A Linux Workbook [1]

Geoffrey Robertson
geoffrey@zip.com.au

October 13, 2002

Contents

37 **171**

Part I

Installing Linux

Chapter 1

Chapter 2

Linux install: RH7.0

This Document: Install Linux from a local CDROM

note 1: This exercise is based on Red Hat 7.0

note 2: The instruction books come with a purchased boxed set of Red Hat Linux. HTML documentation is included on the Red Hat 7.0 cdrom number 5. See section RTFM below.

note 3: The procedures described in this document may result in the loss of all data on your computer. Back up all important data before proceeding.

2.1 system requirements

You will need the following as an absolute minimum:

- a Linux distribution on CDROM
- a blank floppy or two or four
- a suitable computer
  - 486DX2 or better
  - CD ROM drive
  - 8 Mb of RAM or better
  - 150 MB hard disk free, preferably more
- time and patience
- brain turned on

2.2 system information

Gather and record pertinent information about the system
Fill in the sheet "Installation Target Hardware Survey"
Fill in the sheet "Systems Requirement Table"

`/cd5/RH-DOCS/rhl-ig-en-7.0/ch-table.html`

2.3 RTFM

Read the file called README on the first distribution CD.
Extensive documentation is on CD number 5. Open this document in your HTML browser.

`/cd5/RH-DOCS/index-en.html`

- The Official Red Hat Linux Getting Started Guide
- The Official Red Hat Linux Installation Guide
- The Official Red Hat Linux Reference Guide

If you don't have Red Hat 7.0 CD number 5 then you can access all the documentation on the distribution home pages. Read it.

2.4 making space for Linux using the fips utility

This is required if you have a legacy operating system using a fat or fat32 file system taking up all the room on you hard disk and wish to keep it.
Note: Partition resizing may be performed on the fly during installation on many distributions.

- De-fragment the partition that you wish to shrink;
- read the fips documents in `d:\dosutils\`;
- make a DOS boot floppy;
- copy the appropriate files to the floppy from the distribution CD to the floppy - fips.exe, errors.txt and restorrb.exe;
- boot the floppy;
- launch fips: `A:\> fips`;
- follow the instructions and remember that losing the contents of your hard drive can ruin your whole day.

2.5 make a boot floppy

Make a Linux boot disk using "rawrite" if you do not have a BIOS capable of booting the CDROM

- boot the DOG (MS-DOS)
- insert a blank formatted floppy in the a: drive
- at the dos prompt: (assuming D: is the drive letter of the CDROM)

```
C:\> d:
D:\> cd dosutils
D:\dosutils> rawrite
    - D:\images\boot.img
    - a:
```

2.6 starting the installation

- place the Linux distribution CDROM in the drive
- place the Linux boot disk in the a: drive (if needed)
- boot from the CDROM (or floppy)
- answer the questions on your screen correctly ;-)

2.7 installing

The Red Hat 7.0 installer presents the user with a series of screens with questions to be answered and formes to be filled in. The following points might guide you through.

1. Choose a Language: English
2. Keyboard configuration: Generic 101-key PC, US English
3. Mouse configuration: Probably auto detected
4. Read the Welcome screen help
5. Installation Type: Custom System
6. Partitioning: Manual with Disk Druid

   hda1 DOS / WinThing™

   hda5 Linux Swap (128M)

   hda6 Linux / (Available room)
7. Format: /
8. Lilo configuration:
   - Create a boot disk
   - Boot from MBR
   - Winthing™ default
9. Network Configuration These settings are Class C static addresses.

   IP Address 192.168.1.7 *(for the 7th box)*

   Netmask 255.255.255.0

   Network 192.168.1.0

   Broadcast 192.168.1.255

   Hostname box7 *(for the 7th box)*

   Gateway

   Primary DNS

   secondary DNS

   Tertiary DNS

10. Time Zone Selection: Sydney Australia (not UTC if dual boot with WinThing™)

11. Account Configuration:

    Root Password square

    User Account add your login name and password

12. Authentication Configuration: Enable MD5 and shadow passwords

13. Package Group Selection: Choose only the packages listed below and deselect all the others:

    - Printer Support
    - X Window System
    - GNOME
    - Mail/WWW/News Tools
    - DOS/Windows connectivity
    - Graphics Manipulation
    - Multimedia Support
    - Networked Workstation
    - Dial-up Workstation
    - Network Management workstation
    - Authoring/Publication
    - Emacs
    - Development
    - Utilities

14. Monitor Configuration: choose your monitor

15. X Configuration: choose your video card and *deselect* "Use Graphical Login". Test your configuration.

16. About to Install: Next

17. Boot disk Iqnstallation: An emergency rescue disk is your friend.

2.8 check the installation

- check that LILO boots by default into the pre-installed legacy commercial operating systems
- boot the system into your old OS's to check that they're okay
- boot to Linux and log on as root
- setup a user account: `# useradd yourname`
- give yourself a password `# passwd yorname`

- login as you on another virtual terminal `(ALT F2 say)` login with your new user name, explore and enjoy
- try this:
  - `$ cd somewhere` changes directory up to somewhere
  - `$ cd ..` goes back
  - `$ ls -al` lists the details of the files in "."
  - `$ man foo` tells you about "foo"
  - `$ apropos bar` may tell you something about "bar"
  - try `$ info` and graze on the juicy info here

2.9 personal recovery

If you found the installation stressful you may need to discuss it with the "doctor".
Start emacs: `$ emacs`
In emacs type; `[esc] x doctor [enter]`

2.10 Further configuration of X

Set up mouse screen etc. several choices of setup programs:

- `# setup`
- `# XF86Setup`
- `# Xconfigurator`
- `# xf86config`
- hack the config files directly

2.11 start reading and doing

You will probably need a book... have a look at
Running Linux
published by O'Reilly

Part II

Using the `bash` Shell

Chapter 3

Using command aliases

ALIASES allow a string to be substituted for a word when it is used as the first word of a simple command. The shell maintains a list of aliases that may be set and unset with the ‘alias’ and ‘unalias’ builtin commands.

Example

An alias for the command `ls -F` may be set thus:

```
$ lf ↩
bash: lf: command not found

$ alias lf='ls -F'↩

$ lf ↩
install_.sh*            manipulating.text/   test/
cvstest/                misc102.doc         test2*
```

Note that the alias for `$ lf ↩` is only defined in this shell.

3.1 Creating aliases in the shell

In *bash* an *alias* is a user-defined abbreviation for a command.

- Setting an alias from the command line:

  ```
  $ alias m=more ↩
  $ alias ll="ls -l" ↩
  $ alias ls='ls -F'↩
  ```

  The quotes are used to hide the white space—single quotes are preferred.

- Delete an alias:

  ```
  $ unalias ll ↩
  ```

- Temporally unset an alias (use the original command):
  ```
  $ \ls
  ```

3.2 `alias` example

```
$ ls
file0  file1  link.file  mydir
$ ls -F
file0  file1*  link.file@  mydir/

$ alias ls='ls -F'

$ ls
file0  file1*  link.file@  mydir/
$ \ls
file0  file1  link.file  mydir

$ ls
file0  file1*  link.file@  mydir/

$ unalias ls
$ ls
file0  file1  link.file  mydir
```

3.3 Creating aliases in `.bashrc`

Aliases are usually set in the users `bash` configuration file `.bashrc`. This file is sourced when a new shell is started.

```
$ tail -n 5 .bashrc ↩
alias ll='ls -l'
alias ls='ls -F'
alias rm='rm -i'
alias emacs='emacs -font 6x13'
```

3.4 Exercise

In an interactive `bash` session:

1. Create an alias for `ls -alF` called `ls`.
2. Try it out on a few directories.
3. Check that `$ \ls ↩` makes `ls` revert to it's normal unaliased behavior.
4. Delete the alias with the `unalias` command.
5. Check that it no longer works.
6. Append an alias to your `.bashrc` file thus:

   ```
   $ echo -e "\nalias ls='ls -alF'\n" >> ~/.bashrc ↩
   ```

7. check that it got there: `$ cat ~/.bashrc ↩`

8. Try it from your current shell (it won't work yet);
9. Source the `.bashrc` file: `$ . .bashrc` ↩
10. Log out with a `C-D` (that's Control D), log back in and try it from a new shell.
11. How does `ls` differ from `$ \ls`? Why is it different?

Part III

Basics

Chapter 4

Navigating the Filesystem

4.1 Basic Filesystem Commands

There are a handfull of commands required for navigating a filesystem:

- `pwd` Print Working Diretory: returns the Current Working Directory
- `cd` Change Directory: Changes your Current Working Directory
- `ls` LiSt: Lists the files and directories in the Current Working Directory

Commands to create and remove files and directories:

- `touch` Create or update the access and modification time of a file
- `rm` Remove a file (or directory)
- `mkdir` Create a directory
- `rmdir` Remove a directory

4.2 Exercise in navigating a filesystem

Login to the system at a text console as a user. Do not do these or any other exercise logged in as root.

1. Find out where you are with `pwd`
   - Print the Working Directory; `$` **`pwd`** ↩ prints your current absolute path. e.g. `/home/margrert/exercises`
   - after you login as root you will you find yourself in the `/root` directory (or on some systems in the / directory)
   - if you login as ”foo” you will find yourself at `/home/foo`
2. Looking at files and directories with list `ls`
   - change to the root directory `$` **`cd /`** ↩
   - look: `$` **`ls`** ↩ ; you see a simple list of files and directories

- list them one to a line `$` **`ls -l`** ↩ (`-l` for long) Note: `ll` is often included as an alias to `ls -l`. Does it work on your system?
- note that the ones starting with a `d` are directories, the ones starting with a `-` are regular files and the ones starting with an `l` are links
- hidden files (those starting with a `.` (dot)) can be viewed using the `-a` option; `$` **`ls -al`** ↩ to view a long listing including hidden files

3. Changing your current working directory with `cd`

   - the change directory command is used e.g. `$` **`cd /usr/share/doc`** ↩ this will change your working directory to `/usr/shave/doc` (btw what is stored there? you might find it useful later)
   - change back to the root `$` **`cd ..`** ↩ steps back to `/usr/share` and `$` **`cd ../..`** ↩ steps back to `/`
   - change to `/lib/kbd/keymaps/i386/qwerty` and explore: some where here are the key maps for your system. have a look `$` **`ls`** ↩
   - move the root directory with `$` **`cd /`** ↩
   - move to your home directory with `$` **`cd`** ↩ (shortcut)

4. Creating and removing directories

   - change directory to your home directory `$` **`cd ~`** ↩ will also do this
   - make a directory called bush; `$` **`mkdir bush`** ↩
   - admire your bush; `$` **`ls -l`** ↩
   - make a directory under bush called branch `$` **`mkdir ./bush/branch`** ↩
   - make a directory called twig; `$` **`cd bush`** ↩ then `$` **`mkdir twig`** ↩
   - make a file in twig called leaf; `$` **`cd twig`** ↩ then

     ```
     $ cat > leaf ↩
     dum de dum...  ↩
     stuff to go in leaf ↩
     ^D ↩
     ```

   - have a look in leaf: `$` **`cat leaf`** ↩
   - see what you've got `$` **`ls -l`** ↩
   - remove it `$` **`rm -r *`** ↩ (`-r` for recursive, i.e. down the branches)
   - see what you've got left `$` **`ls -l`** ↩
   - back up and remove the bush `$` **`cd ..`** ↩ then `$` **`rmdir bush`** ↩

4.2.1 Questions

1. What does the `-l` do in `$` **`ls -l`** ↩ ?
2. What does rm stand for?
3. List three cd commands that get you to you home directory.

4. pwd stands for:
5. How do you list hidden files?
6. How do you know if a file is a hidden file or not?
7. What command would you use to make a directory called `$` **`/something`** ↩ ?
8. What command would you use to remove a directory called `/something`?
9. What does `$` **`cd /`** ↩ do?
10. What does the command `$` **`cat >some.text`** ↩ do?
11. What does STDOUT usually represent?
12. What does STDIN usually represent?

Chapter 5

Startup and Shutdown

5.1 Startup

There are a few ways of starting Linux

5.1.1 boot floppy

- if you made a boot floppy during installation you may use it to start Linux.
- you can make a new boot floppy at any time
  - find out which kernel you are using:
    ```
    # uname -a
    Linux bim 2.2.5-15 #1 Mon Apr 19 23:00:46 EDT 1999 i686
    ```
  - make a boot disk:
    ```
    # mkbootdisk --device /dev/fd0 2.2.5-15
    ```
  - information about bootdisks:
    ```
    $ man mkbootdisk
    ```

5.1.2 LILO in the master boot record

If you installed LILO in the MBR this easiest way of starting Linux

- At the startup boot prompt type
  ```
  boot: linux
  ```

5.1.3 LILO in the first sector of the boot partition

This method requires a boot manager to be set up to boot to LILO

- Select the Linux partition from your boot loader

5.2 Shutdown

There are a number of acceptable ways of shutting down Linux and two unacceptable ways
Note that you must be root to shut down the system

5.2.1 The shutdown command

- `# shutdown -h now` halts the computer after an orderly shutdown starting now.
- `# shutdown -h 5` halts the computer after an orderly shutdown starting in five minutes. It is possible to broadcast a warning to any users that are logged on at the time.
- `# shutdown -r now` restarts the computer after an orderly shutdown starting now.
- `# halt` halts the computer after an orderly shutdown starting now
- `# reboot` reboots the computer after an orderly shutdown starting now
- `Cntl+Alt+Del` halts the computer after an orderly shutdown starting now.

5.2.2 Ways not to shutdown

- Turning the power off stops Linux now and may damage the file system and involve a loss of data.
- Pressing the reset switch stops Linux now and may damage the file system and involve a loss of data.

Chapter 6

Setting Up User Accounts

Six ways of adding user accounts

1. `useradd, passwd` — Fast command line utility
2. editing `/etc/passwd` — Hack the files
3. `adduser` — Interactive command line utility (not on Red Hat)
4. `linuxconf` — Graphical method
5. RedHat GUI tool: `redhat-config-users`
6. KDE GUI tool: `kuser`

6.1 Using the useradd utility

The useradd utility suitable for adding users quickly (in bulk or often).

- A new user may be added thus:`# useradd jblogs`
- The `m` option in `# useradd -m jbloggs` creates a home directory for jbloggs at `/home/jbloggs` and copies all the files in `/etc/skel` to it
- Control over `/etc/passwd` entries comes from other `useradd` options
  e.g. `#useradd -m -d /home/joeB -g 511 -c ``Joe Bloggs''jbloggs`
  To look at the results of this command in `/etc/passwd`:
  ```
  # grep jbloggs /etc/passwd
  jbloggs:!:1003:100:Joe Bloggs:/home/jbloggs:
  ```
- `# passwd jbloggs` will replace the “!” with a password for joe

6.1.1 Practical Exercise—Add a user account for yourself

Follow these steps to add a user.

1. Add the user and setup the home directory:

   ```
   # useradd quincy
   ```

2. Give the new user a password:

   ```
   # passwd quincy
   Enter new UNIX password:
   Retype new UNIX password:
   passwd: password updated successfully
   ```

3. Logon as the new user to test the setup:

   ```
   $ su - quincy
   ```

6.2 Editing `/etc/passwd`

The text file `/etc/passwd` may be edited directly using the editor of your choice. This method is the most basic and would normally be used only as a last resort or on a specialized simple system.

This does not work as expected if your system uses shaddow passwords. The `/etc/password` file and `/etc/group` files have to be converted to not use shaddow passwords before hand editing. They may then be converted back.

- Start the editor e.g. `# vi /etc/passwd`
- Add a line at the end starting with the new user name, say `jbloggs`
- Add the rest of the details regarding group, shell etc. and save
- Give the user a password: `# passwd jbloggs`
- Warning! Using the `# passwd` by it's self will change the root password... *don't do it!* please.

6.2.1 Practical Exercise for adding a new user manually

Follow these steps to add a user.

1. Open the file in an editor:

   ```
   $ su -c 'vi /etc/passwd'
   Password:
   ```

2. Add [1] a line for joe at the end. Be sure to add a name that does not already exist and choose a UID and GID that are also new.

   ```
   joe:!:510:510:Joe Blow:/bin/bash
   ```

3. Edit the file /etc/group and add a line for joe:

   ```
   joe:x:510
   ```

4. Make a home directory for joe:

[1]In vi `i` enters insert mode and `<ESC> ZZ` will save and exit.

```
# mkdir /home/joe
```

5. Test the new account by loging jo in:

```
$ su - joe
Password:
joe@mintie:~$ pwd
/home/joe
joe@mintie:~$
```

6.3 The adduser utility

This is an interactive utility for adding users and setting up their accounts. Not available in Red Hat distributions. Even if it looks as though it's there.

- In Debian:

```
$ ls -l /usr/sbin/adduser
-rwxr-xr-x    1 root     root         23466 Sep 12 06:08 \
/usr/sbin/adduser*
```

- In Red Hat 7.0:

```
$ ls -l /usr/sbin/adduser
lrwxrwxrwx    1 root     root             7 Feb 16 12:16 \
/usr/sbin/adduser -> useradd
```

6.3.1 Practical Exercise (not on Red Hat)

If your distribution supports it use the adduser utility to add a user account. It should go something like this:

```
bash-2.04$ su -c "adduser fred"
Password:
Adding user fred...
Adding new group fred (1005).
Adding new user fred (1005) with group fred.
Creating home directory /home/fred.
Copying files from /etc/skel
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for fred
Enter the new value, or press return for the default
        Full Name []: Fred Foo
        Room Number []: c222
        Work Phone []: 1234 5678
        Home Phone []: 4321 8765
        Other []:
Is the information correct? [y/n] y
bash-2.04$
```

6.4 The linuxconf GUI utility

`linuxconf` is a graphical utility that can be used to set up user accounts.

- Open Config - User Accounts, select Normal then User Accounts
- Select Add and enter the users details
- Select Accept and enter the users password and then Accept again

6.4.1 Practical Exercise

Use `linuxconf` to setup an account for fred.

1. Start linuxconf in X:

```
# linuxconf &
```

2. Click on the little triangles to open up User accounts / Normal / User accounts
3. Click the add button.
4. Add a login name and the full name.
5. Click the accept button.
6. Give fred a password.
7. Logon as fred from another terminal.

6.5 GUI tools

Click and explore then add two users using each of these commands:

- `redhat-config-users`
- `kuser`

Part IV

Filesystem Basics

Chapter 7

Using DOS floppies with `mtools`

note 1: These tools are in `/usr/bin`

note 2: Documented in `man mtools`, Sobell page 798 and in Linux Unleashed page 1207.

note 3: Obviates the need to `mount` and `umount` a DOS floppy file system

7.1 `mtools`

The `mtools` commands mimic DOS commands and are useful for working with DOS files on a floppy disk—for example

- `mcd` – change directory on the DOS disk
- `mcopy` – copies DOS files between directories
- `mdel` – deletes DOS files
- `mdir` – lists DOS directories
- `mformat` – adds DOS formating to a disk
- `mtype` – cats a DOG file

7.2 Exercise in using `mtools`

This section assumes:

- the current working directory is your home directory
- you have an unmounted floppy in the drive and that the floppy has on it the following directories and files:

```
a:\baz
a:\foo\bar\penguin.txt
a:\foo\bar\rubb.ish
```

- you are logged on as fred and you are in your home directory

7.2.1 mdir

Check what's on the DOS floppy:

```
[fred@fang fred]$ mdir
 Volume in drive A is MTOOLS
Directory for A:/

foo          <DIR>     08-25-1999  17:15
baz          <DIR>     08-25-1999  17:15
        2 files                   0 bytes
                          1 452 032 bytes free
[fred@fang fred]$
```

7.2.2 mcd

Move around the DOS floppy:

```
[fred@fang fred]$ mcd foo/bar
[fred@fang fred]$ mdir
 Volume in drive A has no label
Directory for A:/foo/bar

.            <DIR>     08-25-1999  17:15
..           <DIR>     08-25-1999  17:15
penguin  txt      3998 08-16-1999  10:35
rubb     ish        81 08-25-1999  18:08
        4 files               4 079 bytes
                          1 452 032 bytes free
```

7.2.3 mtype

Look at the contents of a file:

```
[fred@fang fred]$ mtype a:/foo/bar/rubb.ish
This is just a load of rubbish.
Not good for anything..
Just delete it!
Now!!!
```

7.2.4 mcopy

Copy a file to your current working directory:

```
[fred@fang fred]$ mcopy a:/foo/bar/rubb.ish ~/
[fred@fang fred]$ ls
Desktop    latex            nsmail  os2.99-1  perl      tomsrtbt
junk.text  mike.lake.latex  os1     os2.99-2  rubb.ish
```

7.2.5 `mdel`

Delete files from a DOS floppy:

```
[fred@fang fred]$ mdel rubb.ish
[fred@fang fred]$ mdir
 Volume in drive A has no label
Directory for A:/foo/bar

.            <DIR>     08-25-1999  17:15
..           <DIR>     08-25-1999  17:15
penguin  txt      3998 08-16-1999  10:35
        3 files               3 998 bytes
                          1 452 032 bytes free
```

7.2.6 `mformat`

format a floppy with a DOS file system:

```
[fred@fang fred]$ mformat
Mtools version 3.9.1, dated 14 May 1998
Usage: mformat [-V] [-t tracks] [-h heads] [-s sectors] [-l label] [-n serialnumber] [
[fred@fang fred]$ mformat a:
[fred@fang fred]$
```

7.3 Further Information... rtfm

Information about `mtools` *(and many other commands and utilities) can be found in the following manner.*
Try these out on the command of your choice.

7.3.1 usage

A mistake in the usage of a command may result in a usage message.
Here the incorrect option -xxx is used to provoke a usage response:

```
[fred@fang fred]$  mcopy -xxx
mcopy: invalid option -- x
Mtools version 3.9.1, dated 14 May 1998
Usage: mcopy [-tnmvV] sourcefile targetfile
       mcopy [-tnmvV] sourcefile [sourcefiles...] targetdirectory
```

7.3.2 `man` pages

A small section of a `man` *page is shown below:*

```
[fred@fang fred]$ man mcopy
...
...
       If  only  a  single,  MS-DOS  source parameter is provided
       (e.g. "mcopy a:foo.exe"), an implied  destination  of  the
       current directory ('.') is assumed.

       A filename of '-' means standard input or standard output,
       depending on its position on the command line.

       Mcopy accepts the following command line options:

       b      Batch mode. Optimized for  huge  recursive  copies,
...
```

7.3.3 info pages

This facility displays the man pages in emacs running in info mode.

```
[fred@fang fred]$ info mcopy
...
...
       Mcopy accepts the following command line options:

       b      Batch mode. Optimized for  huge  recursive  copies,
              but less secure if a crash happens during the copy.

       /      Recursive copy.  Also copies directories and  their
              contents

       p      Preserves the attributes of the copied files
...
```

7.3.4 `usr/doc/mtools-3.9.1`

Change to the documentation directory and look at some of the documentation

```
[fred@fang fred]$ cd/usr/doc/mtools-3.9.1
[fred@fang mtools-3.9.1]$ ls
COPYING  Changelog  README  Release.notes  mtools.texi
[fred@fang mtools-3.9.1]$ cat README |less
...
```

Chapter 8

Journaling filesystems

8.0.5 Gather information about your filesystems

The command `$ df -hT` shows information about the filesystems. The `-h` option makes the bytecount in human readable form and the `-T` option displays the filesystem types.

```
$ df -hT ↩
Filesystem    Type    Size  Used Avail Use% Mounted on
/dev/hda7     ext2     11G  8.0G  2.5G  77% /
/dev/hda5     ext2     91M  9.7M   76M  12% /boot
/dev/hda1     vfat    5.9G  2.4G  3.4G  41% /win
/dev/hda9     ext3    8.8G  5.4G  2.9G  65% /home
/dev/cdrom iso9660    647M  647M     0 100% /mnt/cdrom
```

Viewing the `/etc/fstab` file will show similar information:

```
$ cat /etc/fstab ↩
```

Note that on this system the `/home` partition has an `ext3` journaling filesystem whereas `/` and `/boot` have `ext2` filesystems.

8.0.6 Convert the root filesystem to `ext3`

1. Determine which partition your root filesystem is on:
2. Add a journal to the root filesystem using the default journal parameters (see `man tune2fs`):

   ```
   $ tune2fs -j /dev/hda5
   ```

Part V

Rescue Disks

Chapter 9

tomsrtbt

9.0 Some small Linux distributions

1. tomsrtbt — the most Linux on one floppy
   `http://www.toms.net/rb/`

2. Offline NT Password and Registry editor
   `http://home.eunet.no/~pnordahl/ntpasswd/`

3. Linuxcare Bootable Recovery Disk (CDROM)
   `http://www.linuxcare.com/`

4. Cyote Linux — Linux Router Project
   `http://www.coyotelinux.com/`

9.1 Installing tomsrtbt onto a floppy

9.1.1 Install from an existing tomsrtbt floppy

- Boot the tomsrtbt floppy.
- Type in `./clone.s` and follow the instructions.

9.1.2 Install from a tomsrtbt download

Download the appropriate tomsrtbt package:

- Linux: tomsrtbt-1.7.218.tar.gz
- DOS: tomsrtbt-1.7.218.dos.zip

Creating a tomsrtbt floppy on Linux

- Unpack your tomsrtbt: `tar zxvf tomsrtbt-1.7.218.tar.gz`
- Change to the tomsrtbt-1.7.218 directory.
- Read the file tomsrtbt.FAQ for installation instructions and general information. `# less tomsrtbt.FAQ` (q to quit)
- Place a blank floppy in the "A:" drive and run the install script: `# ./install`

Creating a tomsrtbt floppy on MSDOS

- Unpack your tomsrtbt: `C:\> pkunzip tomsrtbt-1.7.361.dos.zip`
- Read the file tomsrtbt.FAQ for installation instructions and general information. `C:\> type tomsrtbt.FAQ |more`
- Place a blank floppy in the "A:" drive and run the install script: `C:\> install`

9.2 Booting tomsrtbt

Note: It only really makes sense to use tomsrtbt logged in as root.

- Place the tomsrtbt floppy in the floppy disk drive and turn the computer on.
- Hit ⟨Enter⟩ at the "`boot:`" prompt.
- Hit ⟨Enter⟩ to select a video mode to suit your eyes and screen. (0 is for standard VGA or you could try 6)
- Select the default keyboard at the next prompt: ⟨Enter⟩
- At the "login:" prompt login as root using the password xxxx.
- Check that there are four virtual consoles:
  Alt-F1, Alt-F2, Alt-F3 and Alt-F4

9.3 Explore the virtual consoles

Linux is a multiuser operating system. You can have multiple logins using virtual consoles. Try out the four virtual console on `tomsrtbt`.

- Boot `tomsrtbt` and login (this will be on the first virtual console);
- do something, e.g. `# ls`;
- Change to the second virtual console (VC) by pressing `<Alt-A2>` and login to it;
- start the editor called vi in this new VC:
  - press the letter "i" to enter insert mode (I knew that :p)
  - have a scribble on the page "now is the time dum de dum...)

  - how do you save, where is the help? how do you stop it?
  - intuitive huh? devils work, that's why they call it 6 :)
  - we'll get back to it;

- crank up another VC `<Alt-F3>` and log on;
- start the emacs editor: `# emacs` ... oh *yes*, this looks goood!
- type something: ` ``throw your pants in the air, and pretend you just don't care''`;
- god uses this editor for all her really important work;
- Close you emacs session with `C-X C-C` (that's a Control X followed by a Control C.
- move around between the various virtual terminals by pressing `<Alt-F2>`, `<Alt-F3>`, `<Alt-F4>`
- change back to your original login shell on VC-1 by pressing `<Alt-F1>`

9.4 Command line gibberish

- Occasionally your console may print gibberish after being switched to the upper ASCII character set. To see what this looks like try catting a binary file. For example: `# cat /usr/bin/dmesg`. You can switch between character sets with:

```
# echo -e "\016"      # readable to gibberish
# echo -e "\017"      # gibberish to readable
```

- Try out these commands by typing them at the prompt.

9.5 Replacing LILO in the mbr with tomsrtbt

On occasions you may need to replace LILO in the master boot record of your hard disk. This may happen after a reinstall of WinThing or running the MSDOS command `C:\> FDISK /MBR`.

Normally you place LILO in the master boot record using the boot disk you made during installation. However if your Linux boot rescue disk is lost or damaged you can use tomsrtbt or some other system on bootable media to fix your mbr.

This procedure assumes that your Linux root partition is on /dev/hda5. You may need to run fdisk to determine which partition is root for your system.

1. Boot the tomsrtbt system.
2. Mount the Linux root partition: *(YMMV)*

```
# mount -t ext2 /dev/hda5 /mnt && sync
```

3. Change the root of the filesystem to be the partition you just mounted and start a shell from the system on the hard disk:

```
# chroot /mnt /bin/bash2
```

4. Run the LILO command:

```
# lilo
```

5. Kill the shell running from the system on the hard drive:

```
# exit
```

6. Unmount the filesystem and reboot from the hard drive:

```
# umount /mnt && shutdown -r now
```

The system should now boot using LILO from the hard drive.
Alterntive method, enter: `/mnt/sbin/lilo -r /mnt`

9.6 using fdisk with tomsrtbt

One important administrative task that requires a tool like tomsrtbt is using fdisk to look at and alter disk partitions. It is very easy to trash your hard disk, so be _very_ careful.
Warning: For the purposes of this exercise you must quit fdisk with a q: quit *without* saving changes.

9.6.1 exercise using fdisk

- Start fdisk: `# fdisk /dev/hda`
- Check the menu: m
- Look at partitions on your disk: p
- List the possible partitions types: l
- Quit without saving any changes: q

9.7 Using tomsrtbt to edit a file on a data floppy

9.7.1 mount a data floppy disk

- Once tomsrtbt has booted and you have logged in remove the tomsrtbt floppy and replace it with a blank MSDOS formatted floppy.
- Mount the floppy:

```
 # mount -t msdos /dev/fd0 /fl
```

- Check that it's mounted:

```
# mount
...
/dev/fd0 on /fl type msdos (rw)
```

9.7.2 Editing a file on the floppy

- Change your cwd to the floppy disk and have a look around:

```
# cd /fl
# ls
```

- Add a directory and change to it:

```
# mkdir mydir && cd $_
```

- Open a new file in Chet's emacs

```
# ce answers.txt
```

- Answer the questions in the section headed Questions below by typing the answers into the file `answers.txt` and save it.

9.8 Questions

Answer these questions by typing the answers into a file created on a floppy disk using Chet's emacs.

1. Which Linux kernel does tomsrtbt use? (Hint: `# dmesg | more`)
2. What is different about the sbin directory as compared with the other first level directories?
3. In which directory is the dd command stored?
4. What does the mount command show when issued with no arguments?
5. List the subdirectories under /usr.
6. Are there many man pages on a tomsrtbt? Hint: find them and have a look.
7. In which directories are most of the Linux commands kept?
8. List the files in the directory /usr/doc.
9. What type of files are kept in /etc?
10. What type of hard disk is on your system? Hint: have a look at the boot up messages. `# dmesg |less`

Part VI

Manipulating Text

Chapter 10

UNIX and DOS Line Endings

10.1 Text Files on Various Operating Systems

Unix, DOS and Mac use different characters to end each line of text. So text files have to be "translated" between the three operating systems.

An ASCII carriage return `<cr>` is a hexadecimal 0x0D.

An ASCII line feed `<lf>` is a hexadecimal 0x0A.

10.1.1 End of Line Characters

- A DOS, WINDOWS 3.x, 95, 98, me, NT, 2000 file looks like this:

```
first line<cr><lf>
second line<cr><lf>
<cr><lf>
last line<cr><lf>
^Z
```

- A UNIX / Linux file looks like this:

```
first line<lf>
second line<lf>
<lf>
last line<lf>
```

10.1.2 Text Conversions

Reference: `http://kb.indiana.edu/data/acux.html`

There are many tools that may be used to convert text files from UNIX line endings to DOS line endings and vice versa. The Control-Z required at the end of a DOS text file may be added from the command line thus:

```
$ echo -en "\32" >> dosfile.txt
```

You can view a text file with this command:

```
$ od -bc textfile
```

Should you need to do these conversions using tomsrtbt use awk as the other tools are not available or broken.

UNIX to DOS

sed: `$ sed 's/$/^M/' unixfile.txt > dosfile.txt`
Note: the `^M` is produced by `C-V C-M`

awk: `$ awk 'sub("$", "\r")' < unixfile.txt > dosfile.txt`

Perl: `$ perl -p -e 's/$/\r/' < unixfile.txt > dosfile.txt`

emacs: `M-% C-q C-j RET C-q C-m C-q C-j RET !`
Note Add a Control-Z at the end of the file in emacs with C-q C-z.

mcopy: `$ mcopy -t  unixfile.txt a:/dosfile.txt`

DOS to UNIX

tr: `$ tr -d '\15\32' < dosfile.txt > unixfile.txt`

sed: `$ sed 's/^M//' dosfile.txt > unixfile.txt`

awk: `$ awk '{ sub("\r$", ""); print }' dosfile.txt > unixfile.txt`

Perl: `$ perl -p -e 's/\r$//' < dosfile.txt > unixfile.txt`

emacs: First open the file in emacs using the *find-file-literally* option:

```
$ emacs ↩
M-% find-file-literally ↩
Find file literally: ~/my.MS_DOG.txt ↩
M-% C-q C-m RET RET !
```

Then delete the `^Z` at the end of the document if there is one.

Note: Liberal use of the `<TAB>` key in the emacs mini-buffer will make your life in emacs easier.

mcopy: `$ mcopy -t  a:/dosfile.txt unixfile.txt`

Exercise in Converting Text Files Between Unix and DOS

- Create a text file (`^D` is a C-D)

```
$ cat >text.unix
dum de
dum
^D
```

- View it: (check out `$ man od`)

```
$ od -bc text.unix
0000000 144 165 155 040 144 145 012 144 165 155 012
          d   u   m       d   e  \n   d   u   m  \n
```

- Convert the line endings with sed:

```
$ sed 's/$/^M/' text.unix > text.dos
```

- Add a ^Z to the end:

```
$ echo -en "\32" >> text.dos
```

- Have a look:

```
$ od -bc text.dos
0000000 144 165 155 040 144 145 015 012 144 165 155 015 012 032
          d   u   m       d   e  \r  \n   d   u   m  \r  \n 032
```

- Open emacs.
- Visit the MS-DOG formatted file:

```
M-% find-file-literally ↩
Find file literally: ~/my.MS_DOG.txt ↩
```

- Convert the file back to Unix format:

```
M-% C-q C-m RET RET !
```

- Try out the other tools to convert between Unix and DOS line endings.

10.2 Questions

Answer these questions by typing the answers into a file created on a floppy disk using Chet's emacs.

1.

Chapter 11

Manipulating Text

11.1 Displaying Text

At a bash command prompt in a virtual terminal or in an xterm follow these steps.

- Create a file called letters containing all the letters of the alphabet, lower case then uppercase. The file should be 52 lines long.

```
$ cat > letters ↩
a
b
c
...
Z
^D
```

- Count the lines in the file:

```
$ wc -l letters ↩
```

- Display the file using `cat`:

```
$ cat < letters ↩
```

- Display the file a page at a time using `more` (the space bar displays the next page:

```
$ more letters ↩
    or
$ more letters ↩
```

- Display the file a page at a time using `less` (the arrow keys move both up and down the file, the q key to quit:

```
$ less letters ↩
```

- Display only the first 10 lines:

```
$ head letters ↩
```

- Display only the last 10 lines:

```
$ tail letters ↩
```

- System administrators can monitor updates to the system log files using a command like this (note that I used a \ character to continue my command on the next line:

```
$ su -c 'tail -f -n5 \
/var/log/messages' ↩
```

- List the first 12 lines with line numbers:

```
$ head -n12 letters | nl ↩
```

```
$  ↩
```

11.2 Create a text file and manipulate the text

At a bash *command prompt in a virtual terminal or in an xterm follow these steps.*

- Create a file called fruit containing the following text:

```
$ cat > fruit ↩
blood plum
nashi pear
delicious apple
sugar banana
sultana grape
valencia orange
seville mandarin
^D
```

- Print out the file to STDOUT:

```
$ cat < fruit ↩
```

Note that this shorthand syntax does the same thing:

```
$ cat fruit ↩
```

- Print the sorted fruit to the screen:

```
$ sort < fruit ↩
```

- Direct the sorted fruit to a new file:

```
$  sort < fruit > sorted.fruit ↩
```

- Check out the new file:

```
$ cat < sorted.fruit ↩
```

- Copy the unsorted fruit to a new file called `two.fruit`:

```
$ cp fruit two.fruit ↩
```

- Add a list of sorted fruit to the `two.fruit` file:

```
$  cat < sorted.fruit >>  \
two.fruit ↩
```

- Look at the file:

```
$ cat two.fruit ↩
```

- Look at the second field of each line (apple, banana etc):

```
$ cut -d' ' -f2 fruit ↩
```

- Pipe the descriptions to sort and save the result:

```
$ cut -d' ' -f1 fruit | sort > \
fruit.descriptions ↩
```

- Have a look to check:

```
$ cat fruit.descriptions ↩
```

- List the user's homes from the sixth field the `/etc/passwd` file:

```
$ cut -d':' -f6 /etc/passwd ↩
```

11.3 Questions

1. What is the token used to add (append) to a file?

2. What command would print the contents of the file `hello.c` to the screen?

3. How can you use `cat` to copy the file `one.a` to the file `two.a`?

4. How do you determine how many words there are in a file? (see `$ man wc`)

5. The file a.file contains a list of products, one per line. What command would give a count of the number of products?

6. What command would print a list of the words beginning each line of the file called `stuff` in reverse alphabetical order? (hint: try `$ man sort`)

7. Three numbers are returned by the command `$ wc my.file`. What do the three numbers represent?

8. If a file is made up of lines with fields separated by colons, how would you save all of the third fields to a file called `thirds.text`?

9. Create a file containing the following names.

   ```
   George Pitman
   Jenni Penny
   Joe Blow
   Mary Contrary
   Antonia Lexis
   ```

   (a) Sort the names in reverse order by the family names.

   (b) Display a count of the characters in the file.

   (c) Save a list of the first names in a file called `names.first`

Part VII

Dealing with Files

Chapter 12

File Compression

Document Description: Exercise in using file various compression utilities.

References Read the man pages for `compress`, `uncompress`, `zip`, `unzip`, `gzip`, `gunzip`, `bzip2`, `bunzip2`, `funzip`, `zipcloak`, `zipgrep`, `zip-info`, `zipnote`, `zipsplit`, `zcat`, `bzcat`, `zless`.

Instructions: Read through these notes and do the practical exercises in each section.

File compression is used to minimise the amount of storage space a file occupies and to reduce the time it takes to be transmitted over a network. Commonly used on archived files for backup and long term storage.

12.1 `compress` and `uncompress`

Files compressed with the utility compress are given the .Z extension.

- Make a text file to practice compression on:

```
$  man man -7 >man.txt ↩
```

- check the size of `man.txt`

```
$ ls -l man.txt ↩
-rw-r--r--  1 geoffrey geoffrey 30095 May 13 18:55 man.txt
```

- compress it:

```
$ compress man.txt ↩
```

- check the size of `man.txt.Z`

```
$ ls -l man.txt ↩
-rw-r--r--  1 geoffrey geoffrey  12874 May 13 18:55 man.txt.Z
```

- View the compressed file with `zcat`:

```
$ zcat man.txt.Z |less ↩
```

- Uncompress the file:

```
$ uncompress man.txt.Z ↩
```

What is the compression ratio between `man.txt` and `man.txt.Z`? How does it compare with the compression of a binary file?

12.2 `zip` and `unzip`

The `zip` and `unzip` utilities are compatible with the MSDOG utilities PKZIP and PKUNZIP (Phil Katz zip.

- Archives *and* compresses (similar to `compress` and `tar`);
- Not very efficient or fast; (= to `compress`)
- ported to most platforms:
  - WinNT
  - Atari
  - Mac OS
  - VMS
  - UNIX
  - OS/2
  - Amiga
- See also `funzip`, `zipcloak`, `zipgrep`, `zip-info`, `zipnote` and `zipsplit`.

12.3 `gzip` and `gunzip`

Files compressed with the utility gzip are given the .gz extension. gunzip can currently decompress files created by gzip, zip, compress, compress -H or pack.

- Make a text file to practice compression on (the -7 may not work on your system; leave it out):

```
$  man gzip -7 > gzip.txt ↩
```

- check the size of `gzip.txt`

```
$ ls -l gzip.txt ↩
-rw-r--r--  1 geoffrey geoffrey 18307 May 13 22:19 gzip.txt
```

- compress it:

```
$ gzip gzip.txt ↩
```

- check the results of thegzip compression.

```
$ ls -l .txt ↩
-rw-r--r--  1 geoffrey geoffrey  6404 May 13 22:19 gzip.txt.gz
```

- View the compressed file with `zcat`:

```
$ zcat gzip.txt.gz |less ↩
```

- list the details of the compression:

```
$ gzip -l gzip.txt.gz
compressed  uncompr. ratio uncompressed_name
     6404     18307  65.1% gzip.txt
```

- Uncompress the file:

```
$ gunzip gzip.txt.gz ↩
```

12.4 bzip2 and bunzip2

The `bzip2` compression utility is an advanced high performance compression utility. It produces files with the `.bz2` extension.

- Make a text file to practice compression upon:

```
$  man bzip2 -7 >bzip2.txt ↩
```

- check the size of `bzip2.txt`

```
$ ls -l bzip2.txt ↩
-rw-r--r--  1 geoffrey geoffrey 19367 May 13 22:56 bzip2.txt
```

- compress it:

```
$ bzip2 bzip2.txt ↩
```

- check the results of thebzip2 compression.

```
$ ls -l bzip2.txt.bz2 ↩
-rw-r--r--  1 geoffrey geoffrey  6469 May 13 22:56 bzip2.txt.bz2
```

- View the compressed file with `zcat`:

```
$ bzcat bzip2.txt.bz2 |less ↩
```

- Uncompress the file:

```
$ bunzip2 bzip2.txt.bz2 ↩
```

- The `bz2recover` utility may be used to recover data from damaged `bz2` compressed files.

Chapter 13

`tar` basics

Document Description: exercise in using the tape archive command tar

References `man tar` and `info tar`

Instructions:

- Read through these notes on tar;
- Do the practical exercise in section 13.4

13.1 tar—from the GNU man page

tar is an archiving program designed to store and extract files from an archive file known as a tarfile or tarball. A tarfile may be made on a tape drive, however, it is also common to write a tarfile to a normal file.

13.2 Simple example: creating then extracting a tarball

- Make an archive file called some.tar of all the files and directories recursively under a directory called foo-dir: (`c`—create, `v`—verbose, `f`—file)

  ```
  $ tar cvf foo.tar foo-dir/ ↩
  ```

- Extract all the files and directories from `foo.tar` into the `/tmp` directory: (`x`—extract, `v`—verbose, `f`—file)

  ```
  $ foo.tar /tmp ↩
  $ cd /tmp ↩
  $ tar xvf foo.tar ↩
  ```

13.3 Options

13.3.1 tar Function Letters

The tar options must include one and only one of the following function letters:

- `-A, --catenate, --concatenate` append tar files to an archive
- `-c, --create` create a new archive
- `-d, --diff, --compare` find differences between archive and file system
- `--delete` delete from the archive (not for use on mag tapes!)
- `-r, --append` append files to the end of an archive
- `-t, --list` list the contents of an archive
- `-u, --update` only append files that are newer than copy in archive
- `-x, --extract, --get` extract files from an archive

13.3.2 A few popular tar options

There are dozens of options for tar, these are only a few of them.

- `-v, --verbose` verbosely list files processed— *always use this*
- `-f, --file [HOSTNAME:]F` use archive file or device F (default "–", meaning stdin/stdout)— *always use this*
- `-z, --gzip, --ungzip` filter the archive through gzip—*very frequently used*
- `-j --bzip`—filter the archive through bzip2, use an extention of `.tar.bz2`

13.4 Practical Exercise

1. Peruse the man page and the info pages for the tar command. Memorise the options. (just kidding)
2. Archive your home directory (say as a backup) into a compressed file called mybackup.tar.gz. List the files in your tarball. Then extract the tarball into the `/tmp` directory.
   - Move outside the directory you are going to archive (avoid recursion) then make a tarball in your current working directory:
     ```
     $ cd /tmp ↩
     $ tar -czvf mybackup.tar.gz ~ ↩
     ```
   - List the files in the archive:
     ```
     $ tar -ztvf mybackup.tar.gz ↩
     ```
   - Extract the archive:
     ```
     $ tar -zxvf mybackup.tar.gz ↩
     ```
   - Inspect the extracted files: `$ tree ↩`

Part VIII

Installing Software

Chapter 14

Installing software form a source `tarball`

14.1 the tarball

Open source UNIX and Linux software is frequently supplied archived and compressed in what is coloquially known as a tarball.

A tarball is a source code tree that has been archived with tar and compressed with gzip.

```
$ ls *.tar.gz *.tgz ↩
ls: *.tgz: No such file or directory
wv-0.6.7.tar.gz
```

Follow these steps to install software from a tarball:

- First obtain your tarball, say by googleing for it.
- Copy the tarball to a suitable location, say `/tmp`.
- Uncompress and unarchive the software into a source tree.
- Change directory into the base of the source tree.
- Configure the `Makefile`.
- Compile (`make` the executable)
- Install the software.
- Enjoy :)

14.2 source tree

- Make a copy of your tarball in `/tmp`

  ```
  $ wv-0.6.7.tar.gz /tmp ↩
  $ d $_ ↩
  ```

- Make the source tree:

```
$ tar zxvf wv-0.6.7.tar.gz /tmp ↩
```

- Have a look at the source tree:

```
$ tree -L 3 -d wv-0.6.7 ↩
wv-0.6.7
|-- CVS
|-- Documentation

|-- wingdingfont
|   `-- CVS
`-- xml
    `-- CVS

$ tree -d wv-0.6.7 ↩
...

$ ls wv-0.6.7 ↩
config.h.in   iconv/               sep.c
config.sub*   install-sh*          shd.c
configure*    laolareplace.c       sprm.c
configure.in  laolareplace.old.c   sprmtest
```

14.3 compile and install

- Change the cwd to the source tree:

```
$ cd wv-0.6.7 ↩
```

- Run the `configure` script:

```
$ ./configure ↩
creating cache ./config.cache
checking for gcc... gcc
checking whether the C compiler (gcc) works..yes
...
```

- Compile using the `make` utility:

```
$ make ↩
making oledecod in oledecod
make[1]: Entering directory `/tmp/wv-0.6.7/o
...
```

- Install using the `make` utility:

```
$ su -c 'make install'↩
Password: ↩
gcc -g -O2  -DHAVE_CONFIG_H  -I/usr/include/
glib-1.2 -I/usr/lib/glib/include -ansi -pedantic
...
```

14.4 Using the application

1. Locate a MS Word formatted document:

```
$ ls ↩
MS_Word_File.doc

$ file MS_Word_File.doc ↩
MS_Word_File.doc: Microsoft Word document data
```

2. Have a look at the document with Abiword if it is installed on you r system:

```
$ AbiWord MS_Word_File.doc ↩
```

3. Check which filters are available to you with `wv`:

```
$ wv <TAB> <TAB>↩
wvAbw          wvHtml         wvPS           wvText
wvCleanLatex   wvLatex        wvRTF          wvVersion
wvConvert      wvMime         wvSimpleCLX    wvWare
wvDVI          wvPDF          wvSummary      wvWml
```

4. Convert the file to a plain ASCII text:

```
$ wvText MS_Word_File.doc MS_Word_File.txt ↩
$   ↩ less MS_Word_File.txt
```

5. Convert the file into HTML:

```
$ wvHtml MS_Word_File.doc MS_Word_File.html ↩

$ netscape MS_Word_File.html & ↩
```

6. Convert the file into PostScript:

```
$ wvPS MS_Word_File.doc MS_Word_File.ps ↩

$ gv MS_Word_File.ps & ↩
```

7. Convert the file into a PDF:

```
$ wvPDF MS_Word_File.doc MS_Word_File.pdf ↩

$ xpdf MS_Word_File.pdf & ↩
```

Chapter 15

Using `rpm`s and the RedHat Package Manager

Part IX

Emacs

Chapter 16

emacs tutorial

This Document: Guide to the learn-by-doing built in emacs tutorial.

16.1 what to do

- Read the information in sections 2 and 3.
- Practice opening GNU emacs at a text console and in an X terminal (section 4).
- Work through the practical exercise in section 5.
- answer the questions in section 6.

16.2 emacs key naming conventions

C-x hold the control key while pressing the x key.

M-x either:

- hold down *Meta* or *Alt* key while pressing the x key.
- or press and release the ⟨*ESC*⟩ key then press x key.

16.3 A Few Essential Commands

C-x C-c exit the program

C-x C-s save

C-g cancel what you are doing

C-_ ***or*** **C-/** undo

Delete ***or*** **Backspace** delete the character to the left of the cursor

C-d delete the character under to cursor

C-@ ***or*** **C-space** set the mark

C-w kill text from mark to point to the kill ring

M-w copy text from mark to point to the kill ring

C-y Yank text from the kill ring to the point

C-x C-f find (open) a file

C-x 1 close all but this buffer

16.4 Starting GNU emacs

1. In a Linux text console type the command to start emacs.

   ```
   $ emacs
   ```

2. Close emacs (C-x C-c) and start the X Window System:

   ```
   $ startx
   ```

3. In an X terminal type the command to start emacs. Note the ampersand following the command which runs the program in the background.

   ```
   $ emacs &
   ```

4. Start another copy of emacs from the Window Manager's menu system.
5. Close your emacsen either by killing them or C-x C-c.

16.5 Exercise: Breaking the Ice

- Open emacs `$ emacs`
- Read the survival guide in the scratch buffer.
- Remember: C-g if you make a mistake and C-x C-c to exit.
- Explore the menus by clicking on the menu bar and moving the mouse pointer around.
- Open the calendar from the Tools menu, and move around in it using the arrow keys.
- Close the calendar window: C-x 0 (that's zero).
- Open the emacs tutorial: C-h t. You don't need to read it all just yet.
- Move around the document using the arrow keys and the page up and down keys.
- Delete the first paragraph by highlighting it with your mouse cursor and then hitting the delete key. (C-w will also kill the marked text and copy it to the kill ring)
- Get it back with undo: C-_ or C-/.

- Place your cursor in the middle of a line. Delete five characters to the left with your Backspace key then delete five characters to the right with C-d.
- Get back your ten deletions by pressing C-_ ten times.
- Try a forward search:
  - If you dont have the emacs tutorial open then open it with C-h t
  - Move the curser to very top of the tutorial document.
  - Type C-s to open the I-search:
  - type the letter “b”. Note that the cursor has moved to the first letter b in the document.
  - Type the letter “l”, then an “e”. Note how the search progresses.
  - Cancel it with C_g
- Save the file in your home directory with a new name: C-x C-w and type the name of the file say `/junk.text`.
- Delete a line: Move the cursor the start of a line and press C-k.
- Save the altered file: C-x C-s.
- Close emacs C-x C-c.

16.6 Questions

1. What keys are used to delete characters to the right of the cursor?

2. What keys are used to undo a recent command?

3. What is a kill ring?

4. What does this do? `C-x C-c`

5. What do `C-s` and `C-r` do?

6. How do you save your work?

7. What is meant by the "point"?
   In `emacs` what is ment by the term "yank"?

8. What two commands may be used to set a mark?

9. If you pressed `C-x` and wanted to cancel it what do you do?

10. How do you kill a block of marked text and copy it to the kill ring?

11. What is the difference between `C-x 0` and `C-x 1`

16.7 speedbar

The emacs speedbar is useful for browsing the info system.

```
M-x Info-speedbar-browser
```

16.7.1

16.8 the built in tutorial

Start the tutorial with C-h t.
Answer the question on this sheet while working through the tutorial.

16.8.1 Viewing Screens

1. How do you:
   - Move forward one screenful?
   - Move backward one screenful?
   - Centre the screen about the cursor position?

16.8.2 Basic Cursor Control

1. What are the commands to move the cursor
   - to the previous line?
   - backward along the line?
   - forward along the line?
   - to the next line?
2. When the cursor is placed on the last line of the page what happens when your press C-n?
3. What is the difference between pressing C-f and M-f?
4. What key combinations move the cursor to the beginning or end of a line?
5. What do M-a and M-edo ?
6. What do M-¡ and M-¿ do?
7. Why is it better to learn to use the control keys rather than use the arrow keys?
8. What is C-u used for?
9. Which mouse button is used to move the thumb in the scroll bar?

16.8.3 Cancel the Current Command

1. What key combination is used to reset an emacs command?
2. when would you use it?

16.8.4 Disabled Commands

1. Give an example of a disabled command.
2. What are the choices you are given if you use a disabled command key combination?

16.9 Windows

1. C-x 1 does what exactly?

16.10 Inserting and Deleting

16.10.1 Deleting characters

1. Which two keys delete the character to the left of the point?
2. What key combination deletes the character under the point?
3. How would you type a row of 72 *'s across the screen?
4. M-¡Delete¿ and M-d do what?
5. C-k deletes from the cursor to the end of the line, what does M-k do? What might the k stand for?
6.

16.10.2 Marking sections

1. C-@ is one way of marking the beginning of block, what is the other?
2. To what process does "killing" a block of text refer?
3. What defines a block of text?
4. What key combination kills a block?
5. Waft is the difference between *deleting* and *killing*?
6. What does C-w do?
7. To what process does "yanking" text refer?
8. How do you *yank* the third last thing you *killed*?

16.10.3 Undo

1. What does C-x u do?
2. What other key combination does the same as C-x u?

16.11 Files

16.12 Buffers

16.13 Mode Line

Part X

Inroduction

Chapter 17

using vi

Document Description: Exercise in using the vi (**VI**sual) editor and learning a few relevant commands.

References :

1. Running Linux *by* Mat Welsh and Lar Kaufman
2. A Practical Guide to Linux *by* Mark Sobell
3. Learning the vi Editor *by* Linda Lamb & Arnold Robbins

note1 `[ESC]` means press the escape key, redundant `[ESC]` key-presses just beep at you

note2 Generally you will use a vi clone like elvis or vim or emacs viper mode

note3 vi is pronounced as the letters v then i, not vi like the name

17.1 you must be able to use vi

vi is the single most useful (and used) configuration tool used on all flavours of *NIX. All systems have vi, some have no other editor; if you don't have at least a rudimentary knowledge of vi you will get stuck sooner or later.

Besides, learning vi is a *NIX right of passage.

17.2 writing a new file

- start vi thus:
  `$ vi` ↩ or
  `$ vi filename` ↩
- change from command mode to Insert mode: press ”i”
- enter your text, say some c source, a letter or a novel

17.2.1 entering text

Enter the following C language source code:

```
/* first c using vi */
#include <stdio.h>
main()
{
   printf("hi, vee-eye");
   return 0;
}
```

17.2.2 save your work

- enter command mode
  `[ESC]`
- write the name and path of the file
  `:w hi.c ↩`
- subsequent saves:
  `[ESC] :w ↩`

17.2.3 quit from vi

- enter command mode with `ESC` (if vi is already in command mode then it will harmlessly beep)
- `ESC :q ↩` — if your file has been saved already
- `ESC :wq ↩` — saves (writes) and quits
- `ESC :q! ↩` — forces a quit without saving
- `ESC ZZ ↩` — quick save and exit
- look at the results
  `$ cat hi.c | less ↩`

17.3 view a file without changing it

- open the file `/etc/rc.6` (BTW rc stands for run command)
  `$ vi /etc/rc.6 ↩`
- mmmm... you could hack this around and the shutdown command might not work so well, lets not!
- quit with no changes
  `[ESC] :q! ↩`

17.4 edit an existing file

- get a copy of an existing file to experiment with:
  `$ cp /tmp/tomsrtbt.FAQ .` ↩
- open a file in vi
  `$ vi tomsrtbt.FAQ` ↩
- in case we muck it up save it with a new name:
  `[ESC] :w toms.FAQ` ↩
- open the copy
  `[ESC] :e toms.FAQ` ↩
- in command mode you can use the cursor keys to move around; do that, have a browse
- lets delete a line, move the cursor to the line starting with the number 4) and press `dd` (in command mode) - gone
- delete four lines: move the cursor to '10) Tips' and press `4dd`
- yank (copy) two lines, copy from the '2) Design goals' by `[ESC] 2yy` nothing happened? it's okay
- move the cursor to '14)' and press "p" (p for put) this should place the lines 2) and 3) after the 14)
- want to find something, say the word "ftp"?
  `[ESC] :/ftp` ↩ to find the first "ftp" and press `n` to find the next "ftp"
- replace the word "scratch" with the word "itch"?
  `[ESC] :%s/scratch/itch/`
- want to look at the original? save this file `:w` ↩ and open another `[ESC] :e tomsrtbt.FAQ` ↩
- enough?
  `[ESC] :q!` ↩

17.5 emacs viper mode

The other editor (the editor from hell which includes at least two kitchen sinks) has a vi emulation mode—of course.

Start emacs, read the introduction and edit a text file to check it's operation.

- Start emacs: `$ emacs` ↩;
- enter viper mode: `M-X viper` ↩;
- read, read, read;
- open a file: `ESC :e some-file-name-that-exists` ↩;
- check that the vi editing commands work;
- exit emacs / viper

17.6 want to know more?

- `$ man vi` ↩
- `$ info vi` ↩
- `$ vi -h` ↩
- `$ help vi`↩
- Running Linux *by* Mat Welsh and Lar Kaufman
- Chapter 8 of ”A Practical Guide to Linux” Mark G Sobell
- Learning the vi Editor by Linda Lamb & Arnold Robbins
- `vi` Editor Pocket Reference by Arnold Robbins
- Just about any UNIX or Linux book

17.7 Vi Quick Reference

```
ENTERING vi

        vi name         start vi editor with file   name   .
                        The file is created if it doesn't exist.

LEAVING vi

        ZZ              exit from vi, saving changes.
        :q!             exit from vi, discarding changes.

CURSOR POSITIONING

        h               moves left one character position.
        j               moves down one line.
        k               moves up one line.
        l               moves right one character position.
        0       (zero)  moves to the beginning of a line.
        w               moves right one word.
        b               moves left one word.
        CTRL-u          moves up 1/2 screen full.
        CTRL-d          moves down 1/2 screen full.
        G               moves to the bottom of the file.
        nG              moves to line number   n   .
        CTRL-l          clear screen and re-draw.

TEXT MODIFICATION

        itextESC     inserts   text   to the left of the cursor.
                     Insert doesn't cause the cursor to move;
                     text appears as it is typed, terminate with
                     ESC.
        atextESC     appends (inserts) text to the right of
                     the cursor, terminate with ESC.
        RtextESC     Replaces (overprints) characters at the
                     cursor position, terminate with ESC.
        dd           deletes the line the cursor is on.
        ndd          deletes  n  lines from the cursor position.

        D            deletes characters from the cursor position
                     to the end of the line.
        x            deletes the character at the cursor.
        nx           deletes n characters to the right of the
                     cursor.
        u            undo the last change.

PATTERN SEARCHING

        /pat/        positions the cursor at the next
                     occurrence of the string pattern.

NOTES:
        ESC     represents the ESC key.  Press the ESC key when
```

```
                it is called for in the above commands.

        CTRL-   represents the CTRL key.  Hold the CTRL key and
                press the following key simultaneously.

CURSOR POSITIONING

        }       move down one paragraph.
        {       move up one paragraph.
        mx      save the current cursor position and label it
                with the letter x. (x is any letter)
        'x      return to the cursor position labeled x.

TEXT MODIFICATION

        dw      delete the next word.
        .      (period) repeat last change.
        A       append at the end of the current line.
        P       put back deleted line(s).  Text deleted with D
                and dd commands may be pasted back with the P
                command.  Text is pasted in before the cursor
                position.
        :a,bs/old/new/
                From line number 'a' to line number 'b',
                substitute the pattern 'old' with the pattern
                'new'.  You may use any text string which
                doesn't contain a carriage return in place of
                the 'old' and 'new' strings.  Use CTRL-G to tell
                what line the cursor is on.

PATTERN SEARCHING

        //      search for the next occurrence of a previously
                specified search string.

MISCELLANEOUS

        :w      write out current changes.  The vi editor works
                on a copy of your file.  The :w command causes
                the editor to write its copy over the original
                which is on the disk.
        :w name write out changes to the file  name  .  This is
                like the :w command but the changes are written
                into the file you specify.  (good for making
                intermediate copies)
Cut and Paste   Move to the begining of the text to cut.  Use
                dd to delete (cut) several lines.  Use D to cut
                only the end of one line.  Move to the place
                where you wish to paste the text.  Use P to
                put back the text. You may need to clean up
                the spacing after pasting.

http://vertigo.hsrl.rutgers.edu/ug/vi_qref.html
```

Chapter 18

File Permissions

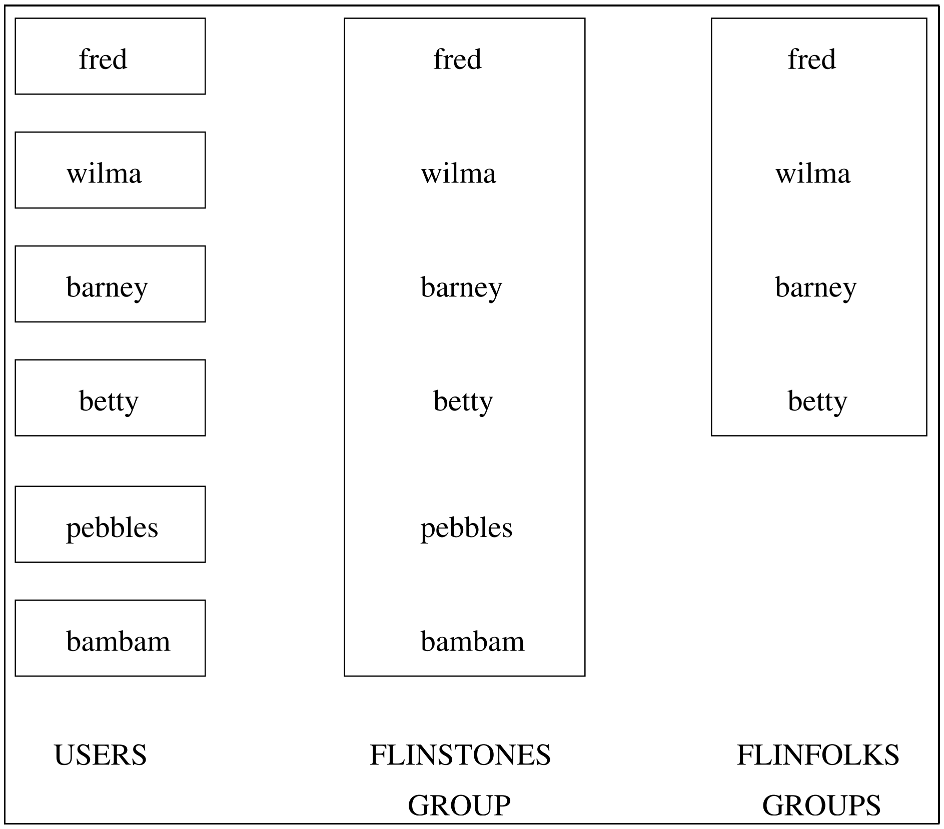
Prepared by Andrew Eager

18.1 File Permissions

- An access control mechanism
- Based on relation between file & user
- Analogy:
  - Documents receive classification
  - Employees receive clearance
  - Access to a particular document is determined by the documents classification and the employees clearance
- A file has 3 modes of access:
  - Read (r) - Can view the file
  - Write (w) - Can change the file
  - Execute (x) - Can run the file (program)
- A file can be accessed by 3 different types of people:
  - The file owner or user (u)
  - A member of the files group (g)
  - Anyone else or others (o)

18.2 Directory Permissions

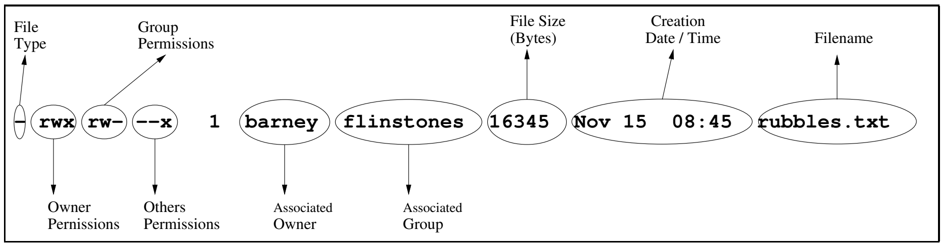
- Directories are treated in the same way as files
- They have an associated owner
- They have an associated group
- The permissions do slightly different things
  - Read (r) - Can view the contents of directory (ls)
  - Write (w) - Can add, delete, rename files
  - Execute (x) - Can 'cd' into the directory and open files in it or its subdirectories



18.3 `ls -l` is your friend

⇒ All of the file's attributes can be examined using the `ls -l` command

```
$ ls -l rubbles* ↩
-rwxrw-r-- 1 barney  flinstones  16345  Nov15  08:45  rubbles.txt
$
```



18.4 Numeric Equivalents

• Each of the permission bits are bitmapped as follows:

FILE TYPE	USER			GROUP			OTHER		
	Read (r)	Write (w)	Execute (x)	Read (r)	Write (w)	Execute (x)	Read (r)	Write (w)	Execute (x)
	400	200	100	40	20	10	4	2	1

- Normal File
d Directory

r	w	x	Value
-	-	-	0
-	-	x	1
-	w	-	2
-	w	x	3
r	-	-	4
r	-	x	5
r	w	-	6
r	w	x	7

18.5 chown & chgrp

- A file's owner can be changed using `chown`:

```
# ls -l rubble.txt ↩
-rw-rw-r--    1 barney   flinstones ...  rubble.txt

# chown fred rubble.txt ↩

# ls -l rubble.txt ↩
-rw-rw-r--    1 fred     flinstones ...  rubble.txt
```

- A file's owner & group can also be changed using `chown`:

```
# ls -l rubble.txt ↩
-rw-rw-r--    1 barney   flinstones ...  rubble.txt

# chown fred:flinfolks rubble.txt ↩

# ls -l rubble.txt ↩
-rw-rw-r--    1 fred     flinfolks  ...  rubble.txt
```

- To change only the group use `chgrp`:

```
# ls -l rubble.txt ↩
-rw-rw-r--    1 barny    flinstones ...  rubble.txt

# chgrp flinfolks rubble.txt ↩

# ls -l rubble.txt ↩
-rw-rw-r--    1 barney   flinfolks  ...  rubble.txt
```

18.6 Practical Exercise

18.6.1 File permissions and the root user

1. Log into the system as root and make sure you are in root's home directory:

   ```
   # cd /root ←
   ```

2. Create a new file called test.txt using touch:

   ```
   # touch test.txt ←
   ```

3. Remove all permissions of test.txt using chmod:

   ```
   # chmod 0000 test.txt ←
   ```

4. Now write something to test.txt:

   ```
   # cat > test.txt ←
   This is root writing to a file without any permissions!
   Can you read this?
   <ctrl-d>
   #
   ```

5. Now try to read the file:

   ```
   # cat test.txt ←
   ```

6. Have a look at the owner, group and permissions of test.txt using ls -l:

   ```
   # ls -l test.txt ←
   ```

18.6.2 File permissions and a normal user

1. Log out from root and log back in as a normal user.
2. Try repeating the exercise above as a normal user.
3. Change the permissions of test.txt to write only:

   ```
   $ chmod 0200 test.txt  (or chmod u=w test.txt) ←
   ```

4. Now try writing something to the file:

   ```
   $ cat > test.txt ←
   This is a user writing to a file with only write permissions!
   Can you read this?
   <ctrl-d>
   $
   ```

5. What do you see when you try to read the file ?

   ```
   $ cat test.txt ←
   ```

6. Have a look at what permissions are set for the file:

   ```
   $ ls -l test.txt ←
   ```

7. Now add read permissions to the file:

   ```
   $ chmod u+r test.txt ←
   ```

8. Look again at what permissions are set for the file:

   ```
   $ ls -l test.txt ←
   ```

9. Can you read the file now?

18.6.3 Umask exercises

1. Log in as a normal user.
2. Have a look to see what your umask is set to:

   ```
   $ umask ←↩
   ```

   Umask = ______________________________

3. Touch a file and have a look at the resulting permissions:

   ```
   $ touch test.txt ←↩
   $ ls -l test.txt ←↩
   $  rm test.txt ←↩
   ```

   Record the permissions: ______________________________

4. Now set your umask to 0000 and try the same again:

   ```
   $ umask 0000 ←↩
   $ touch test.txt ←↩
   $ ls -l test.txt ←↩
   $ rm test.txt ←↩
   ```

   Record the permissions: ______________________________

5. Now set your umask to 0777 and try the same again:

   ```
   $ umask 0777 ←↩
   $ touch test.txt ←↩
   $ ls -l test.txt ←↩
   $ rm test.txt ←↩
   ```

   Record the permissions: ______________________________

6. What do you notice about umask and the execute permission bit?

Chapter 19

Inodes and links

note 1: Every object in the filesystem has an owner ID and a group ID which are by default the ID's of the process which created it.

19.1 Make some files and directories

These directories and files are just examples to experiment with. Follow the steps.

- change to you home directory – `$ cd`
- check where you are – `$ pwd`
- make a new directory – `$ mkdir test.dir`
- change to the new directory – `$ cd test.dir`
- check where you are – `$ pwd`
- check what's there – `$ ls`
- make an empty file – `$ touch a.file`
- make a hard link to the a.file – `$ ln a.file b.file`
- make a soft link to the a.file – `$ ln -s a.file c.file`
- check what's there (the a includes hidden files) – `$ ls -al`
- make a subdirectory – `$ mkdir sub.dir`
- make a soft link to the subdirectory – `$ ln -s sub.dir ln.dir`
- take a look (the i shows the inode numbers) – `$ ls -li`
  Your list should look something like this:

```
[geoffrey@freckle text.dir]$ ls -li %$
total 1
 454378 -rw-rw-r--   2 geoffrey geoffrey     0 Sep  1 14:35 a.file
 454378 -rw-rw-r--   2 geoffrey geoffrey     0 Sep  1 14:35 b.file
 454379 lrwxrwxrwx   1 geoffrey geoffrey     6 Sep  1 14:37 c.file -> a.file
 454380 -rw-rw-r--   1 geoffrey geoffrey     0 Sep  1 14:44 d.file
 454381 lrwxrwxrwx   1 geoffrey geoffrey     7 Sep  1 15:04 ln.dir -> sub.dir
 456433 drwxrwxr-x   2 geoffrey geoffrey  1024 Sep  1 14:35 sub.dir
```

19.2 File permissions

List the details of a particular file with `$ ls -il a.file`. Reading across the line a.file we see:

```
454378 -rw-rw-r--   2 geoffrey geoffrey        0 Sep  1 14:35 a.file
```

1. `454378` which is the inode number
2. `-rw-rw-r--` which are the file permissions
   - the first dash means a.file is a regular file (d for directory and l for link etc.)
   - the following three letters are the owners permissions `rw-` means readable, writable but not executable.
   - the next letters `rw-` indicate the permissions for the file's group
   - the last three letters `r--` show that other users can only read the file
3. `2` is a count of the hard links to the file
4. `geoffrey` is the owner of the file
5. `geoffrey` is the group of the file
6. `0` is a count of the bytes in the file
7. `Sep  1 14:35` is the date and time of last modification of the file
8. `a.file` the name of the file

19.3 Hard and soft links

Note that `a.file` and `b.file` have the same inode number, this indicates that the two names represent the same file. But `c.file` is an alias to that file with two names

```
454378 -rw-rw-r--   2 geoffrey geoffrey        0 Sep  1 14:35 a.file
454378 -rw-rw-r--   2 geoffrey geoffrey        0 Sep  1 14:35 b.file
454379 lrwxrwxrwx   1 geoffrey geoffrey        6 Sep  1 14:37 c.file -> a.file
```

The following should show you that they they are the same file.

- add some text into `a.file` `$ echo "this is going into the in a.file" >> a.file`
- look in a.file `$ cat a.file` ... the text went in?
- look at the `c.file` – `$ cat c.file` – it's a soft link to `a.file`
- nuke the `a.file` – `$ rm a.file`
- take another look at the `c.file` – `$ cat c.file` nothing to link to now
- but what about the `b.file` – `$ cat b.file`

19.4 Groups – `/etc/group`

Share files with a group

- first edit `/etc/group`: `root` will have to do this for you

  ```
  [geoffrey@freckle geoffrey]$ su -
  Password:
  [root@freckle /root]# emacs /etc/group
  ```

- add some users to the `student` group

```
    geoffrey:x:500:
    joe:x:501:
    jbloggs:x:502:
    jblogg:x:503:
    student::504:geoffrey,joe,jbloggs,jblogg
```

- check that you are in the group:

```
$ id
uid=500(geoffrey) gid=500(geoffrey) groups=500(geoffrey)
```

- logout and log back in to register the change in the `/etc/group` file and check your id again:

```
$ id
uid=500(geoffrey) gid=500(geoffrey) groups=500(geoffrey),504(student)
```

- change the group id of the file `a.file` to `student`

```
$ chgrp student a.file
$ ls -l
total 3
-rw-rw-r--   2 geoffrey student        18 Sep  1 17:14 a.file
```

- see if jbloggs can use the file

```
$ su jbloggs
Password:
[jbloggs$ echo "jbloggs waz here" >>a.file
[jbloggs$ exit

$ cat a.file
this is in a.file
jbloggs waz here
```

Chapter 20

mounting file systems

Document Description: Exercise in using file various file system utilities.

References Read the man pages for `mount`, `umount`, `df`, `du`, `fstab`, `mtab`, `tree`.

Instructions: Read through these notes and do the practical exercises in each section.

20.1 The Linux file system and removable media

Unlike MSDOS based operating systems that use named volumes with separate file systems (C: drive, A: drive, etc.) Linux and other unicies have a unified file system with volumes "grafted" in to a single tree at various mount points. The mount points are arbitrary. Typical mount points for removable media are as follows:

- Floppy disk: device /dev/fd0 mounted at `/mnt/floppy`
- ZIP drive: mounted at `/ZIP`
- CDROM disk: device `/dev/hdc` mounted at `/mnt/cdrom`
- CD Writer: device `/dev/sd2` mounted at `/burner`
- Network drive: mounted at `/mnt/nfs/database`
- NT Server network drive: `/mnt/samba-vol`

Exercise: Look at the file system tree on your system:

```
$ tree / | less ↩
```

20.1.1 Mounting and unmounting volumes

1. To access a volume via the Linux filesystem the volume must first be mounted. This example is for a floppy disk.

   ```
   # mount -t msdos /dev/fd0 /mnt/floppy ↩
   ```

   Where:

   - `mount` is the command
   - `-t msdos` is the filesystem *type*
   - `/dev/fs0` is the device node for the filesystem
   - `/mnt/floppy` is the mount point for the filesystem

2. Any files existing at the mount point will be hidden when a volume is mounted at the point.
3. File system types include:

   ext2 Linux standard file system

   ext3 New journaling file system

   riserfs Journaling file system

   iso9660 Standard file system on CDROMs

   msdos Microsoft FAT16 file system

   vfat Microsoft FAT32 file system

   ntfs Microsoft NT file system

   hfs Apple file system

4. Before removing the media the volume should be unmounted:

   ```
   # umount /mnt/floppy ↩
   ```

5. Note that by default the superuser only has mounting rights. Users may be given some mounting rights in the configuration file (see next).

20.1.2 The filesystem table: /etc/fstab

The file /etc/fstab is a table of static mount information. The mount command references this table. Edit this file to reflect your system.

```
$ cat /etc/fstab ↩
# <file system><mount pt><type>  <options>                  <dump> <pass>

/dev/sda1     /           ext2    defaults,errors=remount-ro      0 1
/dev/sda2     /tmp        ext2    rw                              0 2
/dev/sda3     /var        ext2    rw                              0 2
/dev/sda4     none        swap    sw                              0 0

/dev/sr5      /burner     iso9660 defaults,ro,user,noauto         0 0
/dev/hdb      /dvd        iso9660 defaults,ro,user,noauto         0 0
/dev/hdd      /mnt/cdrom  iso9660 defaults,ro,user,noauto         0 0

/dev/fd0      /mnt/floppy auto    defaults,user,noauto            0 0
```

Using this table the mount commands may be abbreviated. Note also that users have access to mounting the floppy and cdroms.

- Mount a floppy

  ```
  $ mount /mnt/floppy ↩
  ```

- Mount the DVD:

  ```
  $ mount /dvd ↩
  ```

20.1.3 /etc/mtab

/etc/mtab is a dynamic table of currently mounted file systems. Do *not* edit this file. Ever.

```
$ cat /etc/mtab ↩

/dev/hda6 / ext2 rw 0 0
none /proc proc rw 0 0
none /dev/pts devpts rw,gid=5,mode=620 0 0
/dev/hda1 /mnt/disk vfat rw 0 0
```

The mount command with no arguments also will report the currently mounted volumes.

```
$ mount ↩
/dev/hda6 on / type ext2 (rw)
none on /proc type proc (rw)
none on /dev/pts type devpts (rw,gid=5,mode=620)
/dev/hda1 on /mnt/disk type vfat (rw)
```

20.1.4 other file system commands

- Disk usage

  ```
  $ du ↩
  24      ./vmware/win2000
  961136  ./vmware/nt4
  961164  ./vmware
  20      ./.netscape/cache
  4       ./.netscape/archive
  968164  .
  ```

- Disk filesystem disk space usage

  ```
  $ df ↩
  Filesystem 1k-blocks      Used Available Use% Mounted on
  /dev/hda6    3755460   3041736    522956  86% /
  /dev/hda1    2044240    614664   1429576  31% /mnt/disk
  ```

20.2 Lab Exercises

20.2.1 Examine the file system

- Check the disk usage on your system. How much of the available disk space has been filled?

  ```
  $ df ↩
  ```

- Look at the dynamic mount table:

  ```
  $ cat /etc/mtab ↩
  ```

- Examine the static mount table:

  ```
  $ cat /etc/fstab ↩
  ```

- Check the disk usage for your home directory (the `-h` option gives a human readable format of the output:

  ```
  $ du -h ˜  ↩
  ```

20.2.2 Floppy disk exercise

Note: If you have an automount demon running it may mount your floppy disk automatically (check with mount. In this case first unmount the floppy disk.

- List the directory `/mnt/floppy`: `$ ls -la /mnt/floppy ↩`
- place an MSDOS formatted floppy in the fd0 device (a.k.a. a: drive);
- mount the floppy so that it can be accessed from the mount point `/mnt/floppy` (note that to use the mount command in it's full form you may need to be root):

  ```
  # mount -t msdos /dev/fd0 /mnt/floppy ↩
  ```

  or using `/etc/fstab` (from a user login)

  ```
  $ mount /mnt/floppy ↩
  ```

- use the df and mount command to see what has happened;
- change the current working directory to the floppy:

  ```
  $ cd /mnt/floppy ↩
  ```

- make a subdirectory on it called `/mnt/floppy/asubdir` and use cat or vi to create a test file on the floppy
- umount the floppy before removing it. Note that the floppy must not be in use when unmounting it.

  ```
  $ cd ↩
  $ umount /mnt/floppy ↩
  ```

- physically remove the floppy.

20.2.3 CDROM exercise

Note: If you have an automount demon running it may mount your CDROM automatically. In this case first unmount the CDROM.

- List the directory `/mnt/cdrom`: `$ ls -la /mnt/cdrom ↩`
- place an iso9660 formatted CDROM in the cdrom device (a.k.a. d:, whatever drive);
- mount the CDROM so that it can be accessed from the mount point `/mnt/cdrom`:

  ```
  # mount -t iso9660 /dev/cdrom /mnt/cdrom ↩
  ```

or as an ordinary user

```
$ mount /mnt/cdrom ↩
```

- use the df and mount command to see what has happened;
- change the current working directory to the CDROM:

```
$ cd /mnt/cdrom ↩
```

- explore the CDROM
- umount and eject the CDROM before removing it:

```
$ cd ↩
$ eject ↩
```

- physically remove the CDROM.

20.2.4 Hard disk partition exercise

- Make a directory on which to mount the windows partition (`/dev/hda1`):

```
#  mkdir /mnt/windows ↩
```

- mount the widows partition (read only) so that it can be accessed from the mount point `/mnt/windows`:

```
$ mount -o ro -t vfat /dev/hda1 /mnt/windows ↩
```

- use the df and mount command to see what has happened;
- change the current working directory to the windows directory:

```
$ cd /mnt/windows ↩
```

- explore the widows file system

Part XI

Programming

Chapter 21

"Hello, world!" with gcc and vi

This Document: Writing, compiling and running a c program using `vi`.

21.1 Make a directory for your c programmes

- Make sure you are logged in as an ordinary user - not as root;
- Change to your home directory:

  ```
  $ cd ↩
  ```

- Make a directory for your programmes:

  ```
  $ mkdir vi.practice ↩
  ```

- Go there:

  ```
  $ cd vi.practice ↩
  ```

21.2 Write a '`Hello, world!` source file

- Write a hello world source file using vi:

  ```
  $ vi hi.c ↩
  ```

  ```
  #include <stdio.h>
  main()
  {
    printf("Hello, world!\n");
    return 0;
  }
  ```

- Save it and exit with `<esc> ZZ`
- Check that you have no typos:

  ```
  $ cat  hi.c ↩
  ```

21.3 Compile to the a.out binary executable with the GNU compiler

- Compile your source (create an executable called `a.out`):

  ```
  $ gcc hi.c ↩
  ```

- check that it happened:

  ```
  $ ls ↩
  a.out    hi.c
  ```

21.4 Running your executable

- Try running the program:

  ```
  $ a.out ↩
  bash: a.out: command not found
  ```

- Check the permissions to see that the 'x' bit is set:

  ```
  $ ls -l ↩
  -rwxrwxr-x  1 yourname   yourname   4150 Mar  9 17:27 a.out
  ```

  Why doesn't it run? By default the current directory '.' is not in the path. Check it `$ echo $PATH`; you will see no '.'.

- To run it you have to give the path to the file:

  ```
  $ ./a.out ↩
  Hello, world!
  ```

21.5 Write a C language program using vi

Copy your program with a new name (`$ cp hi.c hi3.c` ↩) edit it with `vi` so that it is exactly like this:

```
#include <stdio.h>
int main()
{
  char name[100];

  printf("What is your name? =>> ");
  scanf("%s", name);

  printf("Hello, %s\n", name);

  return 0;
}
```

Chapter 22

”Hello would!” with gcc

This Document: writing, compiling and running a c program

22.1 Make a directory for your c programmes

- Make sure you are logged in as an ordinary user - not as root;
- Change to your home directory:

  ```
  $ cd ↩
  ```

- Make a directory for your programmes:

  ```
  $ mkdir c.progs ↩
  ```

- go there:

  ```
  $ cd c.progs ↩
  ```

22.2 Write a ‘`Hello, world!` source file

- Write a hello world source file using cat and ˆD

  ```
  $ cat > hi.c ↩
  ```

  ```
  #include <stdio.h>
  main()
  {
    printf("\n\n\tHello, Linux world!\n\n\n");
    return 0;
  x1}
  ˆD
  ```

- Check that you have no typos:

  ```
  $ cat  hi.c ↩
  ```

- Should you have errors use an editor to fix them.

22.3 Compile to the a.out binary executable with the GNU compiler

- Compile your source (create an executable called `a.out`):

```
$ gcc hi.c ↩
```

- heck that it happened:

```
$ ls ↩
a.out    hi.c
```

22.4 Running your executable

- Try running the program:

```
$ a.out ↩
bash: a.out: command not found
```

- Check the permissions to see that the 'x' bit is set:

```
$ ls -l ↩
-rwxrwxr-x  1 yourname   yourname   4150 Mar  9 17:27 a.out
```

  Why doesn't it run? By default the current directory '.' is not in the path. Check it `$ echo $PATH`; you will see no '.'.

- To run it you have to give the path to the file:

```
$ ./a.out ↩
Hello, world!
```

22.5 Programming from emacs

- Open your file in emacs:

```
$ emacs hi.c ↩
```

- Edit the `printf()` to print "Hello, world from emacs"
- Save the emacs buffer with a new name: (Control-X Control-W then type in the mini-buffer at the bottom of the emacs frame)

```
C-x C-w
hi.emacs.c ↩
```

- Save your emacs edits at any time with C-X C-S.
- Compile your source from inside emacs: (M-! means either Alt-! or [ESC] then !, the shell command is then typed into the mini-buffer window)

```
M-!
Shell command: gcc -o hello hi.emacs.c ↩
```

- Run the program with output to a new buffer:

```
M-!
Shell command: ./a.out
```

- Close the other emacs window:

```
C-x 1
```

- Save your file and exit emacs:

```
C-x C-C
```

22.6 Write out what these 15 commands do

C-X C-s

C-X C-w

C-X C-f

C-X C-c

C-L

C-G

C-K

C-A

C-E

ESC !

C-SPACE

C-X 0

C-X 1

C-X 2

C-X 3

22.7 Write a C language program using emacs

Write, debug and run an ohms law program from command line emacs; user to enter current and resistance program calculates and prints voltage.

Part XII

Networking

Chapter 23

LAN Setup in Room C222 (RH73)

23.1 Overview

There are several layers that have to be built in order to setup a Local Area Network.

1. Load the kernel module (driver) for the NIC (ethernet card) card that is installed on the system.

   On systems using Plug and Play hardware the modudules will be automatically loaded. However, in the case of legacy hardware such as the NE2000 NIC this has to be done by hand.

2. Configure IP address settings.

   - This may be done automatically from a DHCP server,
   - Or configured manually in the case of static IP addresses.

3. Setup and run network services such as:

   - sendmail for email transfer
   - NFS file sharing
   - network printing
   - brousing internet or intranet webservers
   - downloading files using ftp
   - loging into remote hosts using ssh
   - messaging services such as irc etc.

23.2 Nobrainer Network Setup for RedHat

1. Run the ncurses program called `netconfig` from a root prompt:

   ```
   # netconfig ↩
   ```

2. Tap the spacebar to use DHCP, then `<tab>` to `OKAY`

   ```
   [x] Use dynamic IP configuration (BOOTP/DHCP)
   ```

3. Test the network:

   ```
   $ ping 192.168.222.254 ↩
   ```

23.3 Loading a NIC Driver Module

On modern systems this will have been detected and configured automatically during installation and the driver module will have been loaded at boot time.

Exercise:

1. Check that the module used for your Network Interface Card is loaded:

   ```
   # lsmod ↩
   ```

   Unload the module:

   ```
   # rmmod <module_name>↩
   ```

2. Load the module:

   ```
   # insmod <module_name>↩
   ```
   or
   ```
   # modprobe <module_name>↩
   ```

   Note: these two commands both load the module. `modprobe` also loads other modules that are required by the particular module if there any.

23.4 Command Line Tools for Managing Kernel Modules

- List the modules currently loaded:

  ```
  # lsmod ↩
  Module               Size  Used by
  lockd               31176   1  (autoclean)
  sunrpc              52964   1  (autoclean) [lockd]
  ne2k-pci             4652   1  (autoclean)
  8390                 6072   0  (autoclean) [ne2k-pci]
  ```

- Install a loadable kernel module: (if there was a NE2000 ISA NIC)

  ```
  # insmod ne io=0x300 irq=5 ↩
  ```

- Record details of modules so that they may be loaded easily on demanmd

```
# vi /etc/modules.conf ↩
alias parport_lowlevel parport_pc
alias sound-slot-0 maestro3
alias eth0 ne
options ne io=0x300 irq=5
~
~
```

Now when `eth0` is used the `ne` module will load automatically using the parameters shown.

- Unload a kernel module:

  ```
  # rmmod ne ↩
  ```

- depmod handles dependency descriptions for loadable kernel modules. Creates a module dependency list. `$ man depmod ↩` for details.
  Divert the output to STDOUT for viewing:

  ```
  # depmod -n | less ↩
  ```

- The command modprobe (with depmod) provides high level handling of loadable modules. In particular, note that modprobe will not only load a given module but also load all the modules it depends on. Check the man pages.

  ```
  # modprobe ftape ↩
  ```

Exercise **Practice using the commands for modules.**

- List the modules currently loaded: `$ lsmod ↩`;
- Attempt to load the module zftape, and note that there are unmet dependencies. To load the zftape module successfully you would have to load the modules it depends on first.
  Try it: `# insmod zftape ↩`
- Now load the module with modprobe:
  `# modprobe zftape ↩`
- List the modules now loaded;
- unload the driver with `rmmod zftape`;
- again list the modules now loaded;

23.5 Set Network IP Parameters for Static IP Addresses

23.5.1 Configure the `/etc/hosts` file

Exercise

Using your favourite editor edit the file /etc/hosts as shown below.

```
127.0.0.1             localhost
192.168.222.254       foozle.c222     foozle
192.168.222.253       sparkie.c222    sparkie

192.168.222.1         box1.c222       box1
192.168.222.2         box2.c222       box2
192.168.222.3         box3.c222       box3
...
...
192.168.222.22        box22.c222      box22
192.168.222.23        box23.c222      box23
192.168.222.24        box24.c222      box24
```

23.5.2 Network Configuration tools

There are many command line and GUI tools for doing setting static IP addresses.

netconfig RedHat special, old but still supplied.

linuxconf General tool, now generally out of favor.

netcfg Easy to use, reliable, no longer supplied RedHat tool.

neat New RedHat GUI tool. Buggy in RH72 better in RH73.

ifconfig Command line tool always available on all distrobutions.

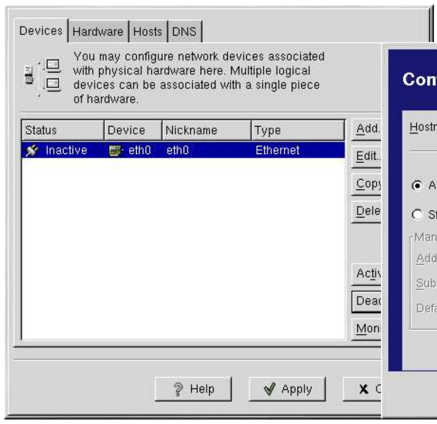
23.5.3 Setting IP addresses Using netconfig

Fill in the numbers and `<tab>` between fields, enter for OKAY.

```
netconfig 0.8.11  (C) 1999 Red Hat, Inc.
          ----------------------- Configure TCP/IP ----------------------
          |                                                             |
          | Please enter the IP configuration for this machine. Each   |
          | item should be entered as an IP address in dotted-decimal  |
          | notation (for example, 1.2.3.4).                            |
          |                                                             |
          |       [ ] Use dynamic IP configuration (BOOTP/DHCP)         |
          |                                                             |
          |           IP address:           192.168.222.1               |
          |           Netmask:              255.255.255.0               |
          |           Default gateway (IP): 192.168.222.254             |
          |           Primary nameserver:   192.168.222.254             |
          |                                                             |
          |            ------                       --------            |
          |            | OK |                       | Back |            |
          |            ------                       --------            |
          |                                                             |
          ---------------------------------------------------------------
 <Tab>/<Alt-Tab> between elements   |   <Space> selects  |   <F12> next scree
```

23.5.4 Setting IP addresses Using the **`neat`** click-o-rama

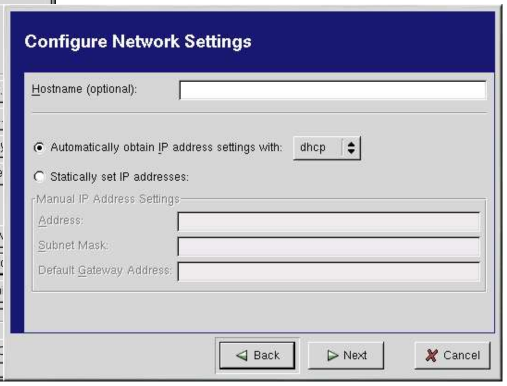
A new Ethernet Device may configured by clicking the add button and following the wizzard. Help may be found by clicking the help button.



You may configure network devices associated with physical hardware here. Multiple logical devices can be associated with a single piece of hardware.

Status	Device	Nickname	Type
Inactive	eth0	eth0	Ethernet

Figure 23.1: Ethernet Device



Hostname (optional):

(●) Automatically obtain IP address settings with: dhcp

( ) Statically set IP addresses:

Manual IP Address Settings

Address:

Subnet Mask:

Default Gateway Address:

Figure 23.2: Ethernet Settings

Exercise

Setup a static IP address for your system. Use `netconfig` then `neat`. Explore `neat` thoroughly.

Chapter 24

Basic Network Commandline Tools

Chapter 25

Chapter 26

Cabling

26.1 Crimping RJ45 connectors onto Cat 5 cable

Reference: Network HOWTO

26.1.1

If you hold the RJ45 connector facing you (as if you were going to plug it into your mouth) with the lock tab on the top, then the pins are numbered 1 to 8 from left to right.

```
                         _____________________
brown           8-------|                     |
white / brown   7-------|                     |
orange          6-------|     _______________ |
white / blue    5-------|     |               |     TAB on top
blue            4-------|     |______________ |
white / orange  3-------|                     |
green           2-------|                     |
white /green    1-------|_____________________|
```

- Do not untwist the pairs any more that necessary
- Do not cut or strip the insulation off the wires
- Trim the ends level after you have them in the right order
- Push the wires to the end of the RJ45
- Make sure the sheath will be caught under the restraint
- Test after crimping

26.1.2 Crosover Cables

Chapter 27

networking text tools

Document Description: Exercise in using a local area network

Instructions:

- read through these notes and the man pages
- do the exercises

27.1 configuration

27.1.1 configuration files

- /etc/HOSTNAME the host name for the localhost
- /etc/resolv.conf stipulates how host names are resolved
- /etc/hosts lookup table matching host names to IP addresses
- /etc/services table of network services and port numbers
- /etc/inetd.conf controls which services are available; xinetd is an alternative system
- /etc/xinetd.conf and the files in /etc/xintd.d/ control which services are available on RH7.0 and up
- /etc/host.conf remote host lookup order.

27.1.2 /etc/resolv.conf

resolver uses the configuration file /etc/resolv.conf provide access to the Internet Domain Name System.

```
$ cat /etc/resolv.conf ↩
search fernbank
nameserver 61.8.0.2
nameserver 61.8.0.5
```

27.1.3 /etc/host.conf

This configuration file controls the host lookup order. In this example the resolver will search all of the `/etc/hosts` file first then use bind (DNS lookup).

```
$ cat /etc/host.conf ↩
order hosts,bind
multi on
```

27.1.4 /etc/hosts

In a small network host names may be resolved into IP addresses form the file /etc/hosts. e.g.

```
$ cat /etc/hosts ↩
127.0.0.1        localhost.localdomain    localhost
192.168.222.1    foozle.zork              foozle
192.168.222.101  box1.zork                box1
192.168.222.102  box2.zork                box2
192.168.222.103  box3.zork                box3
...
192.168.222.116  box16.zork               box16
```

27.1.5 /etc/HOSTNAME

The hostname is read at boot time by the boot scripts from the file /etc/HOSTNAME (RedHat) or /etc/hostname (Debian) on some systems. This file may be edited to permanently change the hostname. The hostname may be displayed:

```
# hostname ↩
foozle.zork
```

or changed temporarily with this command:

```
# hostname blahblah ↩
```

27.1.6 installing a NIC

Normally a plug and play Ethernet Adapter will be automatically detected and set up during installation. A legacy ISA adapter may have to be compiled into the kernel or loaded as a module.

- Check which modules are loaded (here a netgear PCI and a NE2000 NIC are installed)

  ```
  $ lsmod  ↩
  tulip                  30264   1
  ne2k-pci               34757   1
  ```

- A legacy NE2000 ISA card would be loaded thus: (ymmv)

  ```
  # modprobe ne io=0x300 irq=5 ↩
  ```

- To load the module at each system restart the system init scripts will require editing. On RH7.0 the GUI tool kernel.cfg makes this task trivial.

27.1.7 restarting inetd or xinetd

Note: Recent systems such as RH7.0 and above use xinetd to replace inetd.

Services may stopped or started by editing the files in the directory /etc/xinetd/.

After making network configuration changes you may need to restart the network daemon. Three methods of doing this follow:

1. Restart xinetd with the script:

   ```
   # /etc/rc.d/init.d/xinetd restart  ↩

   tarball
   ```

2. Restart xinetd by name with the command:

   ```
   # killall -HUP xinetd ↩
   ```

3. Find the process id number for xinetd and restart the process by PID.

   ```
   # ps afx |grep xinetd ↩
     422 ?        S      0:00 xinetd
   24757 ttyp9    S      0:00  |_ grep xinetd
   # kill -HUP 422
   #
   ```

27.2 GUI configuration tools

There are a range of various GUI tools available on most distributions.

- netcfg network configuration
- linuxconf general RedHat configurations tool
- netconf network configuration
- neat network configuration
- netconfig text (ncurses) network configuration tool
- kernelcfg edits which kernel modules will be loaded
- tksysv edits which services will run at which runlevel

27.3 checking and testing

27.3.1 ifconfig

Some details about the network interfaces may be found with the ifconf command. Note that in this example there are three interfaces:

eth0: the ethernet card for the local area network

lo: loopback address

ppp0: the ppp modem internet connection for this machine

```
# ifconfig ←↩
eth0      Link encap:Ethernet  HWaddr 00:40:05:46:7F:32
          inet addr:192.168.42.1  Bcast:192.168.42.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:24076 errors:0 dropped:0 overruns:0 frame:0
          TX packets:93529 errors:0 dropped:0 overruns:0 carrier:0
          collisions:38 txqueuelen:100
          Interrupt:9 Base address:0x9500
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:3924  Metric:1
          RX packets:29552 errors:0 dropped:0 overruns:0 frame:0
          TX packets:29552 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
ppp0      Link encap:Point-to-Point Protocol
          inet addr:61.8.18.98  P-t-P:203.9.190.192  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:5151 errors:12 dropped:0 overruns:0 frame:12
          TX packets:4591 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:10
```

A NIC can be started with an ifconfig command such as the following:

```
$ ifconfig eth0 192.168.222.15 up ←↩
```

27.3.2 the IP routing table

The route command shows and manipulates the routing table for your system. View the table with the command:

```
$ /sbin/route ←↩
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
localnet        *               255.255.255.0   U     0      0        0 eth0
localnet        *               255.255.255.0   U     0      0        0 eth1
127.0.0.0       *               255.0.0.0       U     0      0        0 lo
default         sherbie.fernban 0.0.0.0         UG    0      0        0 eth0
```

27.3.3 view TCP traffic

TCP traffic on your system may be viewed with the command: (ˆC to exit)

```
# tcpdump ←↩
tcpdump: listening on all devices
01:20:56.210331 eth0 > lollipop.fernbank.ssh >
mintie.fernbank.1870: P 3624801409:3624801517
(108) ack 2624831152 win 32120 <nop,nop,timestamp
61941018 276985734> (DF) [tos 0x10]
....
....
298 packets received by filter
```

27.3.4 ping

ping is used to check if a particular IP address is accessible and to check the timing and reliability of the connections:

- check the connection to the ethernet card

```
$ ping bim ↩
PING bim.fernbank (192.168.42.1): 56 data bytes
64 bytes from 192.168.42.1: icmp_seq=0 ttl=255 time=16.8 ms
64 bytes from 192.168.42.1: icmp_seq=1 ttl=255 time=0.2 ms
```

- connect to a host on the local network

```
$ ping jaffa ↩
PING jaffa.fernbank (192.168.42.3): 56 data bytes
64 bytes from 192.168.42.3: icmp_seq=0 ttl=128 time=2.3 ms
64 bytes from 192.168.42.3: icmp_seq=1 ttl=128 time=0.8 ms
```

- ping a DNS server on the Internet

```
$ ping 61.8.0.2 ↩
PING 61.8.0.2 (61.8.0.2): 56 data bytes
64 bytes from 61.8.0.2: icmp_seq=0 ttl=254 time=122.9 ms
```

27.4 practical exercise

27.4.1 command line tools

Note: After changing network configuration details you may have to restart xinetd or inetd (see previous instructions).

1. Check that the `/etc/hosts` file has entries for the loopback address and the hosts on the network.

   ```
   [fred@box3 fred]$ less /etc/hosts
   ```

   Edit the file if it is incomplete or inaccurate.

   ```
   [fred@box3 fred]$ su -
   Password:
   [root@box3 root]# vi /etc/hosts
   ```

2. Have a look at your host name:

   ```
   [fred@box3 fred]$ echo $HOSTNAME
   box3
   [fred@box3 fred]$ hostname
   box3
   ```

3. Check the binding to your Ethernet card with the command:

   ```
   [fred@box3 fred]$ ifconfig
   ```

   If the IP address is wrong it can be changed with:

   ```
   [root@box3 root]$ ifconfig 192.168.222.xxx eth0
   ```

4. Have a look at the routing table:

   ```
   # route ↩
   ```

   Add a default route if it does not exist:

   ```
   # route add default gw <gateway hostname or IP> ↩
   ```

5. Ping a few hosts near you. What is the packet turn around time? Were any packets lost?

27.4.2 GUI tools

Have a look at the GUI tools on your system. But don't make any changes that may damage the system. Close them with Cancel and Quit without saving changes.

27.5 remote login: telnet

- See notes from previous lesson.
- read the man pages for telnet

27.6 file transfer: FTP

- See notes from previous lesson.
- read the man pages for FTP

27.7 using email

- See notes from previous lesson.
- read the pine help screens

27.8 exercise

After this exercise you should be able to:

- telnet a remote host
- copy files from a floppy to your home directory on the remote host
- make a tarball of a group of files
- use FTP to transfer files between hosts
- compile a c program
- send an email to a user on the local network
- add an attachment to an email
- telnet into the remote host

```
[fred@box1 fred]$ telnet elephant
...
login: fred
Password:
[fred@elephant fred]$
```

- copy all the files on a floppy to your home directory on the remote host

```
[fred@elephant fred]$ mkdir temp
[fred@elephant fred]$ cp -a /mnt/floppy/* temp
```

- tar and compress the files now in temp into a tarball called files.cvfz

```
[fred@elephant fred]$ tar cvfz files.tgz temp
[fred@elephant fred]$ ls  -l files.tgz
-rw-rw-r--   1 fred   fred   219863 Mar 28 10:48 files.tgz
```

- transfer the tarball to your localhost

```
[fred@box1 fred]$ mkdir work
[fred@box1 fred]$ cd work
[fred@box1 work]$ ftp elephant
...
ftp> ls files.tgz
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
-rw-rw-r--   1 geoffrey geoffrey   219863 Mar 28 10:48 files.tgz
226 Transfer complete.
ftp> binary
200 Type set to I.
ftp> get files.tgz
local: files.tgz remote: files.tgz
200 PORT command successful.
150 Opening BINARY mode data connection for files.tgz (219863 bytes).
226 Transfer complete.
219863 bytes received in 0.289 secs (7.4e+02 Kbytes/sec)
ftp>
```

- untar the files

```
[fred@box1 fred]$ tar xvfz files.tgz
```

- find and compile the file ohmslaw.c

```
[fred@box1 fred]$ cd work/temp
[fred@box1 temp]$ ls
begining  lost+found  ncurses  ohmeslaw  ohmslaw.c  test  unleashed
[fred@box1 temp]$ gcc -Wall -o ohmeslaw ohmslaw.c -lcurses
```

- run the program and capture the output

```
[fred@box1 temp] $./ohmslaw >catch.out
> r
```

- check your output

```
[fred@box1 temp] $ cat catch.out
```

- send an email to geoffrey@elephant and attatch the file catch.out

Chapter 28

Login to a Remote Host using `telnet`

Document Description: exercise in using telnet on a local area network

Instructions:

- read through these notes and `man telnet`
- telnet into some hosts where you have user accounts

28.1 warning

Hosts connected to an untrusted network may be vulnerable to cracking and various exploits should they leave the telnet service open. Using a secure shell such as ssh is safer in a hostile environment.

28.2 logging into a remote host

- You must have a network connection to the remote host.
- You will *not* be able to log onto a remote host as root.
- You must have an account with a current passwd on the remote host.

28.3 exercise

- logon to the remote host—it should go something like this (If foozle is not in your `/etc/hosts` file then either add it or use the ip address):

  ```
  [foo@box]$ telnet foozle
  Trying 192.168.222.254...
  Connected to foozle.c222.
  Escape character is '^]'.
  Welcome to foozle.c222
  ```

```
Linux Mandrake release 7.0 (Air)
Kernel 2.2.14-15mdk on an i686
login: foo
Password:
Last login: Tue Aug 15 04:17:27 from bim
[foo@foozle foo]$
```

- check who is currently logged on to foozle

```
[foo@foozle foo]$ finger
```

- get some details about someone on foozle

```
[foo@foozle foo]$ finger fred
Login:  fred                    Name: Fred Dagg
Directory: /home/fred           Shell: /bin/bash
On since Mon Aug  7 00:58 on tty1 4 days 7 hours idle
No mail.
No Plan.
```

- have a look at the home directories for other users

```
[foo@foozle foo]$ ls /home
```

- try to have a look and a play with their files

```
[foo@foozle foo]$ cd /home/foobar
bash: /home/foobar: Permission denied
```

- After enabling the remote host to display on your local host (`$ xhost +foozle` ↩ send some xeyes back to your localhost:

```
[foo@foozle foo]$ xeyes -display box:0.0
```

- create a script in your home directory on the remote host

```
[foo@foozle foo]$ cd
[foo@foozle foo]$ cat >hi.sh
> echo "hello, world at foozle.zork"
> ^D
[foo@foozle foo]$ sh hi.sh
hello, world at foozle.zork
```

- make your self a plan

```
[foo@foozle foo]$ cat > .plan
> I'm here doing this Linux thing...
> ^D
[foo@foozle foo]$ finger foo
```

- logout

```
[foo@foozle foo]$ ^D
Connection closed by foreign host.
[foo@bim foo]$
```

Chapter 29

Transfer Files Between Hosts Using `ftp`

29.1 file transfer protocol

The ftp command is generally used for transferring files between your local host and a remote host, although it is possible to transfer files between two remote hosts.

29.1.1 Notes

- It in not generally possible or advisable to use ftp while logged on as root;
- ftp transmits passwords and data in plain text, so for untrusted networks (internet) use anonymous ftp or scp.

29.1.2 connecting to a remote host with ftp

Establishing a connection from localhost called mybox to a remotehost called otherbox for a user called fred on a network called thisnet

```
[fred@mybox fred] $ ftp otherbox ↩
Connected to otherbox.thisnet
220  FTP server (Version wu-2.6.0(1)
   Tue Jan 4 19:41:20 GMT 2000) ready.
Name (mybox:fred): ↩
331 Password required for fred.
Password: ↩
230 User fred logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

29.1.3 Getting help with ftp commands

- A list of commands is available from the `ftp` prompt:

```
ftp> help ↩
Commands may be abbreviated.  Commands are:

!         debug        mdir        sendport        site
$         dir          mget        put             size
account   disconnect   mkdir       pwd             status
append    exit         mls         quit            struct
ascii     form         mode        quote           system
bell      get          modtime     recv            sunique
binary    glob         mput        reget           tenex
bye       hash         newer       rstatus         tick
case      help         nmap        rhelp           trace
cd        idle         nlist       rename          type
cdup      image        ntrans      reset           user
chmod     lcd          open        restart         umask
close     ls           prompt      rmdir           verbose
cr        macdef       passive     runique         ?
delete    mdelete      proxy       send
```

- The `ftp` man page gives details for the use of each command:

```
$ man ftp ↩
$ /bye ↩
bye    Terminate the FTP session with the remote server  and  exit
       ftp.   An  end  of file will also terminate the session and
       exit.
...
```

29.1.4 finishing an ftp session

```
ftp> bye ↩
221-You have transferred 0 bytes in 0 files.
221-Total traffic for this session was 249 bytes in 0 transfers.
221-Thank you for using the FTP service on otherbox.mynet.
221 Goodbye.
[fred@mybox fred] $
```

29.2 Practical exercise

In these exercises you will establish simultaneous telnet and ftp sessions between your local host (say mybox) and a remote host (here called otherbox).

- Fillin these boxes:
  - Actual name of local host: $\boxed{}$ = `mybox`
  - Actual name of remote host: $\boxed{}$ = `otherbox`
- Open three terminals in X *or* login to three virtual consoles.

first terminal: local login to mybox

second terminal: remote login to otherbox

third terminal: ftp session between mybox and otherbox

29.2.1 Exercise 1

Create two files, one on the local host and one on the remote host, then copy each of the files to the other computer.

- Create a file on the local host:

```
[fred@mybox fred] $ uname -a > ~/local.file ↩
```

- Telnet into the remote host (here called otherbox) and create a file in your home directory over there:

```
[fred@mybox fred] $ telnet otherbox ↩
...
login: fred
Password:
[fred@otherbox fred] $ uname -a > ~/remote.file ↩
```

- Establish an ftp session (see section 29.1.2):

```
[fred@mybox] $ ftp otherbox ↩
...
ftp>
```

- `put` the file `local.file` from the local host to the remote host.

```
ftp> put local.file ↩
local: local.file remote: local.file
200 PORT command successful.
150 Opening BINARY mode data connection for local.file.
69 bytes sent in 0.00 secs (1271.4 kB/s)
```

- Look at the "`.file`'s" on the remote host:

```
ftp> ls *.file ↩
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
-rw-r--r--    1 geoffrey geoffrey       69 May 14 02:09 local.file
-rw-r--r--    1 geoffrey geoffrey       80 May 14 01:59 remote.file
```

- `get` the file `remote.file` from the remote host to the local host:

```
ftp> get remote.file ↩
local: remote.file remote: remote.file
200 PORT command successful.
150 Opening BINARY mode data connection for remote.file (80 bytes).
226 Transfer complete.
80 bytes received in 0.00 secs (137.1 kB/s)
```

- End you telnet session:

```
[fred@otherbox fred] $ ^D ↩
[fred@mybox fred] $
```

- end your ftp session:

```
ftp> bye  ↩
[fred@mybox fred] $
```

29.2.2 Exercise 2

- Establish a telnet session with the remote host;
- copy all the files on the floppy disk mounted on the remote host to a directory in your home directory on the remote host;

```
[fred@otherbox] $ mkdir ~/temp ↩
[fred@otherbox] $ cp -a /mnt/floppy/* ~/temp
```

- tar and compress the files now in temp into a tarball called `files.tar.gz`

```
[fred@otherbox] $ tar zcvf files.tar.gz ~/temp ↩
[fred@otherbox] $ ls -l files.tar.gz ↩
-rw-rw-r--   1 fred   fred   219863 Mar 28 10:48 files.tar.gz
```

- establish an ftp session with the remote host and transfer the tarball to your localhost:

```
[fred@mybox fred] $ mkdir work ↩
[fred@mybox fred] $ cd work ↩
[fred@mybox work] $ ftp otherbox ↩
...
ftp> mget files.t* ↩
mget files.tar.gz? y↩
...
```

- untar the files:

```
[fred@mybox fred] $ tar zxvf files.tar.gz ↩
```

- find and compile the file example.c:

```
[fred@mybox fred] $ cd ~/temp ↩
[fred@mybox temp] $ gcc -Wall -o example example.c ↩
```

- run the program and capture the output:

```
[fred@mybox temp] $./example >catch.out ↩
```

- check your output:

```
[fred@mybox temp] $ cat catch.out ↩
```

Chapter 30

Using Secure Shell—ssh

Document Description: Exercise in using ssh and scp over a network.

References `man ssh` and `man scp`

Instructions:

- Read through these notes on ssh and scp;
- do the practical exercise in Section 30.4.

30.1 Install and setup

OpenSSH was probably installed during your system installation. If not then install the OpenSSH packages using the command line or request your system administrator to do so for you.

30.2 Using an ssh client

In order to login to a remote host:

- You must have an account on the remote host;
- you must have ssh client software installed on the local host;
- and an ssh server (sshd) must be running on the remote host.

30.2.1 Logging in to a remote host using ssh

If your username for your account on the remote host is the same as it is on the local host then you may login thus:

```
geoffrey@mintie:~$ ssh zipper.zip.com.au  ↩
geoffrey@zipper.zip.com.au's password:   ↩
```

If your account name is different then add your remote host username to the ssh login command:

```
geoffrey@mintie:~$ ssh geoff@zipper.zip.com.au  ↩
geoff@zipper.zip.com.au's password:   ↩
```

30.2.2 First time ssh login

The first login requires authenticity encription keys to be set up on the local host. Your first login session may look like this:

```
geoffrey@mintie:~$ ssh zipper.zip.com.au  ↩
The authenticity of host 'zipper.zip.com.au' can't be established.
DSA key fingerprint is fe:12:02:fb:49:de:8a:d1:c2:8f:82:54:63:d1:bc.
Are you sure you want to continue connecting (yes/no)? yes  ↩
Warning: Permanently added 'zipper.zip.com.au,61.8.0.87'
         (DSA) to the list of known hosts.
geoffrey@zipper.zip.com.au's password:  ↩
Last login: Sun May 20 17:14:57 2001
            from cpe-144-132-250-208.nsw.bigpond.net.au

geoffrey@zipperii:~$
```

30.3 Using Secure Copy—scp

30.3.1 Copying files to a remote host

To copy a file called local.txt from the local host to your home directory on a remote host:

```
bar@localhost~$ scp local.txt foo@remotehost.com.au:~/ ↩
foo@remotehost.com.au's password: ↩
local.txt   100% |**********************|   589    00:00
```

30.3.2 Copying files from a remote host

To copy a file called remote.txt from your home directory on a remote host to your current working directory on the local host:

```
bar@localhost~$ scp foo@remotehost.com.au:~/remote.txt . ↩
foo@remotehost.com.au's password: ↩
remote.txt  100% |**********************|   589    00:00
```

30.4 Exercise in using ssh and scp

For this exercise:

- Local hostname:
- Local username:
- Remote hostname:
- Remote username:

Open two xterm's: one will be a local host login; the other a remote host ssh login. Follow these steps.

1. Create a file in your local home directory called `<username>.local` where `<username>` is your user name. e.g.

   ```
   freddy@localbox:~$ id >freddy.local  ↩
   ```

2. Use scp to copy the file to a remote host.

3. Use ssh to login to the remote host.

4. Check that the file arrived:

   ```
   student@remotebox:~$ ls -l  ↩
   student@remotebox:~$ cat freddy.local  ↩
   ```

5. Make a copy of the file on the remote host called `<username>.remote` and append something to it:

   ```
   student@remotebox:~$ cp freddy.local freddy.remote  ↩
   student@remotebox:~$ uname >> freddy.remote  ↩
   ```

6. Copy the file called `<username>.remote` on the remote host back to the local host.

7. Check that it arrived back again:

   ```
   freddy@localbox:~$ ls -l  ↩
   freddy@localbox:~$ cat freddy.remote  ↩
   freddy@localbox:~$ diff freddy.local freddy.remote
   ```

Chapter 31

talk

Document Description: exercise in using the talk utility

Instructions:

- read through these sheets and check `$ man talk`
- have a "talk" with a few people on the network

31.1 talk

Talk is a visual communication program which copies lines from your terminal to that of another user. talk allows the user to have a two way chat session with another user. A typical talk session could look like this:

```
[Connection established]
pretty good... and your self?
okay
bye robbo




|-------------------------------------------------------|
Hi Geoffrey, how are you?
i'm excellent. but i'm very busy i'll talk to
you later
cu later
g
```

The screen divides into two and each person types in the top half and reads what the other person is typing in the bottom half.

31.2 enabling talk

31.2.1 enable the service

For talk to work the service has to be enabled.
As root edit the file `/etc/inetd.conf` and remove the `#` from the beginning of these two lines:

```
talk   dgram   udp   wait   root   /usr/sbin/tcpd  in.talkd
ntalk  dgram   udp   wait   root   /usr/sbin/tcpd  in.ntalkd
```

31.2.2 restart the network daemon

While logged on as root:

```
[root@freckle geoffrey]# /etc/rc.d/init.d/inet restart
Stopping INET services:                    [  OK  ]
Starting INET services:                    [  OK  ]
```

31.3 using talk

31.3.1 establishing a talk connection

```
[fred@box2 fred]$ talk quincy@box13.zork
```

31.3.2 replying to a talk request

```
Message from Talk_Daemon@bim.fernbank at 18:46 ...
talk: connection requested by robbo@freckle.fernbank.
talk: respond with:  talk robbo@freckle.fernbank
[quincy@box13 quincy]$ talk robbo@freckle.fernbank
```

31.3.3 communicating with talk

Type and read, type and read.

31.3.4 finishing a talk session

Just say goodbye and press `^C`.

31.4 multi way talk sessions

Try using ytalk, it is like talk but you can have three way (or more) talk sessions.

Chapter 32

Network File System—nfs

32.1 Using an nfs client to mount directories files on a remote host

This description assumes:

- The local host can access a remote host called remotehost over the network.
- remotehost is an `nfs` server exporting a directory called `/export/`

Remote file systems may be mounted in a number of ways.

- Using the `mount` command; here the exported `/tmp` directory of a remote host is mounted on the preexisting mount point `/nfs/tmp`:

```
# mount remotehost:/tmp /nfs/tmp ↩
# mount ↩
...
192.168.42.10:/tmp on /nfs/tmp type nfs (rw,addr=192.168.1.10)
```

- Mounting of remote file systems at boot time is achieved with an entry in `/etc/fstab`.

```
# tail -2 /etc/fstab ↩
# Server:directory  Mount Point Type  Options           Dump Fsckorder
homesbox:/home      /nfs/home   nfs   soft,timeout=100  0    0
```

- The remote file systems are unmounted in the normal way:

```
# umount /nfs/home ↩
```

- The startup script to mount remote file systems may be run thus:

```
# service netfs restart ↩
```

Note that this script also mount samba and netware remote file systems that are referenced in `/etc/fstab`.

- The modules nfs, lockd and sunrpc should be loded on demand when mounting. Check before and after with:

```
# lsmod ↩
and
# ps aux |grep rpc ↩
```

Chapter 33

nfs

Document Description: Exercise in setting up an using a network file system.

References Read the man/info pages for `nfs`, `nfsd`, `mountd`, `expotrfs`, `showmount`, `nfsstat`, `nhfsstone`.

Instructions: Read through these notes and do the practical exercises.

33.1 Server configuration—`nfsd`

33.1.1 The `nfs` and `nfsd` modules

The `nfs` module is required for mounting an exported file system and `nfsd` is required for exporting a file system. Either they must be compiled into the kernel or dynamically loaded.

- Load the modules:

```
# modprobe nfs ↩
# modprobe nfsd ↩
```

- Check the modules:

```
# lsmod  ↩
Module                  Size  Used by
nfs                    76800   0  (unused)
nfsd                   69984   0  (unused)
lockd                  52336   0  [nfs nfsd]
sunrpc                 62448   0  [nfs nfsd lockd]
...
```

33.1.2 Set exported directories in `/etc/exports`

The `exportfs` command is used to shows which file-systems are currently available for export.

- This example shows the `/tmp` directory may be nfs mounted by any host on the `192.168.1.0` C-class network.

```
# exportfs ↩
/tmp         192.168.1.0/255.255.255.0(ro)
```

- Export all entries in `/etc/exports`:

```
# export -a ↩
```

- Un-export all entries in `/etc/exports`:

```
# export -ua ↩
```

- Export the `/opt` directory to the host `other_box` for reading and writing:

```
# exportfs -o rw other_box:/opt ↩
```

- Note that no part of the file system may be exported more than once. This is broken:

```
$ tail -2 /etc/exports ↩
/home        192.168.1.0/255.255.255.0(ro)
/home/mary   blah.com.au(ro)
```

33.1.3 Services required to run an nfs server

- Check to see if the portmapper is running:

```
# service portmap status ↩
portmap (pid 444) is running...
```

- If the portmapper is not running, start it:

```
# service portmap start ↩
Starting portmapper:                                    [  OK  ]
```

- Start (or restart) nfs:

```
# service nfs start ↩
Starting NFS services:                                  [  OK  ]
Starting NFS quotas:                                    [  OK  ]
Starting NFS mountd:                                    [  OK  ]
Starting NFS daemon:                                    [  OK  ]
```

- Start nfslock

```
# service nfslock start ↩
Starting NFS file locking services:
Starting NFS statd:                                     [  OK  ]
```

- Check the remote procedure call processes:

```
# ps aux |grep rpc ↩
rpc         444  0.0  0.3  1484  168 ?        S    Oct21   0:00 portmap
rpcuser     459  0.0  0.0  1532    4 ?        S    Oct21   0:00 rpc.statd
root       6588  0.0  0.9  1340  424 ?        S    00:25   0:00 rpc.rquotad
root       6593  0.0  1.2  1460  564 ?        S    00:25   0:00 rpc.mountd
root       6600  0.0  0.0     0    0 pts/2    SW   00:25   0:00 [rpciod]
rpcuser    6625  0.0  1.6  1532  756 ?        S    00:27   0:00 rpc.statd
```

- Make sure you have some nfs daemons running:

```
# ps aux |grep nfs  ↩
root       6598  0.0  0.0     0    0 pts/2    SW   00:25   0:00 [nfsd]
root       6601  0.0  0.0     0    0 pts/2    SW   00:25   0:00 [nfsd]
...
```

- See where exported files are mounted on remote hosts:

```
# showmount ↩
Hosts on lapdog2:
192.168.1.2
```

33.2 nfs client

Remote file systems may be mounted in a number of ways.

- Using the `mount` command; here the exported `/tmp` directory of a remote host is mounted on the preexisting mount point `/nfs/tmp`:

```
# mount remotehost:/tmp /nfs/tmp ↩
# mount ↩
...
192.168.42.10:/tmp on /nfs/tmp type nfs (rw,addr=192.168.1.10)
```

- Mounting of remote file systems at boot time is achieved with an entry in `/etc/fstab`.

```
# tail -2 /etc/fstab ↩
# Server:directory  Mount Point Type  Options           Dump Fsckorder
homesbox:/home      /nfs/home   nfs   soft,timeout=100  0    0
```

- The remote file systems are unmounted in the normal way:

```
# umount /nfs/home ↩
```

- The startup script to mount remote file systems may be run thus:

```
# service netfs restart ↩
```

  Note that this script also mount samba and netware remote file systems that are referenced in `/etc/fstab`.

- The modules nfs, lockd and sunrpc should be loded on demand when mounting. Check before and after with:

  ```
  # lsmod ↩
  and
  # ps aux |grep rpc ↩
  ```

33.3 Practical Exercise

1. Explore, check and prepare your system.
   - Read, digest and apply the information in section 33.1 os this document.
   - Check that you have access to your local network—ping some hosts.
   - Load the required modules and check them.
2. Export some part of your file system.
   - Edit `/etc/exports` file; add a line with similar format to the following: (do *not* copy this line exactly)

     ```
     /tmp  192.168.1.0/255.255.255.0(rw) ↩
     ```
   - Use the export command.
3. Mount some remote nfs filesystems.

Chapter 34

Part XIII

Graphics

Chapter 35

The Gimp

The Gimp (Gnu Image Manipulation Program) provides an easy way using script-fu to produce cool and crufty Logos suitable for display on web pages.

35.1 Documentation

Explore the documentation.

- Users manual:
  - Online: `http://manual.gimp.org`
  - Dead tree: GIMP The Official Handbook CORIOLIS Press
- Web page: `http://www.gimp.org`
- Tutorials: `http://www.gimp.org/tut-basic.html`
- Man page: `$ man gimp`

35.2 File Formats

The Gimp can use and save many different file formats, a few of the most useful are:

XCF The Gimp native format; use this to store your images as you work on them, then export to the image format of your choice when you are finished.

TIFF (Tagged Image File Format) Industry standard for file exchange. Lossless.

JPEG (Joint Photographic Experts Group) Highly compressed lossy format suitable for photographs on web pages.

GIF (Graphics Interchange Format) Suitable for transparent Web graphics and GIF animations. Propriety format.

PNG (Portable Network Graphics) Lossless compressed format intended to replace GIFs.

PS (PostScript) Industry standard for printed documents.

35.3 *Exercise:* Make a logo with script-fu

Make a Logo to include on a web page. As a trial run to see how script-fu works follow these steps. Then experiment to make your own original logo for your web page.

1. Open The Gimp by either selecting it from a menu or executing it from an xterm: `$ gimp` ←. The Gimp toolbox should display, see Figure 35.1.
2. From The Gimp extensions menu (Xtns choose a logo script.
   Xtns / Script-Fu / Logos / Alien Neon . See Figure 35.2.

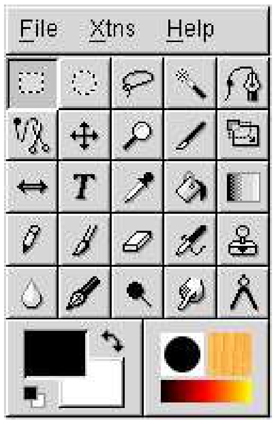


Figure 35.1: The Gimp Toolbox

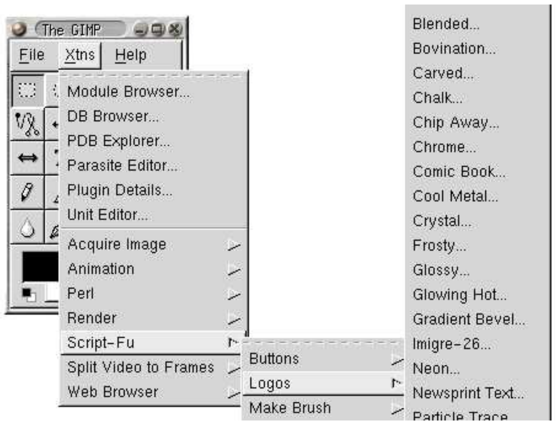
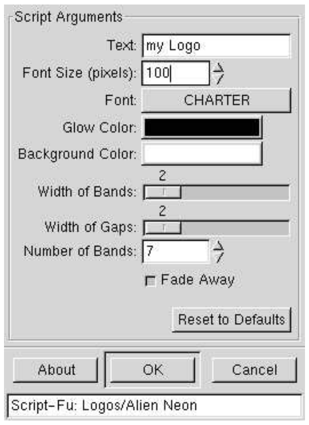


Figure 35.2: Selecting a Logo script

3. Select some appropriate options. See Figure 35.3 for the options chosen to give a simple black and white logo.
4. Right click on the image and choose File / Save As. For a webpage pick a low resolution compressed format like jpeg or png. Figure 35.4 shows a finished logo.



Script Arguments
Text: my Logo
Font Size (pixels): 100
Font: CHARTER
Glow Color:
Background Color:
Width of Bands: 2
Width of Gaps: 2
Number of Bands: 7
Fade Away
Reset to Defaults
About OK Cancel
Script-Fu: Logos/Alien Neon

Figure 35.3: Alien Neon Options

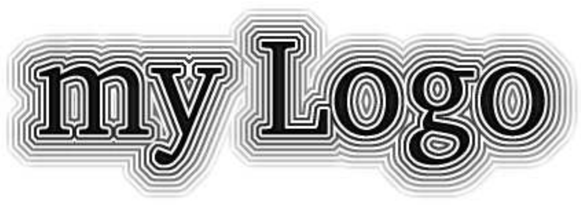


Figure 35.4: Web Page Logo

Chapter 36

xfig: vector drawing

Description: Exercise in learning to use the xfig drawing utility

Instruction: Draw some diagrams with xfig

36.1 documentation

Take a quick look at the documentation.

- Man page:`$ man xfig`
- Users manual: Start netscape and enter the URL
  `file:/usr/X11R6/lib/X11/xfig/html/index.html`
- PDF manual: `$ xpdf /usr/X11R6/lib/X11/xfig/xfig.pdf &`
- PDF xfig-howto: `$ xpdf /usr/X11R6/lib/X11/xfig/xfig-howto.pdf &`

36.2 display

A VGA screen resolution of 640x480 does not give sufficient real estate to run xfig. If X is setup correctly increase the resolution by pressing Cntl-Alt-keypadminus. Screen resolutions may be set up using the setup / xconfigurator or XF86Setup utilities. If all else fails use the virtual screens to use the off screen menus. Resize xfig to best fit your screen.

36.3 starting

- Start a new xfig figure by entering at the command line of an xterm:
  `$ xfig foo.fig.`
  The xfig utility should load and display.
- Set the units to metric. Click mouse button 3 in the box at the intersection of the rulers near the top right corner.
- Explore the menus. Find out how to save, print and open a file. Look at the help

36.4 creating xfig drawings

36.4.1 scribble

Have a scribble on the page. Try out the drawing tools.

36.4.2 draw

See Figure 1 below. Draw a picture of an egg in a truck on a bumpy road. For detailed help see the html howto. Save your drawing.

rectangle: Select the rectangle button; click left; move; click left again.

move: Move the rectangle a bit. Select the move button; left click on a handle; move the object then left click again.

resize: Enlarge the rectangle a bit. Select the resize button; left click on a corner, move to resise and left click again.

splines: Draw a bumpy road. Select a spline button: click; click; click.

rotate: Select ellipse button; left click move ane click again; edit button to edit the pattern and line thickness; rotate button to rotate.

text: Select the button labled T . Select the point size and font then click where you want the text placed and start typing.

36.4.3 draft

See Figure 2 below. Draw a circuit diagram using library parts. Select the library button and place a few parts from the Logic library. Connect the parts.

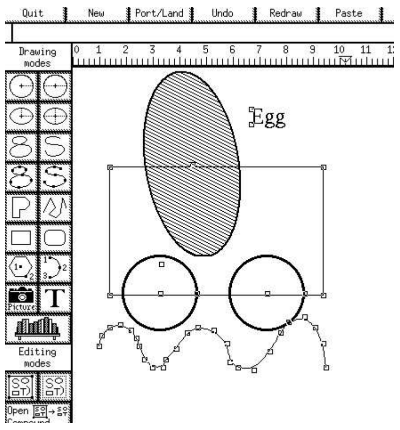
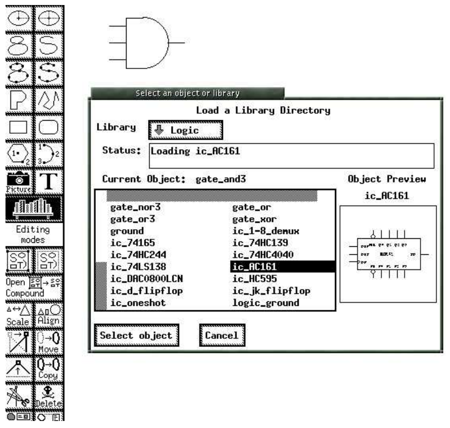


Figure 1



Load a Library Directory

Library: Logic

Status: Loading ic_AC161

Current Object: gate_and3

Object Preview: ic_AC161

```
gate_nor3        gate_or
gate_or3         gate_xor
ground           ic_1-8_demux
ic_74165         ic_74HC139
ic_74HC244       ic_74HC4040
ic_74LS138       ic_AC161
ic_DAC0800LCN    ic_HC595
ic_d_flipflop    ic_jk_flipflop
ic_oneshot       logic_ground
```

Figure 2

Chapter 37