Basic Unix Principles

Jeremy Sanders

October 2011

1 Computer Terms

I'll refer you to the Glossary I have compiled if you have problems with terms such as Memory, Disks, Programs. Simply, memory is what the computer stores programs and their data when they are currently running. Memory contents are lost when the power goes, and memory is limited in size (usually only a few gigabytes in size). Data and programs are stored onto disk for later use and retrieval. A program is a set of instructions for the computer's CPU for manipulating data, including sending it to files, to the screen or to other bits of the computer.

2 The Kernel

In Unix (often spelt UNIX), the basic software controlling the hardware is know as the Kernel. The Kernel does all the difficult and nasty things like managing all the running processes (a name for a running program), the memory, the network connections, disks, tapes and virtually any bit of hardware on the computer.

When you write a program in a computer language like Fortran or C, a 'runtime library' deals with the kernel allowing you to use nicer commands to get memory and write files in your language, like WRITE in Fortran or printf in C.

3 Shells

A shell is a program which allows you to run other programs. Microsoft Windows has a shell called Explorer which allows you to run programs by clicking on them. Although there are graphical shells for Unix, such as Gnome and KDE, they are not installed on the computing cluster. Most interaction is done through a traditional shell, which allows you to type commands into it to run other programs. A 'command line interface' is harder to learn than a graphical shell but much more efficient once you've got the hang of it.

The standard astronomy shell is known as tcsh (Extended C-Shell, as it bears some resemblance to the C language), but other popular ones include the standard sh (Also known as the Bourne Shell) and bash (The Bourne Again Shell). All shells let you run programs by typing in their names, but some have more features. Most shells let you do more complex things like loops and conditional statements, which are very handy when it comes to automating operations. tcsh has lots of flaws, but nevertheless has become the *de facto* astronomical shell. bash is probably the most popular shell.

4 Files

In Unix, like in many environments, data are stored in things called files. A file contains data made up of a particular number of bytes. Each byte can hold a number between 0 and 255. Characters (like letters, numbers and symbols) are encoded as a particular number using ASCII encoding, although things are a bit more complex than this on modern systems. The letter ‘A’, for instance is stored as the number 65, ‘B’ is 66, and the lowercase ’a’ is 97.

Many files in Unix are stored as text; often data in a program are translated into text for saving and loading. Other files, such as those your programs could write, or programs themselves, are encoded such that you can load them back into memory.

Files have particular names, which can be hundreds of characters long, and are case sensitive, so that the file ‘Hello.txt’ isn’t the same as the file ‘hello.txt’. Conventionally files of a particular type have the same ‘suffix’ or ‘extension’, which is the bit at the end back to the dot. For instance the suffix .txt conventionally contains text. Filenames usually contain the normal letters (usually mostly lower case), numbers, and the symbols ‘.’, ‘_’ and ‘-’.

5 Directories

Files are stored on disk (which may be a CDROM or a pretend disk in memory, like the /tmp file system). Files are organised in a ‘File System Hierarchy’ or ‘Directory Tree’ (it looks like a tree), which sounds a bit complex. A directory is a set of files with a name, which can also contain other directories. All the directories can be traced back to the root directory, ‘/’.

In Unix the layout of directories is largely historical and conventional. Fig. 1 contains a diagram how the important directories in the tree are arranged. At the IoA, many other directories exist in the tree, including the data disks and where local packages are installed. It’s important to note that the directories can exist on different disks, and even on other machines on the network. Your home directory is likely to be on a different computer to the one you are using. The Network File System (NFS) allows you to access your files on different computers.

Names of files are built up from the current working directory, or from root if an absolute name is given (starting with /). Directories are specified with slashes after them.

The important one you need to remember is your home directory. It will be at ‘/home/username’, and it’s the one you get when you log in. The ‘cd’ command lets you change the directory you’re currently working in, and pwd tells you where you are. Often you can refer to your home directory quickly as ‘∼/’ or someone else’s home directory as ‘∼username/’.

6 Permissions

Each file and each directory has an owner saying who has control of the file, and a group which can be several users, who also can have access to the file. The important files on the computer are owned by the user ‘root’ who is your local friendly system administrator. Certain programs and files are restricted for use by root.

Each file and directory has permissions saying whether the owner, group or any other users are allowed to read from, write to, or run the file (r, w and x permissions). Root can always do things to a file. ‘ls -l’ lets you examine the ownership and permissions of a file, and chmod lets you modify the permissions if you own it.

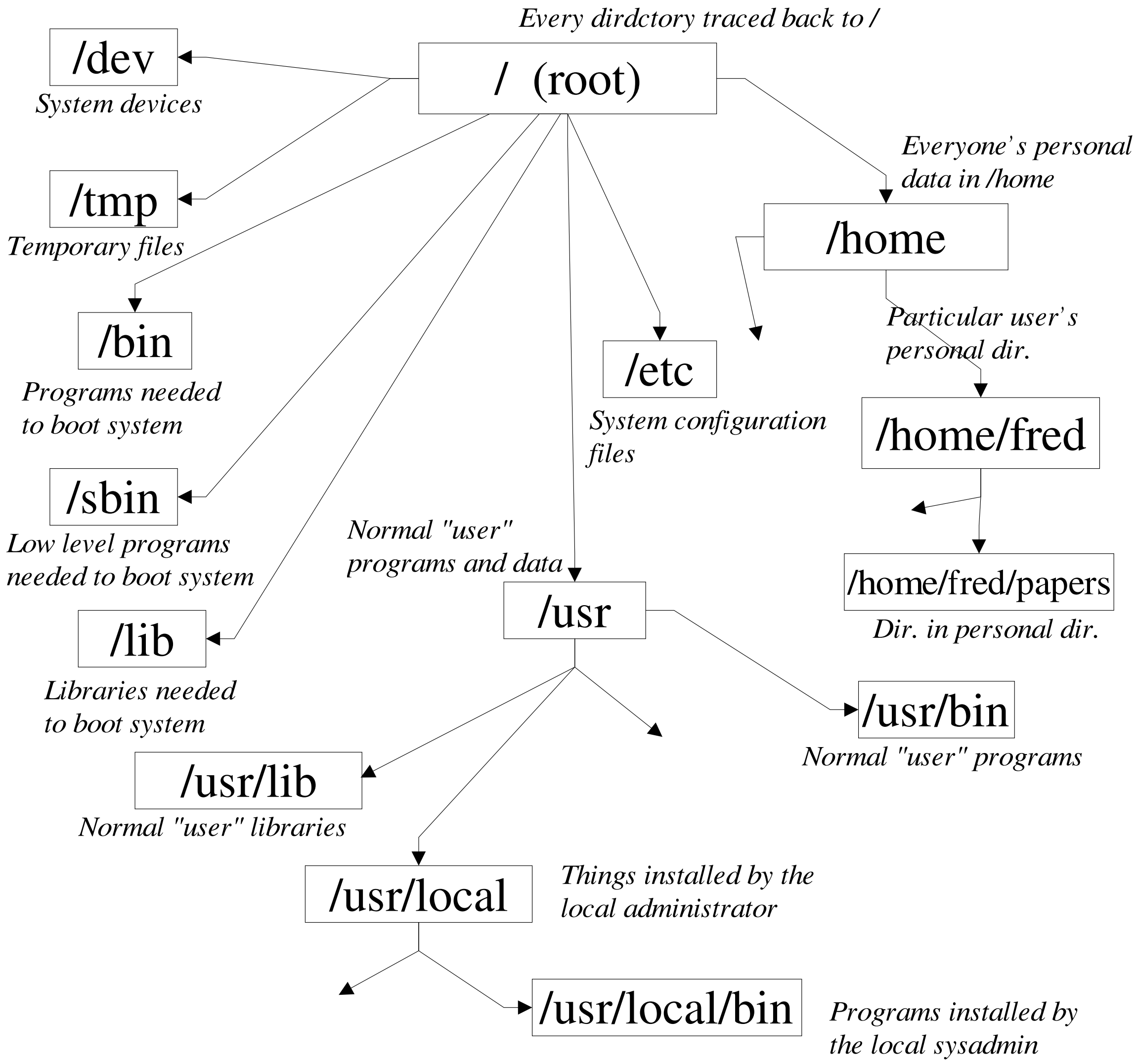
Every dirdctory traced back to /
/ (root)
/dev
System devices
/tmp
Temporary files
/bin
Programs needed to boot system
/sbin
Low level programs needed to boot system
/lib
Libraries needed to boot system
/etc
System configuration files
/home
Everyone's personal data in /home
Particular user's personal dir.
/home/fred
/home/fred/papers
Dir. in personal dir.
Normal "user" programs and data
/usr
/usr/bin
Normal "user" programs
/usr/lib
Normal "user" libraries
/usr/local
Things installed by the local administrator
/usr/local/bin
Programs installed by the local sysadmin

Figure 1: Main directories in the file system hierarchy.

7 Processes

When you run a program, you load it into memory and it starts running as a ‘process’. Processes compete for the CPU time on the computer, according to their priority or ‘niceness’. Each process gets its own identification number (PID), which are recycled eventually. Processes are identified by this number in the ‘top’ and ‘ps’ process examination tools, and can be used to kill or stop the process with the ‘kill’ command.

When you start a process it starts in the ‘foreground’, letting you type input into it and letting you see what its output is. Processes may also be started in the background, letting you carry on using the terminal (where you are typing into the shell).

8 The X Window System

X is a system for displaying graphical applications on Unix (and other) systems. An X server runs on your computer, and receives requests to open windows, draw windows, accept input, and so on from a client application (which is the program you are running).

X is clever because it doesn’t matter where the client application is running. You can run a program in Australia and have it appear on your screen, as if it were running on your local computer (with a bit of slowness).

The program which controls where the windows on your screen appear, and how they can be altered in shap and position, is the *window manager*. There are several window managers, the most common one here being ‘fvwm’.

The way a program knows which X server to talk to (i.e. where to put its window) is to look at the DISPLAY environment variable. You can see what this is set to by typing `printenv DISPLAY`. :0 is the screen on the computer you are using. If it says ‘cassXX.ast.cam.ac.uk:0’, then this is a server on a completely different computer. You can change DISPLAY to point to anywhere, but it won’t necessarily work unless you set up security first. It would be dangerous if you could start programs on any computer at will.

If you ssh to another computer, then ssh normally automatically sets DISPLAY for you, so windows appear on your screen. It forwards X commands over the network, securely encrypting them.