Bomb Lab

30% of lab grade, 15% of overall grade
out: 10/12 in class; in: 10/31 at midnight

The purpose of this lab is to familiarize yourself with machine-level programs and the tools that you can use to understand them. You will do this by defusing a Binary Bomb produced by Dr. Evil, a nefarious Canadian. Each of you will have your own different Bomb to defuse. Bombs consist of a sequence of phases, each of which is diffused by typing a special string (a passphrase). When you diffuse a phase, the bomb will notify Dr. Evil and go to the next phase. If you type the wrong passphrase, the bomb will explode and notify Dr. Evil. Dr. Evil will keep a scoreboard showing, for each bomb, the number of times it has exploded and the maximum number of phases that have been defused. Once all the phases have been diffused the bomb is diffused and you are done with the lab.

What is a bomb?

A bomb is a Linux executable that will only run on a TLAB machine, although you may examine it on any machine. The bomb executable is accompanied with the "bomb driver" source code, which should provide some insight into its operation. A bomb has 6 phases (or maybe more?), each of which is diffused by entering a string. The string for a phase always remains the same – once you've defused a phase, you shouldn't detonate the bomb at that phase again, except accidentally.

Phases get progressively more difficult and phase 6 is intended to be a challenge.

How do I get my bomb?

To get your bomb, send mail to ics2001@grayling.cs.nwu.edu with the following subject:

```
bomb request fall01 your_email_address your_partners_email_address
```

You may work in groups of two or individually. If you are working individually, then just leave the partner email address blank. Dr. Evil will accept your request and email back to you a uniquely numbered custom bomb. He'll keep track of which bomb belongs to whom.

How do I know how I'm doing compared with others in the class?

Dr. Evil will report on the status of all outstanding bombs on the scoreboard web page http://grayling.cs.nwu.edu/bombstats.html

How will this be graded?

Each phase will be worth 10 points, for a total of 60 points. Each time your bomb explodes at a particular phase, you'll lose ¼ of a point. The most you can lose is 10 points per phase.

There is nothing to be handed in. At midnight on the hand-in date, Dr. Evil will freeze his scoreboard and assign grades based on it.

Hints

There are two critical things to realize about this lab. First, you can carefully examine your bomb without running it by using various tools. As you do this, you'll be forced to become familiar with executable file formats, object code, and assembly. Second, you can run your bomb under a debugger and carefully watch what it is doing at the machine level. As side effect, you'll become familiar with debuggers, an essential tool in any programmers toolkit, and you'll develop a feel for what happens when a program executes by observing one in vivo.

Please don't attempt a brute-force passphrase search. You'll lose ¼ point for each miss. You don't know how long the passphrases are, and Dr. Evil assures you that they are long enough that you would run out of points much more quickly than you could ever find the passphrase through a brute-force search.

Here are tools that you will find useful in this lab. You are not limited to these tools.

- man ascii: this will display the list of ASCII characters.
- strings: strings will simply display all of the printable strings in the bomb.
- objdump: objdump lets you examine an executable file in considerable detail. Two options you will find invaluable are –t, which will print out the bomb's symbol table. The function names of the bomb will be in the symbol table and you may learn something from them. The second option is –d, which will disassemble the code in the bomb, letting you take a look at it at the assembly level. You can learn more via "man objdump"
- gdb: gdb is a powerful command-line debugger that's available everywhere. You can use gdb to run the bomb line by line or instruction by instruction. It will be able to show you the contents of memory and registers as the bomb runs. You can use it to set breakpoints which will make the bomb stop running at particular places and you can ask it to watch for when memory values change. You can even script it. Here are some additional tips about gdb:
    - You can also run it with a slightly nicer interface under emacs or xemacs via M-x gdb [return].
    - The gdb manual and a one page cheat sheet are available on the course web page
    - You can get help from the gdb prompt by typing "help"