COMP3430 Winter 2018

# Assignment 1 - Daniel Lovegrove - 7763168

1. After running **ps xf** remotely on the lab machine, I see that the shell I'm using is TCSH, with process id 11080. Following back to pid 1:

| PID | NAME |
|---|---|
| 11080 | tcsh |
| 11079 | sshd |
| 11076 | sshd |
| 1053 | sshd |
| 1 | systemd |

It appears that at the base, systemd, the first process, has an SSH daemon (pid 1053) listening for connections. From the man page for SSHD, the SSH daemon is forked for each incoming connection, creating processes at 11076 and 11079. Then, a TCSH shell is opened with pid 11080 for communicating with the user.

Systemd has a parent process id 0, but there is no actual process with pid 0. This is because systemd is forked from the swapper, which is a direct part of the kernel and so is not a user process, hence why there is no proc/0.

2. Making an exec call within a process overwrites the entire address space of the process and runs the code that is to be executed. It does not make sense to make an exec call in a multi-threaded application because when one thread makes an exec call the whole program with all of its threads is stopped and overwritten to run the new program. If two threads make a call to exec there will be a race condition, since the same address space will be trying to be overwritten for two different programs, the outcome of which will be undefined and likely crash the code.

3. From the man pages for strtok, strtok uses a static buffer while parsing. So, if there are multiple threads running, each using strtok, they will all be using the same buffer and a race condition is created since we don't know which thread is accessing which part of the single static buffer at any time. Strtok_r overcomes this by keeping separate pointers that may point to the same string. Since strtok_r just uses pointers to a string, there is no issue of threads interfering with each other in regards to which part of the string they are processing.

4, 5, 6, 7 are coding questions. See the README.md document to see how to run the code.