83 ec 18      sub    $0x18,%rsp
457 401006: 48 8d 4c 24 08    lea    0x8(%rsp),%rcx
458 40100b: 48 8d 54 24 0c    lea    0xc(%rsp),%rdx
459 401010: be be 1e 40 00    mov    $0x401ebe,%esi
460 401015: b8 00 00 00 00    mov    $0x0,%eax
461 40101a: e8 91 fa ff ff    call   __isoc99_sscanf@plt
462 40101f: 83 f8 01         cmp    $0x1,%eax
463 401022: 7f 05           jg     401029 <phase_5+0x27>
464 401024: e8 14 06 00 00    call   __explode_bomb
465 401029: 8b 44 24 0c      mov    0xc(%rsp),%eax
466 40102d: 83 e0 0f         and    $0xf,%eax
467 401030: 89 44 24 0c      mov    %eax,0xc(%rsp)
468 401034: 83 f8 0f         cmp    $0xf,%eax
469 401037: 74 2c           je     401065 <phase_5+0x63>
470 401039: b9 00 00 00 00    mov    $0x0,%ecx
471 40103e: ba 00 00 00 00    mov    $0x0,%edx
472 401043: 83 c2 01         cmp    $0x1,%edx
473 401046: 48 98           cltq
474 401048: 8b 04 85 a0 1b 40 00 mov    0x401ba0(,%rax,4),%eax
475 40104f: 01 c1           add    %eax,%ecx
476 401051: 83 f8 0f         cmp    $0xf,%eax
477 401054: 75 ed           jne    401043 <phase_5+0x41>
478 401056: 89 44 24 0c      mov    %eax,0xc(%rsp)
479 40105a: 83 fa 0c         cmp    $0xc,%edx
480 40105d: 75 06           jne    401065 <phase_5+0x63>
481 40105f: 3b 4c 24 08      cmp    0x8(%rsp),%ecx
482 401063: 74 05           je     40106a <phase_5+0x68>
483 401065: e8 d3 05 00 00    call   __explode_bomb
484 40106a: 48 83 c4 18      add    $0x18,%rsp
485 40106e: c3             ret
```

图 17

分析汇编代码，程序将 0x401ebe 赋值给了 %esi，用 gdb 打印看看，如图 18 所示。

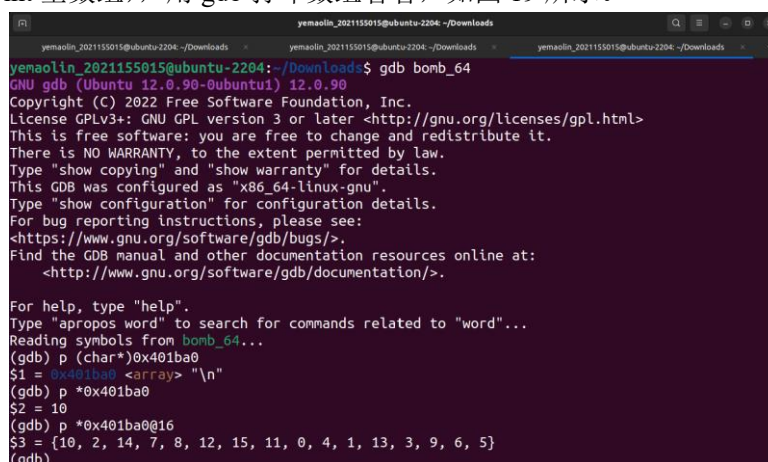


```
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
--Type <RET> for more, q to quit, c to continue without paging--c
Reading symbols from bomb_64...
(gdb) p (char*)0x401ebe
$1 = 0x401ebe "%d %d"
(gdb)
```

图 18

可知程序输入了两个 int 型的数据，存放在地址为 %rsp+8 和 %rsp+12 的内存中，如果输入数据的格式不对或者类型不对则引爆炸弹，然后把第二个数和 0xf 相与提取前 4 位数，如果等于 15 则引爆炸弹，然后是一个循环，每次都将 0x401ba0(%rax,4) 赋值给 %eax，这个 0x401ba0 应该是一个数值的首地址，%rax 存的是输入的第二个数，偏移的单位为 4 个字节，推测是 int 型数组，用 gdb 打印数组看看，如图 19 所示。



```
yemaolin_2021155015@ubuntu-2204: ~/Downloads
GNU gdb (Ubuntu 12.0.90-0ubuntu1) 12.0.90
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from bomb_64...
(gdb) p (char*)0x401ba0
$1 = 0x401ba0 <array> '\n'
(gdb) p *0x401ba0
$2 = 10
(gdb) p *0x401ba0@16
$3 = {10, 2, 14, 7, 8, 12, 15, 11, 0, 4, 1, 13, 3, 9, 6, 5}
(gdb)
```

图 19

- 注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。
2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。

而循环跳出的条件是%eax 等于 15，循环过程%edx 记录了循环的次数，%ecx 是%eax 的累计值，而%edx 不等于 12 会引爆炸弹，%ecx 不等于第一个输入的数也会引爆炸弹。所以输入的第一个数是访问到的数组元素的累加和，输入的第二个数则是第一个被访问的元素的相对位置，由此我们可以从元素 15 的位置进行倒退 12 次，可以知道第一个输入的数是 7，第二个输入的数是 93，输入 7 93 测试，如图 20 所示，测试通过。

```
yemaolin_2021155015@ubuntu-2204: ~/Downloads
yemaolin_2021155015@ubuntu-2204:~/Downloads$ ./bomb_64
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Science isn't about why, it's about why not?
Phase 1 defused. How about the next one?
6 6 6 6 6 6
That's number 2. Keep going!
4 119
Halfway there!
9
So you got that one. Try this one.
7 93
Congratulations! You've (mostly) defused the bomb!
Hit Control-C to escape phase 6 (for free!), but if you want to
try phase 6 for extra credit, you can continue. Just beware!
```

图 20

⑥ 第六关（知识点：寻址）

查看 phase_6 汇编代码，如图 21 所示。

```
*1.txt
~/Downloads
527 0000000004010d9 <phase_6>:
528 4010d9: 48 83 ec 08      sub    $0x8,%rsp
529 4010dd: ba 0a 00 00 00   mov    $0xa,%edx
530 4010e2: be 00 00 00 00   mov    $0x0,%esi
531 4010e7: e8 94 fa ff ff   call   400b80 <strtoul@plt>
532 4010ec: 89 05 8e 16 20 00 mov    %eax,0x20168e(%rip) # 602780 <node0>
533 4010f2: bf 80 27 60 00   mov    $0x602780,%edi
534 4010f7: e8 73 ff ff ff   call   40106f <fun6>
535 4010fc: 48 8b 40 08      mov    0x8(%rax),%rax
536 401100: 48 8b 40 08      mov    0x8(%rax),%rax
537 401104: 48 8b 40 08      mov    0x8(%rax),%rax
538 401108: 8b 15 72 16 20 00 mov    0x201672(%rip),%edx # 602780 <node0>
539 40110e: 39 10            cmp    %edx,(%rax)
540 401110: 74 05            je     401117 <phase_6+0x3e>
541 401112: e8 26 05 00 00   call   40163d <explode_bomb>
542 401117: 48 83 c4 08      add    $0x8,%rsp
543 40111b: c3              ret
```

图 21

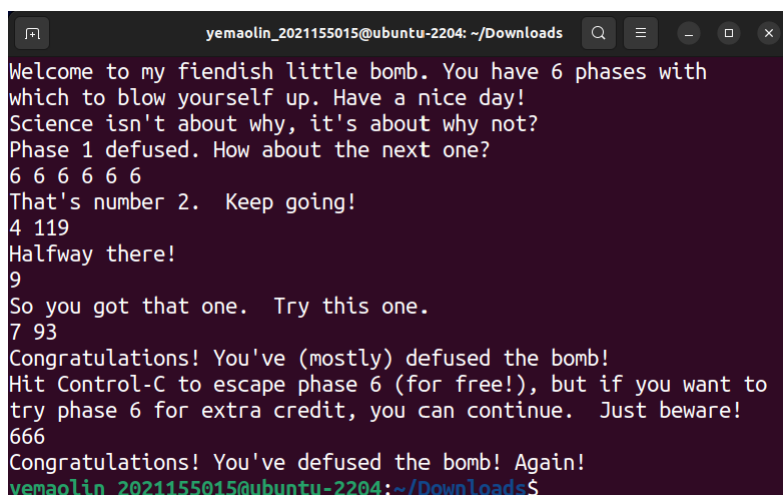
程序首先调用函数将字符串转换为 long 型数据存储在地址为 0x602780 的内存中，我们将地址为 0x602780 的内存值打印出来，如图 22 所示，是一张表。

```
yemaolin_2021155015@ubuntu-2204: ~/Downloads
(gdb) p *0x602780@40
$7 = {0, 0, 6301584, 0, 374, 1, 6301600, 0, 826, 2, 6301616, 0, 370, 3,
6301632, 0, 782, 4, 6301648, 0, 488, 5, 6301664, 0, 673, 6, 6301680, 0, 286,
7, 6301696, 0, 600, 8, 6301712, 0, 529, 9, 0, 0}
(gdb)
```

图 22

- 注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。
2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。

所以程序是将输入的数插入了这张表的表头，然后调用函数 `func6` 进行两层循环降序排序，并且通过三次加 8 偏移寻址更新 `%rax` 使之为表里的第四个数值的地址，然后比较地址为 `%rax` 和地址为 `0x602780` 的值是否相同，即排序后的第四个数值和输入的数值进行比较，如果不同则引爆炸弹。由此可知，程序输入一个数值，并将它插入原有的数据表后进行降序排序，输入的数值必须排在第四个位置才不会引爆炸弹，也就是输入的数值必须介于 600 和 673 之间，输入 666 进行测试，如图 23 所示，测试通过。



```
yemaolin_2021155015@ubuntu-2204: ~/Downloads
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Science isn't about why, it's about why not?
Phase 1 defused. How about the next one?
6 6 6 6 6
That's number 2. Keep going!
4 119
Halfway there!
9
So you got that one. Try this one.
7 93
Congratulations! You've (mostly) defused the bomb!
Hit Control-C to escape phase 6 (for free!), but if you want to
try phase 6 for extra credit, you can continue. Just beware!
666
Congratulations! You've defused the bomb! Again!
yemaolin_2021155015@ubuntu-2204:~/Downloads$
```

图 23

五、实验结论与心得体会：

本次实验运用了 GDB 调试工具和 `objdump` 反汇编工具，通过分析汇编代码分析程序执行的行为，成功通过了六个关卡并拆解了炸弹。

在实验过程中，GDB 调试工具的强大能够帮助我们监视和检查程序运行时的各种状态。同时，对于汇编代码的分析过程，准确理解栈的存储信息、各个使用到的寄存器的变化值，以及所调用函数的参数和返回值等都非常重要，这有助于深入了解程序的内部运行方式。

通过本次分析汇编语言拆解炸弹的实验，我认识到学会分析汇编语言是成长为专业程序员的必由之路，并且掌握相关工具和技巧可以帮助我们更好地理解程序并快速找到和解决问题。

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。

指导教师批阅意见:

成绩评定:

指导教师签字：冯禹洪
2023 年 月 日

备注:

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。
2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。