Dawit Woldegiorgis
COP 4600
Section: 0467
November 17, 2015
Project: Part #4

Learning Experience:

The most significant thing I learnt from this assignment was how syscalls are carried out and managed. Furthermore, I also learnt the basic steps required to compile and install a modified kernel.

Understanding and trying to decipher the ways in which syscalls worked was a bit challenging. Upon initial encounter, I wasn't sure of how to approach the confusing and weird syntax. To tackle this, although not an intentional step, I read through some kernel syntax guidelines and source codes. The most challenging part of this process was trying to figure out what was relevant enough to carry out the most basic of steps, and not to have to learn every ins and outs of the kernel before taking the first step. Dave's example codes were helpful in this case, after figuring out some of the weirdness.

The easy (and fun) part of this assignment was converting the pseudocode algorithm provided by Dave, into an efficient and functional C code. Although there wasn't much thinking involved in how to map the algorithm, as it was already mapped out in psedocode, most of my effort in this section dealt with trying to figure out how to optimize the code, which took me back to the fun days of basic programming.

I did find that going through the steps to recompile the kernel, sometimes unnecessarily, was frustrating and time consuming. There were some tests that could not have been performed in user space alone as they originated when trying to adapt the code to kernel space. I did find the instructions at the bottom of the "Building an OpenBSD Kernel" but first time I read it, I simply skimmed through it as it was labeled a "Shortcut". Had this been on top of the page, I might have saved myself some time and frustration of trying to change the makefile without need.

Suggestion:

Put the "read this first" section at the top of the website. That way, the likelihood of it being read first increases dramatically(?)

Honesty Pledge:

On my honor, I have neither given nor received unauthorized aid in doing this assignment.

PROGRAM INFORMATION

Dawit Woldegiorgis
COP 4600
Section: 0467
November 17, 2015
Project: Part #4

Q - Does the program compile without errors?
A – Yes, the program compiles without errors.

Q - Does the program compile without warnings?
A – Yes, the program compiles without warnings.

Q - Does the program run without crashing?
A – Yes, the program runs without crashing. This claim is supported by the fact that no test runs had crashed the program, nor is it expected to crash under normal conditions.

Q - Describe how you tested the program.
A – The program was tested for correctness in both user space and kernel space. The algorithm implemented was initially tested using a dummy test case before being integrated into the kernel. After confirming that the algorithm works fine, it was then adapted to work with the kernel, and tested robustly using a more integrated test. Test cases consisted of edge cases, “random” texts and texts with certain patterns. True randomness was avoided in the test case to ensure that correctness is guaranteed in all cases.

Q - Describe the ways in which the program does not meet assignment's specifications.
A – The program meets all specifications provided in the assignment.

Q - Describe all known and suspected bugs.
A – There are no known or suspected bugs in the program.

Q - Does the program run correctly?
A – The program runs correctly, as all test cases generated gave out expected results.