Brett DeWitt
CS 415 - Malony, Allen
10/17/2024

Project 1 Report

Unix shells are the focal point of this project. Implementing the pseudo-shell advances the courses learning goals of understanding OS structure and system calls, including computer organization, describing OS services provided to user's processes, and other systems.

A number of UNIX commands have been re-implemented in the pseudo-shell using system calls to the kernel, just as a proper operating system shell would. Certain commands, such as PWD and CD are entirely straightforward, essentially wrapping the system call and directing the output to STDOUT. Other commands such as LS were slightly more technical, requiring the use of a struct supplied by a header file facilitating the reading of directories and their properties. CAT required use of the open() system call in order to read the contents of a file, creating a buffer in which to read the bytes and directing them to STDOUT. CP and MV were the most challenging because of the requirement that multiple cases be considered when parsing the source and destination arguments. The system calls do not accomplish these tasks implicitly. Instead, functions such as stat() must be used to determine whether the provided paths are files or directories and, if necessary, construct a new path to pass to write().

Implementing the design gave me a better understanding of how much of the work of basic shell operations belong to the kernel, and how much is abstracted to the shell. Much more of the work is done by the kernel implicitly than one might expect. However, some commands that are easily taken for granted are made much easier to use by the shell's implementation. Being able to pass a directory as a destination for CP or MV has likely saved a lot of typing and typos over its many invocations. Making typos with filenames where writing is involved can potentially damage sensitive data, so it is important to carefully consider the implications of any particular implementation before setting it loose

on users. It will be interesting to see how processes and threads are created and assigned to system resources through the same sort of practice and implementation.