COMP 3000: Operating Systems

Carleton University
Fall 2017 Mid-Term Exam

November 2, 2017

There are 12 questions **on two pages** worth a total of 20 points. Answer all questions in a separate answer booklet.

This midterm is closed book: no notes, books, computational aids, or other outside resources are allowed. 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. [1] What system call is used to load a program binary in Linux? Does this system call create a new process? (yes/no)

2. [1] Dynamically-linked programs generally make many mmap calls at the start of program execution that a statically-linked version of the program doesn't do. What are those mmap system calls for?

3. [1] Symbolic links map filenames to filenames. What do hard links map filenames to?

4. [1] In the producer/consumer problem, when the producer encounters a filled queue (i.e., there is no room for it to produce anymore), what should it do?

5. [2] Consider the following implementation of fill_rand_buffer():

   ```
   void fill_rand_buffer(rand_state *r)
   {
           ssize_t count;

           count = read(r->fd, (void *) &r->buffer,
                        BUFSIZE * sizeof(unsigned long));

           r->current = (count / sizeof(unsigned long)) - 1;
   }
   ```

   Explain the calculation of `r->current`. Why isn't it just set to the value of `count`?

6. [2] What are two standard uses of signals in Linux? (Do not list uses that depend upon application-specific behavior, e.g. SIGUSR1.)

7. [2] Do pointers in C contain virtual or physical addresses? Why?

8. [2] Does a process make a system call to allocate memory? Why?

9. [2] At the prompt of `shell`, a user types `something > output.txt`. Which program opens the file `output.txt`, `shell` or `something`? Why?

10. [2] 3000test used mmap to compare the contents of two files. Standard tools for comparing files, however, use read rather than mmap. How could you use read system calls to compare the contents of two files of arbitrary size? Does this approach have any advantages over using mmap?

11. [2] The `shared` struct in 3000pc.c is allocated using an mmap call with the MAP_SHARED flag. If this flag is changed to MAP_PRIVATE, how will the behavior of 3000pc.c change? Why?

12. [2] You are trying to port 3000pc to a system that does not provide any implementation of sem_wait() or sem_post(). You search online and you find the following implementation of sem_wait():

```
int sem_wait(int sem)
{
        while (sem < 0) {
           /* wait */
        }

        sem--;
        return sem;
}
```

Does this code implement a correct semaphore? Why or why not? Explain briefly.