Project 2 Grading Rubric

Tyler Allen

February 16, 2018

1. Readme for the documentation and design (10%)
2. The actual design and implementation. Correctness, elegance, and efficiency (60%)
   (a) Nullptr was roughly 20%/60%.
   (b) Shmem was roughly 40%/60%.
3. Test driver (20%)
4. Follow the design and submission instruction (10%)

Every grade has a comment next to it on Canvas. The comment is four numbers. These four numbers indicate how many points you received out of each category. Therefore, if you received a score of 100, then your comment looks like this:

```
10 60 20 10
```

I will only respond to questions about individual mistakes in-person during office hours.

These are a list of the most common mistakes, which also are mistakes that tend to be worth the most points. **Most Common Issues**:

- Fork, Exec, Fork/Exec, Multi-Fork-Exec, etc... Issues: Many people had issues with fork/exec, especially multiple forks or execs. Typically it ended with the equivalent of a segmentation violation. Sometimes parents could write but not children, etc.
- Reference Counter Issues: Some projects had issues tabulating the reference counts. Many people just did not bother decrementing their reference counts.
- Insufficient Testing: Testing only the simplest cases costs points. If the tests were sufficient, the above two cases would have been caught.
- "kfree" Kernel Panic: Projects that received this had issues where processes would die, and the OS would automatically free their shared memory. After the death of two processes, the kernel would panic in a double-free. A perfectly reasonable solution is allocating the memory during OS init and never de-allocating it until system reboot. If you were having this issue, it was *not* your only issue - see the rest of this section. I fixed it and continued to grade.

- The projects this time were graded against my tester in addition to your own. This means that if you strayed from the project description, your project was harder to grade and therefore it was harder for me to evaluate the severity of mistakes. Please follow the project descriptions for your benefit and mine. This includes **not using linux headers.** This is not necessary and causes issues with my build system. If you want `NULL` and it isn't defined, define it. The `NULL` you are importing is part of the GNU/Linux standard library, not the XV6 standard library. Attempting to port unmodified code from other systems can cause all sorts of issues.
- Minor Issues (point-wise): Some points were taken if checks were not added for `argptr` address space and similar minor issues.