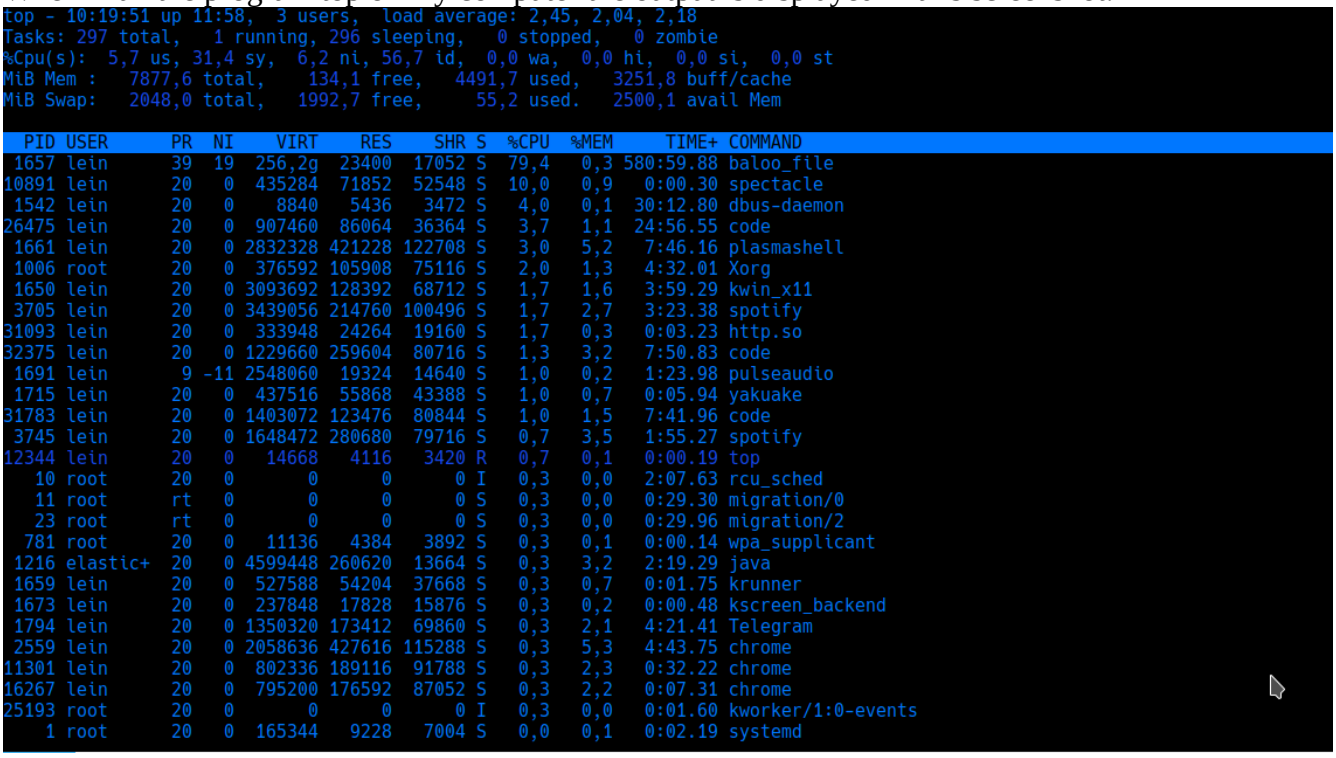
BASIC PROCESS MANAGEMENT WITH LINUX

Introduction.

This week we have cover concurrency and how it can affect the overall performance of the operating system, as well as some basic algorithms that have been used and studied throughout the years of development and research on OS. We were introduced to the Linux shell and some basic terminology used to navigate and perform some operations on the file system.

1. Run the “top” command and explain the output. Which process is using the most CPU and memory?

According to the manual pages, The **top** program provides a dynamic real-time view of a running system. It can display **system** summary information as well as a list of **processes** or **threads** currently being managed by the Linux kernel. The types of system summary information shown and the types, order, and size of information displayed for processes are all user configurable and that configuration can be made persistent across restarts. (man-pages, 2019)
When I run the program top on my computer the output is displayed in this screenshot.



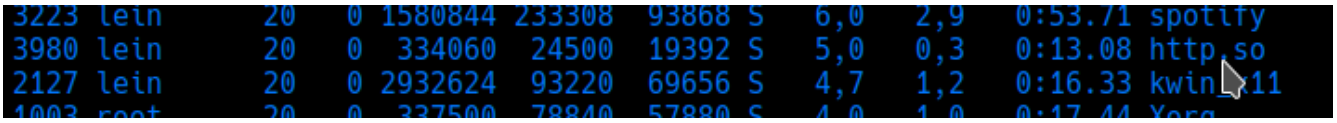
From the screenshot, we can see that the processes accessing the most the CPU are the OS built-in frameworks such as baloo_file which handles the file system in Kubuntu and spectacle that is taking the screenshot. If we notice carefully the header provided by the program top, we can see that the OS has a total of 297 tasks scheduled, however, 296 are already sleeping and only 1 running, the header also provided us with a real-time summary of the CPU and Memory resources. (man-pages, 2019)

We can also notice that the memory access is shared between all the listed processes, we can see that top is using only 0.1% while chrome is taking three chunks of memory and the rest is shared between the running processes, even the ones running on the background such as elastic search server.

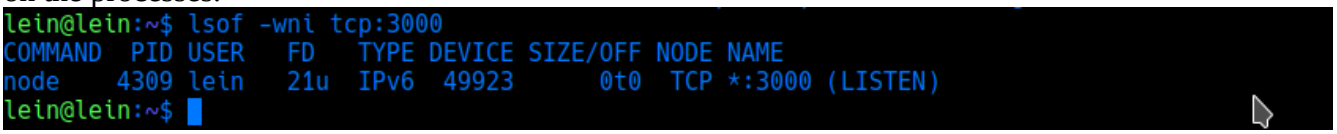
2. How would you kill a process in a Linux terminal?

There are multiple ways to kill a process in Linux, however, there are more effective ways to do so based on the type of process that the OS is running. I will list the two that I have used the most. Xkill works like a charm with processes that somehow have a clickable GUI, as soon as you write xkill on your terminal emulator the mouse pointer becomes an X or a danger sign, and as soon as you click on any open window, it will terminate that process intermediately. (man-pages, 2019)

The second and most used command to kill processes is the kill program which according to the manual it sends a signal to a process, the summary that the man pages provides is "The default signal for kill is TERM. Use **-l** or **-L** to list available signals. Particularly useful signals include HUP, INT, KILL, STOP, CONT, and 0. Alternate signals may be specified in three ways: **-9**, **-SIGKILL** or **-KILL**. Negative PID values may be used to choose whole process groups; see the PGID column in ps command output. A PID of **-1** is special; it indicates all processes except the kill process itself and init." we can easily use this program to terminate processes once we know the id of the process, and as explained above, the top programs provides the PID of the running processes we can easily combine them (man-pages, 2019). As an example, I will shut down my node.js server to show how it can be used.



I can see that the server is running as HTTP with the PID 3980 so I will use kill -9 3980 to signal the process the stop. I will stop the process intermediately. However I have found some other programs that give a more details information of the process we are looking for, I have used to list the running servers on tcp connections the lsof program like so lsof -wni tcp:3000 it will show a more details info on the processes.



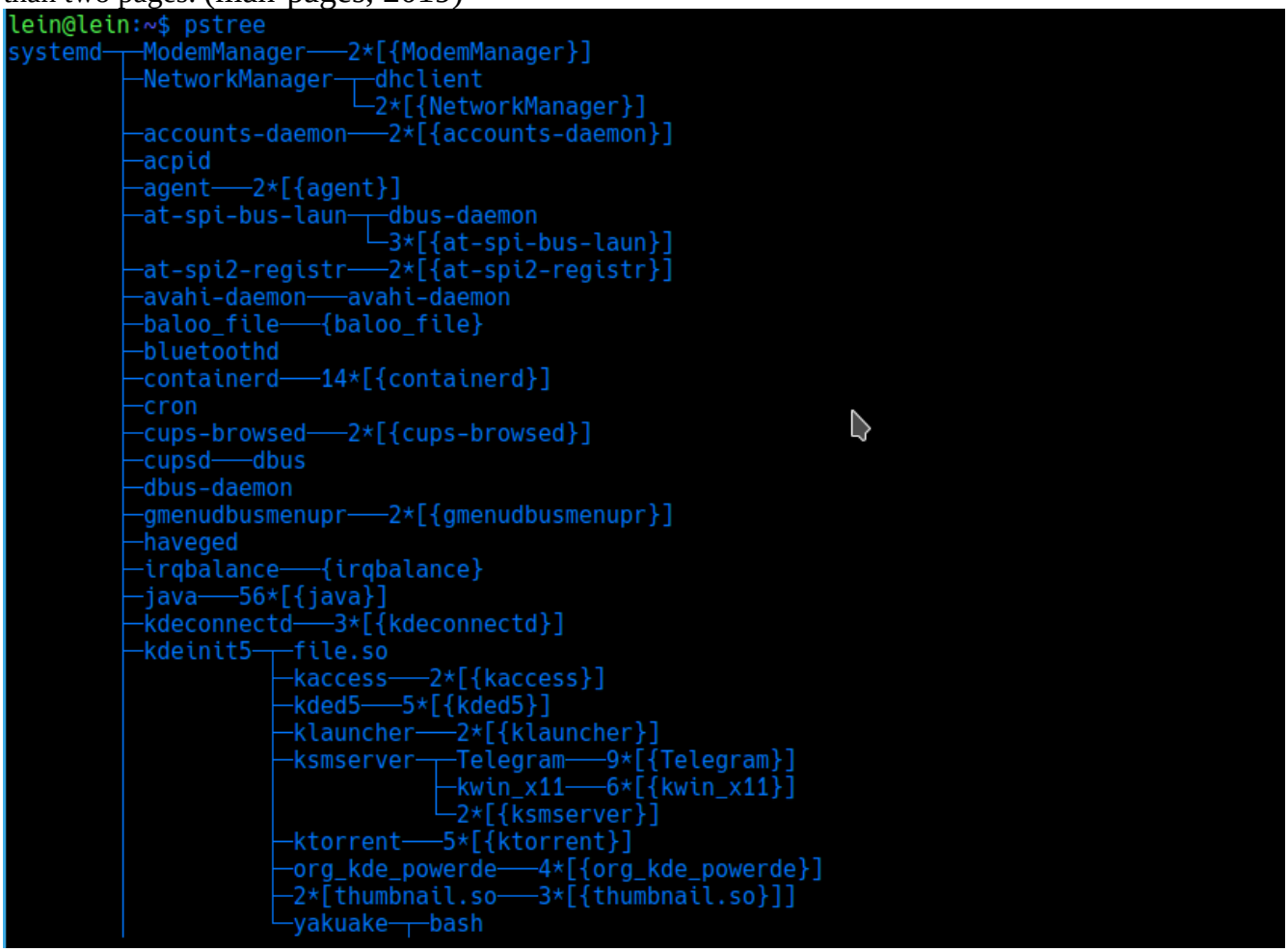
As you can see It has provided me with more detailed info on the process I am looking for and I can easily run the same command used above kill -9 4309.

3. Run the following commands and explain the output and describe the differences
ps: As in the man pages it just reports a snapshot of the current process, which is not very informative

We can see at the moment of the screenshot only the bash and the ps program, even though I have more process running, in the background, I have this text editor, I am listening music on spotify, and there are background servers running, however, ps does not provide these details for that we can use top as we did earlier of we can use the second command for this point…

pstree: According to the man pages this program displays a tree-like structure of all the processes, shows running processes as a tree. The tree is rooted at either pid or **init** if pid is omitted. If a user name is specified, all process trees rooted at processes owned by that user are shown.

If compared with ps, pstree provides more detailed information on what the OS is doing, the list of processes can be pretty extensive depending on the work you are performing. My snapshot covers more than two pages. (man-pages, 2019)



However, you can see everything related to the processes the OS is busy with.

Conclusions:

After reviewing the topic for this week on concurrency and the basic introduction to the Linux shell I am more confident on the importance of writing concurrent programs as well as possible OS optimizations, I had a lot of fun practicing and exploring the file system on my Linux machine, I enjoyed the feeling of knowing exactly what your computer is doing only by making some systems call and without the need for additional software to monitor the OS performance and processes being executed. When comparing this course to the previews course OS 1, where we worked mostly with the Windows OS, the main difference is well marked, most of the power In windows is by using a GUI however, by experimenting with the Linux OS most of the power a user has comes from the CLI which is a more simplistic and better performer.

References:

The Linux man-pages project. (n.d.). Retrieved from https://www.kernel.org/doc/man-pages/