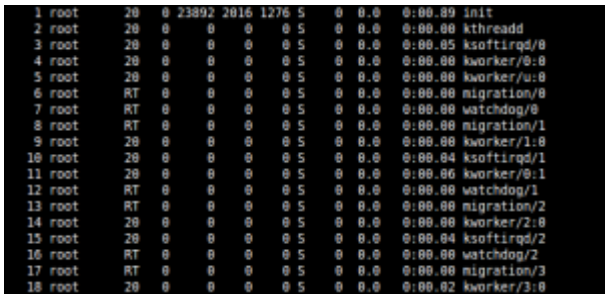
What are PID and PPID?

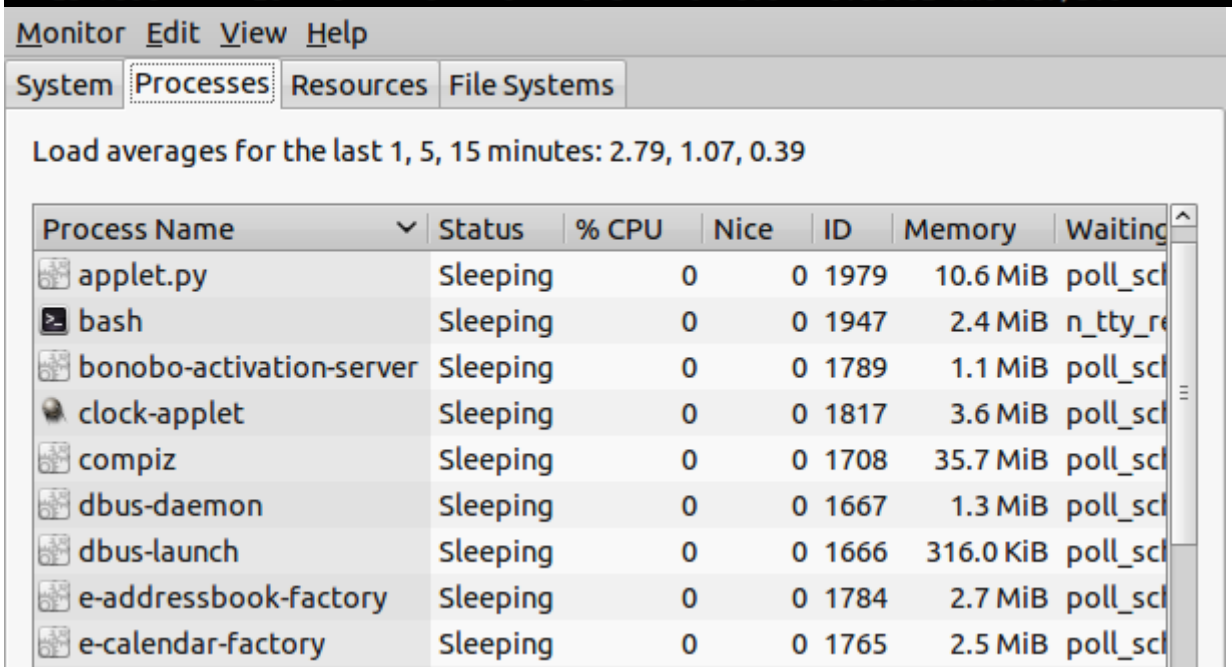


If you have ever opened System Monitor or top you no doubt noticed a column named ID or PID containing a list of numbers. You might even see a value called PPID. What do these numbers mean?

Here is a short explanation of these Linux terms.

In Linux, an executable stored on disk is called a **program**, and a program loaded into memory and running is called a **process**. When a process is started, it is given a unique number called process ID (**PID**) that identifies that process to the system. If you ever need to kill a process, for example, you can refer to it by its PID. Since each PID is unique, there is no ambiguity or risk of accidentally killing the wrong process (unless you enter the wrong PID).
If you open **top** (in a terminal, type **top** and press enter), the PID column lists the process IDs of all processes currently loaded into memory regardless of state (sleeping, zombie, etc.). Both daemons (system processes) and user processes (processes you started either automatically or manually) have their own process IDs. The PIDs are not always assigned in numerical order, so it’s normal to see what appears to be a random selection of numbers.

```
  1 root      20   0 23892 2016 1276 S    0  0.0   0:00.89 init
  2 root      20   0     0    0    0 S    0  0.0   0:00.00 kthreadd
  3 root      20   0     0    0    0 S    0  0.0   0:00.05 ksoftirqd/0
  4 root      20   0     0    0    0 S    0  0.0   0:00.00 kworker/0:0
  5 root      20   0     0    0    0 S    0  0.0   0:00.00 kworker/u:0
  6 root      RT   0     0    0    0 S    0  0.0   0:00.00 migration/0
  7 root      RT   0     0    0    0 S    0  0.0   0:00.00 watchdog/0
  8 root      RT   0     0    0    0 S    0  0.0   0:00.00 migration/1
  9 root      20   0     0    0    0 S    0  0.0   0:00.00 kworker/1:0
 10 root      20   0     0    0    0 S    0  0.0   0:00.04 ksoftirqd/1
 11 root      20   0     0    0    0 S    0  0.0   0:00.06 kworker/0:1
 12 root      RT   0     0    0    0 S    0  0.0   0:00.00 watchdog/1
 13 root      RT   0     0    0    0 S    0  0.0   0:00.00 migration/2
 14 root      20   0     0    0    0 S    0  0.0   0:00.00 kworker/2:0
 15 root      20   0     0    0    0 S    0  0.0   0:00.04 ksoftirqd/2
 16 root      RT   0     0    0    0 S    0  0.0   0:00.00 watchdog/2
 17 root      RT   0     0    0    0 S    0  0.0   0:00.00 migration/3
 18 root      20   0     0    0    0 S    0  0.0   0:00.02 kworker/3:0
```



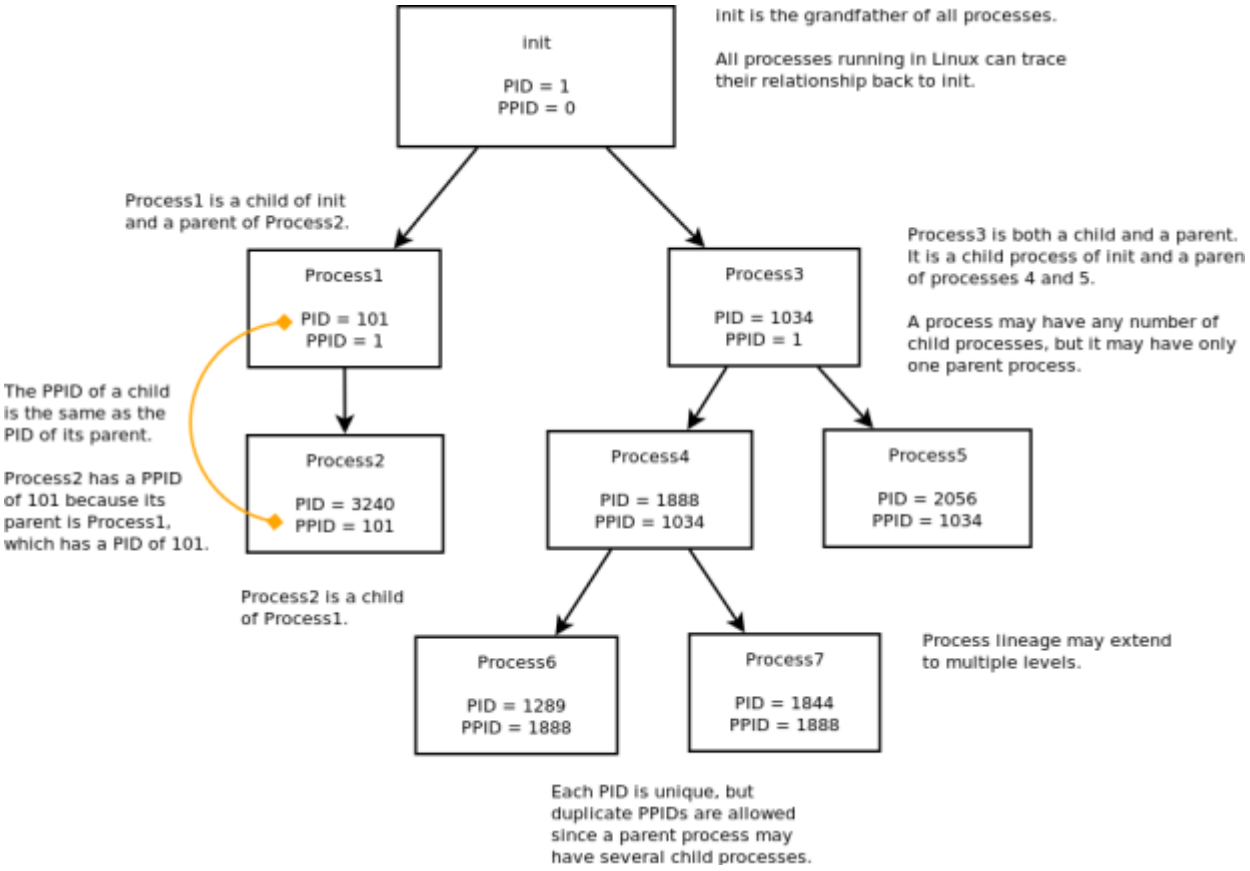
init

One very important process is called **init**. init is the grandfather of all processes on the system because all other processes run under it. Every process can be traced back to init, and it always has a PID of 1. The kernel itself has a PID of 0.

What is the PPID?

In addition to a unique process ID, each process is assigned a parent process ID (**PPID**) that tells which process started it. The PPID is the PID of the process's parent.

For example, if process1 with a PID of 101 starts a process named process2, then process2 will be given a unique PID, such as 3240, but it will be given the PPID of 101. It's a parent-child relationship. A single parent process may spawn several child processes, each with a unique PID but all sharing the same PPID.



Why is the PPID Important?

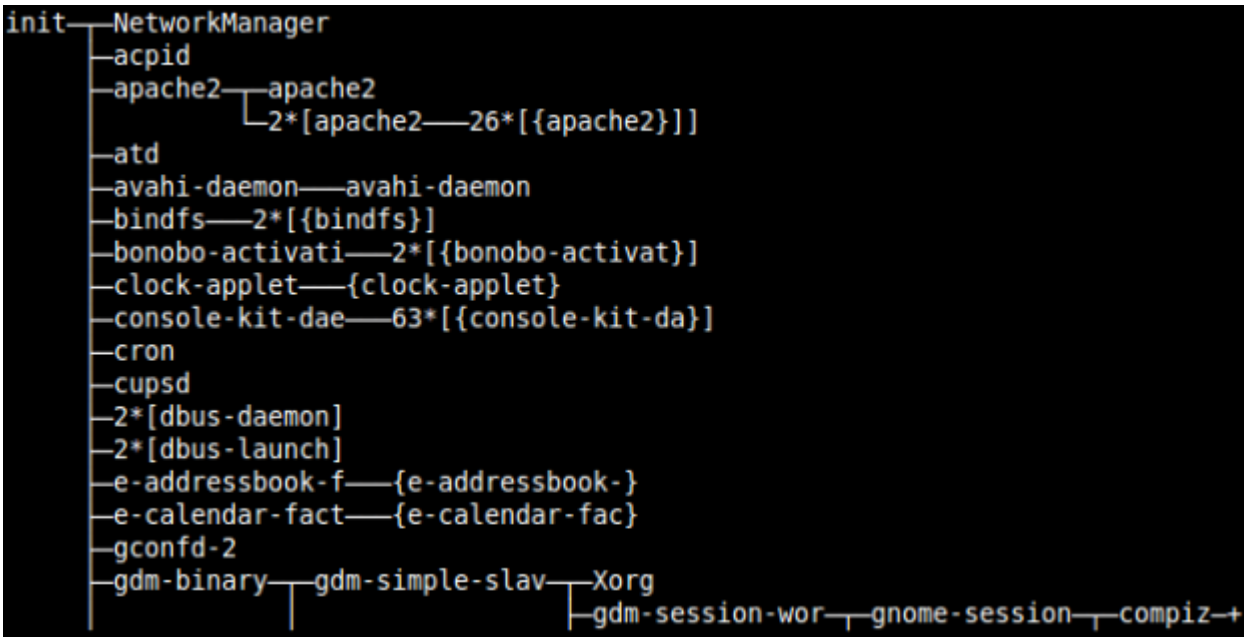
Occasionally, processes go bad. You might try to quit a program only to find that it has other intentions. The process might continue to run or use

up resources even though its interface closed. Sometimes, this leads to what is called a zombie process, a process that is still running, but dead.

One effective way to kill a zombie process is to kill its parent process. This involves using the ps command to discover the PPID of the zombie process and then sending a kill signal to the parent. Of course, any other children of the parent process will be killed as well.

pstree

pstree is a useful program that shows the relationship of all processes in a tree-like structure.



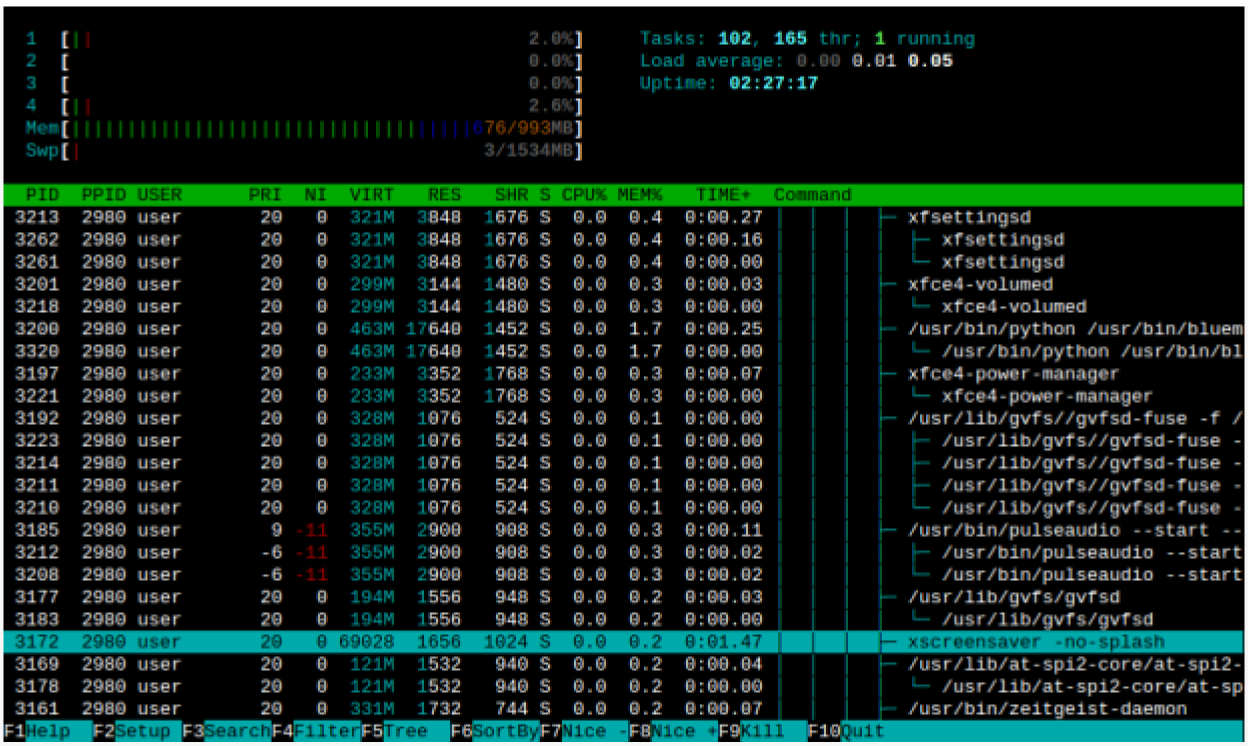
Give it a try to see how processes are arranged on your system. Processes do not float by themselves somewhere in memory. Each one has a reason for its existence, and a tree view helps show how it relates to others.

pstree supports options to adjust the output, so check **man pstree** for more details. Entering the following command lists the PID with each process and organizes processes by their ancestors (numerically) to show their relationship with each other.

```
pstree -pn
```

htop

For simpler process management and a better way to see how processes are organized, have a look at the program htop, which displays PID, optional PPID, process tree view, and much more information in glorious color!



Htop showing processes arranged in tree view along with PID and PPID.

Source : https://delightlylinux.wordpress.com/2012/06/25/what-is-pid-and-ppid/