Tutorial

Processes in Linux

3.1 Processes and Jobs

3.2 Listing suspended and background processes

3.3 Killing a process

3.1 Processes and Jobs

A process is an executing program identified by a unique PID (process identifier). To see information about your processes, with their associated PID and status, type

```
ps
```

A process may be in the foreground, in the background, or be suspended. In general the shell does not return the Linux prompt until the current process has finished executing.

Some processes take a long time to run and hold up the terminal. ***Backgrounding*** a long process has the effect that the Linux prompt is returned immediately, and other tasks can be carried out while the original process continues executing.

Running background processes

To background a process, type an **&** at the end of the command line. For example, the command **`sleep`** waits a given number of seconds before continuing. Type

```
sleep 10
```

This will wait 10 seconds before returning the command prompt %. Until the command prompt is returned, you can do nothing except wait.

To run sleep in the background, type

```
sleep 10 &
```

```
[1] - Done                    sleep 10
```

The **&** runs the job in the background and returns the prompt straight away, allowing you do run other programs while waiting for that one to finish.

The first line in the above example is typed in by the user; the next line, indicating job number and PID, is returned by the machine when the process is finished. Backgrounding is useful for jobs which will take a long time to complete.

Backgrounding a current foreground process

At the prompt, type

```
sleep 1000
```

You can suspend the process running in the foreground by typing **^Z**, i.e.hold down the [**Ctrl**] key and type [**z**]. Then to put it in the background, type

```
bg
```

Note: do not background programs that require user interaction e.g. editors

3.2 Listing suspended and background processes

When a process is running, backgrounded or suspended, it will be entered onto a list along with a job number. To examine this list, type

```
jobs
```

An example of a job list could be

```
[1] Suspended sleep 1000
[2] Running netscape
[3] Running matlab
```

To restart (foreground) a suspended processes, type

```
fg %jobnumber
```

For example, to restart sleep 1000, type

```
fg %1
```

Typing **fg** with no job number foregrounds the last suspended process.

3.3 Killing a process

kill (terminate or signal a process)

It is sometimes necessary to kill a process (for example, when an executing program is in an infinite loop)

To kill a job running in the foreground, type **^C** (control c). For example, run

```
sleep 100
^C
```

To kill a suspended or background process, type **kill %*jobnumber***

For example, run

```
sleep 100 &
jobs
```

If it is job number 4, type **kill %4**

To check whether this has worked, examine the job list again to see if the process has been removed.

ps (process status)

Alternatively, processes can be killed by finding their process numbers (PIDs) and using kill *PID_number*

```
sleep 1000 &
ps
```

```
PID TT S TIME COMMAND
20077 pts/5 S 0:05 sleep 1000
21563 pts/5 T 0:00 netscape
21873 pts/5 S 0:25 nedit
```

To kill off the process **sleep 1000**, type

```
kill 20077
```

and then type **ps** again to see if it has been removed from the list.

If a process refuses to be killed, uses the **-9** option, i.e. type

```
kill -9 20077
```

Note: It is not possible to kill off other users' processes !

Summary

Command	Meaning
ls -lag	list access rights for all files
chmod [*options*] *file*	change access rights for named file
***command* &**	run command in background
^C	kill the job running in the foreground
^Z	suspend the job running in the foreground
bg	background the suspended job
jobs	list current jobs
fg %1	foreground job number 1
kill %1	kill job number 1
ps	list current processes
kill 26152	kill process number 26152