Bomb Lab 实验说明

1. 实验内容和要求

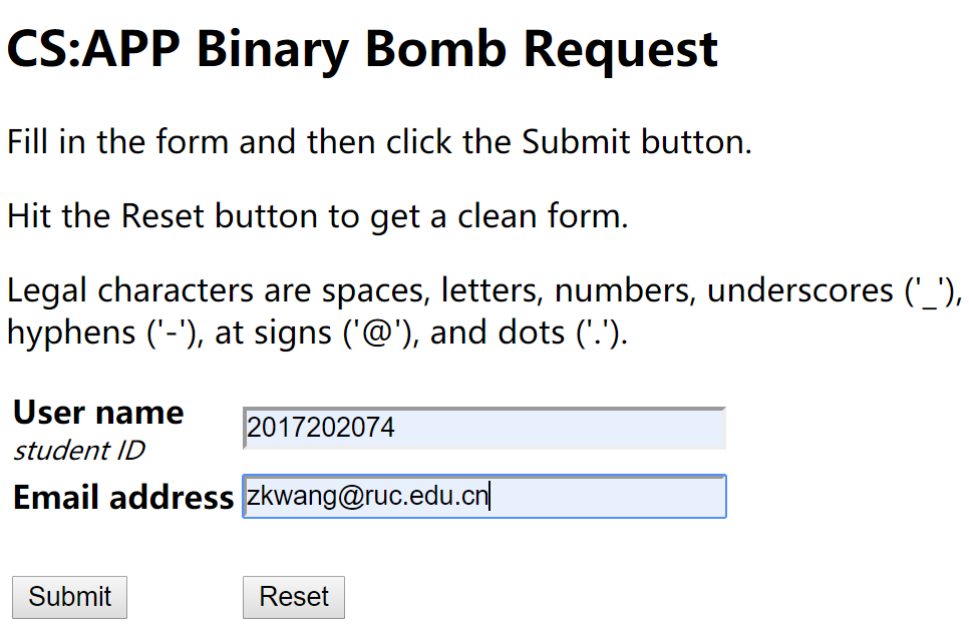
- 完成拆弹
  - 可执行文件bomb包括6道密码，必须全都解开；输错密码会导致炸弹爆炸；
  - 此外还包含一个隐藏关
  - 注意：在线系统随机生成密码，每个人的都不相同；
  - 方法：反汇编、GDB调试（Linux环境）；
- 实验报告：详细求解密码的过程，提交至 obe（只需要提交实验报告）
- 截止日期：2019.11.19 23：55
- 评分标准：
  - 拆弹结果：50%
    - 根据 scoreboard 的成绩
    - 每次爆炸减 0.5 分，减满 20 分后不再扣分（炸够 40 次即可体验无限开火权哦亲~）
    - 隐藏关不计入成绩（强迫症专用）
  - 实验报告：50%

2. 实验内容讲解

2.1 Bomb Lab 实验步骤

2.1.1 下载自己的 Bomb

- 访问网站 http://192.144.238.229:15213
- 用户名为学号，并填写自己的 email
- 会下载一个随机生成的 bomb 压缩包



2.1.2 登录 Linux 服务器

- 服务器 IP：192.144.238.229
- 用户名、密码同 Data Lab 实验
- 将自己下载的 Bomb 压缩包上传到自己的目录下，如 `/home/2017202074`
- 用 `tar -xvf bombN.tar` 命令解压（N 为自己 bomb 文件的编号）

2.1.3 反汇编

- cd ./bombN 进入 bomb 目录
- ls（可以看到三个文件：bomb是二进制可执行文件；bomb.c是C文件，供参考；README是说明文档）
- vim ./README可以查看（esc+:q!不保存退出）
- 反汇编：objdump –d ./bomb > asm（汇编代码保存到asm文件中，同样用vi或vim可以查看）
- 在 vim 中，输入‘/’可以搜索关键词；输入‘:行号’可以调到指定行
- 建议把 asm 文件传回本地看

2.1.4 尝试解密码

- ./bomb可以运行炸弹，输入密码，匹配则进入下一关（不要轻易尝试，每次爆炸自动扣0.5分，最多扣20分）
- 可以把前几关破解的密码放到当前目录一个文件中，例如 input。

  则 `./bomb ./input` 指令可以把文件中前几行依次输入
- 注意：Ctrl+C 终止程序，不要相信提示信息，再等一会儿会退出
- 由于需要随时上传爆炸信息，本程序只能在服务器上运行，在本地执行会报错。

2.1.5 gdb 调试

- gdb ./bomb
- set args input; run
- 分析每一段汇编代码
- 设置断点，运行至断点
- 查看寄存器、内存等

- 查看得分：http://192.144.238.229:15213/scoreboard
  (未必更新及时)

2.1.6 实验报告

- 实验目标
- 实验方法
- 实验结果（拆弹程度、密码）
- 详细过程
  - 每一关的详解，最好引用代码、寄存器、内存内容
  - 最后有得失及问题总结
  - 在你认为必要的地方添加截图

- 由于本次实验部分密码可以靠猜或暴力试出来，所以实验报告所占分数比重较大，你需要在实验报告中体现出**你确实是按照调试过程读懂汇编代码找到的密码**，而不是猜或者试出来的~~(哪怕你真的是这样做的)~~

2.2 GDB 使用方法

2.2.1 GDB是什么?

GDB 是一个由 GNU 开源组织发布的、UNIX/LINUX 操作系统下的、基于命令行的、功能强大的程序调试工具。 对于一名 Linux 下工作的 c/c++ 程序员，gdb 是必不可少的工具。

2.2.2 GDB 的功能

一般来说，GDB主要帮忙你完成下面四个方面的功能：

- 启动你的程序，可以按照你的自定义的要求随心所欲的运行程序。
- 可让被调试的程序在你所指定的调置的断点处停住。（断点可以是条件表达式）
- 当程序被停住时，可以检查此时你的程序中所发生的事。
- 动态地改变你程序的执行环境。

2.2.3 GDB 本实验中常用指令

- 运行命令
  - `gdb ./bomb` 进入 GDB 调试，bomb 即为要调试的程序
  - `set args xxx` 设置程序运行的参数
  - `show args` 可以查看设置好的参数
  - `run(简写为 r)` 开始执行/重启程序
  - `quit（简写为 q）` 退出调试
  - `list(简写为 l)` 显示当前一段代码
  - `disassemble 函数名` 可以反汇编指定的函数，如果不加函数名即显示当前所在部分代码段的汇编语句。

- 断点调试
  - `break(或者简写为 b) *0x.../行号/函数名` 可以在对应地址的汇编指令/程序对应行/程序的对应函数处设置断点
  - `info b` 查看断点情况，后面可以添加断点号，如果不加则显示所有断点信息
  - `disable/enable/delete b n1 n2 n3..` 禁用/启用/删除某(几)个断点
  - `continue` 执行到下一断点或程序结束
  - `next (n)` 单步运行（不会进入函数）
  - `return 返回值` 可以直接结束当前函数并返回指定值
  - `until 行号` 运行到指定行
  - `step (s)` 跟入函数调试
  - `finish` 直接运行到当前函数结束

- 数据命令

- `print(简写为 p) 变量名/寄存器名`可以显示当前变量/寄存器值，显示时会在前面加 $N 标记，再次引用时只需要写作 $N
- 使用 print 指令时，可以用 `print 变量名 格式`来格式化输出，x：十六进制；d：十进制；u：无符号数；o：八进制；c：字符格式；f：浮点数。
- `display 变量名/寄存器名`可以观察变量/寄存器，程序每步执行都会自动显示变量值
- `info display`显示所有要显示的表达式信息
- `delete/disable/enable`删除、禁用、启用某个表达式的显示
- `whatis 变量名`显示变量的数据类型
- `set 变量=变量值`可以改变程序中某个变量的值
- `examine(简写为x)/nfu`查看内存

  n表示要显示的内存单元的个数

  f表示显示方式, 可取如下值

  - x 按十六进制格式显示变量。
  - d 按十进制格式显示变量。
  - u 按十进制格式显示无符号整型。
  - o 按八进制格式显示变量。
  - t 按二进制格式显示变量。
  - i 指令地址格式
  - c 按字符格式显示变量。
  - f 按浮点数格式显示变量。
  - s 按字符串显示变量

  u表示一个地址单元的长度

  - b表示单字节，
  - h表示双字节，
  - w表示四字节，
  - g表示八字节

  例如`x/3dh buf`表示从内存地址 buf 读3个双字节十进制数

  `x/ni`可以显示当前向后的 n 条汇编语句

3. 友情提示

- 调试时在爆炸前设置一个断点
- 在对汇编代码设断点使用`break *0x...`命令时要注意，只能在程序运行时使用的gdb 的 disassemble 命令查看汇编代码及对应代码的地址，不能使用 objdump 反汇编出的代码中的地址。
- 把密码输入 input 文件并在运行时设为参数可以节省时间和防止输错
- 隐藏关的"钥匙"需要在某一关额外输入
- 每个人只有一个使用 bomb 文件的机会，助教会以每个人的第一个炸弹作为评分标准（**不要在爆炸太多次后试图通过换一颗炸弹来洗清罪恶**）
- 拒绝拖延症，尽量不要在最后几天再开始写
- （*无视去年有师兄师姐因为爆炸时显示的“BOOM！！！”焦虑的事实*）**本次实验为本门课程最欢乐的一个，祝大家实验愉快！！！！**