Computer Designs & Concepts

Red Hat Linux

Course Manual

TABLE OF CONTENTS

Table of Contents (Continued)

Table of Contents (Continued)

1 Introduction to Linux

Linux (pronounced LIE-nucks) is the brainchild of the now famous Linus Torvalds. In 1991, as a graduate student of the University of Helsinki, Linus Torvalds started to develop a replacement for the UNIX-like operating system known as Minix. Whatever was created would have to run on Intel-based PCs. On October 5, 1991 Linus introduced the first version of Linux, version 0.02.

Linux is not public domain software. In fact, many components of Linux are copyrighted by many people including Linus Torvalds, who holds the copyright to the basic Linux kernel. The various distributions of Linux are copyrighted by the companies or people that supply the specific version of Linux. Many of the people that have contributed to Linux have protected their work under the GNU General Public Licence (GPL). The basic premise behind GNU is that software should be available to everyone. Without going into a lot of detail about how the licence works, if someone wants to modify the code they have the right to do so and by default, others have the right to the new code and can modify the new code if desired. That's the reason why Linux comes with the complete source code.

1.1 Linux - The Distributions

Linux is distributed by a number of different organizations, including:

- Red Hat (Red Hat's commercial version includes a commercial X server called Metro X)
- Caldera (this vendor uses Red Hat's)
- Slackware Linux
- MCC Interim Linux
- TAMU Linux
- LST
- SLS
- Debian Linux
- Yggdrasil Plug-and-Play Linux CD-ROM and the Linux Bible
- Trans-Ameritech Linux plus BSD CD-ROM
- The Linux Quarterly CD-ROM

Each distribution includes a current release of the Linux kernel (in the case of Red Hat 5.2 the kernel version is 2.0.36), as well as a collection of programs that the distributing company has supplied as a part of their total offering. Some distributions also include experimental kernels with drivers for specific types of hardware. Many of the distributions are supplied both over the Internet and on CD-ROM.

1.2 Linux and UNIX

The first UNIX operating system was developed in the AT&T Bell Laboratories, but many of us probably had our first exposure to UNIX using the Berkeley Software Distribution (BSD) UNIX (released in 1978) which was based on AT&T's Version 7. The BSD UNIX versions had a number of enhancements that made UNIX more user friendly for the casual user. Since that time a PC version of BSD has been distributed both on the Internet and via CD-ROM (BSD386 or FreeBSD).

UNIX System Laboratories (USL) was a spin-off company of AT&T that created its own version on UNIX called UNIX system V (starting in the early 1980s). Novell started working with USL and later purchased the company, selling its version of UNIX called UnixWare. This version of UNIX was later sold to the Santa Cruz Operation (SCO).

Not to be out done, Microsoft developed its own UNIX operating system called XENIX (in the early 1980s). With the advent of the 80386 microprocessor, Microsoft and AT&T merged their two operating systems to form a new UNIX called System V/386 Release 3.6. This operating system is still available today from SCO.

Sun Microsystems developed its own UNIX operating system based on the BSD UNIX. The resulting operating system, known as SunOS, up until recently was a PC workstation standard. (Linux is now reported to be the PC workstation standard.)

IBM also created a derivative of UNIX called AIX (Advanced Interactive Executive). While not as well known as some of the other UNIX operating systems, AIX holds its share of the operating system market.

1.3 An Overview of Linux

Linux is an advanced multi-user, multitasking operating system that is presently used by many Fortune 500 companies. While its power and capabilities are well known by hundreds of thousands of users, it requires that its user has an above average level of computer knowledge. For a brief list of the hardware supported by Linux please see **Table 1-1**.

Table 1.1 Linux Hardware Compatibility

CPU	Intel 386 and later (and compatibles), DEC Alpha, Sun Sparcs, and PowerMacs.
Bus	ISA, EISA, VESA local bus, and PCI; the MicroChannel bus isn't fully supported yet.
RAM	Minimum of 2MB of RAM; 4MB is recommended

Table 1.1 Linux Hardware Compatibility (continued)

Hard Drive	AT standard hard drive controller; Linux supports MFM, RLL, ESDI, and IDE controllers. Linux also supports several popular SCSI drive and CD-ROM drive controllers
Disk Space	Minimum of 20MB; 80MB is recommended.
Monitor	Linux supports Hercules, CGA, EGA, VGA, and SVGA video cards and systems; X Windows has other requirements.
Mouse	Any standard serial mouse (for example, Logitech, Microsoft, or Mouse Systems) or bus mouse from Microsoft, Logitech, or ATIXL
CD-ROM	Any CD-ROM drive that uses a true SCSI interface works; some proprietary CD-ROM drives such as the SoundBlaster series are also supported. CD-ROM drives known to work with Linux include NEC CDR-74, Sony CDU-45, Sony CDU-31a, Mitsumi CD-ROMs, and Texel DM-3042.
Tape Drive	Any SCSI tape drive works; other drives hosted from a floppy controller may also be supported. Now, the Colorado Jumbo 120 and 250 using the QIC 80 format are supported
Printer	If you can access your parallel printer from MS-DOS, you should be able to access it from Linux; some fancy features might not be accessible.
Ethernet Card	If you have access to an Ethernet network, Linux supports several standard Ethernet cards for accessing your network. Cards supported include 3Com's 3C503, 3C509, and 3C503/16; Novell's NE1000 and NE2000; and Western Digital's WD8003 and WD8013.

Some of the features that are included in every Linux distribution are as follows:

- A complete implementation of the TCP/IP networking Protocol
- A complete graphical user interface (GUI), Xfree86
- Unix portability - can be used on a number of different platforms
- Tools for program development (C and C++)
- Full e-mail system
- POSIX (Portable Operating System Interface) compliance - which allows different systems to communicate with each other
- Open system - allowing changes to source code
- A variety of text editors

Some of the challenges associated with Linux include:

- Lack of technical support
- Can be difficult to install with not all hardware being supported
- Cannot run DOS, Windows or Mac applications
- A Linux system must be managed

1.4 The Red Hat 5.2 Distribution

Red Hat has three features that make it more attractive to use than other flavors of Linux: it is one of the easier Linux distributions to install; it has an easy to use Red Hat Package Manager (RPM); and there has been a lot written about Red Hat Linux.

The Red Hat Linux 5.2 kernel is version 2.0.36. There are a number of important packages associated with the Red Hat Linux system's kernel that you will either encounter or need to know how to use.

dev - all UNIX systems use file system entries to represent devices attached to the computer.

initscripts - this package contains all the scripts (small programs) necessary to boot a system, change run levels and shut the system down.

passwd - password changing program.

There are a number of utilities that come with the Red Hat Linux distribution. Some of the utilities that you will encounter in this course include:

linuxconf - system manager with accessibility from text console, web interface, and GUI

gzip - a file compression and decompression program

tar - a powerful archiving package

control-panel - the Red Hat control panel that launches X programs

glint - a graphical interface to the RPM package management tool

LILO - (the LInux LOader) is responsible for loading Linux from either a floppy or a hard drive.

mount - add a new filesystem to the current directory structure. For example, devices must be mounted before they can be accessed.

rpm - powerful package manager used to build, install, query, verify, update and uninstall individual software packages

grep - this is a GNU implementation of the well-known utility that is used to quickly locate strings in a text files

less - an advanced text file viewer that is related to "more"

Some of the applications that you will use or encounter include the following:

minicom - a communications program that allows for complete control over the system's modem

postgresql - database management system

emacs - display editor

emac-X11 - emacs for X Window

netscape-communicator - web browser for X Window

The above lists are just a small subset of the overall list of utilities, tools, libraries, shells, window managers, and games that are included in the Red Hat 5.2 distribution. As you work through this course material, undoubtedly you will run into one or more of the Red Hat packages not listed above.

2 Installing Linux

Most IT professionals will want to run Linux alongside their already working networking operating system, until such time that a migration to Linux can occur without any loss of service or downtime. With that in mind we will load Linux onto a system that is already running Windows 95. While Windows 95 is not Windows NT 4.0, most of us realize that Windows NT 4.0 is actually a little more well behaved than Windows 95, so whatever we do with Windows 95 will work just as easily as it will with Windows NT 4.0. (Many of us have observed how Windows 95/98 loves to take over the whole computer).

2.1 Installation Overview

The objective of this section of the course is to install a fully functional Linux operating system on a computer that already has a Windows based operating system installed and working. In order to complete the task we will have to implement the following plan:

1. Determine what class of Linux operating system installation we want to install.

The base installation is just that. It includes whatever is needed just to get Linux up and running on the computer - which happens to be a lot, but would obviously not be enough for the power user. For example, a base installation would not include all the editors that are available under a workstation or server class installation.

The workstation installation is designed for the power user who not only wants nearly all the bells and whistles, but also wants all the programming and networking development tools that come with the Red Hat distribution.

The server class installation provides the packages necessary to run Linux as a network server.

Generally, there is a lot of overlap between the workstation class of installation and the server class of installation. If you are not quite sure what you want at this point it is okay, because it is relatively easy to add packages later on.

2. Know your hardware.

While Linux has come a long way over the years in terms of ease of installation and supplying drives for various types of hardware, it still has a ways to go before it can handle all the hardware that is available on the market today.

Two areas that require special attention are: the type of bus system used; and the type of monitor that is used.

Linux is generally very adept at handling ISA peripherals, so when possible, we suggest that you use ISA cards. For example, we use only ISA modems in our computers that run Linux.

Secondly, during the installation, the Red Hat installation program will try to probe your video card and monitor. We have found that unless you are fortunate to have exactly what the installation program is looking for, you will either have to select a monitor that is close to the type of monitor that you have or you will have to manually input the required technical information that is requested during the installation procedure. This step is especially critical if one wants to run the X Window system.

It is important to note that if the X Window system is not installed during the Red Hat Linux installation, it will have to be installed before it can be used. In order for the latter procedure to work, you will need to remember which video card was selected at the time of the system installation. In short, it may be easier to re-do the installation.

3. Determine how you want to partition your hard drive.

While a base class installation will require anywhere from 80MB to 200MB (if X Window is included), the workstation installation requires at least 600 MB of free disk space and a server class installation requires approximately 1.6GB of free disk space.

Linux Red Hat provides three different classes, or types of installation:

Workstation
- A 32MB swap partition
- A 16MB partition (mounted as /boot) where the Linux kernel and related files will reside.
- The remaining free disk space will be partitioned (mounted as /) in which all other files will be stored

N.B. this class of installation will automatically use LILO to configure your system for dual-boot.

Server
- A 64MB swap partition
- A 16MB partition (mounted as /boot) where the Linux kernel and related files will reside.
- A 256MB partition (mounted as /)
- A 512MB partition - minimum (mounted as /usr)
- A 512MB partition - minimum (mounted as /home)
- A 256MB partition (mounted as /var)

N.B. This case of installation will destroy all existing partitions on your system. So please remember to save your files before you choose this class of installation.

Custom — As the name suggests, selecting the class of installation provides complete control over the installation process. It will be your decision as to how the hard drive is partitioned; which packages are installed; and whether LILO is used to boot your system.

At a minimum, Red Hat Linux requires that at least two partitions be created - a native partition and a swap partition. In fact, each partition is a specific type of partition:

- One or more "Linux native" partitions must be created
- One "Linux swap" partition must be created

Within the parameters of a minimum swap partition of 16 MB and a maximum swap partition of 127 MB, the size of the swap partition is usually determined by the amount of system memory. For example, if the system has 32MB of RAM, then one should create a 32MB swap partition.

Ultimately one can choose the number of partitions that are created keeping in mind that generally all the files necessary to boot the system are stored in the root directory "/"; a /usr partition is usually where the Red Hat rpm packages are stored; and a /home partition is where users' home directories are created.

4. Determine which method will be used to install Red Hat, CD-ROM, NFS (Network File System), FTP (File Transfer Protocol) or from a hard drive.

While any of the above methods are acceptable, we will be using a CD-ROM to install Red Hat Linux. The only thing that we must remember to do is change the computer BIOS settings so that the computer will boot from the CD-ROM and not one of the computer's drives.

2.1.1 A Brief Description of the X Window System

The X11 Window system (or simply, X Window) was designed to be a machine/OS independent networked program. X Window is composed of two major components, an X Window server that runs off of the computer and interacts with the monitor and video card, and the various clients that can be used as displays for the X Window system. There are more than 50 X Window clients available. These clients range from simple terminal emulators, for example xterm, to very sophisticated X Window managers, for example KDE.

There are a number of different X Window servers as well. Some are free, such as XFree86 Project, which is part of the Red Hat distribution, while other X Window servers are sold commercially.

2.2 Installing Red Hat Linux 5.2 (Hands-on Laboratory)

After having set the computer's BIOS to allow for a CD-ROM boot, make sure that the CD-ROM is placed in the CD reader. Reboot the computer if necessary. The Red Hat installation program will provide you with a series of dialog screens that will guide you through the installation process. Many of the more important screens will be described below.

N.B. If your BIOS cannot be set to boot from a CD-ROM then you can create a boot disk by using the DOS utility rawrite. Go to the dosutil directory of the CD-ROM that contains the DOS utilities that came with the Red Hat distribution. Type the following command **rawrite -f d:\images\boot.img** (replace **d** with the drive letter for your CD-ROM) and then follow any directions that you are given.

You can navigate within a given dialog screen by using the arrow keys the tab key, the space bar and the enter key. Pressing F12 will accept the displayed values and move you to the next screen.

Each dialog screen is self-explanatory. Step through the screens making your choices as you proceed.

2.2.1 General Configuration Guidelines

- Installing from: CD-ROM
- Language used: English
- Type of keyboard: US
- Configuring TCP/IP: Static IP address (note BOOTP and DHCP both require active, properly configured bootp or dhcp server running on your LAN (local area network))
- Installation Path: Install
- Installation Class: Custom
- SCSI Adapters: None

When it comes to selecting a method for partitioning the hard drive, select the method that you are most comfortable using or try both.

2.2.2 Using fdisk to Partition a Hard Drive

If you decide to use fdisk please note the list of commands that you may need to use:

p - show partition information; you may want to copy down this information
d - delete partition
n - add a new partition
l - list
t - change partition's system ID

If fdisk is used, then the amount of space to be allocated for each partition can be specified by the number of cylinders (last cylinder), the number of bytes (+*size*), the number of kilobytes (+*size*K), or the number of megabytes (+*size*M). Remember to use the p command to verify that the changes that you have made are what you want. Also note, do NOT use the Linux fdisk to modify partitions for other operating systems. Doing so may render your hard drive completely inoperable.

2.2.3 Using Disk Druid to Accomplish various tasks

If you decide to use fdisk to partition your hard drive you will then need to use Disk Druid to set the mount points on specified devices. Before proceeding to use Disk Druid you may want to become familiar with the list of Disk Druid terms as shown in **Table 2.1 Disk Druid Terminology**.

Table 2.1 Disk Druid Terminology

Mount Point	Just another term for directory. This is the location where the directory and all the associated subdirectories will be placed.
Device	The name of the physical hard drive and the name of the partition on the hard drive.
Requested size	The minimum size requested for the partition when the partition was being created.
Actual	The amount of space currently allocated to the partition.
Type	Displays the partition type
Geom [C/H/S]	Displays the hard drive's geometry. The number of cylinders, the number of heads and the number of sectors.
Total	The total amount of hard drive space.
Used	The total amount of used hard drive space.
Free	The amount of space which has not been partitioned.
Bar Graph	A symbolic representation of the amount of space used on the hard drive.
Add	Add a new partition
Edit	Edit the selected partition
Delete	Delete the currently highlighted partition
Ok	Writes the changes to the hard drive and continues the installation process
Back	Aborts all changes made and returns to the previous dialog screen.

If you used fdisk to create the hard drive partitions, then you will want to edit each of the partitions so as to provide a mount point or directory name. At least one of the partitions must have a "root" mount point. While in edit mode for the partitions that will contain the root mount point, highlight the Mount Point field and enter / . This will tell Linux that this partition is the root partition. You can proceed to edit any additional partitions in exactly the same way, supplying the desired directory name for the Mount Point field - e.g. /usr for the usr partition.

If you are using Disk Druid to add partitions to the hard drive, you may want to become familiar with the Add screen terminology as shown in **Table 2.2 Add Screen Terminology -Disk Druid**.

Table 2.2 Add Screen Terminology - Disk Druid

Mount Point	Enter the name of the mount point (directory). There must be at least one directory and it is called / .
Size (Megs)	Enter the size of the partition. The default is 1MB.
Growable	Indicate if the size of the partition is a minimum size (growable) or a maximum size. Use the space key to toggle the check box on or off.
Type	Select the appropriate partition type using the arrow keys.
Allowable Drives	Shows the hard drives that are in the system. Linux can be directed to use a specific hard drive for the partition being added or Linux can be allowed to make the selection on its own. In any case, if at least one box is not selected, the partition will never be created.
Ok	Selected when all partition information has been entered.
Cancel	Cancel the add partition process.

2.2.4 Activating the Swap Partition

The Red Hat installation will automatically ask if you wish to initialize the swap partition and whether you want to check for bad sectors while it formats the swap partition.

2.2.5 Location of the Red Hat Directory Tree

If there is more than one partition then you will have to indicate where the Red Hat directory tree is located. This is accomplished in the Select Partition screen.

2.2.6 Formatting Partitions

The Partitions to Format screen allows selection of partitions for formatting. All new partitions should be formatted and all old partitions that contain data that you no longer want should also be formatted. Partitions that you may not want to reformat include /home and /usr/local since they may contain important data that you wish to keep. At the same time, one can also select to have the installation program check for bad blocks during formatting.

2.2.7 Selecting and Installing Packages

The next step is to select the components that you want to install. At the same time, it is possible to indicate whether you want to select individual packages within a given

component. Many of the packages are dependent on other packages being selected. The Red Hat installation program checks for these dependencies and alerts the user of any problems.

If you want to install all of the components and their associated packages, you will need more than 700MB of free hard drive space.

Once the components and packages have been selected the installation program will proceed to the Install Status screen. In addition to displaying real time installation information, the installation program will create a log file /tmp/install.log that will list all of the packages that are installed.

2.2.8 Mouse Configuration

After the packages that have been successfully installed, the computer is then probed to determine if a mouse is present. You can select a mouse from the provided list of mice or you can select a generic entry. While it is reported that selecting to emulate a three-button mouse allows for an easier use of X Window, we have not had any problem using a two-button mouse. (The three-button mouse emulation allows a middle mouse button to be emulated by clicking the left and right mouse buttons simultaneously.)

You will be asked to select the serial port to which your mouse is connected.

N.B. The Red Hat Linux mouse configuration can be changed at any time using the /usr/sbin/mouseconfig command.

2.2.9 XFree86 Server Configuration

N.B. Special care must be taken to try to select a monitor and video card that is as close to your computer's hardware configuration as possible. Selecting the wrong chipset for your video card can lead to destroying both the video card and the monitor.

The installation program will automatically launch the Xconfigurator utility and probe your system in an attempt to determine the type of video card that is being used. If this autoprobe fails, you have to select a video card from the list of video cards displayed. If you have enough knowledge about your video card you can opt to choose Unlisted Card and match the card's video chipset with one of the available X Window servers.

N.B. If you have no idea what clockchips are on the video card, then we suggest that you choose the No Clockchip Setting.

Once a video card has been selected, the installation program installs the appropriate XFree86 server and a list of monitors appears on screen. Select your monitor or select Custom. If Custom is selected the horizontal and vertical sync ranges will have to be supplied. This information can usually be gathered from the documentation that came with the monitor.

N.B. It is not recommended that you select a monitor that is similar to your monitor unless you know that the monitor that you have selected does not exceed the capabilities of your monitor. If you do, you may damage or destroy your monitor.

The next screen prompts for the amount of memory on the video card. An incorrect answer at this point will not cause hardware damage, but the XFree86 server may not start correctly.

Finally, you are prompted for the screen resolution you wish to have displayed. You will want to select a resolution that you know will work on your monitor.

2.2.10 Network Configuration

Here you have the opportunity to keep the present network configuration, reconfigure the network, or not setup a network at this time. Electing to configure the network leads to being provided with a number of dialog windows. At this stage we will select a Static IP address. **Table 2.3 Sample Networking Information** should be used as a guide to help input your network information.

Table 2.3 Sample Networking Information

Field	Example Entry
IP Address	192.168.0.3
Netmask	255.255.255.0
Default Gateway	*leave blank for now*
Domain Name	*leave blank for now*
Hostname	SS1

2.2.11 CMOS Clock Configuration

Select Greenwich Mean Time (GMT) if you want your hardware clock to properly handle daylight savings time.

N.B. With a dual boot system where both OSs are allowed to alter the clock, more than likely the clock will show the incorrect time when daylight savings time is compensated for.

2.2.12 Start on Reboot Services

Please skip this section if you are installing a workstation or server class installation.

The services screen allows for the selection of services that will start automatically every time the Red Hat Linux system boots. For more information about each service, highlight the service of interest and press the F1 key.

It is possible to change what services will start on reboot after the installation is complete by running /usr/sbin/ntsysv or /sbin/chkconfig.

2.2.13 Printer Configuration

If you choose "Yes" to configure a printer, you will have to select how the printer is connected.

- Local - The printer is connected directly to your computer.
- Remote lpd - The printer is connected to your LAN either through another computer (or directly) and is capable of communicating via lpr/lpd.
- LAN Manager - The printer is connected to another computer which shares the printer via Lan Manager (or SMB) networking.

N.B. SMB (Session Message Block or Samba) protocol services allows for the sharing of files and printers between Windows 95, 98 or NT and Linux.

SMB will be discussed in some detail in **Section 6** of this manual.

Local printer setup continues with the installation program attempting to determine which printer ports are available. Enter the name of the printer device in the field provided. Please skip to the paragraph on completing the printer installation.

Remote printer setup is accomplished by providing the name of the computer to which the printer is connected (e.g. comp1.redhat.com) and the name of the print queue on the remote computer to which the printer is connected.

Completing the printer installation requires selecting the appropriate choices shown in the Configure Printer screen, paper/resolution screen, and the Verify Printer Configuration screen.

2.2.14 Choosing a Root Password

The installation program will prompt you for a password for your root directory. Once you start using Linux you will quickly see how important it is to have root directory accessibility password protected. You will have to select a password that is at least 6 characters in length. Also remember that Linux is case sensitive, so your password is case sensitive.

2.2.15 Boot Disk Creation

Creating a boot disk can be advantageous for a number of reasons, not the least of which is to reinstall LILO if another operating system should happen to overwrite the master boot record (MBR).

N.B. The creation of a boot disk is highly recommended. Remember to create a new boot disk each time changes are made to the kernel. More information on this subject is available from the mkbootdisk man page. (More on man pages later.)

2.2.16 LILO Installation

Workstation and server class installations have this completed for them automatically.

LILO can be installed in one of two locations. As previously mentioned, LILO can be installed in the MBR. If a Windows OS has already been installed on the system, its loading program is already located in the MBR. LILO will replace the Windows loader and upon booting the system, LILO will provide an option to boot whichever operating system you want.

Selecting to place LILO in the first sector of the Linux boot partition is advantageous when the system already has a boot manager (e.g. System Commander) already located in the MBR.

Next, the installation program will ask for any default options that are to be passed to the kernel at boot time. Also, select linear mode if the computer's BIOS uses LBA mode to access the computer's hard drive.

N.B. LBA (logical block addressing) is a method of linearly addressing sector addresses. Each sector on the drive is numbered starting at 0, with the maximum ending number of 268,435,456. LBA was new in ATA-2, but has always been used in SCSI.

Finally, the installation program will display all the possible bootable partitions, including partitions of other operating systems. This is where one can change boot label names and which OS will be the default.

N.B. At boot start, if you forget the boot labels, press the ? key and LILO will display the list of possible boot labels.

3 Linux Basics

Now that the Linux operating system is up and running, you will probably want to start finding your way around and even start using the system for something useful. If you have used only Windows operating systems then you are more than likely in for a big change when you start using Linux. On the other hand, if you have a good knowledge of DOS, or even better still, some knowledge of UNIX, then the learning curve should not be as steep. In any case, the goal of this section is to provide you with enough knowledge to use Linux effectively.

3.1 Notational Conventions for Linux and Unix

As you might have guessed, the need for documenting UNIX and Linux has resulted in the development of a clear and concise system of communicating the command syntax for these two operating systems. This notational convention specifies such things as what options or parameters you must use and what options or parameters you can or cannot use.

Basically, all you need to remember are six rules in order to understand the syntax of any Linux or UNIX command.

1. Text which is shown by itself and not written within {}, [ ], or <>, must be typed as shown.

2. Text within square brackets [ ] is optional and may be typed or not typed, as desired. In either case the square brackets themselves are not typed.

3. Text within angle brackets <> must be replaced by appropriate text, usually a name. Again, the angle brackets themselves are not typed.

4. One value must be chosen when one or more values are provided within curly brackets {}. The values are separated by | which in this case means or, not piped. The curly brackets are not typed.

5. The ellipsis ... is used to indicate "more" or "and so on". It is used most often when wanting to indicate, for example, one or more files (<filename>...).

6. Brackets may be combined as necessary in order to communicate the command syntax. For example, [<filename>...] indicates that the use of one or more filenames is optional.

3.2 The Shell - bash

Linux is command line driven. Just like DOS, which had a command interpreter called COMMAND.COM, Linux uses a command interpreter as well, called bash (bourne again shell - clone of the shell bourne). Unlike DOS, however, Linux can make use of a number of different shells (Red Hat includes sh, tcsh, csh, pdksh, zsh, ash and mc shells in its distribution), the default shell is bash. In short, a shell is simply a program that accepts commands and executes them. bash is located in /bin/bash.

To determine which shell you are presently using enter

echo $SHELL

The echo command prints whatever follows the word echo to the terminal screen. **SHELL** is a variable that is maintained by whatever shell is running; **$SHELL** is the value of the variable **SHELL**.

Let's digress for a moment and talk about device drivers. It is important to understand that Linux looks at every device, or piece of hardware (e.g. keyboard, CD-ROM, monitor, etc.) just like every other device, and all devices look like files. It is up to the device drivers for each piece of hardware to accomplish the task of, for example, making a keyboard look like a CD-ROM. The device drivers are actually programs that lie between the hardware and the shell.

The computer terminal, which in the case of PCs consists of a monitor and a keyboard, is defined as a tty (teletype) device. Since the terminal is always connected to the computer, the device driver for the terminal is designed to allow the user to define control characters that can be sent to a running process such as a shell or an application program. These controls can be defined and displayed by the stty (set teletype) command. To see present settings, enter **stty -a**.

Depending on what the computer is being asked to accomplish, the device driver is either passing each typed key directly to the shell (e.g. running an editor or a spreadsheet) or it's interpreting every typed key before passing the information onto the shell. The two different modes of operation are called "raw" and "cooked" respectively.

Now let's return to our discussion about the shell. The shell that is used is just one of the things that goes into define the shell environment or simply, the environment. By typing **env** one can see the values of the various parameters that determine the environment. For a list of the parameters please see **Table 3.1 bash Environment Variables**.

Table 3.1 bash Environmental Variables

Home=/home/login	Home sets the home directory when you first logon. The login is replaced by your login ID.
LOGNAME=login	Automatically set to your login ID.
PATH=path	List of directories that are used by the shell to find commands and programs
PS1=prompt	The primary shell prompt. For non-root user this set to $ (unless changed by the user).
PWD=directory	Displays the current directory.
SHELL=shell	Identifies the location of the shell that is presently being used.
TERM=termtype	Sets the name of the terminal as defined by the terminal database.

The **PATH** variable can be modified easily by supplying a colon-delimited list of directories that should be sequentially searched by the shell. For example: **PATH= /usr:/bin:/usr/bin:/usr/local/bin**

The **PS1** variable can be changed in a variety of ways using special characters as listed in **Table 3.2 Prompt Special Characters**.

Table 3.2 Prompt Special Characters

\!	Displays the history number of this command.
\#	Displays the command number of the current command.
\$	Displays the $ sign unless the user is root. The # signifies the root user.
\\	Display a backslash.
\d	Display the current date.
\h	Display the host name of the computer on which the shell is running.
\n	Prints a newline character.
\nnn	Displays the character that corresponds to the octal number nnn.
\s	Displays the name of the shell that is running.
\t	Displays the current time.
\u	Displays the user name of the current user.
\W	Displays the base name of the current working directory.
\w	Displays the current working directory.

These special characters are used by typing quotations around them. For example: **PS1="\t"** will change the prompt so that the current time is displayed as the prompt.

bash has a feature called command line completion. When you start to type a command at the command line prompt, bash will try to complete the command for you. The guess can be seen by pressing the **Tab** key. bash will either provide its suggestion and hitting enter will accept the guess, or bash will beep, signifying that bash does not know what command you want to enter and more information is required.

bash also makes use of the following wildcards:

*	used to match any number of characters
?	used to match any single character
[…]	matches certain characters or ranges of characters within the brackets

bash supports command history. In other words, it keeps track of the commands that are entered and the order in which they are entered. The number of commands it will track is set by the shell variable **HISTSIZE**. Setting the file name of where the command history is stored is accomplished using the **HISTFILE** bash variable.

Previous commands can be accessed using the up- and down-arrow keys. It is a simple and quick way to scroll through the commands that have been previously entered. Note that the command, once recalled, can be edited using the left- and right-arrow keys, the delete key, the backspace key, and re-entered using the enter key.

Another feature that is useful when using bash is the ability to set aliases. Alias commands are usually abbreviations for longer commands that you don't wish to type over-and-over again. A new alias is defined by typing the following line command

alias <new command>="<old command>"

In order to display a list of all the aliases, type alias. To delete an alias, type **unalias** <new command>.

The bash shell provides input and output redirection through the use of the < and > characters respectively.

The bash shell provides pipelining using the | character. Pipelining allows the user to designate the output from the first command to be the input for the next command in the pipeline.

3.3 Linux Commands

There are many Linux commands that are not built into the bash. There are so many different commands that it would be virtually impossible to list all of the possible commands in the text. For example, the **ls** command (list directory command) has over 75 command line options. The following tables provide a summary of various commands and some of their options. For more information please refer to the appropriate manual page.

Table 3.3 Working with Directories

cd /usr/bin	Change directory to /usr/bin.
cd..	Move up one directory. In this case to /usr
cd	Return to your home directory
pwd	Print working directory - displays current working directory

Table 3.4 Finding Files

find /usr -name spell -print	Find spell in the /usr directory and display
find /usr -name '*.ps' -print	Find all PostScript files in the /usr directory and display
whereis <filename>	Find the location of the file's binary, source and manual pages
locate <filename>	Must build the database first using updatedb. The database is called locatedb and resides in the /var/lib directory
whatis <filename>	Must build the database first using makewhatis. The database is called whatis and resides in the /usr/man/man1 directory
apropos <keyword>	Used to search your whatis database for the keyword that was entered.

Table 3.5 Listing File and Directory Information

ls	List directory - lists nearly all files and directories in the current directory
ls -F	Indicates directories by / and executable files by *
ls -a	Lists all directories and files
ls -l	List files and directories in long format
ls <path>	List files and directories with a specified directory
ls <*****.extension>	List all files with a specified extension
ls -R	List all the files and directories on your system
dir	See ls above
vdir	See ls -l above
tree	Displays a graphical representation of the current directory
tree -d	Displays a graphical representation listing all directories

Table 3.6 Creating, Removing, Moving, and Copying - Files and Directories

touch <filename>	Creates a new file or updates a file's date and time
touch -t 1120090099 <filename>	Specify file date and timestamp
rm <filename1> <filename2>	Delete one or more files
rm -i <filename*>	Delete specified file(s) interactively
rm -f <filename>	Forces the deletion of a file

Table 3.6 Creating, Removing, Moving, and Copying - Files and Directories (continued)

rm -fr <filename*>	Forces file(s) to be deleted recursively
mkdir <dir1> <dir2>	Makes one or more directories
mkdir -p <dir1>/<dir2>	Makes a parent directory and a child directory
rmdir <dir>	Removes a specified directory
rmdir -p <dir1>/<dir2>	Removes specified parent and child directories
rmdir <dir1>/*	Removes parent directory and all subdirectories
mv <filename1> <filename2>	<filename1> is renamed <filename2>
mv <dir1> <dir2>	<dir1> is renamed <dir2>
mv -b <filename> <filename>	<filename1> is renamed <filename2> and a backup of <filename2> is made
mv -i <filename> <filename2	<filename1> is renamed <filename2> interactively
cp <filename1> <filename2>	Copies <filename1> to <filename2>
cp -i <filename1> <filename2>	Copy <filename1> to <filename2> interactively
cp -bi <filename1> <filename2>	Copy <filename1> to <filename2> interactively and backup <filename2>
cp <dir1>/***** <dir2>	Copy all the files in <dir1> to <dir2>
cp -r *dir1 dir2*	Copy <dir1> and all of its files into <dir2>
cp -p <dir1>/<dir2>/<filename> <dir3>	Copies the <filename> and its directories to <dir3>

Table 3.7 Linking

ln <filename1> <filename2>	Creates a hard link between <filename1> to <filename2>
ln -s <filename> <name>	Creates a symbolic link between the <filename> and <name>
ln -s <dir1>/<dir2>/<dir3> <name>	Links <dir3> to <name>

Table 3.8 Concatenate Files

cat <filename>	(concatenate) Display contents of the file <filename>
cat -n <filename>	Display contents of the file <filename> with line numbers
cat <filename1> <filename2>	Display contents of both <filename1> and <filename2>
cat <filename***** > <filename3>	Combines data from all specified files into a new <filename3>
cat <filename1> **>>** <filename2>	Appends data from <filename1> to data in <filename2>
cat > <filename>	Use cat as a simple editor. Type Ctrl-D to quit and save.

3.4 grep - Global Regular Expression Parser

grep, along with egrep and fgrep, allows you to search through a file or a series of files, looking for patterns of text that match. Each "grep" has over 20 different command-line options that can be used. Specific examples of various **grep** commands are provided in **Table 3.9 grep - Command Summary**.

Table 3.9 grep - Command Summary

Specific Example	Return Description
grep help <filename>	Returns each line in <filename> that contains the word "help"
grep .ay <filename>	Matches any single character (except a line return) combined with "ay"
grep [mM]ake <filename>	Matches both "make" and "Make"
grep [1-5] <filename>	Matches "12345"
grep [+] <filename>	Search for the "+" character
grep \+ <filename>	Search for the "+" character using \ as an escape character
grep \+ -n <filename>	Search for the "+" character, display line numbers
grep ^704 <filename>	Matches "704" only if at the beginning of the line
grep 704$ <filename>	Matches "704" only if at the end of the line
grep 7\{2,3} <filename>	Matches "77" or "777"
grep 'more' /temp/*	Matches "more" in all files located in the temp directory

3.5 Processes

Linux is a multitasking operating system and is therefore capable of running two or more programs at the same time. Each program that is running under Linux is called a process. The Linux operating system assigns a process ID (PID) to each running process. To see which processes are running and their associated PIDs, type the command **ps**. To display more information about the running processes, type the command **ps -guax**. ps has a number of different flags or command options that can used, to see a list of some of the more useful options, see **Table 3.10 ps - Summary of Command Options**.

Table 3.10 ps - Summary of Command Options

ps	Displays running processes and their associated PID
ps -m	Displays the running processes and the memory that is used by each process

Processes can be stopped, if necessary, by using the kill command. The kill command has a number of command line options. While using the **-9** (SIGKILL) option is the most drastic choice, a list of the other options can be displayed by typing **kill -l**.

3.6 Getting Information and Help

3.6.1 man Command

There are a number of different places where you can get help with Linux. The Red Hat Linux distribution comes with what are called manual pages. These manual pages cover nearly every utility, command and programming system that comes with the distribution. This information can be accessed by typing **man** <topic>. For a listing of the various directories that contain manual page information, see **Table 3.10 Directories Containing Manual Page Information**.

Table 3.10 Directories Containing Manual Page Information

Directory	Description of Contents
/usr/man/man1	Commands run from within a shell
/usr/man/man2	Documented kernel functions
/usr/man/man3	Documented library calls (libc functions)
/usr/man/man4	Information on files in the /dev directory
/usr/man/man5	File format information
/usr/man/man6	Information about games
/usr/man/man7	Description of the man pages and Linux file system
/usr/man/man8	Information on root operator utilities
/usr/man/man9	Documentation on the Linux kernel source routine

N.B. To exit from viewing a man page press the **Ctrl-**,which is the terminal device driver command for quit.

3.6.2 --help

Many of the Linux commands have help available by simply typing the command followed by **--help**. For example, typing **dir --help** will display the help page for the command dir. Many of the bash commands (e.g. **ls** or **ps**) have both a **--help** and a **man** page, while Linux utilities (commands), such as **dir**, only have a **--help** screen.

3.6.3 Documentation in /usr/doc

There are a number of subdirectories filled with helpful information found in the /usr/doc directory. Some of the more commonly used subdirectories are the FAQ (frequently asked questions) directory and the HOWTO directory.

3.6.4 On-line Help

There are a number of websites, too many to list, that offer excellent help and information about Linux. A short list of sites that one might want to try include:

http://www.redhat.com
http://sunsite.unc.edu/linux
http://www.linux.org
http://www.mcp.com

3.7 Linux Basics (Hands-on Laboratory)

3.7.1 Working with the Environment

1. Confirm which shell you are using by typing:

   echo $SHELL

2. Check your shell environment by typing:

   env

   See how many of the environment variables you recognize in the env listing.

3. See if the sh shell is available to you by typing:

   sh

4. Check which processes are running by typing:

   ps

   You should see **sh** as one of the running processes. See if you discover where the **sh** shell is located by just using the echo command along with **$SHELL**.

5. To exit the sh shell and check the processes, type:

   exit
   ps

   You should be back to the bash shell.

6. Once you are back in bash, use the set teletype command to see the present terminal environment.

   stty -a

7. Try changing the kill key to ^C

   stty kill '^C'

   Check for the new setting.

8. To return to the default values, type:

   stty sane

9. Display your terminal environment settings to see if the default values are now being used.

10. Check the value of your path variable by using the **env** command or:

    echo $PATH

11. Append a new path to the present path variable by typing:

    PATH=$PATH : /etc

    Use one of the above commands to verify the change to the path variable.

12. Next, let us work with changing the primary prompt. Change the prompt to just display the date.

    PS1='\d'

    Note the difference between this prompt and the command

    PS1=\d

    Try some of the other special characters and see if you can create a prompt that you like.

13. Finally, see if you can create a prompt that includes the following information and look

    user name: *hostname* / *base name of the working directory* #:

3.7.2 Working with alias

1. Display the aliases that are already defined for your system

   alias

2. In order to display the manual page for the man command, type the following:

   man man

3. Use the alias command to change the man command to help

   alias help="man"

   Now try using the new help alias to display the manual page information on manual pages, ls and any other commands that you may be interested in viewing. For example:

   help man

4. Design an alias for the **ls** command that you will call "timedir". Determine what flags for the **ls** command you will need in order to list all the files, their size, and sorted horizontally. Create an alias called sortdir assigning this series of commands to the new alias.

5. Also, unalias the help alias from the **man** command. Check to make sure that it works.

3.7.3 Working with Files and Directories

1. Make a new directory, use your first name as the name of the new directory - we will refer to this directory as the first-name directory. Use the **cd** command to go to your new directory.

   Make a second directory, using your last name as the new directory name - we will refer to this directory as the last-name directory. Go to the directory that you just created.

2. Using the **cat** command, create a file called memo1 and input some lines of text. You decide what goes in the new file. You can use the **^D** command to indicate the end of file once you have input the text that you want.

   Create a second file called memo2. Place some text into the new file.

Create a third file called copymemo1 using the **touch** command. Check the contents of copymemo1.

3. Use the copy command to copy the contents of memo1 into the file copymemo1. Check the contents of copymemo1.

   Add more text to the file copymemo1 by using the following command:

   cat >> copymemo1

   Once again check your results using the **cat** command.

4. Return to your home (root) directory and attempt to remove your first-name directory.

5. Go to your last-name directory and move all the files to your first-name directory using the **mv** command.

   Take the necessary steps to delete your last-name directory using the **rmdir** command.

6. Create a hard link between hardlink and memo1. Delete memo1 and then list all the files in your first-name directory. Display the contents of the hardlink file.

   Create a soft link between softlink and memo2. List the files in the directory using the **-l** flag. Display the contents of softlink. Next, delete memo2 and then try to display the contents of softlink. List the files in the directory using the **-l** flag.

7. Take the necessary steps to remove the first-name directory.

3.7.4 Finding files

1. Use **whereis** to find **man**.

2. Use the following command to find the spell file.

   find /* -name spell -print

3. Search for all the text files present on your hard drive. Try different level of searches by varying the number of **/*** used and see how your results change.

3.6.5 Working with grep

1. Go to /usr/man/man1 directory. Using the man.1 file or any file which you choose, complete the following grep searches:

   Search for the word **help**
   Search for the words **make** or **Make**
   Search for the numbers **1** or **2** or **3** or **4**
   Search for lines containing the character **/**
   Search for lines which have the word **will** as the first word
   Search for the number combinations **77** or **777**
   Search for the word combination **look for**
   Search for the character string **.B** and display the lines

4.0 System Back-up and Recovery

4.1 The Use of tar and gzip

4.1.1 tar

tar (tape archiver) is a very useful archiving utility that can be used to back up files to tape or disk. The **tar** command format is as follows:

tar <action> [optional functions] <file(s)/directory(ies)>

If the last parameter is a directory, then all the files in that directory will be tarred.

A list of actions can be found in **Table 4.1**, for a more complete list refer to the appropriate man page.

Table 4.1 Short List of tar Actions

c	Create an archive
r	Append files to an existing archive
t	List the files in an archive
u	Update archive with files that are newer than the archive files
x	Extract files from the archive

A list of some of the optional functions can be found in Table 4.2, for a more complete list please refer to the appropriate man page.

Table 4.2 Short List of tar Optional Functions

f	Specify archive media device name
l	For unresolved links, display error messages
n	Archive media is not a tape
p	Extracts files with their original permissions
v	Verbose output, displaying filenames
w	Waits for user confirmation of archive action

N.B. It is important to remember that **tar** remembers paths. If you tar a directory or directories, then when you extract the tarfile, all the files and their associated directories will be recreated.

4.1.2 gzip

Another commonly used utility is the GNU gzip program. Like its predecessor compress, gzip is used to compress files using some of the most current compression algorithms.

One can even specify the amount of compression through the use of a numerical switch (**-1** to **-9**; with **-9** providing the greatest amount of compression). The command format for gzip is as follows:

gzip [numerical switch] <filename>

To uncompress a gzipped file you need to use the gunzip utility.

N.B. gzip automatically appended a .gz extension to the compressed file and automatically deletes the original file. Also, if a numerical switch is not specified then gzip will use its default settings.

It is also possible to **gzip** a tarfile. The resulting file would have a .tar.gz file extension to indicate that one should uncompress first and then untar. The reverse is also used, where the gzipped file is tarred. In this case the file extension show appear as .gz.tar.

4.2 Hands-on Laboratory (tar and gzip)

1. If you are not in your home directory then return to your home directory.

   Create a directory called archive and use the **cd** command to move to your new directory.

   Use whichever method you like to create a text file called archtest.

2. Use the following command to **tar** the archtest file

   tar cvf archtest.tar archtest

   Display the file information in the archive directory. What do you see?

3. Remove the archtest file and again display the file information for the archive directory. What do you see?

4. Using the following command, extract the archtest.tar file

   tar xvf archtest.tar

5. Display the file information for the archive directory. What do you see?

6. Using the following command, gzip the archtest.tar file

   gzip -1 archtest.tar

7. Diasplay the file information for the archive directory. What do you see?

8. Using the following command ungzip and untar the archtest.tar.gz file

   gunzip archtest.tar.gz | tar xvf -

   Note that the hyphen after the tar actions replaced the filename. The is typical UNIX terminology which translates to "use as input, the output from the pipe".

4.3 Recovering a Lost root Password

If you ever have the misfortune of forgetting a root password then you will realize how important it is to be able to remedy this problem. Use your boot floppy to boot the system. Mount the root partition and edit the /etc/passwd file to remove any password for the root. Reboot your system from the hard drive.

Once your system reboots you will be prompted for a new root password.

N.B. The ability to use this technique is a security risk.

5.0 The Linux Kernel

The Linux kernel is really what Linux is all about. It is the program that acts as the interface between the user's applications and the computer hardware. Linux, like all operating systems, is responsible for performing multitasking, disk read/writes, memory management, networking and a host of other operations.

5.1 Operating Systems

When you think about it, there are relatively few operating systems to choose from in today's marketplace. Most of today's operating systems have a number of features in common, it is usually a question of system performance, i.e. how efficiently a task is performed, and system stability, how frequently the operating system crashes.

One of the critical features of any operating system is how it handles one or more tasks at a time (multitasking). Linux and Windows NT both operate using preemptive multitasking where the operating system, more specifically, the kernel, allocates system resources on an application by application basis. One of the keys to system stability is the ability of the operating system to isolate tasks (processes) so that if there is a problem with a given process, that problem does not bring down the entire system.

Another key feature of any operating system is its portability or the ability to run the operating system on any number of different computers (platforms) with little-to-no platform specific code having to be supplied. UNIX operating systems are well known for their portability and this is a feature that Microsoft Windows NT has tried to incorporate, but falls short of in such areas as networking and security.

For operating systems that act as the heart of a LAN or WAN, their networking ability is critical. UNIX, and as a result Linux, was designed for use by multiple users, both local and remote. The TCP/IP (Transmission Control Protocol/Internet Protocol) network protocol is at the core of the UNIX multiuser operating system and one of the reasons why UNIX is the Internet operating system of choice.

As more and more computers are linked together, system security at every level is becoming increasingly more important. We take a closer look at security when we discuss the Linux file system later in this manual.

5.2 Configuring and Compiling the Kernel

To check the version of the kernel that is presently on your system type the command

uname -a

The version is displayed in the format of:

major-version-number.minor-version-number-patch-number

Linux is constantly being modified and it would undoubtedly be a full-time job to try to keep up with the latest modifications. That being said, there may be times when a new patch will allow you to run a new piece of hardware, or when you simply may want to remove unwanted features that are taking up precious memory resources. In these cases you will want to have the ability to build a new kernel.

5.2.1 New Kernel Configuration

The first step in building a new kernel is to make a backup copy of the current kernel. The source code for the current kernel is located in the /usr/src/Linux directory. Copy the files in this directory to a separate backup directory of your choice. If you have not already done so, you will need to load the C compiler package. The command below will accomplish this task:

rpm -i kernel -source-2.0.34-0.6.i386.rpm
rpm -i gcc-2.7.2.3.11.i386.rpm

More than likely you will download the most recent version of Linux from the Internet. After you have completed your backup of the current kernel, place the new compressed version in the /usr/src directory and unpack the kernel using the following command:

tar xzf linux-x.x.x.tar.gz

where linux-x.x.x.tar.gz is the name of the new kernel (e.g. linux-2.2.0.tar.gz). This command will decompress and untar the kernel into the /usr/src/linux directory.

The next step toward getting your new kernel running is to configure the kernel. There are three ways that you can configure a new kernel.

make config a text based non-menu driven interface

make menuconfig a text-based menu driven interface

make xconfig a graphical based menu driven interface using the X Window system

In each case the command of your choice is entered while in the /usr/src/linux directory.

While we are not going to go through each step in the process of configuring a new kernel, there are a couple of key items which we should mention.

Linux supports the modular concept. What this means is that the whole kernel does not have to be loaded into memory at boot time. Rather, some portions of the kernel will

only be loaded when needed. The advantage of this is that computer resources are preserved for more important processes up until the time that a particular process is requested. For example, if you decide to modularize the PPP connectivity, then it will only be loaded by the kernel when you use PPP connectivity.

N.B. If you want the kernel to automatically load modules then make sure that the Kernel daemon support is tagged Yes.

5.2.2 Compiling the New Kernel

The compiling process may take anywhere from 10 minutes to 2 hours, depending on your system. The process is started by issuing the following commands:

make dep;make clean;make zImage

The make dep command takes the configuration information and builds an associated dependency tree. During this stage it is determined as to what will be compiled and what will not be complied. The next command, make clean, deletes all of the old compilation so as to avoid an overlap between the new and the old versions. make zImage actually does full compilation.

If you are making use of modules, then before you can use the new kernel, you must compile the modules using the following command:

make modules

5.2.3 New Kernel Installation

Check the current/boot directory to see which kernels are presently installed. You can usually tell that a file is a kernel by the vmlinuz string at the beginning of the filename. If this is not what is displayed, then check the information in the /etc/lilo.conf file to see what kernels are installed and what their names are.

Next, copy the file /usr/src/linux/arch/i386/boot/zImage to the /boot directory and give it an appropriate name. For example:

cp /usr/src/linux/arch/i386/boot/zImage /boot/vmlinux-2.2.0-latest

Now you can install the modules. Before doing this, you should rename the directory containing the old modules. For example:

cd /lib/modules
mv 2.0.36 2.0.36-old

Change directories to /usr/src and type the command:

make modules_install

This will install the modules into a new directory called /lib/modules/*version of kernel just compiled*.

As a final step, add the new kernel's name to the /etc/linux.conf file. Do not remove the old version's name until you are sure that your newly complied kernel and modules work.

5.2.4 Installing Patches

Rather than replacing the whole kernel, you may decide to apply a patch. If this is the case then download the patch into the /usr/src directory and run the following commands:

cd /usr/src
gzip -cd <patch name> **| patch -p0**

where <patch name> is the name of the patch that you are using (e.g. patch-2.2.0.gz). Once this procedure is complete it is a good idea to look in the /usr/src/linux directory for any files with the file extension **.rej**. This is a flag that indicates that the patch for that file failed. If you have done everything correctly, it must be the patch that is the problem. If the patch is the problem, then keep looking on the Internet for the posted fix.

5.3 Recovering from a Bad Kernel

In the event that your newly installed kernel does not work you have to have some means of getting back to your original configuration, if that is what you want to do.

Reboot the system and while at the LILO prompt, select your old kernel, but attach the parameter **single**. You will undoubtedly see a lot of errors displayed as Linux boots up. These errors are due to mismatched modules in the /lib/modules directory.

Once you have logged in, go to the /lib/modules directory and erase the current module installation. For example, if your new modules are in 2.2.0 then use the command:

rm -rf 2.2.0

Then rename your old modules with their original name. Following the example from above, use the following command:

mv 2.0.36-old 2.0.36

Once you reboot you should gain control over your Linux system.

5.4 File Systems

The Linux file system is both versatile and complex. Perhaps the best way of looking at the system is to realize that Linux uses a single directory structure. Regardless as to how many disks, devices or partitions you have, everything is handled as a part of one large directory. At the very top of the directory system or "tree" is the root directory and the next set of directories branch out from the root directory. This branching continues from one set of directories to the next.

Figure 5.1 Linux File Tree Structure - Example

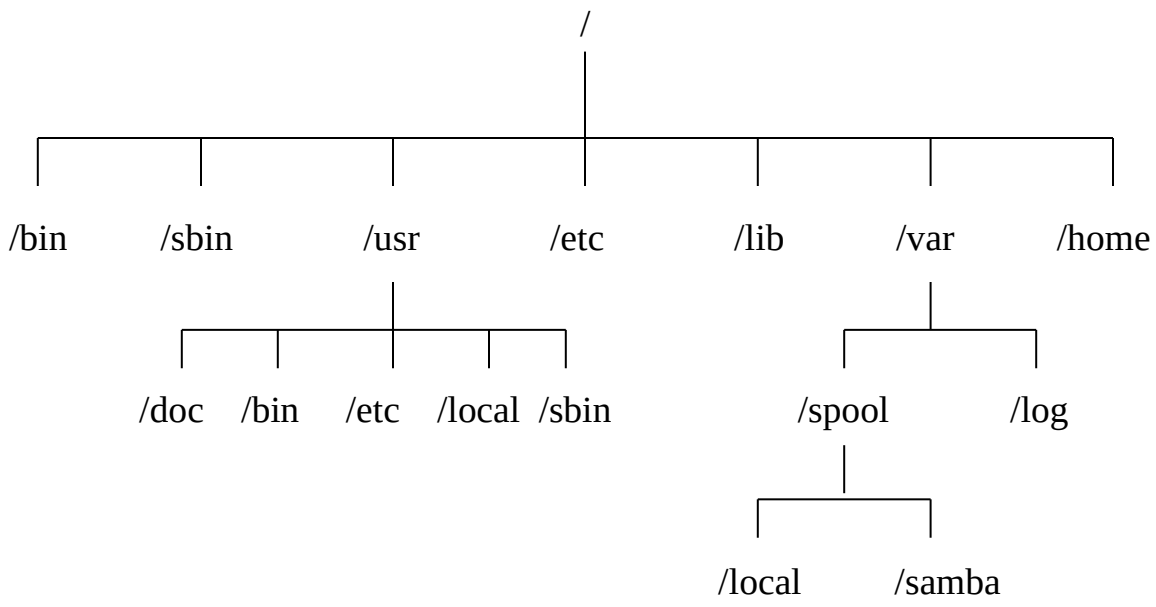

Figure 5.1 shows how the various directories are linked within the Linux file structure. From a user's perspective the directory system is seamless with no need to know the physical location of each directory. For example, the usr directory may be located on a different partition, on a different hard disk or even on a different computer. In each case that portion of the file tree is connected to an overall file tree at a directory known as a mount point. The mount point and all of its attached directories and files is known as a file system. All of this is transparent to the user.

N.B. Linux doesn't care where you mount a file system, just so long as it is mounted in a directory that is located in the root file system. You cannot mount a file system in the root directory.

The syntax for the mount command is:

mount <device name> <mount point>

where the device name is the name of the hard disk, CD-ROM, partition, etc. and the mount point is the name of the directory to which you want to mount the device.

The syntax to unmount a file system is:

umount <device name>

or

umount <mount point>

Either command will work.

N.B. The unmount command is **umount**. This is not a typing mistake. Also note that during the shutdown process Linux will manage any required unmounting.

During the boot process Linux must be told which file systems to mount. This information is setup in the /etc/fstab file. For more information on this important file, please refer to the manual page for fstab.

5.4.1 The /bin Directory

The /bin directory (bin is short for binary or binaries) contains many of the basic Linux programs. For example, this is where the various shells are stored (e.g. bash is located in /bin). Many of the Linux command programs, such as **ls** and **mount** are also located in the /bin directory.

5.4.2 The /sbin Directory

The /sbin directory usually contains files that are necessary for booting the system, system recovery and executables used only by the system administrator (root user). Such files as init and telinit are located in the /sbin directory.

5.4.3 The /usr Directory

The /usr directory contains file shared across the whole system. In a typical LAN, the /usr directory is very large and usually has its own partition with read only permission (more on permissions later in this manual). This particular directory typically only contains other subdirectories. **Table 5.1** provides an incomplete listing of the subdirectories found in the /usr directory:

Table 5.1 Subdirectories of the /usr Directory

X11R6	Directory for the X Window System
bin	Directory contains user-oriented Linux binaries
doc	Directory containing non-manual page documentation
etc	Directory containing site-wide configuration files - usually text files
lib	Directory for libraries
local	Directory used to store local software - these files protected during upgrades
man	Directory for man pages
sbin	Directory of system administration binaries that are run after the boot process
src	Directory containing source code

The Red Hat control panel stores information in the /usr/lib/rhs directory.

5.4.4 The /etc Directory

The /etc directory holds many of the local system's configuration files. Some of the more important files in this directory include fstab (the file containing boot time device mounts), profile (file containing the setup for the system environment), and smb.conf (the samba configuration file).

Red Hat stores a number of files associated with the Red Hat control panel in the /etc/sysconfig directory. Many of the files in this directory are run at boot time.

5.4.5 The /lib Directory

The /lib directory contains those files necessary to execute the system files in the /bin and /sbin directories.

5.4.6 The /var Directory

The /var directory is used to store temporary files, log files, mail files, and print spooler. This directory was designed to work in concert with the /usr directory; the /usr directory containing read only files and the /var directory containing read/write files.

Red Hat stores files used by the RPM (Red Hat Package Manager) in the /var/lib/rpm library.

5.4.7 The /home directory

The /home directory (called "root" if you logged on as the root) is used to store the user's home directories.

5.4.8 File and Directory Access Levels and Access Permissions

Access to UNIX files and directories is controlled by access levels and access permissions.

There are three levels of access: the owner level (also called the user); the owner's group level (called the group); and the rest of the world level (also called other). Within each group one can set the access permissions or simply, which operations the specified level is allowed. The operations are denoted **r** for read, **w** for write, **x** for execute. When one uses the command **ls -l** one sees a list of files and directories that looks like this:

-rwxrwx--x 3 root root 1024 Apr 24 07:34 myfile

The first column displayed is the permission block. The first character in the permission block indicates the file type as outlined in **Table 5.2**.

Table 5.2 Permission Block File Types

-	ordinary file
b	block mode device
c	character mode device
d	directory
l	link

The next set of characters are arranged in groups of three; each access level has its three permissions displayed in the order of read, write, execute with a - used to indicate that a particular access is denied. In our above example, the file type is ordinary, the user (the owner of the file) has read, write, and execution permission, the user's group has read, write, and execution permission and the world has execution permission only (for directories this means that the world, anyone else on the system, has the ability to change into that particular directory).

N.B. The permissions for directories have slightly different meanings than the meanings of permissions for files. The read permission allows for the listing of the directory. The write permission allows the writing of files to the directory. The execute permission allows entry onto the directory (using the cd command).

When you create a file for the first time, the file is automatically given a set of default permissions. These permissions are determined by the particular file creation mask for that owner of the file called umask (the "user's permission mask"). Every person on the system either has a permission mask which is set for them (by the system administrator) or has the system default permission umask. The permission mask for the current user is displayed by entering the **umask** command. The block of three numbers that is returned

is simply an octal representation of the permissions that are displayed in a file's permission block. See **Table 5.3** for a description of each number.

Table 5.3 Mask Block Number Descriptions

0	read and write (and execute for directories) - the highest level of permission
1	read and write (not execute for directories)
2	read (and execute for directories)
3	read
4	write (and execute for directories)
5	write
6	execute
7	no permissions - the lowest level of permissions

Prior to creating a file, the owner can change their umask to automatically set the permissions for the new file. For example, the root user has a default mask of 022 (read, write, and execution permission for the owner of the file and read permission for the user's group and other users on the system). Typing **umask 077** will set the permissions for any newly created files (or directories) to read, write, and execute for the owner of the file and now permissions for anyone else on the system (the permission block would be rwx------). The new umask value will stay in effect until the user changes the umask value or reboots the system. To permanently change the umask of a user, the system administrator would have to make changes to one or more start-up configuration files.

5.4.9 Changing Permissions

Once a file or directory has already been created, the owner of the file or directory can change permissions by using the **chmod** command. The syntax of the chmod command is as follows:

chmod <level of access><change><permissions> <filename>

The level of access that you want this change to apply to is the first value or values that you input (**u** for user, **g** for group, and **o** for other). The second value that is entered is the type of change that is to be made (**-** is used to revoke permissions, **=** is used to explicitly set the permissions or **+** is used to add permissions). The permissions value(s) have already been discussed in detail (**r**, **w**, and **x**). For example, typing:

chmod go=rwx myfile

would change the file "myfile" permissions for the "group" and "other" levels of access to read, write, and execute.

N.B. If you do not specify which access levels you are referring to, then all three levels will be changed.

N.B. To learn about the chmod command's absolute mode, see the appropriate man page.

5.4.10 Owner and Group Changes

Every file and every directory in the Linux file system has both an owner and a group. These are displayed in columns 3 and 4 when the **ls -l** command is used. The owner of the file and the group level access can be changed by using the **chown** and **chgrp** commands respectively. The syntax for the chown is as follows:

chown <new owner name> <filename or directory>

where the new owner name is the name of a valid user (the system will check the /etc/passwd file to insure that the new owner name is valid and that you are the owner of the file).

The syntax for the **chgrp** command is as follows:

chgrp <new group name> <filename or directory>

where the new group name is the name of a valid group (the system will check the /etc/group file to insure that the new group name is valid and that you are a member of the group that currently has ownership of the directory).

N.B. If you know the UID (user identification number) or the GID (group identification number), where appropriate you can use these in place of the new owner name or the new group name.

N.B. Be aware that you can change ownership's, only to realize afterwards that you no longer have access to that particular file or directory.

N.B. Removing the write permission for a file does not prevent a user (owner) from deleting the file. If you want to prevent others from deleting files, then set the directory permission for both the group level and the other level to read and execute only.

5.4.11 More working with Files and Directories (Hands-on Laboratory)

1. Check to see if there is a /mnt/cdrom directory and if not, create a directory called /mnt/cdrom. Mount the CD-ROM device on the /mnt/cdrom directory (make sure that you have a CD in your CD-ROM before trying to mount the device).

2. Go to the appropriate directory and display the list of files found on your CD. If there are directories on the CD, display the files in one or more of these directories.

3. Unmount the CD-ROM when you have completed steps 1 and 2 above.

4. Take some time to see what other directories are located in the /usr directory. Display a list of files in one or more /usr subdirectories. See if you can locate some of the files that have been mentioned in this section.

5. Return to your home directory and display a full listing of the files and directories that are located within your home directory.

6. Display your umask setting. Change this setting to allow you, the owner of any newly created files and directories, to be the only user to have read, write, and execute permission (all other users should have no permission to access your newly created files or directories).

7. From within your home directory, create a directory called temp. Go to your newly created temp directory and create three new files using whichever method you prefer.

8. Check the permission blocks of your newly created files.

9. Following the steps necessary to mount the CD-ROM, mount your floppy disk on the directory /mnt/floppy. Copy the files that you created in Step 7 and check to make sure that the files were copied on to the floppy disk.

10. Using the linuxconf command from a shell window in X Window, create a new group called devproject. Write down the GID on a piece of paper.

11. Change the ownership of the temp directory to devproject.

5.5 Red Hat Package Manager

The Red Hat Package Manager (RPM) is a utility that is provided to users and developers under the terms of the GPL. It gives the user the ability to upgrade or install whole or parts-of-whole software applications and systems using one line commands. Some of RPM's more important features include:

Unlike the tar.gz way of distributing software, RPM allows for the upgrade of individual components so that a complete rebuild is not necessary. It automates the process, making sure to preserve configuration files so that customized settings remain intact.

An RPM package can be easily uninstalled. RPM automatically removes multiple files in different locations, providing a clean software removal.

Once a package is installed, RPM has the ability to verify that the new software is working. The user is notified of any variances and if necessary, reinstallation can proceed without altering configuration files that may contain important custom information

RPM gives the user the powerful querying functionality. The user can quickly find out which file belongs to what package. Each RPM package is stored in a compressed archive format that has attached a binary header containing all types of package specific information.

In addition to the five major modes of operation of the RPM, **Table 5.4** provides a short list of general use RPM commands.

Table 5.4 Listing General Use rpm Commands

rpm --version	Displays the system's current RPM version
rpm --help	Displays RPM help
rpm	Displays a short version of RPM --help
rpm --showrc	Displays a list of variables that can be set in the files $HOME/.rpmrc and /etc/rpmrc
rpm --rebuilddb	Rebuilds the RPM database that keeps track of all the installed packages

5.5.1 RPM Install Mode

The format for the RPM install command is:

rpm -i [options] [package]

The options can include one or more of the options listed in **Table 5.5**.

Table 5.5 Common rpm Install Options

-v	Displays what RPM is doing
-h or **--hash**	Displays 50 hash marks (#) as the package is installed
- -percent	Displays percentages complete as files are extracted from the package
- -test	Goes through the install process without actually installing any files
- -excludedocs	Does not allow the installation of any document files
- -includedocs	Insures that the document files are installed
- -nodeps	Dependency is not checked prior to installation
- -replacefiles	Installed files are replaced with files from the package being installed

Table 5.5 Common rpm Install Options (continued)

- -replacepkgs	Installed packages are replaced with the packages that are being installed
- -oldpackage	Allows a newer version of an installed package to be replaced by an older version
- -force	Forces a package to be installed

Typically, the package name will be of the following format:

name-version-release.arch.rpm

where name is the name of the package, version is the package's version number, release is the release level of the package, arch is short for computer architecture, and rpm is the default extension. An example might be vim-4.6-4.i386.rpm.

If the package was in a directory on the computer's CD-ROM, then change directories to the appropriate CD-ROM directory and then issue the install command using the desired options.

N.B. Options that start with a single - can be blocked together, e.g. -ivh. Options that start with two - - must be listed separately, e.g. - -includedocs - -replacepkgs.

5.5.2 RPM Upgrade Mode

The format for the RPM upgrade command is:

rpm -U [options] [package]

The options can include any of the general or install options previously listed.

The upgrade mode first uninstalls any old version of the package, if it exists, and then installs the new package. This particular mode has the advantage of saving all of the configuration files.

5.5.3 RPM Uninstall Mode

The format for the RPM uninstall command is:

rpm -e [options] [package]

The options for this command are listed in **Table 5.6**. The package name is merely the name of the package (e.g. vim), not the installation package name (e.g. vim-4.6-4-i386.rpm).

Table 5.6 Useful Uninstall Options

- -nodep	Forces RPM to ignore dependencies
- -test	Runs an uninstall test; usually use -vv to display results of test

Many packages install multiple files, potentially in several different directories. The RPM uninstall mode automatically uninstalls all of the files associated with the package that is being removed from the file system.

5.5.4 RPM Query Mode

The format for the RPM query command is:

rpm -q [options] [package]

The options can include one or more of the options listed in **Table 5.7**. As with the uninstall mode, the package name is the name of the package, not the installation package name (if necessary, see **Section 5.5.3** for more clarification).

Table 5.7 RPM Query Options

-a	Display all installed packages
-c	Display all files in a package that are marked as configuration files
-d	Display all files in a package that are marked as document files
-f	Display the package that owns a specified file
-i	Display the complete information for a package
-l	Display all the files in a package
-p	Display the package name
-s	Display the state of files in a package

5.5.5. RPM Verify Mode

The format for the RPM verify command is:

rpm -V [package]

If RPM detects that the package has been installed properly, no output will be displayed. On the other hand, if there is a difference between the RPM database record and the installed package, RPM will display an eight-character string where a period represents a test pass and an appropriate character to indicate a particular test failure. See **Table 5.8** for a list of test failure characters.

Table 5.8 Character/Test Failure Chart

D	Device
G	Group
L	Symlink
M	Mode
S	File Size
T	Mtime
U	User
5	MD5 Sum

Other options that can be added to the verify command include: **f** which will verify a specified file; **a** which verifies all installed packages; and **p** which will verify an installed package against a corresponding not-installed RPM package.

6 Configuring Linux (Hands-on Laboratory)

This section will deal with some of the complexities that are encountered in setting up Linux as a server (i.e. connecting Linux to a local area network and printing) and customizing the look and functionality of your windows environment.

6.1 Setting up User Accounts

Before connecting your Linux computer to your local area network (LAN) let us first add a user account. This is an important first step to the extent that only those clients with accounts will have access to the Linux server.

One of the easiest ways of adding a new account to Linux is to use linuxconf, which is included in the Red Hat distribution. Start the X-windows environment by typing startx. From an x-terminal type in the command linuxconf. If you are not already logged on as having root access, you will be prompted for the root's password (please enter the root's password if prompted). Using the tree menu interface, complete the following steps:

1. Open **Config -- User accounts -- Normal -- User accounts**.

2. Select **Add**.

3. Enter the new account's login ID. Make sure that the login exactly matches the login ID used to login on the Windows workstation.

4. Enter the account's full name.

5. Enter the other optional information as required.

6. Select **Accept**.

7. Enter a six (6) character password. Make sure that the password exactly matches the password used to login on the Windows workstation.

8. Reenter the password to confirm that it is correct and select **Accept**.

To check that the host information is correct, before leaving linuxconf, open **Config -- Networking -- Client Tasks -- Basic host information**. Here is where you should see much of the information that you entered during the Linux installation procedure for networking (**Section 2.2.10**).

6.2 Connecting to the LAN

If you will recall, you installed the samba package during the installation of Red Hat Linux 5.2. The samba package contains two important daemons, smbd, the Samba server, and nmbd, the netbios name server. Both of these daemons are started at boot time. To assure yourself that they are running, at a console prompt, type **samba status**. If they are not running, type **samba start**.

Samba is really an open source implementation of what Microsoft now calls the Common Internet File System (CIFS). One of the most important protocols within the CIFS is the Server Message Block (SMB) protocol. The SMB protocol (smbd) is responsible for print and file services.

As mentioned above, the nmbd daemon is also a part of Samba and is responsible for the NetBIOS (Network Basic Input Output System) nameserving and browsing support. The NetBIOS serves as the connection between the network protocol stacks (in the case of connections between Linux and Windows, TCP/IP) and the operating system. When the operating system requests network resources the NetBIOS translates the request to the appropriate transport stack, which in turns translates the request to the LAN's physical layer.

The Samba configuration file smb.conf is found in the /etc directory. This file has a number of different named sections, with each section performing a different function. While a detailed discussion of each section is beyond the scope of this manual, we will take this opportunity to look at one parameter within the global settings.

If not already in the /etc directory use the **cd** command to go to /etc

Type **pico smb.conf** to use the pico editor to view and edit the smb.conf file

Note that most of the lines have been commented out using either **#** or **;**

In the Global Settings section, the default workgroup name is **WORKGROUP**. If has not already been changed to **WKGRP1** then delete the original entry and type **WKGRP1**.

Press **Ctrl-x** and then select to save the changes made to smb.conf

Type **samba stop**

Type **samba start**, this insures that the smbd reads the changes that have been made to the smb.conf file

6.3 Using a Windows Print Server

Start the X Window environment by typing **startx**. If the control panel is not displayed then display the control panel using the appropriate steps for the window manager that is being used.

Click on the **Printer Configuration** icon to display the Red Hat Linux Print System Manager.

Click on the **Add** pushbutton and select **Lan Manager Printer (SMB)**

Enter the necessary information as requested by the LAN Manager Printer (SMB) Entry window using as a guide **Table 6.1 Sample LAN Manager**.

Table 6.1 Sample LAN Manager Printer Entries

Names	lp
Spool Directory	/var/spool/lpd/lp
File Limit in Kb (0 = no limit)	0
Hostname of Printer Server	Intel200mhz
Printer Name	CANON
User	Administrator
Password	******
Input Filter	*select appropriate printer filter*

N.B. Selecting the appropriate printer filter can be challenging given that there are relatively few printers to choose from. Even when you have selected what you think is the correct printer, you may have to try selecting different color depths in order to get your printer to work. For example, I found that my Canon BJC-4550 does not work when I select a Printer Type: **Canon BJC-600 and BJC-4000** with a Color Depth: **1 Normal B&W printing**, but if I change my Color Depth to**: 8 Floyd-Steinberg B&W printing ...**, it works fine.

6.4 Window Managers and Environments

As mentioned earlier in this manual, X Window does not dictate what your window desktop will look like. While the default window manager is fvwm, there are literally dozens of GUIs (graphical user interfaces) to choose from. **Table 6.2** provides a short list of some of the window managers and environments that are available.

Table 6.2 List of Window Managers and Window Environments

fvwm2	Default virtual window manager for Red Hat
fvwm95	Windows 95 look alike virtual window manager
olwm	Window manager by Openlook
olvwm	Virtual window manager by Openlook
CDE	Common Desktop Environment, a complex GUI environment
KDE	K Desktop Environment, a complex GUI environment

It is somewhat important to draw a distinction between a window manager and a window environment. Perhaps the easiest way of understanding the difference between a window manager and a window environment is to realize that window environments contain, as one of their components, a window manager. In its simplest terms a window manager determines the look and "feel" of the desktop and the window frames. A window environment not only determines the look and feel of the desktop (i.e. a window manager), but also supplies a number of other components, providing the user with a completely functional desktop environment. These other components usually include a file manager, an applications manager and an assortment of tools and utilities.

6.4.1 Fvwm2

As already stated, Fvwm2 (F virtual window manager 2) is the default window manager for Red Hat 5.2. Fvwm2 is popular for a number of reasons, not the least of which is its small memory footprint. Like all virtual window managers, it provides a large virtual desktop in which each desktop is made up of a number of pages. Each page is the size of the physical screen and the user can move to an adjacent page by simply moving the mouse to the edge of the current page. In addition, there are multiple disjoint desktops. In other words, there is more than one desktop available to the user. The user can manage the desktop pages and which desktop is currently active through use of the pager module.

Regardless of which window manger one uses, Red Hat provides a number of enhancements to the window manger.

Click the mouse anywhere on the desktop to display the **Start Menu**.

Click **Programs -- Administration -- Control Panel** to display the Red Hat Control Panel.

Use Window Operations to "stick" the control panel to the screen. See what happens when you move to a new page; when you move to a new desktop.

Click on the **New Shell** menu item to start an xterm session. One can enter console commands from this terminal just like you would if you were using a typical Linux virtual terminal. Try some bash commands and see what happens.

6.4.2 CDE

CDE (Common Desktop Environment) is a commercial window environment that includes all of the functionality of a complete desktop environment (front panel of application and tools, file manager, window manager, application manager, and multiple virtual workspaces).

6.4.3 KDE

KDE (K Desktop Environment) is freely available under the Open Source code concept. It is a fully "integrated desktop environment" which includes a file manager, window manager (KWM), help system, configuration system, and various tools and utilities. Version 1.1.1 was recently released. For more information on KDE you may want to visit www.kde.org.

6.4.4 mwm

The mwm (Motif window manager) is the window manger that is supplied either as a standalone application or as part of Motif, and up until recently, CDE (CDE 2.1.20 is using dtwm). Both Motif and CDE are window environments.

N.B. Motif is actually the name of the most common commercial UNIX window environment.

6.4.5 olwm and olvwm

Sun Microsystems created a window environment called OpenWindows. Sun also created two window managers to choose from for their OpenWindows. The olwm (Open Look window manager) is their standard window manager. It is included with all versions of OpenWindows and is FTP down loadable from Sun. The olvwm (Open Look virtual window manager) is the virtual desktop edition of olwm.