# *RMIT School of Computer Science*
## COSC1131/1133 Unix Systems Administration
## Lab Session 2 - Advanced Installation

**NOTE:** *This lab is a continuation of Lab 01*. If you have not successfully completed a Linux installation in Lab 01, you should probably complete that before proceeding to the advanced exercises.

# Linux installation (again)

First you will need to install Linux again. Follow the same procedure as in Lab 1, partitioning your hard disk anew, installing Linux and making it bootable. Install packages as described in lab sheet 1. (Though make sure that the `groff` package is installed, as it is needed by the manual system.) In general it is best to accept the default package selection since it will maks sure that necessary dependencies are satisfied.

# Adding a new partition

In this exercise, you will be creating a new partition. You will be doing this without booting from the installation CD. (In general, you should only boot from the installation CD when installing a new system or when a system will not boot.) You will be using the `fdisk` utility for making the new partition.

First of all, note down the permissions that the /tmp directory has (hint: use `ls` with the `-l` switch); when you create your partition, you will need to set its permissions to the same ones, otherwise things won't work. (Note that we already have a mounted /tmp partition – we will creat a new partion for the home directory of Fred Dagg – username fred)

Next, as root, invoke `fdisk` on the first hard disk, with the "`fdisk /dev/hda`" command. You will see the `fdisk` command prompt. Enter the command to print the partition table; it should show a list of all partitions on the disk and their sizes.

Now add a new logical partition, 100Mb in size, starting at the default first cylinder. Then print the partition table again; your new partition should appear. Note down its device name. Then write the partition table to disk and exit.

As you are creating a partition on the disk currently in use, you may need to reboot before Linux can be guaranteed to properly recognise the new partition table. Try the next step, but if `mke2fs` complains, then you will need to reboot the machine, with the **reboot** command.

Once the machine is rebooted, if necessary, (try this first without a reboot)make a Linux file system on the new partition. (This is equivalent to 'formatting' a disk in DOS/Windows.) The standard Linux file system type is known as "ext2", which stands for "second extended"; to make an ext2 filesystem, you use the `mke2fs` program. You will need to specify the partition to make the filesystem on; for example, if your new partitions is named `/dev/hdx23`, you would use the command `mke2fs /dev/hdx23`. (If you don't understand how mke2fs works, look at the manual page, which can be called up with *`man mke2fs`*.)

Use `cd` to change directory to `/home`. Now create a new directory that we will use to mount our filesystem with the command `mkdir /home/fred`. Go into the directory and create a file called test with `vi`. "`vi test`" Enter the text "`This is a test file`".

Change directory out of the /home/fred directory. "cd .." will do it.

Now you can mount the formatted the partition, mount it at /home/fred, with the mount command. (See the manual page for details on how to do this.) Once you have done this, call mount without any arguments to list the partitions mounted. It should say something like:

```
/dev/hdax on /home/fred type ext2 (rw)
```

Now change directories to /home/fred and use the ls command. You should only see a directory called lost+found. What has happened to the test file?

Create a new file called test2 in the /home/fred directory in the same way that you created test. Put some different text in it.

Now check the permissions of the current directory, with ls -ld .. How do they differ from the permissions of /tmp?

In /home/fred, change the permissions of . with the chmod command, until they are identical to the permissions you noted down from the original /tmp directory. The permissions of /home/fred, should look something like:

```
drwxrwxrwt    8 root     root      4096 Mar 13 16:58 /home/fred,/.
```

Notice the t at the end of the permissions; this is called the "sticky bit". The reason for this is to allow users to create files in /tmp whilst preventing users from deleting or modifying other users' files without permission.

Now try unmounting the filesystem with the umount command. Check the contents of /home/fred with ls. Is the test file there? Is the test2 file there?

Finally temporarily mount your new filesystem at /mnt with the mount command. (/mnt is specifically designed to be a temporary mount point on the system.) Look in /mnt with ls and see what files are there.

## fstab

By now, you have created and mounted your new /home/fred partition, and it should work. However, the next time the system is rebooted, it will not be mounted automatically; until you do so manually, the system will use the old /home/fred directory in the home partition.

To make a partition mounted automatically, you will need to put it in /etc/fstab. This is a system configuration file which contains a list of the filesystems automatically mounted when the system boots. Each line of the file describes one filesystem.

Lines in the file look like:

```
device-name     mount-point     fs-type options flags
```

Where *device-name* is the device on which the filesystem exists (usually the disk device of the partition), *mount-point* is where it is mounted, and *fs-type* is the filesystem type (i.e., ext2 for most Linux partitions). Look at /etc/fstab on your machine for an example.

Now edit /etc/fstab, adding a new line for your new /home/fred partition. You may want to copy the line from an existing partition and change the device name and mount point.

Reboot the machine; then log in as root and type mount. The list of mounted partitions printed should include your new /home/fred partition.

# Tweaking the boot sequence

In this exercise, you will be making adjustments to the way Linux boots. On the system you are working on, Linux uses the `lilo` system for booting. The lilo configuration lives in `/etc/lilo.conf`. The format of the file is described in the `lilo.conf` manual page.

Firstly, you may want to make a copy of `/etc/lilo.conf`, so that you can have a copy of the original to look on. Copy it to `/etc/lilo.orig`.

1. Firstly, you will be removing the boot prompt. Given that this machine will be used as a dedicated Linux server, it makes no sense to prompt for which operating system to boot. Modify /etc/lilo.conf to remove the prompt; then run lilo. If lilo reported any errors, fix them and run it again; if not, reboot. Watch the booting sequence to see if it works as desired.
2. Now you will be optimising the video display mode. Modern graphics cards and monitors can display text at resolutions much greater than 80x25, and can consequently display much more information on the screen. However, for reasons of compatibility with older systems, the standard VGA text mode is set by default.

Looking at the manual page, put a line into the LILO configuration file to ask which video mode you want to use when you boot. Then run lilo and reboot, trying it out. Select the highest-resolution mode and let the system boot. If you don't like that mode, reboot again, selecting another one until you find one you like. Once you do that, change the line you added to /etc/lilo.conf to set the mode to that without asking you.

(Aside: how can you change which video mode Linux boots into on a once-off basis without changing the LILO setup?)

# APPENDIX: Disk and partition naming under Linux

A Linux system may have a number of physical hard disks attached, each one of which is divided into a number of partitions. To Linux, a partition is a device like a disk.

Hard disk devices have names starting with `hd` (or `sd` for SCSI disks), and are represented by device files in `/dev`. Linux identifies physical disks with letters, from `a` onwards; hence, the first physical hard disk is `/dev/hda`, the second is `/dev/hdb`, etc.

Partitions are identified with numbers, following the name of the disk on which the partition is. Partition 1 of disk `a` is named `/dev/hda1`; partition 2 of disk `c` is `/dev/hdc2`.

On a PC, partition numbers 1 to 4 are reserved for the four primary partitions. If you have logical partitions, they will haved numbers from 5 upward; in addition, one of your primary partitions will be an extended partition and will contain them.