Processes and Job Control

# Listing Processes and Displaying Information

To display the currently running processes use the ps command. If no options are specified, ps displays the processes associated with your current session. To see every process including ones that are not owned by you, use ps -e . To see processes running by a specific use, use ps

-u username .

ps - Display process status. Common ps options:

-e - Everything, all processes.

-f - Full format listing.

-u username - Display processes running as username.

-p pid - Display process information for pid. A PID is a process ID. Common ps commands:

ps -e - Display all processes. ps -ef - Display all processes. ps -eH - Display a process tree.

ps -e --forest - Display a process tree.

ps -u username - Display processes running as username.

$ ps

PID TTY TIME CMD

19511 pts/2 00:00:00 bash

19554 pts/2 00:00:00 ps

$ ps -p 19511

PID TTY TIME CMD

19511 pts/2 00:00:00 bash

$ ps -f

UID PID PPID C STIME TTY TIME CMD bob 19511 19509 0 16:50 pts/2 00:00:00 -bash

bob 19556 19511 0 16:50 pts/2 00:00:00 ps -f

$ ps -e | head

PID TTY TIME CMD

1 ? 00:00:02 init

2 ? 00:00:00 kthreadd

3 ? 00:00:19 ksoftirqd/0

5 ? 00:00:00 kworker/0:0H

7 ? 00:00:00 migration/0

8 ? 00:00:00 rcu\_bh

9 ? 00:00:17 rcu\_sched

10 ? 00:00:12 watchdog/0

11 ? 00:00:00 khelper

$ ps -ef | head

UID PID PPID C STIME TTY TIME CMD

root 1 0 0 Dec27 ? 00:00:02 /sbin/init

root 2 0 0 Dec27 ? 00:00:00 [kthreadd]

root 3 2 0 Dec27 ? 00:00:19 [ksoftirqd/0]

root root root root root root

5

7

8

9

10

11

2 0 Dec27 ?

2 0 Dec27 ?

2 0 Dec27 ?

2 0 Dec27 ?

2 0 Dec27 ?

2 0 Dec27 ?

00:00:00 [kworker/0:0H]

00:00:00 [migration/0]

00:00:00 [rcu\_bh]

00:00:17 [rcu\_sched]

00:00:12 [watchdog/0]

00:00:00 [khelper]

$ ps -fu www-data

UID PID PPID C STIME TTY TIME CMD

www-data 941 938 0 Dec27 ? 00:00:00 /usr/sbin/apache2 -k start

www-data 942 938 0 Dec27 ? 00:00:00 /usr/sbin/apache2 -k start

www-data 943 938 0 Dec27 ? 00:00:00 /usr/sbin/apache2 -k start

Here are other commands that allow you to view running processes.

pstree - Display running processes in a tree format.

htop - Interactive process viewer. This command is less common than top and may not be available on the system.

top - Interactive process viewer.

# Running Processes in the Foreground and Background

Up until this point all the commands you have been executing have been running in the foreground. When a command, process, or program is running in the foreground the shell prompt will not be displayed until that process exits. For long running programs it can be convenient to send them to the background. Processes that are backgrounded still execute and perform their task, however they do not block you from entering further commands at the shell prompt. To background a process, place an ampersand (& ) at the end of the command.

command & - Start command in the background.

Ctrl-c - Kill the foreground process.

Ctrl-z - Suspend the foreground process.

bg [%num] - Background a suspended process. fg [%num] - Foreground a background process. kill [%num] - Kill a process by job number or PID. jobs [%num] - List jobs.

$ ./long-running-program &

[1] 22686

$ ps -p 22686

PID TTY TIME CMD

22686 pts/1 00:00:00 long-running-pr

$ jobs

[1]+

$ fg

Running

./long-running-program &

./long-running-program

When a command is backgrounded two numbers are displayed. The number in brackets is the job number and can be referred by preceding it with the percent sign. The second number is the PID. Here is what it looks like to start multiple processes in the background.

$ ./long-running-program &

[1] 22703

$ ./long-running-program &

[2] 22705

$ ./long-running-program &

[3] 22707

$ ./long-running-program &

[4] 22709

$ jobs

1. Done ./long-running-program
2. Done ./long-running-program [3]- Running ./long-running-program &

[4]+ Running ./long-running-program &

The plus sign ( + ) in the jobs output represents the current job while the minus sign (- ) represents the previous job. The current job is considered to be the last job that was stopped while it was in the foreground or the last job started in the background. The current job can be referred to by %% or %+ . If no job information is supplied to the fg or bg commands, the current job is operated upon. The previous job can be referred to by %- .

You will notice that jobs number 1 and 2 are reported as being done. The shell does not interrupt your current command line, but will report job statuses right before a new prompt is displayed.

For example, if you start a program in the background a prompt is returned. The shell will not report the status of the job until a new prompt is displayed. You can request a new prompt be displayed by simply hitting Enter .

To bring a job back to the foreground, type the name of the job or use the fg command. To foreground the current job execute %% , %+ , fg , fg %% , fg %+ , or fg %num . To foreground job number 3, execute %3 or fg %3 .

$ jobs

[3]- Running

[4]+ Running

./long-running-program &

./long-running-program &

$ fg %3

./long-running-program

$

To pause or suspend a job that is running in the foreground, type Ctrl-z . Once a job is suspended it can be resumed in the foreground or background. To background a suspended job type the name of the job followed by an ampersand or use bg followed by the job name.

$ jobs [1]

[2]-

[3]+

$ fg

Running Running Running

./long-running-program &

./long-running-program &

./another-program &

./another-program

^Z

[3]+

$ jobs [1]

[2]-

[3]+

Stopped

./another-program

Running Running Stopped

./long-running-program &

./long-running-program &

./another-program

$ bg %3

[3]+ ./another-program &

$ jobs

[1]

[2]-

[3]+

$

Running Running Running

./long-running-program &

./long-running-program &

./another-program &

You can stop or kill a background job using the kill command. For example, to kill job number 1

execute kill %1 . To kill a job that is running in the foreground, type Ctrl-c .

$ jobs

[1] Running

[2]- Running

[3]+ Running

./long-running-program &

./long-running-program &

./another-program &

$ kill %1

[1]

$ jobs [2]-

[3]+

Terminated

./long-running-program

Running Running

./long-running-program &

./another-program &

$ fg %2

./long-running-program

^C

$ jobs

[3]+

$

Running

./another-program &

# Killing Processes

Ctrl-c - Kills the foreground process.

kill [signal] pid - Send a signal to a process.

kill -l - Display a list of signals.

The default signal used by kill is termination. You will see this signal referred to as SIGTERM or TERM for short. Signals have numbers that correspond to their names. The default TERM signal is number 15. So running kill pid , kill -15 pid , and kill -TERM pid are all equivalent. If a process does not terminate when you send it the TERM signal, use the KILL signal which is number 9.

$ ps | grep hard-to-stop

27398 pts/1 00:00:00 hard-to-stop

$ kill 27398

$ ps | grep hard-to-stop

27398 pts/1 00:00:00 hard-to-stop

$ kill -9 27398

$ ps | grep hard-to-stop

$

# Deep Dive

 [Bash Documentation on Job Control](http://www.gnu.org/software/bash/manual/html_node/Job-Control.html) [http://www.LinuxTrainingAcademy.com](http://www.linuxtrainingacademy.com/)