The University of Queensland
School of Information Technology and Electrical Engineering

CSSE2310/CSSE7231 — Semester 1, 2023
Assignment 2 (version 1.0)

Marks: 50
Weighting: 10%
Due: 6:00pm 6 April, 2023

Introduction

The goal of this assignment is to ensure you have gained familiarity and skills with both the C programming language and using a debugger (such as `gdb(1)`) to examine various characteristics of running programs. These will be essential skills in other assignments for this course. For this assignment you will be given an executable program (the "bomb") which you have to "defuse" by entering the correct defusing phrases into the program for each of the 10 bomb phases.

Student conduct

This is an individual assignment. You should work on defusing your own bomb by yourself. You should feel free to discuss aspects of C programming and the use of debuggers with your fellow students, but you shouldn't actively help (or seek help from) anyone with the defusing of particular phases. Do not share your approaches to solving the bomb phases – even after the assignment deadline (as extensions may be given to other students).

You should note that each student will receive a different bomb and the strings that defuse your bomb will be different to the strings that defuse another student's bomb.

In short – **don't risk it!** If you're having trouble, seek help early from a member of the teaching staff. Don't be tempted to cheat. You should read and understand the statements on student misconduct in the course profile and on the school website: `https://www.itee.uq.edu.au/itee-student-misconduct-including-plagiarism`

Obtaining the "Bomb"

While logged in to `moss.labs.eait.uq.edu.au`, you should type the following command:

```
getbomb
```

This will create a subdirectory within your current directory named `csse2310a2` and place the bomb files into that directory. The files will include an executable called `bomb` and a number of source files (.h and .c files). Your bomb (executable and source) will be different to the bombs for all other students. You will not receive all of the source files – just some of them – so you will not be able to build the bomb yourself. There is enough information contained within the bomb executable and the supplied source files in order for you to successfully defuse all phases (although some of them are more difficult than others). You should note that some of the bomb's modules have been compiled with debugging support (`-g` flag to `gcc`) and some haven't.

Running the "Bomb"

The bomb program will only run on moss.labs.eait.uq.edu.au and you are the only user who can run your bomb program. Any attempt to run the program on another host or to run another user's bomb will cause the bomb to exit immediately. While in your csse2310a2 directory, you can execute the bomb by typing

```
./bomb
```

You may not want to do this until you are ready to try defusing the bomb. When you start the bomb program, it will print out details of any phases you have already defused and it will print your current mark (out of 50) and the maximum mark you can obtain based on your attempts to date.

The bomb will then prompt you to enter the number of the phase to defuse next, followed by the string that you believe defuses that phase (or a test string). You will be prompted for confirmation before that string is tested. If you confirm your attempt and the string is incorrect then the bomb will "explode" and exit. If the

string is correct, then that phase is defused and you will not be able to solve it again. You will lose marks for every time the bomb "explodes".

The goal of this assignment is to enter the correct defusing strings for each phase, not to "hack" the bomb so that it thinks the phases have been defused. The bomb checks for attempts to tamper with the bomb and will "explode" if tampering is detected (e.g. if internal variables or data structures are modified in a way that causes the required defusing string to change). We also log all defusing attempts and will adjust marks afterwards (i.e. reduce your marks) if the bomb has not been properly defused.

Hints

There are two demo phases that do not count for marks. You may attempt these as many times as you like by entering either "demo1" or "demo2" when prompted for a phase to defuse. There is no mark penalty if either of these demo phases "explodes."

You should carefully read the supplied source code and be familiar with the use of `gdb` before attempting to run the bomb. It is suggested you run the bomb from within a debugger rather than standalone. Note that you may get a warning message about "`Missing separate debuginfos...`" when you run the bomb within `gdb` – you can safely ignore this message.

All phases have associated code and some debugging information and you will need to use a debugger to set breakpoints, examine various variables etc. in order to determine the defusing strings. You may need to learn about and use a number of features of gdb including (but possibly not limited to) watchpoints, automatic display, conditional breakpoints, and/or breakpoint command lists to solve the phases more efficiently.

You should note that the code that determines each defusing string is not executed until AFTER the defusing text is read from the user so you may need to enter some arbitrary text, debug the code to determine the defusing string, quit the program and then run it again to enter the defusing string for that phase.

The bomb is deterministic – the same sequence of inputs from program startup will result in the same operation each time, so the defusing string for each phase will not vary between runs of the bomb. **However**, many of the functions within the bomb are not deterministic – they may return something different each time they are called, and they may change internal data structures. This means that a defusing string may change over time within a single run of the bomb program if you call functions from within `gdb`. Your goal is to determine the defusing string that would have to be entered from a fresh run of the bomb – not the defusing string that might apply after you have perhaps modified the internal state of the bomb by using `gdb` commands.

Submission

Every time you run the bomb, a record is kept of your interactions with it and your success/failure at defusing each phase. Your submission time for the assignment will be considered to be the **time of your last attempt to defuse any phase of the bomb**. You must make at least one attempt in order to be considered to have made a submission. An attempt means either that the bomb explodes or a phase is defused.

Late penalties will apply as described in the CSSE2310/CSSE7231 course profile. Any attempt to defuse the bomb after the deadline will result in a late penalty being applied to your whole assignment mark.

Marks

There are 10 phases, each worth 5 marks. The mark you achieve for each phase is determined by the number of attempts taken before you successfully defuse that phase. If you do not defuse a phase you will receive zero marks for that phase. If you defuse a phase on the first attempt, you will receive 5 marks for that phase. If it takes you more than one attempt, your mark for that phase will be

$$4 \times 0.75^{(number\_of\_attempts-2)}$$

i.e. if it takes you 2 attempts, your mark for that phase will be 4 out of 5; 3 attempts gives you 3 out of 5; 4 attempts gives you 2.25 out of 5; etc. There is no limit on the number of attempts you can make at any phase before succeeding. You should note that although each phase is worth the same number of marks, they are not of equal difficulty. All marks are subject to an audit of our logs to ensure that you have correctly entered the defusing strings and haven't tampered with the bomb to defuse it in some other way. If you have not defused a phase with the expected defusing string then you will receive zero marks for that phase.