CIT 371 Lab 8: Managing Processes

Some parts of this lab must be done with the Web Console. See the VM Access Document for more information on accessing your VMs.

We continue our look at managing processes. Log into Coivcenter, start your VM and log in as Student. Open two terminal windows.

1. Using ps
   1. Start top in your first terminal window and leave it running. In your second terminal window, type **ps**. *What is the result?* Make a note of the terminal number (it will appear under the TTY column and look something like pts/0 or pts/1). The ps program provides a snapshot of the running processes. ps with no options only tells you the processes running in the current window. Switch to the window running top, type **control+z** to suspend it and type **ps** in the same window. *How does this output differ from the previous ps?* Make a note of the TTY number for this terminal. Type **fg** and go back to the other window. One option, -t, allows you to specify the terminal that you want to use ps on. Issue the command ps –t *#*, *#* where #, # are the two numbers you noted earlier, such as **ps –t 0,1** (but use the two numbers you noted earlier). You should see both bash sessions, the ps command and top. Now type **ps a**. You not only get more processes but more detail on the two processes. *What extra detail are you given? What tty’s are the other processes associated with? Research ps’s man page to determine what the a option does and describe it here.*
      1. **The result of this are the bash and the ps with titles of PID, TTY, TIME, and CMD. What is different is the top process is in this one. The extra details that are given are Stat and then it also shows more processes. Tty 2 is listed first then goes to tty3, 4, 5, 6, and then tty1. The -a option when looked up says that it will print all process that are in the terminal.**
   2. Another variation of ps is ps x. This version shows all processes, not just those that correspond to a terminal (tty). Type ps x and count the number of processes. How many are there? (note: use **ps x | wc –l** to get an easy count). Another variation is ps u, which shows “user format”. The output gives you more detail. Compare **ps u** and **ps ux**. *What new fields appear for the “x” version?*  The command **ps aux** applies a, u and x all together to give you all processes in user format.
      1. **The new fields in the x version compared to the u is the Stat. The ps u command has the %cpu, %mem, VSZ, RRS and much more compared to the ps x.**
   3. Another useful variation of ps is to use f to denote, using \ the parent child relationships between processes. Type **ps xf** to list all processes and their relationships. *Describe the parents of top* (that is, what is top’s parent, what is its parent’s parent, what is its parent’s parent’s parent, all the way up to gnome-terminal). You can exit out of top now (type **q** in the window running top).
      1. **The list of parents I see is /usr/bin/ gnome-terminal**
   4. Again, review ps’s man page under the entry for option u (ps u). *What do VSZ, RSS and STAT mean?* For STAT, you may find processes listed as Ss+, Ss, R+, S, Sl, and Ssl. *What do each of these mean?*
      1. **VSZ means virtual memory size of the process in KiB. RSS means resident set size. STAT means multi-character process state. Ss+ means that interruptible sleep (waiting for an event to complete) and is a session leader that is in the foreground process group. Ss means that interrupted sleep (waiting for an event to complete) and is a session leader. R+ means running or runnable in the foreground process. S means interruptible sleep. Sl means interruptible sleep and is in a multi thread. Finally, Ssl means that interruptible sleep (waiting for an event to complete) and is a session leader who has multiple threads.**
2. There are two GUI resource monitor programs (one from Gnome, one from KDE), both are found under **Applications** 🡪 **System Tools**. Start them both. They are somewhat similar to Windows Task Manager. *Compare the two resource monitor programs in terms of features that are similar and features that are different. Which would you prefer to use, if either?* 
   1. **Comparing the two features, one uses graphs and live feedback of CPU history, memory and swap history, and network history. Looking into this system it looks very appealing to the eye using the GUI. The other resource monitor is broken down into a list stating specific username, cpu %, memory, shared memory, etc. I would use the first system monitor because of the live feedback and the graphs that help you read and understand more in real time.**
3. Process priority – Linux runs processes in a multitasking way. You can control the number of cycles the CPU spends on a given process before moving onto the next by altering the process’ priority. In Linux, we refer to priorities as niceness. The nicer a process (the higher its niceness), the lower its priority because it is nicely giving away cycles. As a user, you can raise a process’ niceness value but not lower one (only root can).
4. Type **ps axl** (lower case L). The 6th column is the niceness value. *How many processes do not have a niceness of 0?* To change the niceness value, launch the process using **nice –n *value command*** where *command* is the command you are executing. Type **nice –n 10 find ~ -name “.txt”**. Try the same command with a niceness of -10. It still runs find but without changing its nice value. *What message do you get (look at the top of the output).*
   * 1. **There are 7 niceness values that are not 0 and then there are 6 niceness values that have a –. The message I received was nice: cannot set niceness: permission denied.**
5. You can also change the niceness value of a running process using renice. In one of your terminal windows run top and change it to show only Student’s processes, make a note of its PID (process ID). Type **renice –n 5 -p *PID*** where *PID* is the PID of top. *What does this command do?* *What message did you receive?* You may exit top.
   * 1. **This command reniced the 3061 student process id from 0 to 5. The message I got was 3861: old priority 0, new priority 5.**
6. Killing processes with kill and killall, usually kill. The syntax of kill –s *SIGNAL* *PID*. SIGNAL is the type of kill command. We usually use –s 9, which is the highest level of “killing”.
7. In one window, start top, obtain it’s PID. In the other window, type **kill –s 9 *PID*** where *PID* is the PID of top. Restart top, make a note of its PID. Kill top again but this time use a signal of 1 instead of 9. *How does the result differ? What does signal 1 represent versus signal 9?* (see page 149, NOTE: to see all of the available signals, go to man7.org/linux/man-pages/man7/signal.7.html). Run the **ps –ef|grep bash** command **to** obtain the PID of one bash session. Kill it. *What command did you enter? What happened?* 
   1. **When you kill -s 9 you kill the whole program, it doesn’t allow you to type or anything anymore. The kill -s 1 will stop the top and will prompt you with a new command line in the terminal to keep typing. Signal 1 represents hang up, which will end the running process but keep you in the terminal. Signal 9 represents kill, meaning this will kill the whole program. I used the command kill -s 9 3630. This exited out of the terminal and killed my program.**
8. From your remaining terminal window, open two xterm windows (using &) by using the command **xterm & xterm &**. Enter the command **killall xterm**. *What happens?* Open two more xterms. Now type **killall –i xterm**. *What is the difference?* You can also issue **killall –i –u *username*** to iterate every process owned by that user and ask you if you want to kill it or not. *Why might a system administrator use killall?*
   1. **This killed all the xterm windows that we opened. This command allows you to type y/N to manually close the xterms. A system administrator might use killall when he/she is permanently done with what they are doing.**

Shut down your VM if desired, and disconnect from the VPN if you are using it. Submit your lab report.