Mawlana Bhashani Science and Technology University

Lab -Report

Report No:06
Course code: ICT-3110
Course title: Operating System Lab
Date of Performance:19-08-2020
Date of Submission:26-08-2020

Submitted by

Name: Ruku Shikder
ID: IT-18057
3th year 1st semester
Session: 2017-2018
Dept. of ICT
MBSTU.

Submitted To

Nazrul Islam
Assistant Professor
Dept. of ICT
MBSTU.

Experiment No :06
Experiment Name: Linux command for process.

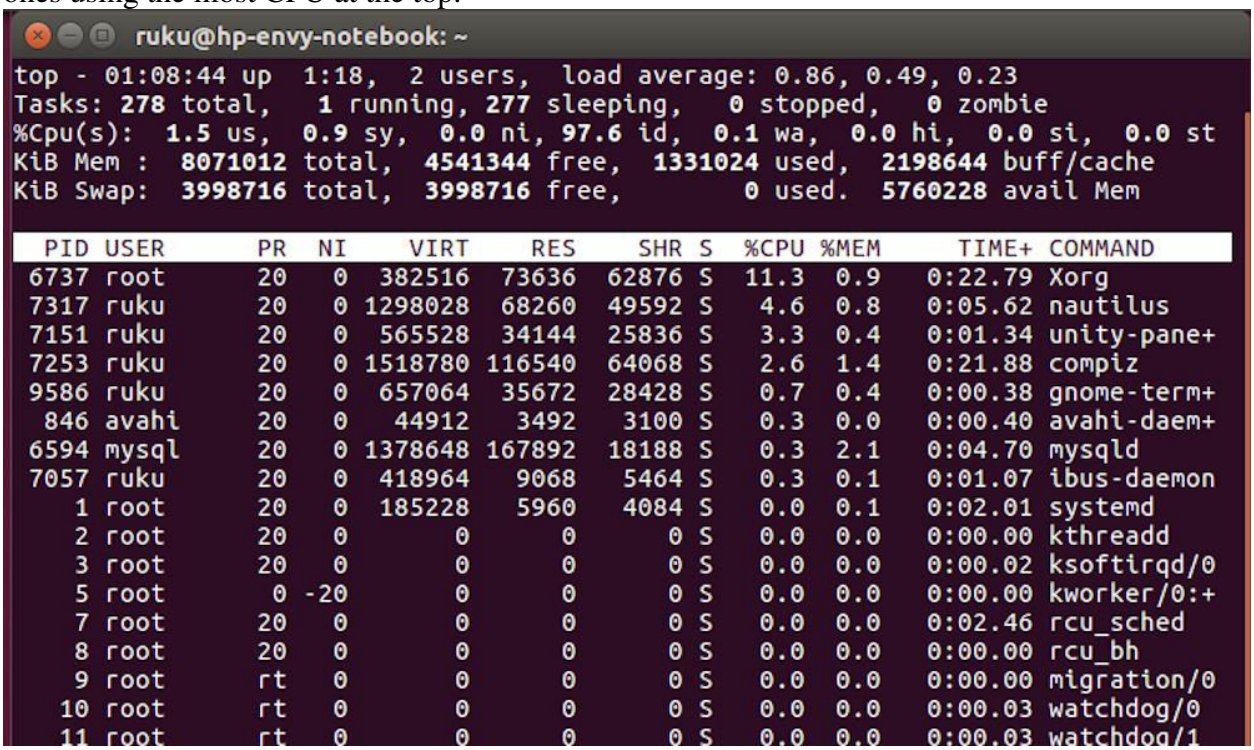
- How to Manage Processes from the Linux Terminal?
- Run the following process commands in Linux.
    - Top, htop, Ps, pstree, kill, pgrep, pkill ,killall, renice, xkill,

Manage Processes from the linux Terminal:

The procedure to monitor the running process in Linux using the command line is as follows:

- Open the terminal window on Linux
- For remote Linux server use the ssh command for log in purpose
- Type the ps aux command to see all running process in Linux
- Alternatively, you can issue the top command or htop command to view running process in Linux

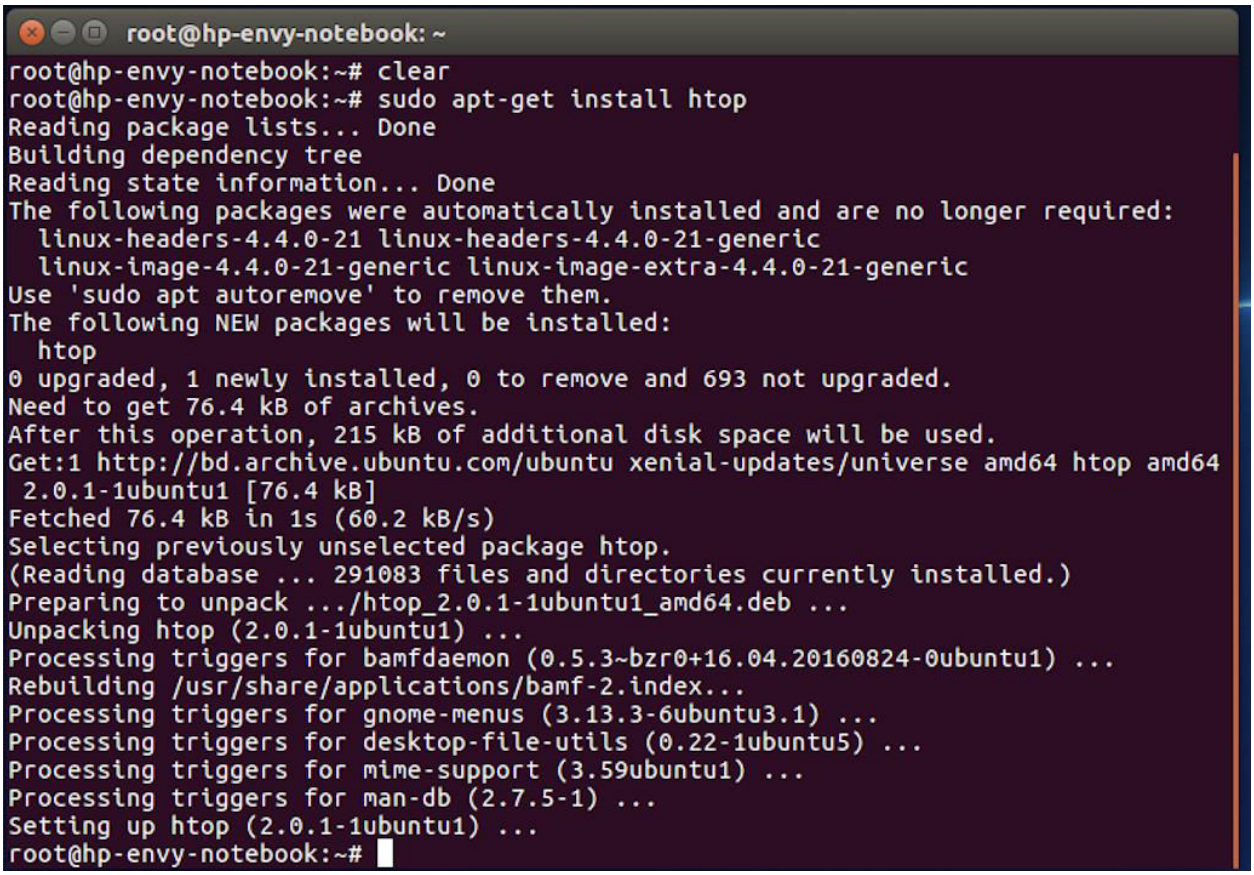
top: The top command is the traditional way to view your system's resource usage and see the processes that are taking up the most system resources. Top displays a list of processes, with the ones using the most CPU at the top.

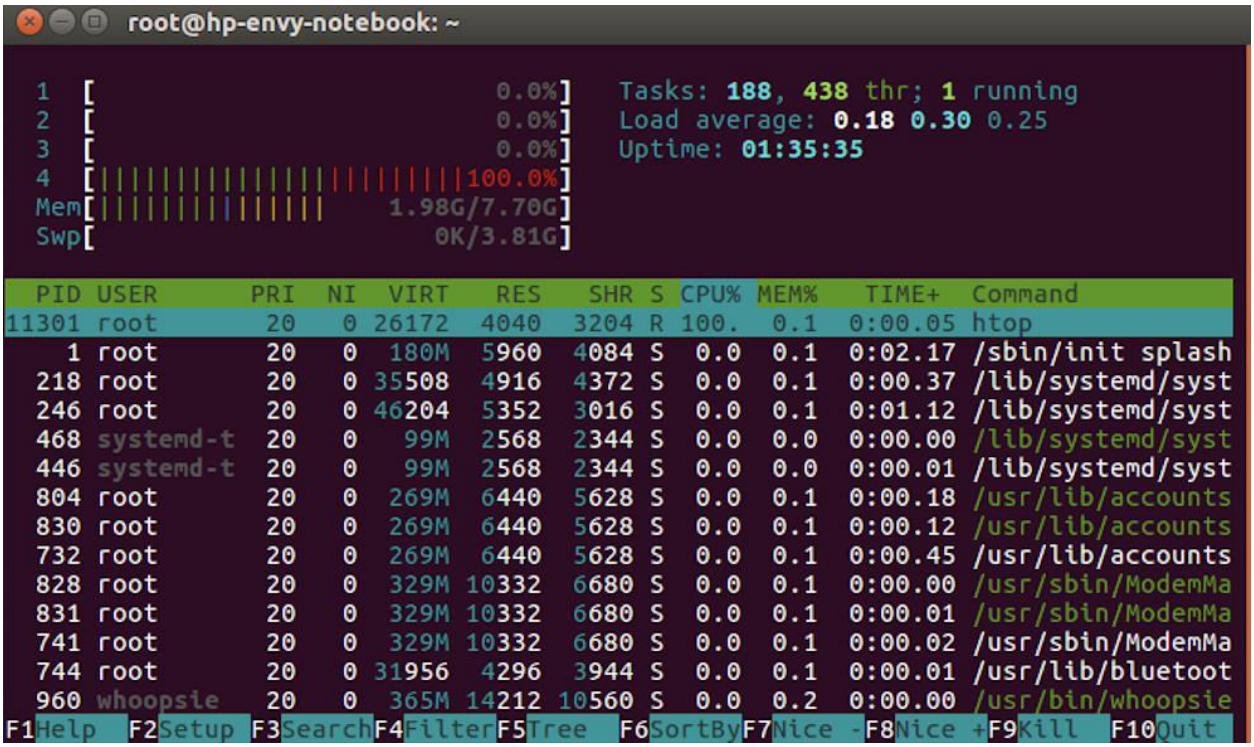


To exit top or htop, use the **Ctrl-C** keyboard shortcut. This keyboard shortcut usually kills the currently running process in the terminal.

htop: The **htop** command is an improved top. It's not installed by default on most Linux distributions — here's the command you'll need to install it on Ubuntu:

sudo apt-get install htop

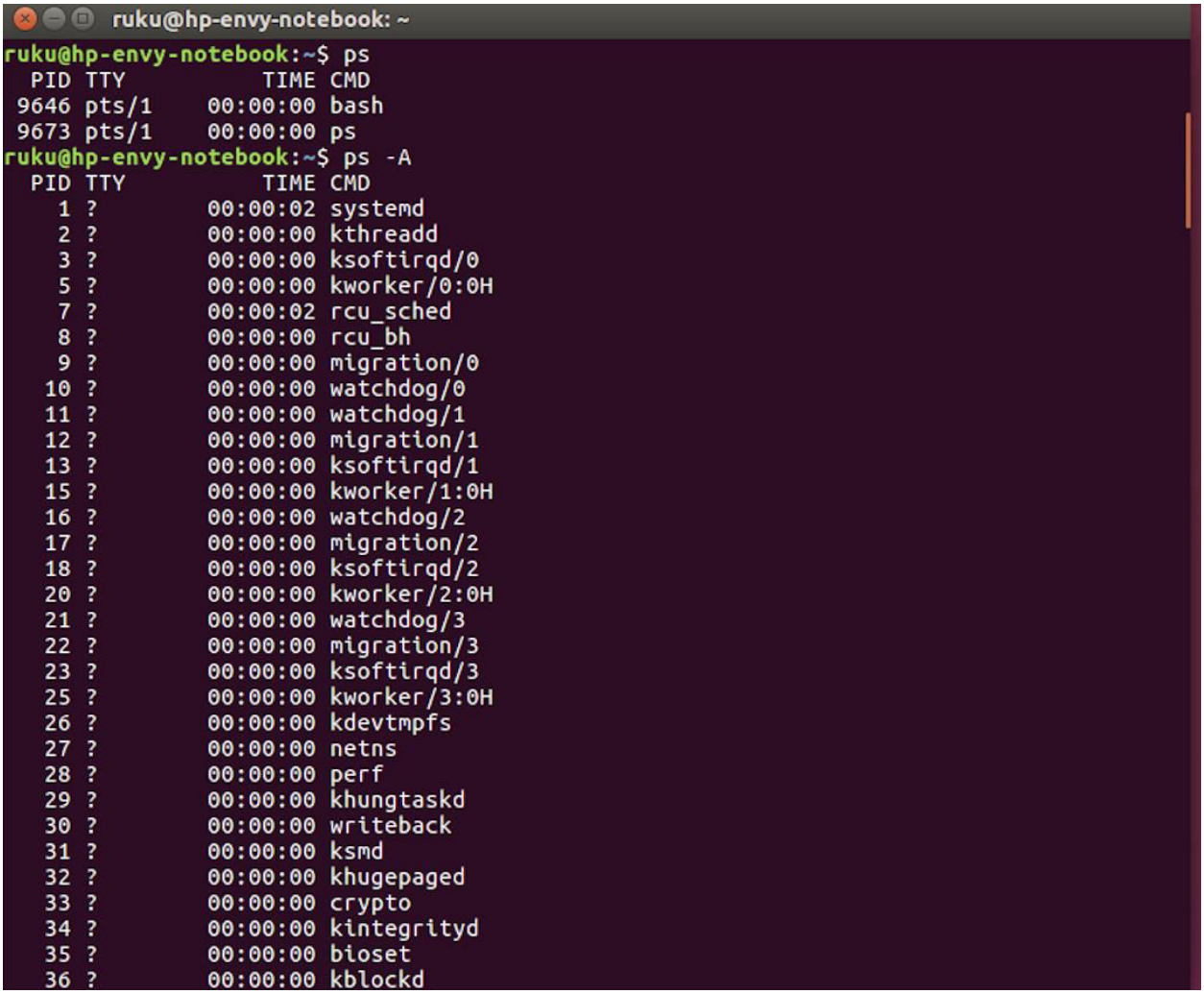



root@hp-envy-notebook: ~
1 [0.0%]
2 [0.0%]
3 [0.0%]
4 [||||||||||||||||||||||100.0%]
Mem[|||||||||||||| 1.98G/7.70G]
Swp[0K/3.81G]
Tasks: 188, 438 thr; 1 running
Load average: 0.18 0.30 0.25
Uptime: 01:35:35
PID USER PRI NI VIRT RES SHR S CPU% MEM% TIME+ Command
11301 root 20 0 26172 4040 3204 R 100. 0.1 0:00.05 htop
1 root 20 0 180M 5960 4084 S 0.0 0.1 0:02.17 /sbin/init splash
218 root 20 0 35508 4916 4372 S 0.0 0.1 0:00.37 /lib/systemd/syst
246 root 20 0 46204 5352 3016 S 0.0 0.1 0:01.12 /lib/systemd/syst
468 systemd-t 20 0 99M 2568 2344 S 0.0 0.0 0:00.00 /lib/systemd/syst
446 systemd-t 20 0 99M 2568 2344 S 0.0 0.0 0:00.01 /lib/systemd/syst
804 root 20 0 269M 6440 5628 S 0.0 0.1 0:00.18 /usr/lib/accounts
830 root 20 0 269M 6440 5628 S 0.0 0.1 0:00.12 /usr/lib/accounts
732 root 20 0 269M 6440 5628 S 0.0 0.1 0:00.45 /usr/lib/accounts
828 root 20 0 329M 10332 6680 S 0.0 0.1 0:00.00 /usr/sbin/ModemMa
831 root 20 0 329M 10332 6680 S 0.0 0.1 0:00.01 /usr/sbin/ModemMa
741 root 20 0 329M 10332 6680 S 0.0 0.1 0:00.02 /usr/sbin/ModemMa
744 root 20 0 31956 4296 3944 S 0.0 0.1 0:00.01 /usr/lib/bluetoot
960 whoopsie 20 0 365M 14212 10560 S 0.0 0.2 0:00.00 /usr/bin/whoopsie
F1Help F2Setup F3SearchF4FilterF5Tree F6SortByF7Nice -F8Nice +F9Kill F10Quit

htop displays the same information with an easier-to-understand layout. It also lets you select processes with the arrow keys and perform actions.

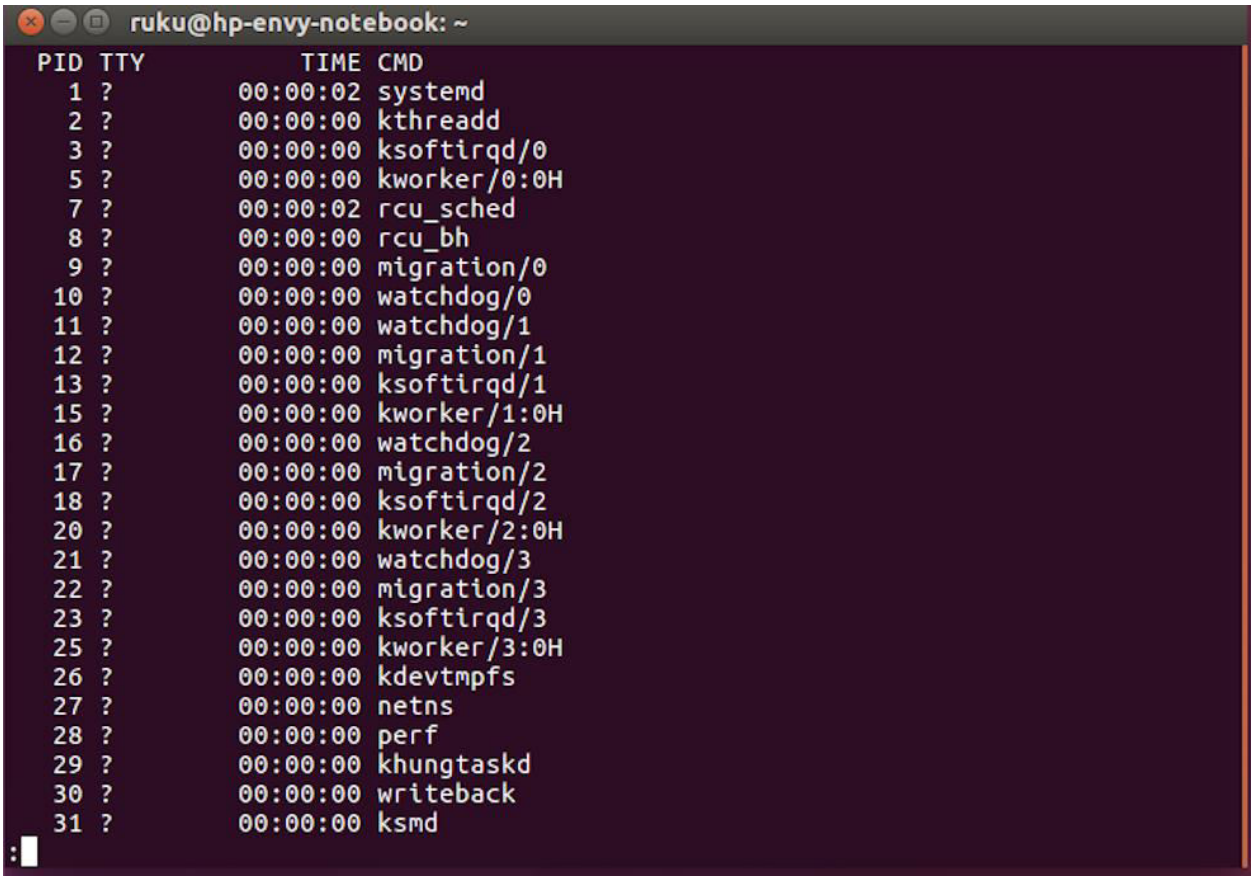
ps: The **ps** command lists running processes. The following command lists all processes running on your system:

ps -A



This may be too many processes to read at one time, so you can pipe the output through the **less** command to scroll through them at your own pace

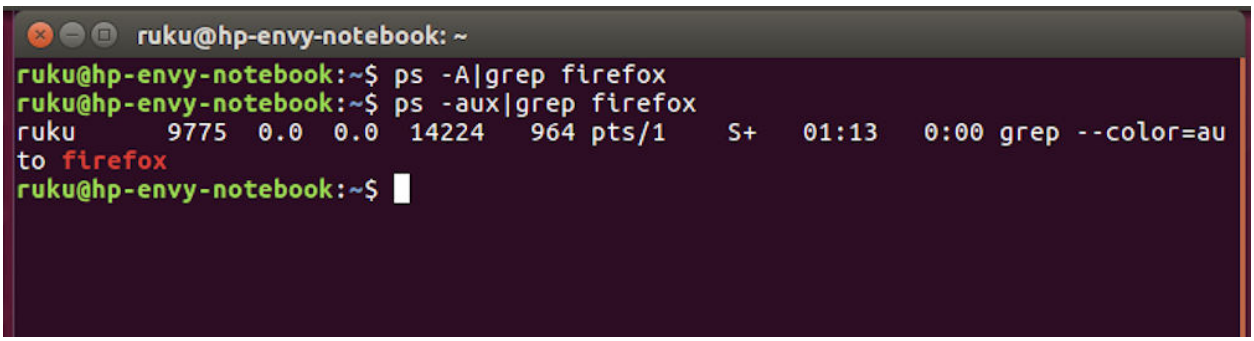
ps -A | less



Press **q** to exit when you're done.

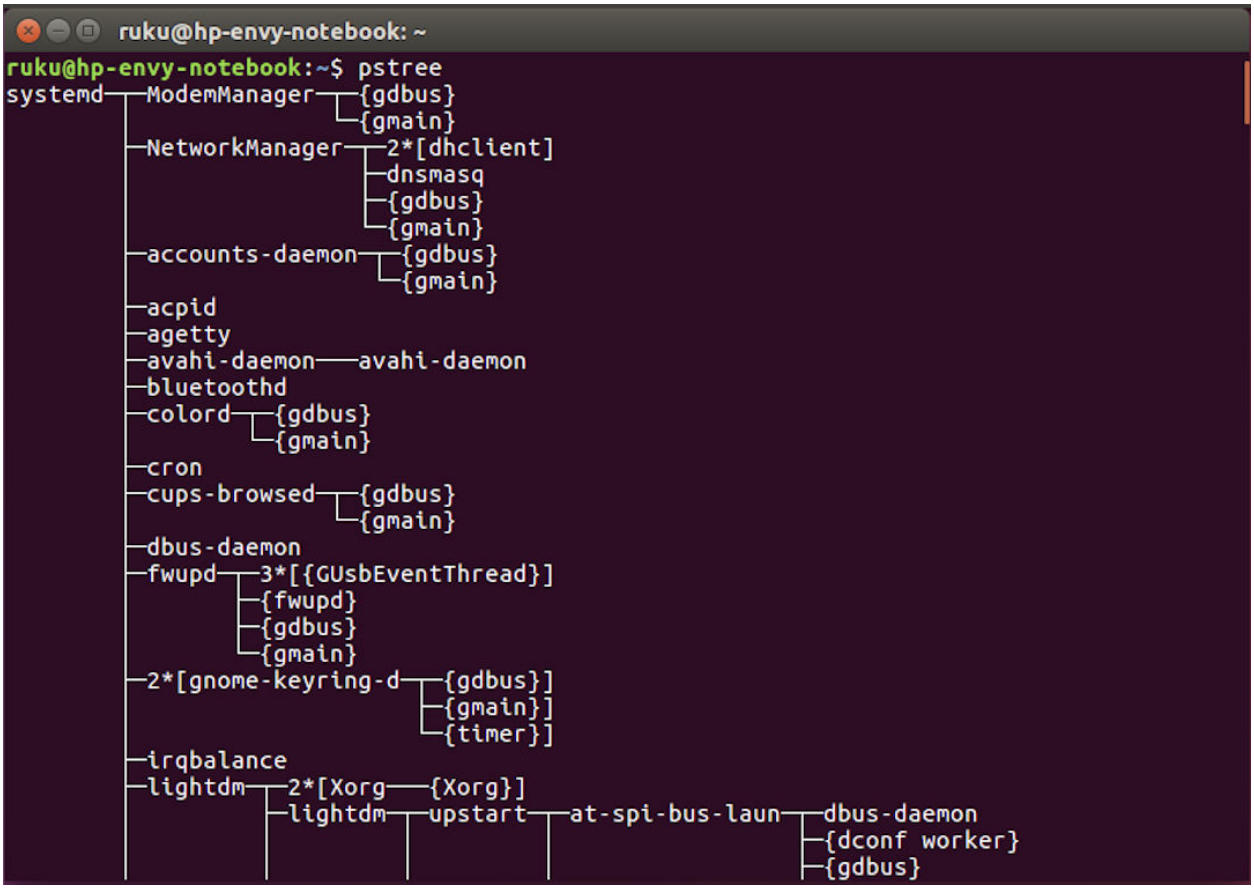
You could also pipe the output through **grep** to search for a specific process without using any other commands. The following command would search for the Firefox process:

ps -A | grep firefox



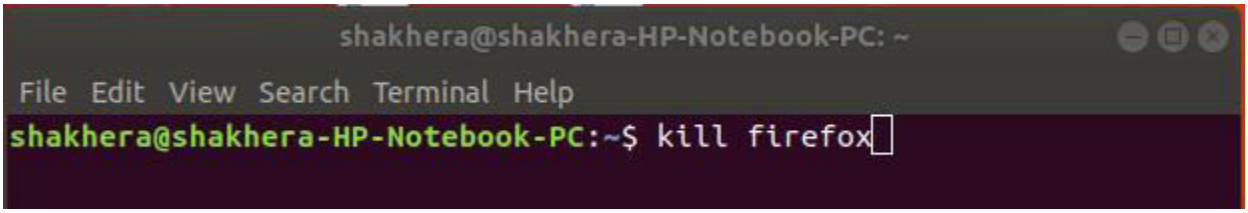
pstree: The **pstree** command is another way of visualizing processes. It displays them in tree format. So, for example, your X server and graphical environment would appear under the display manager that spawned them.

pstree

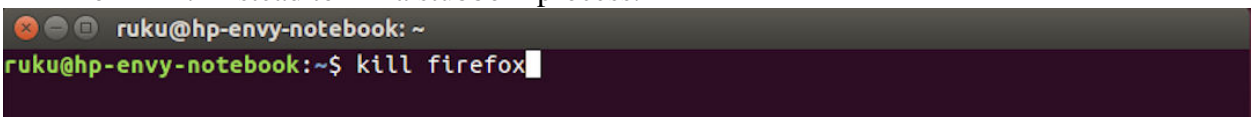


kill: The **kill** command can kill a process, given its process ID. You can get this information from the **ps -A**, **top** or **pgrep** commands.

kill PID



Technically speaking, the kill command can send any signal to a process. You can use **kill -KILL** or **kill -9** instead to kill a stubborn process.



pgrep: Given a search term, **pgrep** returns the process IDs that match it. For example, you could use the following command to find Firefox's PID:

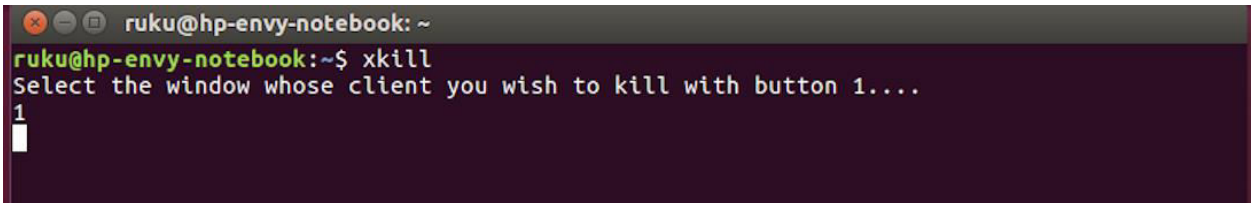
pgrep firefox

pkill & killall: The **pkill** and **killall** commands can kill a process, given its name.

Use either command to kill Firefox:

pkill firefox
killall firefox

xkill: The **xkill** command is a way of easily killing graphical programs. Run it and your cursor will turn into an **x** sign. Click a program's window to kill that program. If you don't want to kill a program, you can back out of xkill by right-clicking instead.



You don't have to run this command from a terminal — you can also press Alt-F2, type **xkill** and press Enter to use it from a graphical desktop.

Discussion:

In this lab, I will walk through a basic understanding of processes and briefly look at how to manage processes in Linux using certain commands.

A process refers to a program in execution; it's a running instance of a program. It is made up of the program instruction, data read from files, other programs or input from a system user. There are fundamentally two types of processes in Linux-Foreground processes (also referred to as interactive processes) – these are initialized and controlled through a terminal session. In other words, there has to be a user connected to the system to start such processes; they haven't started automatically as part of the system functions/services.Background processes (also referred to as non-interactive/automatic processes) – are processes not connected to a terminal; they don't expect any user input.