# Linux Lab 6 - Process Commands

# Reading

*The Linux Command Line*, chapter 10, “Processes”, pp 95 - 106 in the printed text or pp 110 - 124 in the pdf.

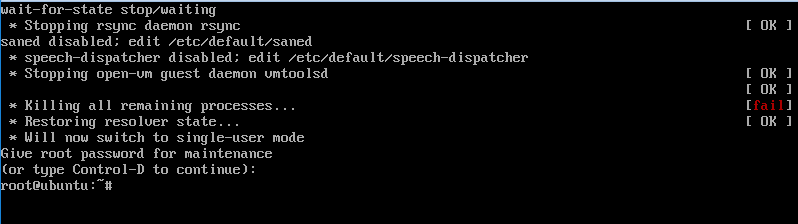
# Lab

## Terminals

We will practice using the ps, top, and kill commands from Chapter 10, but first we will look at some commands for manipulating terminals. You may wonder why Linux has a “less” command for paging through text data when you can scroll through the data using the scroll bar at the right of the terminal window. Why does Linux have ways to launch a process in the background (see the & added to the end of the xlogo command on page 101, xlogo & instead of xlogo) and ways to move a process back and forth between background and foreground? The answer is that for years (and to this day on most servers) there was no GUI. The console was all you had.

Review what happens in a terminal without a desktop GUI as we did in Lab 5, Apps and Services. Here are the systemd versions of the init 2 and init 5 commands we used before:  
sudo systemctl isolate multi-user.target  
and to return to the GUI:  
sudo systemctl isolate graphical.target

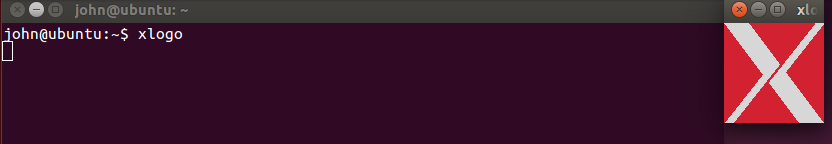
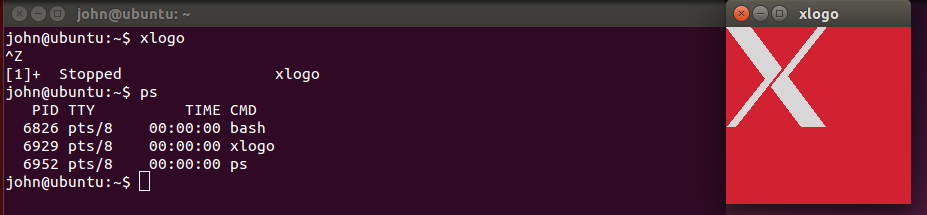
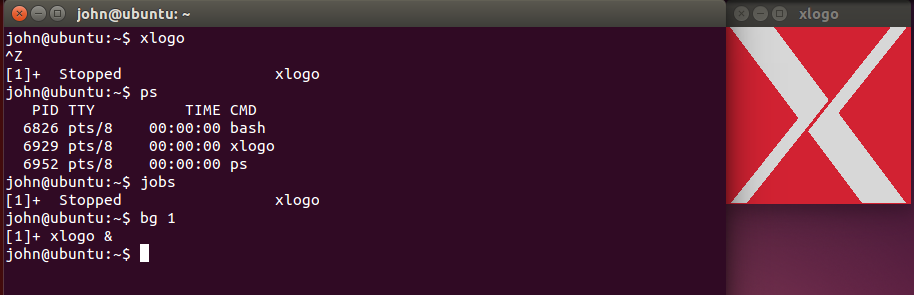
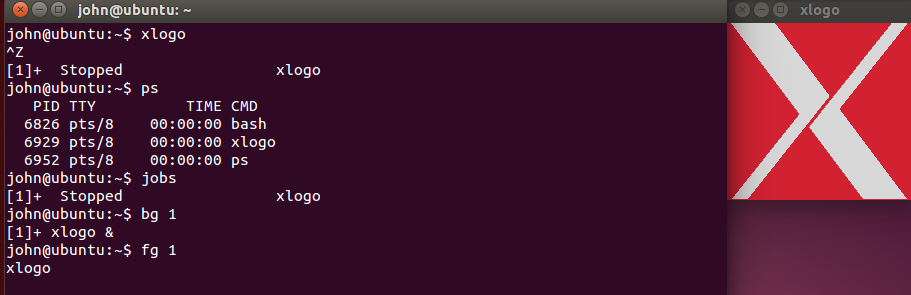
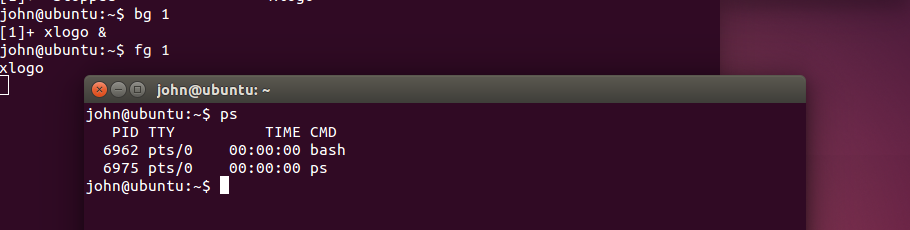
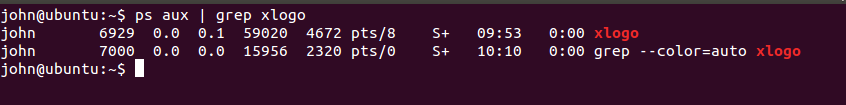
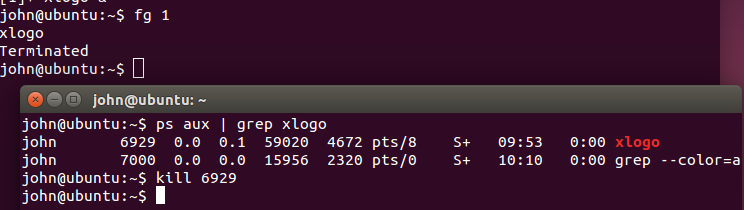
Note: if for some reason your VM locks up, you can just restart it from the VMware Player task bar.



This is a console. There are no scroll bars, and you can only do one thing at a time. Now the reason for having text paging (more and less), ways to switch between foreground and background, and ways to have multiple terminals is apparent.

## The ps command

The ps command allows you to see the running processes. It has one unusual aspect, in that it tries to follow the option formats for three different distributions, AT&T Unix, BSD Unix, and Gnu Linux. In the ps command, AT&T options start with a dash ( - ), BSD options do not have a dash, and Gnu Linux options have two dashes ( -- ). I usually use the BSD options aux. We will run the exercise from *The Linux Command Line*.

1. In one terminal window, run the command xlogo. It should bring up a small window with the XWindows logo. Note that your cursor has gone away and you can no longer enter commands, as the window is running xlogo.  
   
2. Type control-Z to pause xlogo and get your terminal back. Then use the ps command without options. The xlogo process is stopped (paused) and you should see the Process ID (PID). Note that the ps command without options only shows processes with your UID and running in that terminal; there are many processes running on your VM that aren’t shown. You can tell that the xlogo process is paused when you resize the window, as the window size changes but the “X” does not.  
   
3. The jobs command will show you what processes are in your terminal, and their job numbers. Make the xlogo job start running again in background by typing bg 1 (stands for “run job number 1 in the background.) The X resizes.  
   
4. Now bring the xlogo job to the foreground. We’ve lost the terminal again.  
   
5. Open a new terminal by right-clicking on the terminal icon on the left of your VM. Enter the ps command. Why doesn’t it see the xlogo process? (Hint: If you do not remember what was in the lab a few steps ago, type man ps. You will find the answer in the paragraph that starts, “By default.”)  
   
6. Now use ps aux, which will show all processes running on the OS. Can you find xlogo?
7. There is a lot of output to sort through. We can make that easier by using a pipe ( | ) and a search program called grep. Note that whenever you search for processes using grep, the grep process shows up, as well as the process we are searching for.  
   
8. In this case, the PID of the xlogo process is 6929. Yours will probably be different. Let’s kill the xlogo process. The logo window disappears, and the old terminal is back.  
   
9. This time run xlogo in one terminal, and then stop it with control-Z. Then type jobs, and you will see the process for xlogo is still there, but it is listed as stopped (docs may call this “suspended” as well.) Find the PID and try to kill it.   
   kill 2328 (or whatever the PID is on your system.)  
   Text

   Description automatically generated with medium confidence  
   You will probably find that it does not terminate, as in the example above. The process is still there after the kill command. The program did not hear the terminate signal from the kill command because it was suspended or stopped. You can force xlogo to quit by using the KILL signal, as below. (Your PID will be different.)  
   kill -KILL 2328 or kill -9 2328
10. The ampersand ( & ) immediately puts a process in background. This is handy when you open a text editor, and you don’t want to bother with opening another terminal. Try this command:  
    gedit &

# Hand in

The chapter also covers the command called “top”. What does it do, and why might you use it?