BOMBLAB:README

For Hongyi Honor College
TA: Mao Wenyue Zhang Bowen

1 要求

进入 educoder 网站后，你就能看见所有的要求了。
所以在此不做赘述。
如果有兴趣可以看看《东方永夜炸》文档。

2 实验报告

实验线上的提交截止时间为 2020 年 11 月 7 日。
实验报告截止提交的时间为 2020 年 11 月 8 日 24 时。

实验报告的要求：
要求能对你的代码进行分析，格式不做要求，即，我们希望你能将你做题的思路写下来，至于是怎样的思路都可以，只要是你自己做的，你自己能得到的思路，写在哪里，都可以。

实验报告的提交：
- 命名格式：学号-姓名-bomblab 实验报告.pdf
- 发送到邮箱 hyxt_csapp_2020@163.com
- 截止时间：2020 年 11 月 8 日 24 时。

3 关于 gdb

gbd 用来对得到的汇编文件进行调试，以下代码也许用的上。

b *0x114514　在地址 0x114514 处设置断点（建议设置在炸弹处，防止炸弹炸了）
print (char*)0x114514　输出地址 0x114514 的内容
r　运行
c　继续运行

4 关于 vim

可以直接查看文件内容的方式

5 特殊提交

根据 educoder 的要求，你的提交需要在程序语言里边进行输出。
即，输出你能破除炸弹的方式。

6 例子

下面以第一阶段（第一关）为例介绍实验步骤：首先调用"objdump –d bomb > bomb_disas.txt" 对 bomb 进行反汇编并将汇编源代码输出到"boomb_disas.txt"文本文件中。查看该汇编源代码文件，我们可以在 main 函数中找到如下语句，从而得知第一关的处理程序包含在"main()"函数所调用的函数

“phase_1()”中，判断的过程可以参照 bomb.c 文件源码。汇编代码中地址 400ca7 处调用了 phase_1 函数，

```
400ca0:        48 8b 45 f8              mov    -0x8(%rbp),%rax
400ca4:        48 89 c7                 mov    %rax,%rdi
400ca7:        e8 a8 00 00 00           callq  400d54 <phase_1>
400cac:        bf e8 19 40 00           mov    $0x4019e8,%edi
400cb1:        e8 ca fc ff ff           callq  400980 <puts@plt>
400cb6:        e8 8d 09 00 00           callq  401648 <read_line>
400cbb:        48 89 45 f8              mov    %rax,-0x8(%rbp)
400cbf:        48 8b 45 f8              mov    -0x8(%rbp),%rax
400cc3:        48 89 c7                 mov    %rax,%rdi
400cc6:        e8 b1 00 00 00           callq  400d7c <phase_2>
400ccb:        bf 11 1a 40 00           mov    $0x401a11,%edi
400cd0:        e8 ab fc ff ff           callq  400980 <puts@plt>
400cd5:        e8 6e 09 00 00           callq  401648 <read_line>
400cda:        48 89 45 f8              mov    %rax,-0x8(%rbp)
400cde:        48 8b 45 f8              mov    -0x8(%rbp),%rax
400ce2:        48 89 c7                 mov    %rax,%rdi
```

我们在反汇编代码中寻找这个子函数 phase_1：

```
0000000000400f57 <phase_1>:
  400f57:       48 83 ec 08              sub    $0x8,%rsp
  400f5b:       be 2d 19 40 00           mov    $0x40192d,%esi
  400f60:       e8 b7 01 00 00           callq  40111c <strings_not_equal>
  400f65:       85 c0                    test   %eax,%eax
  400f67:       74 05                    je     400f6e <phase_1+0x17>
  400f69:       e8 74 02 00 00           callq  4011e2 <explode_bomb>
  400f6e:       48 83 c4 08              add    $0x8,%rsp
  400f72:       c3                       retq

0000000000400f73 <phase_4>:
```

可以看到这个子函数比较小，只有几行汇编代码，可以进行简单阅读（如果汇编代码较多，不建议逐句阅读，而是借用 gdb 调试工具进行辅助）：我们看到（教科书中已经提到过调用函数的过程），……，还调用了 string_not_equal 函数，接着测试%eax 是否为零，如果是就跳转到+0x26 出否者就调用 explode_boPRINT mb，可以判定这是一个判断两个字符串是否相等的过程，使用 gdb 调试 bomb 二进制文件：gdb bomb 后，输入(gdb)print (char *)ox40192d 输出是

```
Reading symbols from /home/allen/work/bomb...done.
(gdb) print (char *)0x40192d
$1 = 0x40192d "Public speaking is very easy."
(gdb)
```

于是去设置断点去监测这个答案是否正确，我们在 explod_bomb 处设置断点：(gdb)break *0x400f69 然后(gdb)run 运行按照提示输入这个字符串

```
(gdb) print (char *)0x40192d
$1 = 0x40192d "Public speaking is very easy."
(gdb) break *0x400f69
Breakpoint 1 at 0x400f69: file phases.c, line 26.
(gdb) r
Starting program: /home/allen/work/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Public speaking is very easy.
Phase 1 defused. How about the next one?
```

第一关解除。