Thomas Chung

COEN 177 Assignment 2

In this lab we modified the typical priority scheduling algorithm that Minix uses for processes by introducing a small degree of randomness. Minix's priority scheduling is implemented by iterating each scheduling queue in the pick_proc function from highest priority to lowest priority while checking for their ready state. Then, the function assumes that the head of this selected queue is runnable and returns a pointer to the process. The lab requires a specific way for us to implement randomness in this priority scheduling algorithm. The requirement was as follow: for processes 0-6, choose a process as you normally do, but for processes 7-15 randomly choose a process to run. So to modify the scheduling algorithm, we went into "proc.c" which contains the scheduling code for Minix. We modified the pick_proc function so that when it iterates each scheduling queue, for queues 0- 6 it continues as it always does but when the iterator variable is greater than or equal to 7, it lets a variable 'i' take its place. The variable 'i' randomizes itself so that it takes on a value between 7-15. Then 'i' replaces all instances of where the iterator variable was throughout pick_proc.

I tested my code by recompiling the source code by calling "make world." Then I rebooted the VM to see if my scheduling algorithm works. I could tell that it worked because when it rebooted, the OS started a lot slower than it was. This is because typically a lottery system isn't as efficient as a priority selection (although it's not as if Minix's priority scheduling algorithim is that great.) If my code had been impractical, I would have tested and re-compiled my Minix source code through small additions over time. Minix's "Make World" crashes if it isn't capable of being compiled so this would be the easiest way for me to find mistakes. And of course, I understand that the compiling process itself actually takes time. But it's a risk I'm

willing to take. Besides Minix “Make World” crashing when it turns out the source code can’t be compiled, it also helps a lot to retain the original priority scheduling process for the higher level processes. This way, Minix can still run as normal upon boot even if you make a small change.