COMP 3000A: Operating Systems

Carleton University
Fall 2022 Midterm Exam

October 13, 2022

There are 14 questions on **2 pages** worth a total of 30 points. Answer all questions in the supplied text file template (available on Brightspace, titled `comp3000-midterm-template.txt`). Please do not corrupt the template as we will use scripts to divide up questions amongst graders. Your uploaded file should be titled `comp3000-midterm-username.txt`, replacing username with your MyCarletonOne username. Make sure you submit a text file and not another format (such as MS Word or PDF). Use a code editor, not a word processor! (You won't be graded for spelling or grammar, just try to make it clear.)

This test is open book. You may use your notes, course materials, the course textbook, and other online resources. If you use any outside sources during the exam, you **must cite** the sources. Your citation may be informal but should be unambiguous and specific (i.e., if you refered to the textbook, indicate what chapter and page you looked at rather than just citing the textbook). You **may not** collaborate with any others on this exam. This exam should represent your own work.

Do not share this exam or discuss it with others who have not taken it. Some students will be taking it at other times due to accommodations. Solutions will be released once everyone has finished th exam.

All explanations should be concise and to the point (generally no more than a few sentences, sometimes much less). If you find a question is ambiguous, explain your interpretation and answer the question accordingly.

You have 80 minutes. Good luck!

1. [2] Answer the following questions about x86-64 assembly language:

   (a) [1] What instruction is used to return from a function?

   (b) [1] What instruction is used to make a system call?

2. [2] On Linux, processes must make the `sbrk` or `mmap` system call to allocate memory from the kernel. Does a C program make one of these system calls every time it calls `malloc()`? How do you know?

3. [2] Why are calls to `fork()` normally followed by an if statement that tests the number it returns? Explain briefly.

4. [2] What is a C statement or declaration that could have generated the following assembly language code? Explain how each line is accounted for in your C code.

```
.LC0            .string  "fox"
.LC1            .string  "chicken"
.LC2            .string  "wolf
animals         .quad    .LC0
                .quad    .LC1
                .quad    .LC2
                .quad    0
```

5. [4] Consider the following x86-64 assembly code and C code.
   Assembly code:

```
cmpl %edi, the_number(%rip)
je .L7
ret
```

   C code:

```
#include <stdio.h>

extern int the_number;

void check_guess(int g)
{
        if (g == the_number) {
                puts("Got it!\n");
        }
}
```

   (a) [2] What part of the C code does this assembly code implement? Be specific.

   (b) [2] Do you think `the_number` appears elsewhere in the assembly code for this C code? Why? Explain briefly.

6. [2] How could you execve `/bin/ls`, giving it the command line argument of "-l"? Assume that you can use `environ` for the environment. Be sure to specify the exact arguments you would give to execve, defining any necessary data structures using C code.

7. [2] In 3000shell2.c, there are two calls to `wait()`. The one in `run_program()` can cause 3000shell2 to pause before continuing, while the one in `signal_handler()` will never cause 3000shell2 to pause. Why does a call to the same function potentially produce different behavior?

8. [2] On the current version of Ubuntu Linux, what system call does the `fork()` library call make? Could it make a different system call and still work? Explain briefly.

9. [2] Are the internal commands of bash the same as 3000shell2? Are the external commands the same? Explain briefly.

10. [2] What two things must you do make a program setuid root? Give each command along with a brief explanation of what it does.

11. [2] When a process creates a file, the file's owner is set to the filesystem uid of the creating process. The filesystem uid is normally set to be equal to the process's effective uid. If you make a program setuid root, will that change the default ownership of the files it creates? Explain.

12. [2] If 3000shell2 is setuid root, will every program it runs have root privileges? Why or why not?

13. [2] If you wanted to change a user's uid, what is one file that you would definitely have to change? Why?

14. [2] In bash, if I type "`ls > logfile`", what program opens `logfile`, bash or ls? Why?