

# **Linux Administration**

## **Managing processes**

Xavier Belanger

**This work is licensed under  
a Creative Commons Attribution-ShareAlike 4.0 International License.**

<http://creativecommons.org/licenses/by-sa/4.0/>

**You are free to:**

- **Share** — copy and redistribute the material in any medium or format
- **Adapt** — remix, transform, and build upon the material for any purpose, even commercially.

The licensor cannot revoke these freedoms as long as you follow the license terms.

**Under the following terms:**

- **Attribution** — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
- **ShareAlike** — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.
- **No additional restrictions** — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

# 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 in 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.