Linux Administration

Managing processes

Xavier Belanger

Definitions

- A process is a program running in memory, and using other resources as needed (network, graphical display, etc).
- The main states for a process are: running, waiting or blocked, and then terminated.
- Processes are linked to a user, and all related to the main system process: init.

Checking processes

- *ps* is the most versatile command to check on processes.
- *pstree* can give you a “visual” representation of all processes (parent/child).
- *top* is an interactive and real-time view of all processes running.

The ps command

- Without any option, *ps* will display all processes for your current shell.
- *ps ux* will display all your processes, across multiple shells.
- *ps aux* will display all processes for all users.
- *ps u -u <username>* will display all processes for a specific user account.

The pstree command

- By default *pstree*'s output includes all processes running on the system.
- You can add the *-p* option to display the PID for each process.
- *pstree <username>* will restrict the output to a specific user.

The top command

- *top* will display a list of all processes running on the system, plus some basic information with an automatic refresh every three seconds by default.
- Use the ‘*q*’ key to quit the command.

Managing processes

- If a program becomes unresponsive, using too much resources or otherwise causing issues, you can terminate it with the *kill* command.
- *kill* requires the process identifier (PID) to target the proper process; you can obtain the PID with the *ps* command.
- *killall* is another command that can be used to send a signal to multiple processes with the same name.

The kill command

- *kill -l* will list all type of signals that you can send to a process.
- By default the TERM signal is sent to terminate the targeted process.
- Other signals can be used; but the result will depend on how the targeted program has been set to process a given signal.

The killall command

- The *killall* command works is a similar way as the *kill* one, the main difference is that multiple processes could be impacted.
- *killall -i* will ask for a confirmation before terminating each process.