Unix Processes

Objective: Use of the UNIX ps command to examine the behavior of processes in UNIX

1. Download a copy of the files in the ProcLab folder of our class website’s files section (make sure that no extra extensions like .html are added to the files). You can download them directly to a folder on your Knoppix desktop, or put them into a separate folder on a diskette or your USB flash device. Start Knoppix, open a terminal window, and change directories to the folder you created. For example, if you downloaded the files into a folder called ProcLab on your Knoppix desktop, you would use the command:

   cd /home/knoppix/Desktop/ProcLab

2. Examine the contents of the ProcLab directory with the full directory listing command:

   ls -Fal

3. Run a shell script called create_executables.sh to create some executable programs by typing:

   sh create_executables.sh

   at the Unix prompt.

4. Examine the contents of the ProcLab directory again with the full directory listing command:

   ls -Fal

   What new files exist, and how are they different from the other files in the ProcLab directory?

5. Use the man command to read documentation about the ps command:

   man ps

   What code is used for each of the following system states?

   ______ Running or Runnable (ready state)
   ______ Waiting for an Event to Complete (blocked state)

   What character is added for each of the following situations?

   ______ The process is low priority (nice)
   ______ The process is in the foreground (visibly running in a window)

6. Get a listing of all the processes on the system that belong to the knoppix user:

   ps U knoppix

   Make a note of which programs are currently in the system.

7. Get a listing of all the processes on the system:

   ps aux

   Write the name of one of the processes not owned by the knoppix user. What user does own it?

8. Run the loop program in the background:

   loop &
9. Run the `loop` program again in the background, but do it so that it is lower priority (nice to other users):

   `nice loop &`

10. Do another `ps` command, and note how much processor time has been used by the two loop processes:

        ___________ The nice loop
        ___________ The not-nice loop

11. Open another terminal window, change to the ProcLab directory, and run the `do_some_input` program, but do not answer it’s question.

   `do_some_input`

12. Open another terminal window, change to the ProcLab directory, and run the `loud_loop` program.

   `loud_loop`

13. In your original terminal window, get a listing of all the processes on the system that belong to the knoppix user:

   `ps U knoppix`

   How much processor time has now been used by the two loop processes?

        ___________ The nice loop
        ___________ The not-nice loop

   Why is there a difference?

   What other new processes are in the system?  What is the status (running, ready, sleeping, etc. – NOT JUST CODES!) of these new processes, as well as of the loop processes?

14. Type `ps` again. The infinite loop processes are still alive. Obtain the process ids corresponding to them, and kill them using the command `kill <pid>`

   What are the process ID’s of the two loop processes?

        ___________ The nice loop
        ___________ The not-nice loop
        ___________ The loud loop

15. Answer the question in the `do_some_input` program, and let it terminate. Kill the loop processes using the command `kill <pid>`. Do another `ps` command to make sure they are dead.

   **Submit results**

   Turn in this completed lab handout. It is due at the beginning of class on Monday, February 12, 2007, and is the second part of Assignment #1.