Operating Systems, Security and Networks (207SE)

Lab 14: **Linux command-line manipulation of processes**

**Your task**

1. Demonstrate some Linux command-line manipulation of processes: Start, suspend, background (make run in the background), foreground, and kill processes. Use typescript or screenshots to show evidence. There are a few ways to do each of these.

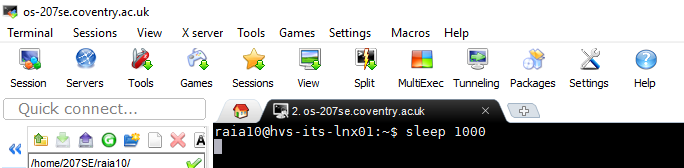
Demonstrate more ways to get the full 5/5.

* Example(s) of how to start process

[How to started process]

The way to start a process on Linux shell is to command. I am going to start the ‘sleep’ process.

[Screenshot or section of Linux script showing process starting]

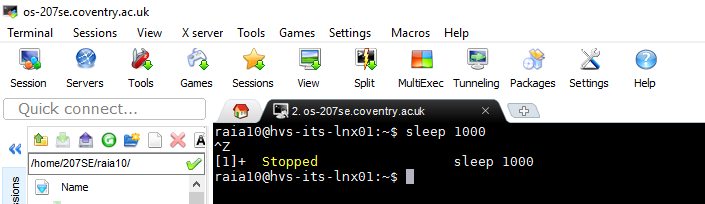


* Example(s) of how to suspend process

[How to suspended process]

To suspend a process : CTRL + Z

[Screenshot or section of Linux script showing process suspended]

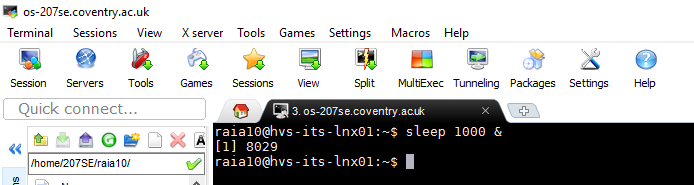


* Example(s) of how to run process in background

[How to run process in background]

To run a process in background : Command &. For this task I am going to run the sleep process in background

[Screenshot or section of Linux script showing process running in background]

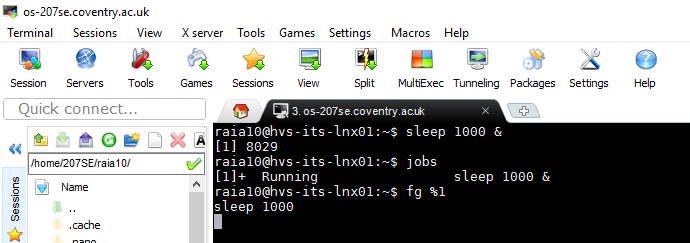


* Example(s) of how to run process in foreground and bring from background

[How to run process in foreground and bring from background]

To bring a process in foreground from background : fg %1

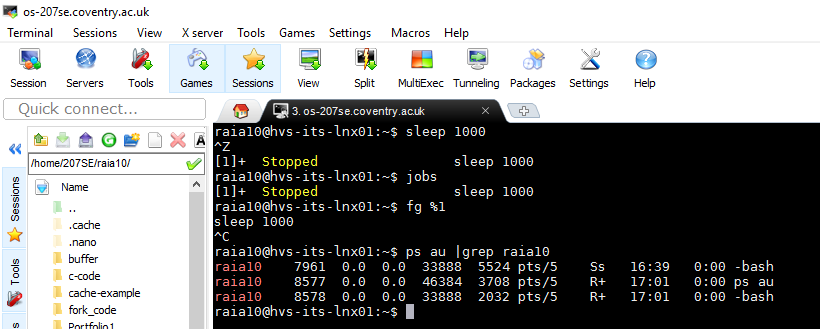
[Screenshot or section of Linux script showing process running in foreground or brought from background]



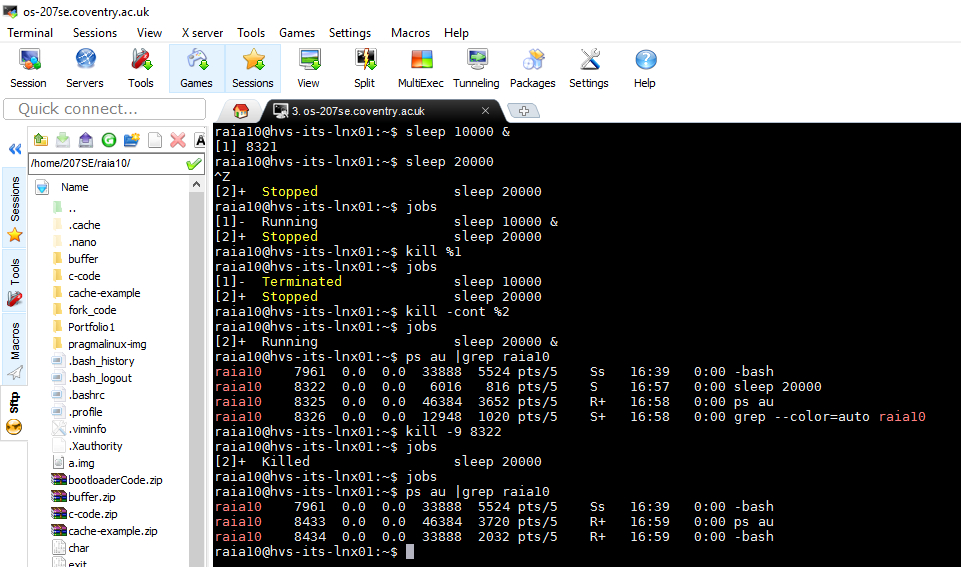
* Example(s) of how to kill a process

[How to kill process]

There are different ways to kill a process: CTRL + C [Screenshot or section of Linux script showing process killed]



**Demonstrating more ways**



1. I ran the sleep process in the background.
2. I then started a sleep process in foreground.
3. I then suspended sleep process in foreground using **CTRL + Z**.
4. I then used **Jobs** command to show the two sleep processes.
5. I then killed the first sleep process using **Kill %1** which was running in background.
6. I then used **Jobs** command to show the first sleep process was terminated and the second was suspended.
7. I then used kill **-cont %2** to continue the second process in foreground.
8. I then used Jobs command to show the process was running continually in foreground.
9. I then used command **ps au |grep raia10** to show a list of all my processes running
10. I then killed the remaining sleep process using **kill -9 8322**
11. Investigate the unix commands disown and nohup . Write a paragraph on how they are used and how they differ from one another.

Paragraph on disown and nohup command

[paragraph here]

**NOHUP & Disown**

Nohup is a command which allows a process to continue executing even after you have logged out. Nohup also known as ‘No hang up’, when the Nohup command is executed, it ignores all the hang up signals in the terminal so that the user can exit the terminal and the process will still be running in the background. This is done by separating the command from the shell and therefore allowing the process to continue after the shell is terminated. By default, the standard output will be redirected to nohup.out file which contains the standard output and error messages from the script that you’ve executed using Nohup command. The command to run Nohup is **‘nohup command &**’.

Disown on the other hand, removes the job from the shell’s job list and all the sub points such as the process like SIGNUP sent by the shell don’t apply and it is still connected to the terminal. If the terminal is destroyed and the controlling program is terminated by closing xterm or SSH connection, the program will fail as soon as it tries to read from standard input or write to standard output.

Inconclusion, disown removes the process from the shell’s job control but still leaved it connected to the terminal and Nohup disconnects the process form the terminal redirecting its output to nohup.out. Both commands supress SIGNUP(hangup) signals so that the program isn’t automatically killed when controlling terminal is closed. With Nohup, you must ‘nohup’ before a job/process begins however, if you did not then you can use ‘disown’ to modify the running job/process to keep executing even after you have logged out.

[linux.101hacks.com/unix/nohup-command/](https://Linux.101hacks.com/unix/nohup-command/)

<http://unix.stackexchange.com/questions/3886/difference-between-nohup-disown-and>

1. Describe what the watch command can be used for and give an example using the watch command. You don’t actually have to show this running, just describe the purpose and the commands you would type.

**Watch** command executes a program periodically, showing output(changes) of a program. When the command is executed it runs repeatedly, displaying its output from the first screen full which allows users to watch the program output change overtime. By default, when the command is executed, it will run programs every 2 seconds. This can be changed to using **-n**( where n is time) or **–interval** to specify a different interval. The -**d** or **---differences** will highlight the differences between updates. **–cumulative** will run a display of all positions changed, **-t or –no—title** wo;; turn off headers, time and blank line. **Watch** keeps running until interrupted.

**Examples**

To watch for mail = **watch -n 60 from**

To watch for directory change of a content = **watch -d ls -1**

Run **ls** command every 1 second = **watch -n 1 ls -l**

http://www.tutorialspoint.com/unix\_commands/watch.htm