LINUX NEWBIE ADMINISTRATOR GUIDE

ver. 0.193 2002−12−14 by Stan, Peter and Marie Klimas

The latest version of this guide is available at http://sunsite.dk/linux−newbie.

Copyright (c) by Peter and Stan Klimas. Your feedback, comments, corrections, and improvements are appreciated. Send them to linux_nag@canada.com This material may be distributed only subject to the terms and conditions set forth in the Open Publication License, v1.0, 8 or later http://opencontent.org/openpub/ with the modification noted in lnag_licence.html.

Intro. We are relative Linux newbies (with Linux since Summer 1998). We run mostly RedHat and Mandrake −> the solutions might not be directly applicable to other Linux distributions (although most of them probably will be). Hope this helps; we try to be as practical as possible. Of course, we provide no warranty whatsoever. If you spotted a bad error or would like to contribute a part on a topic of your choice, we would like to hear from you.

General description of this Guide. A complete reference for new Linux users who wish to set up and administer their own Linux home computer, workstation and/or their home or small office network. The answers are meant to be simple, with just sufficient detail, and always supported with a readily usable example. The work is still in progress, but we hope the Guide can be helpful already. We welcome your corrections, advice, criticism, links, translations, and CONTRIBUTIONS. Pls note that there are no ad banners on our pages.

Conventions:
<> = single special or function key on the keyboard. For example `<Ctrl>` indicates the "control" key.
italic = name of a file or variable you probably want to substitute with your own.
`fixed width` = commands and filenames.

Part 0: For the Undecided (Linux Benefits)
If you are wondering what the Linux pros and cons are, and whether Linux is for you.

Part 1: Before Linux Installation
What distribution should I use, how to obtain it, Linux hardware requirements, how to partition your hard drive, about dual boot, which packages to install, which graphical user interface (GUI) to install (gnome or kde?), and how to login for the very first time.

Part 2: Linux Resources, Help and Some Links
How to access the Linux documentation (from under MS Windows or Linux), what are Linux help commands, where to find the geek dictionary, + pointers to some Linux newsgroups and websites.

Part 3: Basic Operations FAQ
After you installed Linux, here are answers to some questions that Linux newbie users/administrators may have when trying to perform every−day tasks: what are the file name conventions, how to run a program, shut down my computer, set up the path, add users, remove users, make your passwords and system more secure, work with file permissions, schedule jobs with "at" and cron, change your shell prompt, print symbols in the text mode, use color in the text mode, redirect input/output, write a simple shell script, install a new program ...

Part 4.1: Boot−time issues
Some info on LILO and GRUB boot managers, how do I choose the operating system which boots on default, hints on configuration of the boot loaders, "uninstalling" Linux ...

Part 4.2: Drives
Where are my drives, how to access them, configure user access, get the zip drive recognized, set 32−bit hard drive IO, increase the limit on the number of opened files, add a new hardrive, manage the swap space ...

Part 4.3: X−windows
How to switch between text and graphical consoles, set up my video card, monitor and mouse for the X−server, setup a graphical login prompt, change a default desktop, have multiple sessions of Xwindows running at the same time, use Xwindow remotely, install TrueType fonts from my MS Windows partition to Linux, how to copy−paste under X and in the text mode, how to use VNC.

Part 4.4: Basic Configurations
Real basics on how to configure the printer and soundcard, bits about configuration files, daemons, and device files.

Part 4.5: Networking
Setting up a network, ppp (connection over the phone), remote access to your computer, ftp and html server, e−mail, how my computer can get hacked ...

Part 5: Linux Shortcuts and Commands
Maybe this should have come first. A practical selection of Linux shortcuts and commands in a concise form. Perhaps this is everything that a computer−literate newbie Linuxer really needs. **Highly recommended.**

Part 6: Linux applications (proprietary or not)
Essential and/or famous Linux applications with some hint/comments: word processing, spreadsheet, database, latex. Extensive info on how to set up and use a CD recorder to write data, audio, and mixed mode CDs.

Part 7: Learning with Linux (commands for more esoteric work or programming)
Review of some more advanced or less useful commands/tools to get you started with fancier text processing, encryption, digital signatures (gpg), simple programming plus some info on the Linux console tools that can help you learn about computers. Under development so perhaps not so good: grep, regular expressions, sed, gawk, sort, ascii codes, linux built−in c compiler and tools, perl, python, tcl/tk, "Reverse Polish Notation" (RPN) calculator, scilab, wine ... working on it.

Appendix A: How to upgrade the kernel (by Alesh Mustar)
All you need to know to upgrade the Linux kernel (currently unmaintained)

Licence, Acknowledgments and log of changes.

The master copy of this page is http://sunsite.dk/linux−newbie/ (Denmark, Europe) hosted free by SunSite at Aalborg University. There are some official mirrors listed below.
Mirror: http://slayernetworking.com/newbie (Seattle, Washington, USA) hosted by Joe Morthland (Skull) (new).
Mirror: http://dbstreams.ca/mirrors/linux−newbie/ (Canada) hosted free by Kenan Bektas of DB Streams Inc.
Mirror: http://www.linsup.com/newbie/ (Australia) hosted free by linsup.com.
Only the link to the title page (e.g., http://sunsite.dk/linux−newbie/) is (rather) guaranteed not to change. Links to individual chapters may break because filenames may need to change in the future (as they did in May 2002).

Translations

A Hebrew translation (partial, pdf only) is available locally here (new).
A Portugese translation is available at http://www.onlinux.com.br/dicas/lnag/index.htm (Brasil) and locally here contributed by Ronaldo T Morais <rtoledo@onlinux.com.br>.
A French translation (in progress, Part 1−3 so far) is available at http://www.gerelia.asso.fr/lnag.html (France). Thanks to David <David.Lecat@gerelia.asso.fr> (new).
A Russian translation (new) is avialable at http://linuxbegin.by.ru/lnag_ru/ . An older Russian translation (ver. 0.10) is available here locally or at http://www.college.balabanovo.ru/rider/book/ (Russia).
A Polish translation is available at http://www.wzz.org.pl/~lnag/pl/ (Poland).
A Chinese translation (Big−5) of the "Linux Shortcuts and Commands" (ver. 0.32) is available (locally) here.
A Chinese translation (GB) of 3 parts is avialable (locally) here.

Downloading for Printing and Viewing Off−Line

The following (automatically generated using htmldoc) files are available for downloading:
*The **pdf** version of the Linux Newbie Guide is here (~600 kB, best for printing and off−line reading) and same zipped here, (slightly smaller, ~400 kB).*
*The **postscript** version of the Linux Newbie Guide is here. (big, ~ 1 MB) and zipped here (smaller, ~400 kB).*
*The **postscript** "2−logical−pages−on−1−physical−page" is here and zipped here (environmentally friendly for printing but very small print!).*
The pdf "2−logical−pages−on−1−physical−page" is here and zipped here (environmentally friendly for printing but very small print!).
*The **html zip** version of the Linux Newbie Guide is here (~300 kB, includes all the local html files).*

Table of Contents

Part 0: For the Undecided (Linux Benefits) **1**
0.1 Fundamentally, why Linux? 1
0.2 Is Linux for me? 1
0.3 Linux is difficult for newbies. 2
0.4 What are the benefits of Linux? 2
0.5 What are the differences between Linux and UNIX? 4
0.6 What are the differences between Linux and MS Windows? 4
0.7 I don't believe in free software, etc. 4
0.8 "There ain't no such thing as a free lunch" 5
0.9 I need high security. With commercial software, I can sue them if things go wrong. 5
0.10 I need standards. Big software corporations (Microsoft) provide standards. 5
0.11 I Need MS Windows for Reading Writing MS Word Documents 6
0.12 MS Windows popularity insures that it is "here to stay". 6
0.13 But LINUX may fork into many different systems 6
0.14 Linux is a cult 7
0.15 The total cost of ownership (TCO) of Linux is high 7
0.16 Linux is idealistic "dreaming"; it is business that rules the world nowadays 7
0.17 Linux sux etc. 8

Part 1. Before Linux Installation **9**
1.1 Which Linux distribution should I use? 9
1.2 What are the Linux hardware requirements? 11
1.3 Will my hardware work under Linux? 12
1.4 How do I download Linux? 13
1.5 How do I get a Linux CD? 13
1.6 I have Linux Installation CDs but no install floppy. What do I do? 13
1.7 What do I need to read before installation? 14
1.8 Can I have MS Windows and Linux installed on the same computer? 15
1.9 How do I partition my hard drive? 15
1.10 The MS Windows partition occupies my whole harddrive. Can I shrink/split it without a re−install? 18
1.11 How do I start the installation? 18
1.12 Is the Linux installation difficult? 18
1.13 Which packages should I install? 18
1.14 Which GUI desktop should I install, KDE or GNOME? 19
1.15 I finished the installation. How do I log−in for the very first time? 19
1.16 How do I crash Linux? 19
1.17 Can I use Graphical User Interface (GUI) all the time? 20
1.18 How do I upgrade a Linux distribution? 21

Part 2. Linux Resources, Help and Some Links **22**
2.1 Any Linux reading materials? 22
2.2 Is there a help command? 23
2.3 Any dictionary of terms? 23
2.4 Web Search 24
2.5 Newsgroups 24
2.6 Any Linux Internet links? 25
2.7 Source code−−the ultimate resource 25

Part 3: Basic Operations FAQ **27**
3.1 Basics 27
3.1.1 Filenames 27
3.1.2 What are the different directories for? 29
3.1.3 How do I run a program? 31
3.1.4 How can I change the PATH? 32
3.1.5 How can I shutdown my computer? 32
3.1.6 How do I deal with a hanged program? 34
3.1.7 Command options 35
3.2 Users, passwords, file permissions, and security 36
3.2.1 Home directories, root, adding users 36
3.2.2 About password security 37
3.2.3 I forgot the root password 38

Table of Contents

Part 3: Basic Operations FAQ
3.2.4 I forgot my user password....39
3.2.5 Disabling or removing a user account....39
3.2.6 I have file permission problems. How do file ownership and permissions work?....40
3.2.7 My mp3 player chokes. The sound is kind of interrupted (how to set suid).....42
3.3 Job scheduling with "&", "at", "batch", and cron....43
3.3.1 How do I execute a command in the "background"?....43
3.3.2 How do I execute a command at specified time (using "at" or "batch")?....43
3.3.3 How do I set up cron?....44
3.4 Shell....45
3.4.1 What is a shell and do I want to use a different one?....45
3.4.2 How do I customize my shell prompt?....46
3.4.3 Colour on text terminal....47
3.4.4 How do I print symbols on the console, in a text mode application, and in X?....48
3.4.5 How do I write a simple shell script?....48
3.4.6 Meaning of quotes....49
3.4.7 Input/output redirection....50
3.4.8 Shell special characters (metacharacters)....51
3.5 Package installation and rpm package manager....52
3.5.1 How do I install a program I downloaded from the Internet?....52

Part 4: Linux Newbie Administrator FAQ....55
4.1: Startup Issues (LILO and GRUB)....55
4.1.0 LILO and GRUB....55
4.1.1 Linux cannot detect all my memory....56
4.1.2 LILO displays only LI (or LIL) and hangs....56
4.1.3 How can I change the operating system that LILO boots on default?....57
4.1.4 The LILO prompt stays too short (or too long) on the screen during the bootup....58
4.1.5 Uninstalling Linux....58
4.2: Accessing my drives....59
4.2.1 Where are my drives?....60
4.2.2 How can I access my CDROM?....60
4.2.3 How to mount a floppy, zip drive, DOS/Windows partition, or a network drive?....61
4.2.4 How to mount a remote MS Windows filesystem through Samba?....63
4.2.5 Any quick way to access a file on a DOS/Windows floppy?....63
4.2.6 Mounting works when I am root. Can a normal user mount?....64
4.2.7 Mounting command is too long, how can I simplify it with an alias?....65
4.2.8 Can I mount automatically?....65
4.2.9 How do I get my parallel−port (external) Zip drive recognized?....66
4.2.10 Can I set 32−bit hard drive I/O?....66
4.2.11 I reached the limit on the number of opened files (error message)....67
4.2.12. I attached a new hard drive. What do I do to start using it?....68
4.2.13 Swap space....70
4.3: Working with X−windows....71
4.3.1 How to switch between text and graphical consoles?....71
4.3.2 How do I setup video card, monitor and mouse for the X−server?....72
4.3.3 Can I have a GUI login prompt?....72
4.3.4 How do I install kde (e.g., on RedHat 5.2)?....73
4.3.5 How can I change my default desktop to KDE (or Gnome or yet another)....74
4.3.6 Can I have multiple sessions of X running at the same time?....75
4.3.7 Can my sister have second GUI login prompt so she does not have to kill my X−session to start hers?....75
4.3.8 How to X−window remotely?....76
4.3.9 How do I install TrueType fonts from my MS Windows partition?....77
4.3.10 How do I copy−paste?....78
4.3.11 How do I Display and Control a Remote Desktop using VNC....78
4.4: Basic Configurations....80
4.4.1 How to setup my soundcard?....80
4.4.2 How do I setup my printer?....81
4.4.3 Word Perfect 8 does not have a driver for my printer....82
4.4 4 Where are the setup and configuration files?....82
4.4.5 What are all the device files?....83

Table of Contents

Part 4: Linux Newbie Administrator FAQ
Some Linux daemons....84
4.5: Networking....85
4.5.1 Would it be worth it to set up my home network?....86
4.5.2 How to set up my home network?....86
4.5.3 I have problems configuring my ppp dial out....88
4.5.4 How to browse the net from my networked computer without a modem?....90
4.5.5 How to use Samba?....91
4.5.6 Sendmail....92
4.5.7 Simple web server (running Apache)....93
4.5.8 Simple ftp server....94
4.5.9 How can one access my computer from the outside world when I am on the net using phone connection?....95
4.5.10 Can my home computer get hacked?....95

Part 5: Linux Shortcuts and Commands....99
5.1 Linux essential keyboard shortcuts and sanity commands....100
5.2 Help commands....102
5.3 System info....103
5.4 Basic operations....106
5.5 File management....106
5.6 Viewing and editing files....107
5.7 Finding files....108
5.8 Basics of X−windows....109
5.9 Network apps....109
5.10 File (de)compression....111
5.11 Process control....112
5.12 Some administration commands....113
5.13 Hard Drive/Floppy Disk Utilities....117
5.14 Management of user accounts and files permissions....118
5.15 Program installation....120
5.16 Accessing drives/partitions....121
5.17 Network administration tools....122
5.18 Music−related commands....123
5.19 Graphics−related commands....124
5.20 Small games....126

Part 6: Some Essential Linux Applications....127
6.1 Word processing....127
6.1.1 OpenOffice.org /StarOffice Suite....127
6.1.2 abiword....130
6.1.3 kword....130
6.1.4 lyx and latex....131
6.1.5 WordNet (dictionary / thesaurus /synonym / antonym finder)....133
6.2 Spreadsheet....133
6.2.1 gnumeric....133
6.2.2 kspread....133
6.3 Databases....134
6.4 CAD....134
6.5 Web browsers: Mozilla, Konqueror, Galeon, and Lynx....134
6.6 Writing CD−Rs: cdrecord and cdparanoia....134
6.7 Automating creation of graphs with gnuplot....139

Part 7: Learning with Linux....141
7.1 Linux Advanced Text Processing Tools....141
7.2 Simple Programming under Linux....150
7.3 Math Tools....158
7.4 Miscellaneous....160
How do I run an MS Windows Application (using "wine")?....160
Can I have a RAID if my computer has two or more IDE (or other) harddrives?....160
Network traffic shaping using shapecfg....161

Table of Contents

Licence, Acknowledgments, etc. **162**
LNAG LICENCE 162
Acknowledgments 162
Other Matters 163

Part 0: For the Undecided (Linux Benefits)

LINUX NEWBIE ADMINISTRATOR GUIDE

ver. 0.193 2002−12−14 by Stan, Peter and Marie Klimas

The latest version of this guide is available at http://sunsite.dk/linux−newbie.

Copyright (c) by Peter and Stan Klimas. Your feedback, comments, corrections, and improvements are appreciated. Send them to linux_nag@canada.com This material may be distributed only subject to the terms and conditions set forth in the Open Publication License, v1.0, 8 or later http://opencontent.org/openpub/ with the modification noted in lnag_licence.html.

Contents of this section:

0.1 Fundamentally, why Linux?
0.2 Is Linux for me?
0.3 Linux is difficult for newbies
0.4 What are the benefits of Linux?
0.5 What are the differences between Linux and UNIX?
0.6 What are the differences between Linux and MS Windows?
0.7 I don't believe in free software, etc.
0.8 "There ain't no such thing as a free lunch"
0.9 I need high security. With commercial software, I can sue them if things go wrong.
0.10 I need standards. Big software corporations (Microsoft) provide standards
0.11 I Need MS Windows for Reading Writing MS Word Documents
0.12 MS Windows popularity insures that it is "here to stay"
0.13 But LINUX may fork into many different systems ...
0.14 Linux is a cult
0.15 The total cost of ownership (TCO) of Linux is high
0.16 Linux is idealistic "dreaming"; it is business that rules the world nowadays
0.17 Linux sux etc.

0.1 Fundamentally, why Linux?

If you truly enjoy working with computers, Linux is the operating system of your dreams. It is more fun than any other computer operating system around. However, the reason why Linux is truly revolutionary is that it is Open Software. Our science and technology works owing to the free availability of information and peer review. Would you fly a plane that was based on secret "science" and an unreviewed design, a plane at the internals of which nobody but the manufacturer could look? Then why would you trust a computer program containing secret parts and algorithms? Open−source Linux is ideally suited for a mission−critical application−−its security and power are based on robust solutions which anyone can view, criticize, or improve on. It is the implementation of the scientific method in computing.

The making of horseshoes, good glass, or measuring time were once closely guarded trade secrets. Science and technology exploded 500 years ago thanks to the sharing of knowledge by the means of printing. In the early days of printing, many of those who dared to share were assassinated for revealing "trade secrets." Linux is for the computer age what Gutenberg was for writing. Hopefully there will be no assassinations this time :−) . Linux does clash with those who claim the "ownership" of information, trying to push time back 500 years.

0.2 Is Linux for me?

Only you can answer this question. Linux is a mature, powerful, secure and extremely versatile UNIX−like operating system. The power and versatility come with a price−−you may need to be computer−literate in order to set−up and maintain Linux. Linux is relatively easy to use once the operating system and applications are set up properly. So, your mother will also be able to use Linux, if you set up an easy graphical account for her and put the proper icons/menus on her GUI desktop. Linux is secure, so your mother will not be able to damage the system no matter how hard she tries−−unless it's with a hammer :−) .

Linux is quite different from MS Windows, so do not expect that if you can get around MS Windows, Linux will be straightforward for you. You may need to learn. On the other hand, if you come from UNIX, Linux will be easy for you. If you don't know much about computers or you don't enjoy them, chances are Linux administration is not for you. If you don't know your hardware, Linux installation may be a challenge.

0.3 Linux is difficult for newbies.

This may be true. But the real question is: do you really want to learn it? None of the authors has a computer science background, yet we use Linux everyday and we love it.

0.4 What are the benefits of Linux?

Linux can give you:

o A modern, very stable, multi−user, multitasking environment on your inexpensive PC hardware, at no (or almost no) monetary cost for the software. Linux is a rich and powerful platform−−don't think of it as a "poor people" operating system. Out−of−box Linux has as much capability as MS Windows NT with $5000 in software add−ons, is more stable, and requires less powerful hardware for comparable tasks.

o Standard platform. Linux is VERY standard−−it is essentially a POSIX compliant UNIX. (Yes, Linux is a best−of−the−breed UNIX. The word "UNIX" is not used in conjunction with Linux because "UNIX" is a registered trademark.) Linux includes all the UNIX standard tools and utilities.

o Unsurpassed computing power, portability, and flexibility. A Linux cluster recently (April 1999) beat a Cray supercomputer in a standard benchmark. Linux is most popular on Intel−based PCs (price of the hardware), but it runs very well on numerous other hardware platforms, from toy−like to mainframes. One distribution (Debian) expresses the idea like this: "Linux, The Universal Operating System." Linux can be customized to perform almost any computing task.

o Advanced graphical user interface. Linux uses a standard, network−transparent X−windowing system with a "window manager" (typically KDE or GNOME).

o Dozens of excellent, free, general−interest desktop applications. This include a range of web browsers, email programs, word processors, spreadsheets, bitmap and vector graphics programs, file managers, audio players, CD writers, some games, etc.

o Thousands of free applets, tools, and smaller programs. "Small is beautiful" goes well with Linux philosophy. The small Linux tools and applets often work in tandem to perform more complex tasks.

o Hundreds of specialized applications built by researchers around the world (astronomy, information technology, chemistry, physics, engineering, linguistics, biology, ...). In many fields, Linux seems like "the only" operating system in existence (try to find out what your friend astronomer runs on her computer). The software in this category is typically not very easy to use, but if you want the power, it is the best software that humanity has in these areas. Doubtful? Have a look at:
http://SAL.KachinaTech.COM/Z/2/index.shtml for examples.

o Scores of top−of−the line commercial programs including all the big databases (e.g., Oracle, Sybase, but no Microsoft's). Many (most?) of these are offered free for developers and for personal use.

o A truly great learning platform. If you are a parent, you should be really glad your daughter/son does Linux−−s/he will surely learn something of lasting value. If you are a teacher, you should consider the installation of Linux at your school. "It is indeed a strange world when educators need to be convinced that sharing information, as opposed to concealing information, is a good thing" (http://edge−op.org/grouch/schools.html). You select Linux if you care to provide education, not training. The better the university, the greater the chance their computer department uses Linux in teaching. For example, under Linux, you can immediately begin modifying and compiling for yourself a spreadsheet application which is in every bit as advanced and capable as MS Excel. Linux puts you right on the cutting edge (in technology, project management, QA, methodology of science). Many teachers won't use Linux in schools because they are lacking in computer education themselves (at least that's what I see).

o Excellent networking capability built into your operating system. You think you don't need a network? Once you try home networking, you will never be able to live without it! How about connecting the two or more computers that you have at home and sharing your hard drives, CDROM(s), sound card(s), modem, printer(s), etc.? How about browsing the net on two or more machines at the same time using a single Internet connection? How about playing a game with your son over your home network? Even your old 386 with Win3.11 may become useful again when connected to your Linux Pentium server and when it is able to use your network resources. All necessary networking software comes with standard Linux, free, just setup is required. And it is not second−rate shareware−−it is exactly the same software that runs most of the Internet (the Apache software runs more than 50% of all Internet web servers and Sendmail touches some 70% of all e−mail). The pleasure of home networking is something I was able to discover only owing to Linux.

o Connectivity to Microsoft, Novel, and Apple proprietary networking. Reading/writing to your DOS/MS Windows and other disk formats. This includes "transparent" use of data stored on the MS Windows partition of your hard drive(s).

o State−of−art development platform with many best−of−the−kind programming languages and tools coming free with the operating system. Access to all the operating system source codes, should you require it, is also free. The "C" compiler that comes standard with Linux can compile code for more platforms than (probably) any other compiler on earth. Perl, Python, Guile, Tcl, Ruby, powerful "shell" scripting, and even an assembler also come as standard with Linux.

o Freedom from viruses, "backdoors" to your computer, software manufacturer "features," invasion of privacy, forced upgrades, proprietary file formats, licensing and marketing schemes, product registration, high software prices, and pirating. How is this? Linux has no viruses because it is too secure an operating system for the viruses to spread with any degree of efficiency. The rest follows from the open−source and non−commercial nature of Linux: Linux evolved itself by "bazar−like" mechanisms to encapsulate the best computing practices, code legibility and correctness, security, flexibility, usefulness, coolness, performance.

o The operating platform that is guaranteed "here−to−stay." Since Linux is not owned, it cannot possibly be put out of business. The Linux General Public License (GPL) insures that development/maintanance will be provided as long as there are Linux users. There is a great number of highly−educated Linux users and tens of thousands of actively developed projects.

o A platform which will technically develop at a rapid pace. This is insured by the modern, open−software development model which Linux implements: "build−on−the−back−of−the−previous−developer" and "peer−review−your−code" (as opposed to the anachronistic closed−software model: "always−start−from−scratch" and "nobody−will−see−my−code"). Even if the current "Linux hype" died out, Linux will develop as it did before the media hype started. Open source development does have its peculiarities: the development appears rather slow (vertically) but it proceeds on a very wide front, dangerous security bugs are fixed almost upon discovery, there are typically several alternatives for a program of similar functionality. Linux depth cannot be overestimated.

If you wanted to learn first−hand about the General Public License, check these famous GNU documents:

http://www.gnu.org/copyleft/gpl.html
http://www.gnu.org/gnu/linux−and−gnu.html
http://www.gnu.org/philosophy/categories.html#TheGNUsystem

In a nutshell, the GNU General Public Licence (GPL) allows anybody to:

− use the software at no charge, without any limitations,
− copy, and distribute or sell unmodified copies of the software in the source or binary form,
− modify, and distribute or sell a modified version of the software as long as the source code is included and licenced under the GPL,
− sell support for the software.

What this license *does not* allow to do is to modify the software and then distrubute a binary−only version of the software (without the source code). Speaking plainly, the GPL licence just forbids stealing somebody else's software for incorporation into a closed, commercial−only product. However, you may incorporate GPL software in a propriatory computer program if you obtain a permission from the author. Excluded from the use of GPL are persons who have been found to violate GPL.

The license under which Linux is distributed is probably the most important part of it. It is designed to perpetuate the freedom of information. Other important open−source projects include science and law (hardly a joke). The Linux method is really nothing new−−it is simply the application of the scientific method to software: you get information free, you add your ideas and make your living, and finally, you leave it free. However, some big corporations and their lawyers seem to be trying hard to change this, to push us back in time, to the dark ages, when information was kept "proprietary." Hence, you see in newspapers some famous Linux−connected persons involved in all kinds of struggles.

> To get a flavour for the value of Linux, here are some prices for commercial software as listed at www.amazon.com. All prices are in $USA, as listed on 2001−02−03, with discounts. Roughly equivalent Linux software is included on almost any Linux CD (but with no restrictions on the number of clients). In addition, the hardware for Linux is MUCH cheaper, since Linux can run all services on a single server:
>
> Microsoft Windows 2000 Server (5−client)−−$848.99; Microsoft Exchange 2000 Server (5−client)−−$1,279.99; Microsoft Outlook 2000 (1−client)−−$94.99; Systems Management Server 2.0 (10−Cals)−−$994.99; Proxy Server 2.0−−$886.99; Microsoft SQL Server 2000 Standard Edition (5−client)−−$1,229.99; Microsoft SQL Server 2000 Standard Edition (1−user License)−−$4,443.99; Microsoft BackOffice Small Business Server 4.5 NT (Add−On 5−CAL)−−$264.99; Windows NT Server Prod Upgrade From BackOffice SBS Small Bus Server (25−client)−−$558.99; Microsoft Windows 2000 Advanced Server Upgrade (25−client)−−$3,121.99; Microsoft FrontPage 2000−−$129.99; Microsoft Internet Security and Acceleration Server −−$664.99; Site Server Commerce 3.0 (25−client)−−$4,092.99; Visual C++ 6.0 Professional Edition with Plus Pack−−$525.99; Microsoft Visual Basic Enterprise 6.0 with Plus Pack−−$1,128.99; Microsoft Visual Sourcesafe 6.0 CD−−$469.99; Microsoft Office 2000 Standard (1−client)−−$384.99; Adobe Photoshop 6.0−−$551.99; Microsoft Plus Game Pack−−$19.99.

The word "free" has two quite different meanings in the English language, and it sometimes leads to misconceptions about the free nature of Linux. These two meanings follow the Latin adjective "liber" and the adverb "gratis," and they are often illustrated with the phrases "free speech" and "free (of charge) beer." Most Linux software is free in both senses, but it is only the first sense which is essential to Linux.

0.5 What are the differences between Linux and UNIX?

Command−line−wise, almost none, although this has been changing (for better or worse). Linux has a much larger market appeal and following than any commercial UNIX. GUI−wise there are also no major differences−−Linux, as most other UNICES, uses an X−Windowing system.
The major differences:

− Linux is free, while many UNICES (this is supposed to be plural of UNIX), cost A LOT. The same for applications−−many good applications are available on Linux free. Even the same commercial application (if you wanted to buy one) typically costs much more for a commercial UNIX than for Linux.
− Linux runs on many hardware platforms, the commodity Intel−x86/IBM−spec personal computers being the most prominent. A typical UNIX is proprietary−hardware−bonded (and this hardware tends to be much more expensive than a typical PC clone).
− With Linux, you are in charge of your computer, whereas on most UNICES you are typically confined to be an "l−user" (some administrators pronounce it "loser").
− Linux feels very much like DOS/Win in the late 80s/90s, but is much sturdier and much richer, while a typical UNIX account feels like a mainframe from the 60s/70s.
− Some UNICES may be more mature in certain areas (for example, security, some engineering applications, better support of cutting−edge hardware). Linux is more for the average Joe who wants to run his own server or engineering workstation.

0.6 What are the differences between Linux and MS Windows?

Mouse−click−wise, almost none, once Linux is properly installed. Linux installation can be a challenge though, whereas MS Windows comes pre−installed with your computer.

The major differences:

− Linux is free, while MS Windows costs money. Same for applications.
− Linux file formats are free, so you can access them in a variety of ways. On MS Windows, the common practice it to make you lock your own data in secret formats that can only be accessed with tools leased to you at the vendor's price. How corrupt (or incompetent?) must the politicians who lock our public records into these formats be! "What we will get with Microsoft is a three−year lease on a health record we need to keep for 100 years" [http://news.bbc.co.uk/hi/english/health/newsid_1694000/1694372.stm].
− With Linux, you are unlikely to violate any licence agreement−−all the software is happily yours. With MS Windows you likely already violate all kinds of licenses and you could be pronounced a computer pirate if only a smart lawyer was after you (don't worry, most likely none is after you).
− MS Windows tries to be the "lowest−common−denominator" operating system (for better or worse), whereas Linux is built for more sophisticated, feature−hungry computer users (for better or worse).
− MS Windows is based on DOS, Linux is based on UNIX. MS Windows Graphical User Interface (GUI) is based on Microsoft−own marketing−driven specifications. Linux GUI is based on industry−standard network−transparent X−Windows.
− Linux beats Windows hands down on network features, as a development platform, in data processing capabilities, and as a scientific workstation. MS Windows desktop has a more polished appearance, smoother general business applications, and many more games for kids (these are not better games though−−Linux games tend to be more sophisticated).
− Linux is more feature−rich than you could imagine. Heard on the Internet: "Two big products came from the University of California: UNIX and LSD. And I don't think it's a coincidence."

0.7 I don't believe in free software, etc.

And do you believe in the Internet? The Internet and Linux share underlying ideas and have common roots. Do you remember the disbelief about the Internet a few years ago, the endless, seemingly unbeatable arguments that free Internet cannot exist? "Who pays for that, anyway?"

The reality is simple. Cooperation and good will can benefit many at the same time: your gain is not my loss. The Internet works fine and is expanding at a rapid pace. So does Linux.

Here is the opinion of an IBM executive: "The reason we are so excited about Linux is we believe Linux can do for applications what the Internet did for networks" (http://linuxtoday.com/news_story.php3?ltsn=2000−08−17−001−04−PS−EL). IBM just (May 2002) spent 1 billion dollars making Linux run on all their hardware platforms (mainframes, workstations, PCs, laptops).

0.8 "There ain't no such thing as a free lunch"

"The economic paradigm which makes this true depends upon scarcity of resources. Software resources are only scarce because we all keep software proprietary and secret. But not Linux! When I give you my software, it may create an opportunity cost for me, but I get to keep it even after I've given it to you. It is a free lunch only rivaled in history by the loaves and the fishes." (Brett Bazant <bbazant@shaw.wave.ca> (http://linuxtoday.com/cgi−bin/showtb.pl?tbsn=12450&sn=5418).

0.9 I need high security. With commercial software, I can sue them if things go wrong.

Don't count on suing. Things go wrong on many MS Windows NT machines every day, and there are no damages awarded by courts. Read your MS Windows license agreement to find out that there is no guarantee whatsoever that ANYTHING will work. Trying to sue would be a waste of your money.

Linux also provides no guarantees, although it is far more secure than any version of MS Windows. If you are really security−sensitive , you can use high−security tools built by companies that rely on the availability of the source code to design and test their security features (e.g., Kryptokom in Germany provides high security firewalls). The "security in obscurity" implemented in MS Windows has repeatedly been demonstrated to be a naive approach.

"Risk aversion is what dictates you use Linux and other open products, rather than NT. The risks with NT are entirely out of your control, and there is nobody you could sue if anything goes wrong. Why people still believe the myth that Windows in any form offers any bit of accountability "more" than Linux remains a complete riddle to me." (David Kastrup, Research Engineer, Bochum, Germany, "Internet Week," http://www.techweb.com/se/directlink.cgi?INW19990329S0050).

0.10 I need standards. Big software corporations (Microsoft) provide standards.

Perhaps that's what people would expect from large corporations, but the reality is rather different. Once, big companies loved inventing nuts that could be undone only by their own service shops. Did these nuts become standard? Hardly. They didn't because there was no public benefit involved, and they couldn't because they were patented. Luckily, now we have open and free standards for nuts. A "proprietary standard" is such a ridiculous oxymoron that it is hard to believe that educated people can believe in it. (Currently, marketing types use the term "de facto standard" or "industry standard" to cover up the ugliness of the lack of standards.)

An example from the computer field. The "standard" MS Word file format has changed numerous times over the recent years. This keeps happening probably for a good business reason: as soon as other companies "reverse−engineer" the current Word format, Microsoft changes it. There are even sub−formats (an MS "fast−save" anybody?). It is also completely closed−−Microsoft does not publish any specifications. How can the user benefit from this in a longer term? What is the Microsoft guarantee that MS Word 6.0 file format will still be legible in 2020?

"... Microsoft's standards are both proprietary and arbitrary− the stealth incompatibility of Office 97 file formats with older versions of Office or the subversion of Open standards like XML with proprietary extensions that require Internet Explorer 5, MS Active server and so on, are sober reminders of what the company does to a market." (Xavier Basora, http://www.osopinion.com/Opinions/XavierBasora/XavierBasora47.html).

"... Microsoft's monopoly doesn't guarantee that your current MS Office will work with any previous or future MS Office. This is in spite of any number of Microsoft apologists arguing that the benefit of Microsoft's monopoly has been a standard for productivity applications." (Wesley Parish, http://www.osopinion.com/Opinions/WesleyParis/WesleyParish10.html).

To add to the confusion, companies typically do not "standardize" on file formats but on the applications that are supposed to produce them. It is like standardizing on a manufacturer of nuts instead of on nuts. How is this supposed to work if the file manufacturer keeps changing the specification to drive their sales?

"We need standardized, open file formats so that users can exchange documents between platforms. The actual word processing software used to generate these documents shouldn't even be an issue." (Ted Clark, http://linuxtoday.com/news_story.php3?ltsn=2000−09−29−004−06−OP−MR−0010).

There are a few text/document oriented file formats that are quite definitely more standard than MS Word file format: ASCII, XML (with non−propriatory stylesheets), HTML, SGML, LaTeX, TEX, PostScript, pdf, dvi ... and all of them have excellent support under Linux. The MS Word file format can also be read/written very well under Linux by OpenOffice (and a number of other applications) to cover your current needs. Advanced, "universal," open−source document formats (XML−based) is being worked on by an independent organization. The story is similar like with other proprietary computing "standards" (*.giff vs. *.png anyone?).

Linux, by its very nature, is based on true, published and free standards because "open source" makes the full specifications available to everybody (competitors or not). I believe that the urge for open standards is the very driving force behind Linux. Some people feel that they cannot afford to trust their algorithms and data to a commercial entity, let alone one that repeatedly demonstrated it is untrustworthy.

Have a look at a draft of this Argentinean law for a taste of the future. It sounds like the Argentineans may be the first to decide that their public records cannot be held hostage by a commercial entity: (source: http://slashdot.org/articles/01/04/28/010216.shtml): "... Public National Organizations mentioned in article 1 of this law, will not be allowed to use programs that store data in non−public format ...". Several other counteries are contempleting or enacting legistlations requiring storage of data in public file formats.

There is a strong perception in the Linux community that there is a serious problem with the computing "standards" championed by large software vendors. This includes their standards for our "static" data , as well as the knowledge embedded in our computer codes. Can we afford to have somebody decide for us when, how, and at what cost we can access our work? This problem is ignored and even aggravated by people who are paid to take care of it. Linux is a grass−root answer.

0.11 I Need MS Windows for Reading Writing MS Word Documents

In a large corporate environement, you may have little choice−−they locked themselves by cheerful productions of non−portable forms, templates, visual basic−driven web pages and other "tools".

In a smaller environment, you can use OpenOffice.org suit (OO) that runs on Linux, Windows, Mac, Solaris, with full file−level compatiblity. It can be downloaded and installed for free (no restrictions whatsoever) so nobody should really complain about the file format (some control freaks still will). Just to make sure, it can import and export MS Word and Excel documents of reasonable complexity very well. However, its native file format is fundamentally much better (and non−propriatory). Feature−by−feature, it can do almost anything MS Office can, plus some extras. Depending on whom you ask, the ease of use veries between "50% more difficult" to "20% easier" (measured on experienced MS Office users). Very complex documents are best transferes as pdf, and OO can make them on the fly.

So, probably you do not need MS Office any more. Download your OO for MS Windows and Linux at: http://www.openoffice.org/

0.12 MS Windows popularity insures that it is "here to stay".

This is likely true. Nintendo is probably also "here to stay." However, I like computing; therefore, I choose a computer with a powerful operating system, not a lowest−common−denominator piece designed for "everybody."

Linux is quite positively here−to−stay because of its open−source nature (Linux cannot possibly be put out−of−business). It is a standard selected for countless projects that are not going to go away, and some of them are quite "mission−critical." Try the International Space Station, for which Linux is the operating system (http://www2.linuxjournal.com/lj−issues/issue59/3024.html).

Plus, never underestimate the strength of the Linux community. Linux is "here to stay" at least for the computer avant−garde. Many Linuxers do not even want Linux to become very popular because they fear it could "dumb down" the elite Linux platform.

0.13 But LINUX may fork into many different systems ...

This is a typical argument of the type spread by those specializing in the marketing tactics known as "fear, uncertainty and doubt" (FUD) [about the competing product].

"Forking" in this context means "branching a computer program," so as to create parallel "subversions" of the program, and consequently fragment Linux.

There is very little (if any) evidence of harmful forking of any software included with a typical Linux distribution. Where forking did occur, it has always turned beneficial. Quite possibly, this is because although there are no artificial barriers to fork software under Linux, there are also no artificial barriers to merge the best pieces back.

The theoretical background on how forking software can be good for its development might have been actually given quite some time ago by the German philosopher Georg Wilhelm Friedrich Hegel (1770−1831), with his concept of dialectic development. E.g., in "Phenomenology of Spirit", Hegel concludes: "... the schism incipient in a party, which seems a misfortune, expresses its fortune rather."

0.14 Linux is a cult

The Linux community has repeatedly been labeled "religious zealots" by journalists whose well−established computer magazines received massive feedback after they had published highly unfair articles on Linux. So yes, the Linux community is numerous, literate, and willing to express its opinions. And many computer journalists/magazines know that Linux means less money for them (users pay less for their computing and the associated advertisement, while expecting more). Does this explain the "zealots"?

Face it, you salespeople pretending to be journalists. There is hardly any integrity left in the computing press. How many words on Linux did your PC Magazine (or whatever) publish by 1999−01−01? Wasn't Linux at least an interesting technology by that time? It surely was, yet you conspired to keep your readership in the dark, selling your journalistic integrity for money. And now, after Linux has surfaced in the mainstream (non−computer) media, you keep writing misleading articles about it saying "yah, but it will/cannot/may ...." whatever (trying the "fear, uncertainty and doubt" tactics to kill it). And adding "Microsoft is already ...", continuing to write about the vaporware and the future paradise in the face of the increasingly stealthy, unstable, pricey, architecturally unsound computer platform, whose greatest achievement has been exhorting unheard−of−before money by denying inter−operatibility, and killing any existing or proposed standard (by "embracing" and then proprietary−extending it). Whom do you serve? Surely not your readers.

I worded it as strongly as I could. Am I a zealot? Or am I just trying to voice my disapproval for the self−serving actions of the computer "powers−that−be"?

You think "self−serving" is ok in business? How pathetic must your business be! I always thought that business was a social contract in which we exchange good values, for a mutual benefit. As I read history, societies use to hang / guillotine / electrocute those members who really persisted in their self−serving business. Well, times have changed. A bit for the better, a bit for the worse :)))

0.15 The total cost of ownership (TCO) of Linux is high

Nobody really knows how to calculate the "total cost of ownership" of a general piece of software. So a good lawyer + accountant can prove whatever point they are paid to make, and they regularly do.

Let me try a simple estimate of how much the average total cost of the ownership of MS Windows is. Let's add the fortunes accumulated by all the MS Windows software makers. Add all the salaries of all generic Windows programmers, consultants, support and training personnel, IT management, etc. Now, add the losses customers must surely have suffered while the software corporations were presenting them with "features" so as to achieve their current monopolistic status. Divide this figure by the number of years (whatever timeframe you selected), and the number of MS Windows users (only in the countries in which software is normally paid for). Here is the TCO of MS Windows. However you count it, it will be many thousands of good US dollars per average joe per year. You didn't pay that much money? Well, you must have, it has just been hidden from you. Yes, developed countries waste billions every year on software.

How much did Linux cost? Hardly anything. The number of users is much lower, too, but you will be hard pressed to come up with $10 per user per year.

Yet, in my opinion, the total cost is not what matters the most. What value did I receive for my money? You would have to calculate the total value of ownership (TVO?), then subtract from it the total cost of ownership (TCO) to obtain the "net benefit of the ownership."

I guess accountants only talk about the TCO for software "necessary for doing business," and thus skip the issue of value and benefit. There is no value in the normal commercial software, it is just the necessity for doing business these days. Well, Linux satisfies my computing necessities at zero monetary cost, and the personal pleasure and learning value is just great.

0.16 Linux is idealistic "dreaming"; it is business that rules the world nowadays

Think of Linux as a consortium. Businesses/individuals get together to address a common computing need or problem. They may chip in labour or money, hire a technical leader, or otherwise cooperate to make Linux address their requirements. The solution is totally theirs for keeps, and it does not have to cost a lot−−they can re−use the pre−existing Linux software pieces. They may cooperate to overcome the advantage that a big "industry leader" may have and use against their interests.

Linux is the end−product of activities of many such loose "consortiums" who "scratch their needs." So Linux is a business, but it is not necessarily about selling software−−it is about access to reasonably−priced software that matches your need, solves your problem, sells your hardware or service, and which is totally yours (the licence never expires, and you will never be cut off from the source code).

0.17 Linux sux etc.

Then do not use Linux. You are not doing anybody a favor by using Linux. GNU/Linux is free and powerful software, but only for those who like or need it. There are alternative operating systems for you to choose from and they may better match your requirements. Although most Linuxers enjoy the growth and welcome new users, some are not very happy about it because "the crowd and commerce can spoil the hackers' paradise we created." Therefore, you really aren't doing anybody a favour by using Linux.

In this context, it may be worthwhile to briefly summarize Linux strengths and weaknesses: Linux is owned by its fans (your piece of ownership comes free with your free subscription to the fan club), definitely very powerful and feature−rich, highly configurable, as flexible as you want it to be (comes with complexity), low on the cost of hardware, comes with any networking bell−and−whistle known to man, requires a computer literate administrator, some essential desktop applications are still behind commercial offerings on other platforms (e.g., spreadsheet and word processing), a number of excellent end−user applications come "standard" and free with the operating system, the graphical user interface is very nice but still not as polished as Apple or MS offerings, Linux is highly standard (UNIX, POSIX), open file formats used all along, thousands of programs available for free download (although the ease of use and quality of these varies vastly). And most of all, Linux is enjoyable!

Next: Before Linux Installation
Back to Top Page

Part 1. Before Linux Installation

LINUX NEWBIE ADMINISTRATOR GUIDE

ver. 0.193 2002−12−14 by Stan, Peter and Marie Klimas

The latest version of this guide is available at http://sunsite.dk/linux−newbie.

Copyright (c) by Peter and Stan Klimas. Your feedback, comments, corrections, and improvements are appreciated. Send them to linux_nag@canada.com This material may be distributed only subject to the terms and conditions set forth in the Open Publication License, v1.0, 8 or later http://opencontent.org/openpub/ with the modification noted in lnag_licence.html.

Contents of this section:

1.1 Which Linux distribution should I use?
1.2 What are the Linux hardware requirements?
1.3 Will my hardware work under Linux?
1.4 How do I download Linux?
1.5 How do I get a Linux CD?
1.6 I have RedHat CD but no install floppy. What do I do?
1.7 What do I need to read before installation?
1.8 Can I have MS Windows and Linux installed on the same computer?
1.9 How to I partition my hard drive?
1.10 The MS Windows partition occupies my whole harddrive. Can I shrink/split it without a re−install?
1.11 How do I start the installation?
1.12 Is the Linux installation difficult?
1.13 Which packages should I install?
1.14 Which GUI desktop should I install, KDE or GNOME?
1.15 I finished the installation. How do I login for the very first time?
1.16 How do I crash Linux?
1.17 Can I use Graphical User Interface (GUI) all the time?
1.18 How do I upgrade a Linux distribution?

1.1 Which Linux distribution should I use?

Linux distribution is a coherent collection of free software with the Linux kernel (operating system) at its center.

The differences between the various Linux distributions ("distros") are minor: the installation program, choice of the bundled tools/applications, arrangement of a few things on the hard drive (most of Linux is still at the same, standard hard drive location in all distributions). Whichever distribution you decide to install, you will end up with essentially the same Linux.

We mostly use "Red Hat Linux" (also called RedHat or RH) and Mandrake (sometimes called MDK) for the following reasons:

1. They are both very popular (both an advantage for a newbie and a testimony to their quality).
2. They are both general−purpose distributions.
3. They both come with relatively easy setup programs.
4. Both Mandrake and RedHat contributions to Linux are "open software" (this means that all the software written by the packaging corporations and included on the distribution CDs is licensed under the General Public License, GPL, so that it can be legally copied, given away, reused, etc.).
5. Both Mandrake and RedHat can be obtained very cheaply or free if you don't care for commercial support. This is a consequence of (4).
6. Mandrake is based on RedHat, so both are very similar. Software packages for RedHat typically work on Mandrake (and vice versa) without problems. However, Mandrake is a bit more automatized and makes a somewhat nicer desktop than RedHat, and requires Pentium processor on default (RH will run on a i386).

In short, as a newbie, you can safely bet on "RedHat" or "Mandrake" unless you like something else or have specialized needs, or your environment suggests using something else (e.g., if you have an experienced guru nearby, or a bunch of friends who are using Linux, you may want to use the same distribution − makes getting help a whole lot easier).

The most recent distributions we recommend (Nov. 2002) is RedHat 8.0 or Mandrake 9.0. These are excellent distributions. Be sure to specify the most recent version if ordering your software from a dealer−−many dealers like to clear their inventory by sending you an older version (this applies not only to Linux). Generally, development under Linux is fast, and you don't want to waste your time with

older distributions. The authors of this guide have no connection to RedHat, Mandrake (or any other Linux distributor) whatsoever.

Our recommendation of Mandrake and RedHat for newbies does not mean that other distributions don't offer benefits or unique features which may surpass Mandrake or RedHat in specific areas. We do believe that we benefited from exposure to a different distribution because it helped us understand Linux better.

We tried Debian and we liked it very much. It was probably as easy as RedHat, but Debian seems less common (hence, being newbies, we picked up RedHat). The great benefit of Debian is that it is 100% non−commercial (put together by volunteer hackers, the true Linux way) and it probably most strictly adheres to Linux standards (it probably sets the standards too). Another great benefit is that Debian crams on their numerous distribution CDs thousands of tools and applications−−easily much more than any other distribution. All these tools/apps are nicely "packaged" (for ease of installation) and tested for compatibility. This makes Debian distro look monumental, safe, conservative, and always slightly outdated. So yes, we would not have a problem recommending Debian as a great general−purpose Linux distribution. Debian calls itself "The Universal Operating System" for a good reason. At any time, Debian carries 3 versions. (1) The "stable" version (sometimes called "potato"), and we would not recommend it, unless you are really paranoid on stability and don't mind quite outdated packages. (2) The Debian "testing" version (sometimes called "woody") is probably as stable as the latest RedHat, and more stable than your current Mandrake. It is much more up do date than Debian "stable". Debian Woody is the version we like. (3) If you don't mind occasional trouble, you can also the the third branch called "unstable", which is likely quite up−to−date.

Corel was once working on their own Linux distribution apparently geared towards a nice and easy platform to run the Corel suite of office applications: WordPerfect wordprocessor, QuattroPro spreadsheet, Corel Presentations, Paradox database, CorelDraw artist package.... The Corel Linux was based on the Debian distribution. It looked initially quite promissing, but it is unclear to me what Corel has done with it (was paid by Microsoft to drop it?). In brief, Corel Linux is dead now, and I would never recommend it to anybody because it it a dead−end. The only reason to mention it here is that Corel Linux once received lots of publicity, so you may still hear about it.

Slackware seems to be favorite among "cutting−edge hackers" who like being close to the operating system and hardware−−we did not use it so this is hearsay. We would have trouble recommending Slackware for Linux newbies. Our reviewer Bill Staehle says: "The real 'reason' for a newbie to avoid Slackware is that it is much more command line oriented, and lacks some of the 'cutsie slick and drool' tools that the other distributions have." However, we received feedback from Linux newbies who use Slackware and it works very well for them. It seems that Slackware is relatively simple and cool because of the lack of automation. Therefore, with a bit of effort, a computer−literate administrator can actually understand what is going in her operating system (this is not something I can always say about Mandrake, or MS Windows for that matter). Perhaps Slackware is to Linux what DOS is for MS Windows :) .

S.u.S.E distribution (http://www.suse.com) is very popular in Europe. It surely looks German−−a solid, general−purpose distro with an easy setup and an excellent reputation. Many users swear by SuSe. We couldn't find cheap Suse CDs though but it appears you can download it.

Caldera is another, well−known and respected distribution. It is said to be aiming at corporate users, have the most fancy installation program, a set of advanced (and pricey) remote configuration tools, and other corporate goodies. In Aug.2000, Caldera purchased SCO Unix (the original UNIX, including the UNIX trademark) which gives them an even more "corporate" look in my eyes. Caldera does not seem to be putting too much of their work into the Linux community, nor to care too much about the home Linux users, so I would not consider it for my home use.

There are "localized" versions of Linux for specific countries or languages (Korean, Chinese, Japanese...)−−they likely contain (on default) all the hacks and docs (documentation) that the users in these countries want to see. Says Bill Staehle: "You may want to mention the Conectiva Linux distribution, loosely based on RH from Brazil. As such, it is in Portugese, and is also available in Spanish. Try: http://www.conectiva.com.br/". I heard several good things about Conectiva, so if Portugese or Spanish was my language, I would probably give it a try.

There are also "special purpose" distributions, e.g. the "real−time" editions of Linux (might be useful if you are in for automation, robotics, fast speed data acquisition, etc.), very small distros (if you like the idea of running Linux from a single floppy which can be useful for system security or recovery), Linux for embedded systems (if you wanted to customize Linux as a small "special purpose" device, which could be good for the next−generation stereo, MP3 player, palm computer, or a fancy cellular phone), parallel computing and clustering systems (might be great if you plan to do your own weather forecasting :−) or at least nuclear explosion simulations :p), etc. Here the differences will be larger, but these distributions are not meant to be "general purpose". As a newbie, you likely don't want to start with any of these, although you might be tempted to. (They surely show Linux strength and viability−−Linux runs on toys, even a wrist watch, as well as computer clusters that make the currently fastest systems in the world.)

The distribution you need is of course specific to the hardware platform you have. This means that for your PC hardware containing an Intel 386 processor, or Intel 486, or Intel Pentium, or Intel 586, or Intel 686, or Cyrix, or K6, AMD, or similar, you need the binary distribution called "Intel" or "386" or x86. [Unless you are prepared to start with your own compilation of the Linux source code, which is not typical for a newbie :−)] . This happens because there are binary distributions for other hardware platforms too: PowerPC, Alpha, Apple, IBM mainframe, "Intel StrongARM", Transmeta, and perhaps a dozen more−−you don't want to get those binaries for

your PC clone; they surely will not work on a PC machine with an "Intel" or "AMD" processor inside. If you have no−Intel hardware, you may want to search the Internet to find who supports it (chances are Debian does, they seem to support even the most exotic ones. Then, you need to obtain "Debian ARM" or "Debian Motorola 680x0"or "Debian PowerPC" or "Debian SPARC ", ...).

In short, although newbies get confused with the multiple Linux distributions, there are reasons to have different distros. They should be viewed as a Linux strength rather than weakness. Linux is simply filling all application and hardware platform niches.

This guide concentrates on RedHat and Mandrake for the PC (Intel) platform. Many of the answers will work fine on other distributions or platforms, but we did not try them.

> Which Linux Distribution should I select for my old computer(s)? Quick answer: Debian, Slackware, or perhaps BasicLinux (current version), or an older version of RedHat, Mandrake, or SuSE. Justification: RedHat, Mandrake, SuSE, Caldera, and TurboLinux are optimized/suitable for hardware current at the date of their release. They may be difficult or impossible to install on older machines mostly due to the memory contraints and speed. Debian and Slackware are suitable for most older hardware as well.

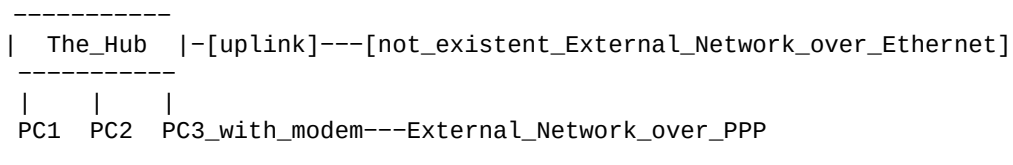
1.2 What are the Linux hardware requirements?

"Out−of−box" Linux should run on a 386SX−based PC with 8 MB of memory, but such a low−end computer is practical for text−only applications (no X−window). A 486 with 16 MB memory and 600 MB free (unpartitioned) hard drives work under X−windows but don't expect it to fly at all. My 586−133 MHz with 64 MB of memory runs acceptable under Linux with X. My 1.33 GHZ "Athlon" (AMD processor) with 256 MB of memory is a real pleasure to run with an instantaneous response even when running many large applications concurrently. I would not buy today a computer with less than 256 MB of memory (Dec.2001).

My 486−33 MHz with 8 MB memory and 1 GB hard drive has too little memory to run adequately stand−alone under GUI, but is still useful in my home network environment running as an X−terminal (a 486−class machine also performs just adequately stand−alone if it has at least 16 MB of memory but sometimes memory for old computers is hard to obtain at the price you would think it is worth). My old portable 386−SX−20 MHz Toshiba with 9 MB memory and 120 MB hard drive runs "legacy applications" under MS Windows 3.11 and it connects to our Linux home network and is thus still useful. We tried older Debian Linux on this Toshiba too, and it runs fine in text mode. (Pls note that Mandrake requires a Pentium processor.)

If you are willing to jump a few extra hoops, you should be able to install and run Linux on as little as 4 MB of memory, but this is probably not worth the effort for the general purpose home Linux machine. I would say: get at least 32 MB of memory, and if possible 128 or 256 MB −−more memory can make a difference in performance when running several large GUI applications concurrently. Memory is cheap these days. Please note that many current distribution have problems running their installation programs on older computers with a small amount of memory (although once installed, Linux will typically run just fine). If you require more help on installing Linux on a low−memory computer, try: http://7thguard.net/files/DebianHOWTO.txt

Networking is where Linux really shines, so consider getting 10−base−T Ethernet cards−−they are not very expensive and will be perfect to connect your two or more home computers together. Also, look around for old Ethernet cards which MS Windows deems obsolete−−they can be bought for a really low price and they will work great under Linux. To connect just two computers, a cross−over cable for direct Ethernet−card to Ethernet−card connection is sufficient ("networking for the poor"). To connect more than 2 computers together, you need a hub (~US$30 to US$80) and normal (not cross−over) cables. (If you have extra Ethernet cards, you may also consider installing more than 1 Ethernet card on a computer, use direct connections using the cross−over cables, and save the expense of a hub. But it adds a configuration complexity to your system. The 10−base−T system uses "giant phone" (RJ45)−type connectors and all machines are connected to one box (called the hub). The hub has an extra connection (called "uplink") which I will use if I ever have a permanent "over−Ethernet" connection to the outside world. Here is a schematics for a straight−forward home network arrangement:



> Here, I show a local private network consisting of PC1, PC2 and PC3, connected through a hub. Since I do not have "External_Network_over_Ethernet" on my home hub "uplink", PC3 provides my connection to the outside world (over a modem). Therefore, PC3 is called the "gateway" for all computers on my local private network (except PC3 itself). I enable the firewall software on PC3 PPP network interface, and let PC3 know how to dial out and connect to the outside. The outside world can only see PC3. As far as they can tell, PC1 and PC2 do not exist. My local ethernet network is "trusted" because only trusted people have physical access to PC1, PC2 and PC3. ("PPP" stands for "Point−to−Point Protocol" and it is a standard for communicating over phone lines.)
>
> You can, of course, build a more complicated network with Linux. A PC can have 2 (or more) ethernet cards. It may then work as bridge between 2 (or more) networks. The PC will act as a gataway for all traffic between

between the 2 networks. The networks do not have to be known to the outside world ("local private networks") and sit behind a firewall enabled on a gateway computer. The outside world will only know about 1 computer of mine, the "gateway" to the external network. Other computers will still be able to communicate with the outside world, but all the traffic will appear outside to originate from one, very busy computer−−the gateway.

Here is another suggestion on setting up a different kind of network, using a very much older type hardware, which uses coaxial cables (like for the cable TV). For this, no hub is necessary. Because this networking scheme is older, it can be assembled using cards and parts that are sometimes available for free:

(edited for space) From: John.Edwards@brunel.ac.uk Subject: Linux Guide−a suggestion

Hi. Many older 10Mbps network cards (and some newer ones as well) have a BNC connector and you can usually pick up old co−axial cabling when companies upgrade to UTP. Add a T piece for each machine and a 50−ohm terminator at each end (about 1 pound or $1.50 each) and you have a home network that will happily support more machines than you probably have room for. And most importantly−−no expensive hub (or cheap hub that can cause trouble). There are other advantages to co−ax as well, it's tougher to break and more resistant to noise from other equipment.

Disadvantages: There is a limit of 185 meters per network segment of thin co−ax, 30 machines per network, and you're stuck at 10Mbps, but I don't see any small home network needing more than that. Also if one cable goes down then the whole network stops, this shouldn't happen often unless someone unplugs a cable section. You can disconnect the T piece from a PC without harming the rest though.

Quick diagram, T for a T piece and Term for a terminator:

```
Term−T−−−−−T−−−−−T−−−−−T−Term
     |     |     |     |
     PC    PC    PC    PC
```

The various parts connect together using BNC connectors similar to a TV & video connector but with a bayonet that secures the two sockets together. For more details see the /usr/doc/HOWTO/Ethernet−HOWTO

The most straight forward and modern, however, is to get 10−base−T ethernet cards for your computers and a hub to connect them.

1.3 Will my hardware work under Linux?

Not every piece of PC hardware is supported under Linux, but most are, particularly the more standard, older, and popular ones. This applies to SCSI adapters, CDROMs, writable and rewritable CDs (CD−R and CD−RW), video cards, mice, printers, modems, network cards, scanners, Iomega drives, etc.

The most notable exceptions are the so−called Winmodems (=MS Windows modems also called "software modems"). Avoid these like fire−−they are a bit less expensive than full modems, but they are crippled (some processing is done by the main computer CPU instead of by the modem), and there is little chance you will have a Winmodem running on Linux right away (for more info on Winmodems, see http://www.idir.net/~gromitkc/winmodem.html). External modems are never "Winmodems" so if in doubt, purchase an external modem (external modems are more expensive, but they don't drain your PC power supply, are easily portable between machines, look better, and show modem activity). Additional points to consider with modems: "Older externals using a Rockwell Protocol that don't work too well. Also, the newer USB modems are not currently (March 2001) well supported. See the winmodem page." [source: B.Staehle].

Another area of potential problems is the video card. If you have a recent "cutting edge" 3D or uncommon card, you may want to check its compatibility at http://www.Xfree86.org.

Zip drives of all kinds are supported fine.

I wouldn't count on Linux supporting a parallel port (non−SCSI) scanner, no matter if the manufacturer claims TWAIN (="Technology Without An Interesting Name", no joke here) compatibility.

So the short answer is yes, in all likelihood your standard PC will run Linux with no problems. You don't invest much when trying Linux, so probably the easiest way to make sure is to attempt an installation on your existing hardware. There are Linux hardware compatibility lists at http://hardware.redhat.com/hcl/genpage2.cgi and http://metalab.unc.edu/LDP/HOWTO/Hardware−HOWTO.html if you want to check your newer or less popular hardware.

When purchasing new hardware, I would always check its Linux compatibility on the above lists. You can also ask your supplier if the hardware is supported under Linux, but I would take the answer with a grain of salt−−too many companies have incompetent sales personnel/technical support. When purchasing a new computer, I would consider a system with Linux pre−installed. A number of major suppliers offer systems (particularly large ones) with Linux, but many don't. You can always get a system with

Linux−preinstallled from a smaller vendor.

If you are an adventurous person, as I am, I would pay no attention to the remarks above, chances are 90−10 that the hardware will work.

If a piece of hardware of yours is (apparently) not supported in your current Linux distribution, don't give up. Chances are that: (1) It is supported, but you don't know how to set it up. (Solution: stay around with Linux for a few weeks, don't waste your time, when you get some understanding of how your system works, then you may be able to set it up.) (2) You have to go through a more complex setup to support the hardware (for example some cryptic command or a kernel re−compile, which is not as difficult as it seems). (3) An updated (different?) distribution already supports it "out−of−box" (you can usually order it for US$1.99). (4) There is already an upgrade somewhere on the Internet, you have to find it, download it, and figure out how to install it. (4) The upgrade will be available next month−−Linux development goes really fast!

1.4 How do I download Linux?

Do yourself a favor and do not download Linux. Buy an installation CD instead. Linux can be downloaded completely from the Internet, but it is a very large and sophisticated operating system. The download may take hours or days of download time, and you may encounter problems and frustrations, e.g., due to errors in the downloaded files.

If you do have a speedy Internet connection (definitely not a 56k−modem, but perhaps cable modem) and you are not a complete newbie, a Linux download may be an option to you after all. Try http://www.linuxiso.org/ for ready−to−burn CD images (ISO format) of your selected Linux distributions.

1.5 How do I get a Linux CD?

Many possible ways. (1) Buy the "RedHat" or Mandrake CDs from Linuxmall (http://www.LinuxMall.com/), or Cheapbytes (http://cart.cheapbytes.com/cgi−bin/cart)−−last time I checked, "the unofficial" RedHat GPL was US$2.99 or something like that for a 2 CD package + shipping and handling. They will mail you bare CDs. You get no printed manual, no support, no boot diskette, but the price is right, and the manual and tools to make a boot diskette are on the first (installation) CD. I purchased several packages from "Cheapbytes" and they always arrived fast, were of good quality, and there were no problems with my credit card charge (the authors have no connection to "cheapbytes" whatsoever). (2) Buy the boxed "official Red Hat" or "Mandrake" from the same place on the Internet or from your favorite software supplier; prices start at around US$40−−you will get the printed manual, e−mail or telephone installation support (60 days?), the boot diskette, additional CDs with "bundled" commercial applications, and perhaps other goodies (free updates?). If your time is worth lots of money, you may opt for more advanced technical support at higher price. (3) Copy the installation CDs from your friend. This is perfectly legal and ok−−Linux is free. If you have a Linux CD, don't be shy to loan it to your neighbor. (4) Check your library, local bookstore, or http:/www.amazon.com. Several Linux handbooks come with an attached CD containing a full Linux distribution. This is a good way to start with Linux because it is definitely a good idea to have a nice Linux handbook. With Linux' countless utilities, I need a handbook all the time. The drawback is that the books often include versions of Linux which are quite dated. Perhaps consider the "official" Linux with a handbook? (5) Visit a Linux "installfest" when one is organized in a place near you. Local Linux "gurus" will install Linux on your computer free (bring the computer) and you will likely be able to get a Linux CD too (why don't you bring some empty CD−R to the fest?). Check for the Linux User Group on the net to see when the nearest to you plans an installfest. Good way to meet other Linuxers too.

> Here is a more comprehensive list of places to obtain Linux CDs with their location, so you can find something near to you (after B.Staehle) : http://www.ixsoft.de (low price CDs in Europe); http://linuxservice.de (another source in Germany); http://www.polo.demon.co.uk/emporium.html (The Linux Emporium); http://www.linux−emporium.co.uk; http://definite.ukpost.com/ (Definite Linux Systems); http://www.mallind.demon.co.uk/ (GPL + official distros); http://www.amush.cx/linux/ (GPL distributions in UK only); http://www.linux123.co.uk/ (GPL + official distros in .uk); http://www.kihi.com.au/bowtie/ (Bowtie Software − cheap CDs in OZ); http://EverythingLinux.com.au/ (cheap CDs in OZ); http://www.lsl.com.au/ (cheap CDs in OZ); http://www.arles.ns.ca (official distros and BSD in .ca); http://www.affinity−systems.ab.ca/ (Official distros − hardware in .ca); http://www.warpedsystems.sk.ca (GPL + official distros, custom built system CA); http://www.softcopy.on.ca/ (cheap CDs in CA); http://www.libranet.com (Libranet Linux Vancouver, BC, CA); http://www.linuxwarehouse.co.za/ (low cost and official in South Africa); http://linuxcentral.com (Clinton Township MI 48035); http://www.lsl.com (Chesterfield, MI 48047, USA); http://www.cheapbytes.com (Lodi, CA 95241, USA); http://www.infomagic.com (Flagstaff AZ 86004, USA); http://www.tummy.com/krud/ (Fort Collins, CO 80525, USA); http://www.pieceby.com/ (Hudson, NH 03051, USA); http://www.linuxmall.com (Aurora, CO 80046−0190, USA); http://www.linux−now.com (Clarion, PA 16214, USA); http://www.xcomputing.com (San Francisco, CA 94134, USA); http://www.ccsoft.cc/linux/ (Santa Rosa, CA 95401, USA); http://www.storeanywhere.com/ (Brooklyn, NY 11235, USA); http://www.linuxcomponents.com (Owings Mills, MD 21117, USA).

1.6 I have Linux Installation CDs but no install floppy. What do I do?

If your computer can boot from the CD drive (most older computers cannot), you don't need a boot diskette to install Linux. Have a look at your BIOS setup; the boot sequence can often be set up there (the default is often floppy followed by hard drive). My computer has the CD drive specified as the first boot device in the BIOS yet still cannot boot from the CD drive. So the BIOS setup does not necessary reflect the capability of your machine. If you can boot from the CD drive, just insert the RedHat CD into the CD

drive and reboot the computer to enter the RedHat Linux installation program.

If you don't know how to access your BIOS setup, read this paragraph. The BIOS setup can typically be entered at boottime by pressing the proper key at the right moment (often when a prompt is briefly displayed). Most often, it is the <Del> key. Here is a list of key combinations used by popular BIOSes: Acer notebooks: <F2> during Power−On Self−Test (POST). American Megatrends (AMI): <Del> during Power−On Self−Test. Award: <Del>, or <Ctrl><Alt><Esc>. Compaq: <F10> after the square appears in the top right corner of the screen during boot−up. Dell: <Ctrl><Alt><Enter>. DTK: <Esc> during Power−On Self−Test. IBM Aptiva 535: <F1> while the square with the wavy lines is displayed in the upper right corner during power−on. IBM PS/2: <Ctrl><Alt><Del>, then <Ctrl><Alt><Ins> when the cursor is in the top right corner Mr. BIOS: <Ctrl><Alt><S> during Power−On Self−Test. Packard Bell: For some models, <F1> or <F2 > during Power−On Self−Test. Phoenix: <Ctrl><Alt><Esc> or <Ctrl><Alt><S>, or <Ctrl><Alt><Enter>.

If your computer cannot boot from the CD drive, make an install boot diskette from under DOS or the MS Windows DOS mode. (You have go to to "Shutdown" and "Restart in MS−DOS mode", not just run a DOS window).

It is important that you have a perfectly good floppy (without even one bad cluster). The program that makes the diskette does not check if the floppy was written properly. Also, don't count on the DOS `FORMAT` utility finding a faulty floppy−−it probably won't. If I were you, I would make two or three boot floppies at once−−you may be surprised how many diskettes have problems. For me, the third floppy worked! If your install diskette does not boot, make another one−−it definitely should boot.

Here are the commands. To make the boot floppy run:

```
F:\dosutils\rawrite.exe −f F:\images\boot.img −d a: −n
```

To make the supplemental (optional) diskette run:

```
F:\dosutils\rawrite.exe −f F:\images\supp.img −d a: −n
```

This assumes your CDROM is the DOS "F:" drive, and your floppy is "A:", adjust the commands if the drive letters are different on your system.

The commands above run the utility "rawrite" and specify the input file ("disk image", after the option "−f") and the destination drive (after the option "−d"), and suppresses the prompt to insert a floppy (option "−n"). You may find it easier to run `rawrite` without any argument−−it will interactively prompt you for the input image (pick the file name as in the commands above) and the destination drive letter.

From under Linux, you can make a boot disk by mounting the RedHat CDROM and typing the commands (as root user):

```
cd /mnt/cdrom/images/
dd if=boot.img of=/dev/fd0
```

[The ``dd" command copies files. The above command specifies that the input file ("if") is ``boot.img" and the output file ("of") is /dev/fd0, which is the first floppy drive, i.e. the floppy drive number zero (if you want to write to your second floppy drive, use /dev/fd1).]

1.7 What do I need to read before installation?

It is VERY helpful to get some UNIX orientation if you don't have any. Buy a good Linux manual or dust your old Unix handbook. Almost all Unix concepts apply in Linux, and almost all UNIX commands will run fine under Linux. I find manuals for MS Windows useless (click this, click that, look at the screenshot), but manuals for Linux/UNIX are typically great (give you an understanding of the system, a lasting benefit).

You may want to learn about your hardware: how many and what size hard drives you have, the type, number, order and size of all partitions on each drive, where your DOS/Windows partitions are, which one is the DOS/Windows boot partition (if you plan to have dual boot), what type of mouse you have, what video card and with how much memory, what monitor (max synchronization frequencies), etc.

Go to BIOS setup to see the number and geometry of your hard drives. Run DOS "fdisk" to display your hard drive(s) partition table(s), and print it. Watch your system boot to learn about the type of your video card and the amount of video memory. Boot MS Windows, go to the control panel−devices and write down the sound card, modem, network card types and settings (name, type, IRQ, i/o address, DMA channel). Read the label underneath your mouse to see the type of mouse you have. (Next time you buy a mouse, get a Linux−ready 3−button Logitech or similar−−Linux makes good use of all three buttons.) Dust off your monitor manual to find out the maximum synchronization frequencies (vertical and horizontal) that your monitor supports. Never use frequencies out of the monitor specification−−this may damage your monitor.

You may want to browse the RedHat or Mandrake manual. If you don't have the printed copy, an html version is on your CD so you can read it using any web browser, e.g. Netscape for Windows. Look here to see how to access this manual and some additional reading material which is on your Linux CD.

1.8 Can I have MS Windows and Linux installed on the same computer?

Yes, you can. Many Linuxers use a dual boot. This is typically achieved by installing MS Windows on one hard drive partition and Linux on another partition. Linux comes with a simple boot manager called LILO (or a more sophisticated one called GRUB), which will let you choose, at boot time, the operating system you boot. Install MS Windows first and Linux only afterwards or else the MS Windows installation program will disable your access to Linux. Have a Linux boot floppy ready if you need to re−install MS Windows−−MS Windows will surely disable your access to Linux and you will have to boot Linux from the floppy and then re−run the command `lilo` to be back in business.

From under Linux, you will be able to read from and write to your MS Windows drive partitions so that the data exchange between MS Windows− and the Linux−based program is seamless. You will also be able to use your existing MS Windows−based resources: sound files, backgrounds, pictures, fonts, etc. (First check if it does not violate your license agreement though, smile. For products that are on rent to you from Microsoft, it probably does. With my Linux computers, I am proud to have no pirated software on my system whatsoever.)

1.9 How do I partition my hard drive?

Before Linux installation, you might really want to know what a hard drive partition is. The concern is that you delete your MS Windows partition when you really don't want to−−you want two separate partitions to dual boot. This means: MS Windows is on one partition, Linux is on a separate partition. You do not normally install Linux on free space on your MS Windows−allocated partition(s). It is possible to install Linux on a MS Windows partition, but we do not recommend it.

If you plan a dual boot (Linux and MS Windows on the same computer), first use your DOS/Win utility `FDISK` to make the MS Windows partition(s). Leave part of the hard drive(s) unpartitioned for Linux. You will make and format the Linux partitions during your RedHat (or Mandrake or whatever else) installation. Linux will recognize the free space on the harddrive.

Make the MS Windows partition "primary" and "bootable". Install, configure, and test your MS Windows before Linux installation. If you plan to run Linux only, you need just a clean hard drive (no partitions) to start with.

It is possible to have only one Linux partition (plus one for MS Windows if you dual−boot). But it is better to have more partitions so that you can keep users' data separate from the rest of the operating system. This way, if something ever goes wrong, or if you have to reformat or re−install the operating system, you don't lose the users' data. (You can perform a complete Linux re−install without losing the contents of the `/home` directory that contains all user data if you skip the "re−format" option given to you during installation. But for that, the `/home` directory must be on its own partition.)

During the Linux setup, you will be asked to partition the available space on your hard drive(s). There are many possible ways to partition, depending on your hard drive space, requirements, and taste. I like Linux hard drive partitions like this (for a modest total of 2 GB of hardrive space which I give to Linux in this example):

```
mount point    type     size
/              ext2     300 MB
/usr           ext2     1200 MB
/home          ext2     380 MB
swap           swap     120 MB
```

In the above example, I dedicate 300 MB for the root partition that holds the base of the Linux operating system. I allocate 1200 MB to the mount point that will be visible on my filesystem as the `/usr` directory and will contain the user's programs (the programs that don't come with the base operating system and I install later, for example StarOffice). I dedicate 380 MB for the partition that will be visible as the directory `/home` and will contain the setting and data of all users on the machine. And I allocate 120 MB to a "raw" partition for the operating system to use as the virtual memory (extension of the physical, silicon memory on the hard drive, so−called swap). If your kernel is lower than 2.2 (this is the case with standard RH5.2 and earlier), your swap partition cannot be larger than approximately 127 MB. The rule of thumb is that the swap should be about twice the amount of the physical memory (RAM). If you need more (e.g. if you have lots of physical memory, or you expect to run custom programs with really large data structures) you might want to create a larger swap partition during the installation (or several smaller swap partitions) or add a swap file(s) later.

2 GB is a respectible amount of disk space and should be sufficient for users who like having many applications. (This is because Linux applications tend to be slimmer than their MS Windows equivalents). However, if you try to install everything that's available on the modern distribution CDs, you will surely run out of disk space. My experience is that however large the hard drive space, it will get filled and I regret I don't have more :−) .

If my space on the hard drive is really restricted, I may consider a two−partition setup like this (for a lean 650 MB total dedicated to Linux):

```
mount point    type     size
/              ext2     600 MB
swap           swap     50 MB
```

In this example, I dedicate 600 MB to the base of the operating system, applications, and user documents/data, and allow 50 MB for the swap partition (for the operating system to use as the virtual memory). The 50−MB swap should be quite sufficient for medium duty operations. The limitation of 600 MB for the operating system, applications and user data means that you will have to be very selective as to which applications you install or else you risk running out of hard drive space. Try pressing <F1> when installing the optional software that comes on the Red Hat CD−−it will give you a short description of what the software does so you could perhaps decide if you really need it. (Don't worry too much if you miss something you need, you can install the missing parts later). You can easily finish the RedHat installation with 200 MB free on your Linux partition (out of 600 MB used in this example) if you make reasonable choices. Please note that "bundling together" the root partition "`/`" and the `/home` directory will likely save you some disk space, but it is not the safest solution.

It is possible to install Linux on even less disk space than in the example above, but you will have to be really picky as to what you install.

For a larger available hard drive space, I may consider the following setup (for a comfortable total of 15 GB dedicated to Linux):

```
mount point    type     size
/              ext2     800 MB
/usr           ext2     5000 MB
/usr/local     ext2     3000 MB
/home          ext2     5200 MB
swap           swap     1000 MB
```

Please note that the the mount points can reside on different physical hard drives. Linux agglomarates all the hard drive space into a single directory tree.

Another consideration when setting up the partitions on older computers (486?). Many older BIOSes have the restriction that the boot partition cannot extend beyond the 1024th cylinder on your first physical hard drive. To overcome this limitation, simply make the first (bootable) partition so that it ends before the cylinder number 1023 (this makes this partition max approximately 512 MB in size, which is plenty for the "/" root partition). Once Linux boots, the BIOS restriction does not matter any more because Linux takes over the hardware management and it can access the partition(s) beyond the cylinder number 1023.

When installing and using Linux, your drives appear as devices with the following names: hda−−first IDE drive (stands for "hard drive a", i.e. the master drive on the first IDE interface), hdb−−second IDE drive (i.e., the slave drive on the first IDE interface), hdc−−third IDE drive (i.e. the master drive on the second IDE interface), hdd−−fourth IDE drive (i.e. the slave drive on the second IDE interface). The numbers mean the partitions on the physical drives: "hda1" means the first IDE hard drive (hd a), first partition (1); "hda2" is the first IDE hard drive, second partition; "hda3"−−the first IDE hard drive, third partition; (and so on if you have more than 3 partitions on the first IDE hard drive); "hdb1"−−second IDE hard drive, first partition (or just "hdb" if it is the CDROM installed as a slave on your first IDE interface). "hdc1"−−third IDE hard drive, first partition, etc. SCSI drives have analogous names but start with the letters "sd" (="SCSI drive"), followed by the letter indicating the SCSI interface and by the number indicating the SCSI device id. For example, "sda4" means "first SCSI interface, id number 4". If you have an external zip drive attached to your parallel port, it will appear as SCSI device "sda4" (zip drives work in a SCSI−emulation mode).

The listing of partitions that your Linux setup program presents to you during installation will include any MS Windows partitions which you have. For example, I have the following MS Windows partition:

```
mount point    type     size        comment

[no mount]     vfat     1200 MB     ["Win C drive, hda1]
/mnt/dos_hdd2  vfat     1600 MB     ["Win D drive, hdd2]
```

Don't erase these MS Windows partitions during your Linux installation if you want a dual boot. If you erase the MS Windows partition, MS Windows is gone from your system! If not sure, backup your data from your MS Windows partitions before Linux installation. "msdos", "fat" and "vfat" and "ntfs" are typical filesystems used by DOS and MS Windows 3.x/95/98/NT.

As a quick reference, here is a brief summary of the standard linux partition types ("filesystems") with a short description. I copied the info from the linux manual pages: `man fs` and `man mount` (with some additions after I had a look at the source code files at

`/usr/src/linux/fs`). The **underlined** filesystems are the ones that you are more likely to use. Other filesystems (not listed below) are available as add−ons (for example journaling filesystems, compressed, encrypted, ...).

minix is the filesystem used in the Minix operating system, the first to run under Linux. It has a number of shortcomings: a 64MB partition size limit, short filenames, a single time stamp, etc. It remains useful for floppies and RAM disks.

ext is an elaborate extension of the minix filesystem. It has been completely superseded by the second version of the extended filesystem (ext2) and will eventually be removed from the kernel.

ext2 is the high performance disk filesystem used by Linux for fixed disks as well as removable media. The second extended filesystem was designed as an extension of the extended file system (ext). ext2 offers the best performance (in terms of speed and CPU usage) of the filesystems supported under Linux. In short, ext2 is the main (default, typical) Linux filesystem.

ext3 is an extension of the ext2 filesystem with journaling. It is backwards and forward compatibile with ext2. It means that ext2 can be converted into ext3 without reformatting or data loss (just re−mounting the partion is required). ext3 can be changed back to ext2, also without data loss. I use ext3 extensively since Oct.2001−−it is simple and trouble−free. It is included as an installation "option" since RedHat 7.2 and Mandrake 8.0. It is highly recommended that you use this filesystem.

xiafs was designed and implemented to be a stable, safe filesystem by extending the Minix filesystem code. It provides the basic most requested features without undue complexity. The xia filesystem is no longer actively developed or maintained. It is used infrequently.

msdos is the filesystem used by DOS, Windows, and some OS/2 computers. msdos filenames can be no longer than 8 characters followed by an optional period and 3 character extension.

umsdos is an extended DOS filesystem used by Linux. It adds capability for long filenames, UID/GID, POSIX permissions, and special files (devices, named pipes, etc.) under the DOS filesystem, without sacrificing compatibility with DOS.

vfat is an extended DOS filesystem used by Microsoft Windows95 and Windows NT. VFAT adds capability for long filenames under the MSDOS filesystem.

proc is a pseudo−filesystem which is used as an interface to kernel data structures rather than reading and interpreting /dev/kmem. In particular, its files do not take up disk space. See `man 5 proc`.

iso9660 is a CD−ROM filesystem type conforming to the ISO 9660 standard. Two extensions (listed below) are automatically supported.

High Sierra −−Linux supports High Sierra, the precursor to the ISO 9660 standard for CD−ROM filesystems. It is automatically recognized within the iso9660 filesystem support under Linux.

Rock Ridge −−Linux also supports the System Use Sharing Protocol records specified by the Rock Ridge Interchange Protocol. They are used to further describe the files in the iso9660 filesystem to a UNIX host, and provide information such as long filenames, UID/GID, POSIX permissions, and devices. It is automatically recognized within the iso9660 filesystem support under Linux.

hpfs is the High Performance Filesystem, used in OS/2. This filesystem is read−only under Linux due to the lack of available documentation.

sysv is an implementation of the SystemV/Coherent filesystem for Linux. It implements all of Xenix FS, SystemV/386 FS, and Coherent FS.

nfs is the network filesystem used to access disks located on remote computers.

smb is a network filesystem that supports the SMB protocol, used by MS Windows for Workgroups, Windows NT, and Lan Manager. To use smb fs, you need a special mount program, which can be found in the ksmbfs package, found at ftp://sunsite.unc.edu/pub/Linux/system/Filesystems/smbfs. [Standard linux command "smbmount" will also do.]

ncpfs is a network filesystem that supports the NCP protocol, used by Novell NetWare.

devpts−−is a pseudo file system, traditionally mounted on /dev/pts. In order to acquire a pseudo terminal, a process opens /dev/ptmx; the number of the pseudo terminal is then made available to the process and the pseudo terminal slave can be accessed as /dev/pts/<number>.

fat −−is not a separate filesystem, but a common part of the msdos, umsdos and vfat filesystems.

UFS −−is a file system widely used in different operating systems.

swap−−is a special partition type used for swapping data from memory to hard drive.

raiserfs−−is a brand new journaling filesystem available as standard with Linux kernel version 2.4.1 up (January 2001).

hfs (=hierarchical files system)−−MacIntosh filesystem. It is a late beta version., i.e., not recommended for use with critical data, unless read−only.

ntfs−−MS Windows NT filesytem. It is still "experimental" under Linux, i.e. not recommeded for production machines, unless read−only (Aug.2001).

1.10 The MS Windows partition occupies my whole harddrive. Can I shrink/split it without a re−install?

Possibly. There is a utility called `FIPS.EXE` on my RedHat CD that does just that. Check the directory `\dosutils\fipsdocs\` on your RedHat CD for documentation. If I were you, I would back−up my essential data before doing anything to my partitions. There are also commercial utilities to change the partition size without destroying its contents.

My personal preference is to do a clean re−install of MS Windows on a single, dedicated partition. I leave some space on the hard drive unpartitioned so I can use it later for Linux. My fair division of hard drive space between MS Windows and Linux is 50/50. Linux programs tend to be smaller but they include (as standard) components that MS Windows offers only with many thousand of dollars of add−ons: e.g., servers (not just clients) for telnet, ftp, http, and mail, several databases, programming languages, graphics processing programs ...

1.11 How do I start the installation?

Insert the installation boot diskette into your floppy drive, the RedHat CD to the CDROM, and reboot. If you can boot from the CD, insert the RedHat CD into your CDROM drive and reboot.

You may also start the installation from DOS (or real DOS mode under MS Windows), by running `EZSTART.BAT` which is on my RedHat CD from Cheapbytes.

Most installers give you an option between text and graphical install. You need to select "text" if your computer memory is restricted.

1.12 Is the Linux installation difficult?

It was not for me. It seems that for most newbies, it is fairly straight forward and painless. Depending on your hardware and installation choices, it typically takes 1 h to 2.0 hours. [Expect longer or possible problems for slow systems with very restricted memory−−it took a whole night to install RH6.0 on my 486−33 MHz with 8 MB memory, the system pausing for 5 minutes at a time appearing to do nothing, yet it installed ok.] Upgrades from previous installations take longer and tend to be more problematic.

However, some newbies reported that the installation was a "total nightmare" to them (hardware problems? lack of experience?). If you encounter problems, my advice would be to install a plain−vanilla system, without struggling with the highest resolution on your fancy video card or other bleeding−edge hardware which you might have. Anything can be added/configured later, after you get more understanding of how things work on your system. Even a re−install is always an option for a newbie (it seems Linux gurus think it is a shame to ever re−install). It seems that many newbies have problems because they specify too high screen resolutions (which may be not supported or supported only with some extra tune−up). Again, it may not be wise to break your whole installation for support of a single device−−the support can be added/tuned−up later.

1.13 Which packages should I install?

Linux software comes in "packages". For example, my Linux Mandrake 7.0 installation CD contained 1002 packages. Mandrake 7.2 packs 2 CDs of software−−my installation of Mandrake 7.2 put 1123 packages on the hard drive. Mind you, this is not all the software available for Linux−−just a selection made by people who put the Mandrake distribution together. Mandrake tends to pack more software than RedHat.

No matter what distribution or version, the CD contains packages that make the base operating system (kernel, libraries, a selection of command−line configuration and maintenance tools, etc) a rich selection of networking "clients" and servers" with appropriate configuration and monitoring tools, some end−user text mode applications, base X−windowing system, at least one GUI desktop (most likely several), and likely a slew of GUI applications.

The installation program (either RedHat or Mandrake) will ask you which packages to install. If you select "workstation installation", then the packages normally found on servers will be omitted from your installation (for example, the Apache web server may be skipped). If you choose "server installation", then typically the end−user desktop applications will not be installed (for example, the GIMP graphical utility may be omitted). You can also choose to install "everything", and this is my favorite option for a home computer installation. Finally, you may opt to make your own selection of packages to install−−read on.

It is definitely a very bad idea to select installation of packages/programs on the basis of how interesting their names sound−−some packages have rather unusual names and I would never guess what they do. You could cripple your system by omitting the installation of an essential package (e.g., a library). You might also be disappointed when insisting to run some cool−named, cutting−edge piece of

software ("version 0.1") that happened to be included on the distribution CD. In general, you might be annoyed by the functionality (or lack of it) that your "customized Linux" exhibits. Being a newbie, it sometimes pays to trust the defaults selected by your distribution creator.

Therefore, for my final "production" installation, I would stay away from the tempting installation option "expert install−−select packages manually" unless I wish to install everything anyway. For starters, I like the safe "max default installation", however this installation option is called on your CD.

If you don't install a package and later find that you need it−−don't panic. It can easily be installed later. Read on.

1.14 Which GUI desktop should I install, KDE or GNOME?

Disk−space permitting, definitely both. You can later decide if you prefer KDE, GNOME or another desktop, but whatever your choice, you definitely want both the KDE and the GNOME libraries installed. Once you have the libraries installed, KDE programs can be run under GNOME and vice versa, which is great because there are nice applications written using either library. As far as the amount of disk space is concerned, the "desktop" is only a small part of the KDE and GNOME systems so you don't save much space by omitting the desktop and trying to install "libraries only". Both GNOME and KDE come with a set of nice programs and tools, so it is definitely worth it to install both desktops in full. I never heard that the two adversely interfered with each other. For every−day work, I use the KDE desktop, because it feels more solid than GNOME. If you like more "cutting edge" and "cooler", go GNOME, but don't complain if things don't always work quite that well.

I would also install the other "alternative windows managers". They hardly take any space (some are really tiny) yet they can be useful under some circumstances. You can run any KDE or GNOME application from under any of them, as long as KDE and GNOME libraries are installed.

KDE is more power hungry. On older hardware (e.g., 133 MHz Pentium) I prefer GNOME to KDE. Other windows managers are ligther than either KDE or GNOME. Therefore, on really modest hardware, I would choose one of the "alternative" windows managers.

1.15 I finished the installation. How do I log−in for the very first time?

As root. "root" is the only account that exists after the initial installation (newer installation programs do prompt you to create a regular user account during the installation). Example text mode login:

```
my_machine_name login: root
Password:  my_password
```

In the example above, I typed the word "root" at the login prompt. After that, I entered the password that I chose during the initial Linux installation. The password did not appear on the screen when I typed it (for security). After I login, I find myself in a text−mode terminal.

If you installed the GUI login screen, the login procedure looks similar, but occurs on an X−window screen (if you occasionally have problems typing there, perhaps remember that your mouse cursor must be above the dialog box. The X login screen implements the "focus−follows−mouse" policy). After a successful login, my default GUI desktop is launched.

"root" is a special account with an absolute power over the system, and it is used for system administration. You surely want to create at least one more "user" account later to do regular (not system administration) work. Read on to learn how to do it.

1.16 How do I crash Linux?

As root, you can do whatever you want, including an accelarated system crash. Try (reconsider if you really want to crash):

```
cp /dev/zero /dev/mem
```

As root, you can even erase all the files on your system with a similarly innocuously looking one−liner (don't do it):

```
rm −fr /
```

This is not to say that Linux is an easy crash, but that the system administrator ("root") has the complete power over the system. You can make MS Windows unusable by trashing some files from C:\WINDOWS OR C:\WINDOWS\SYSTEM directory. An important distinctions that makes Linux resiliant is that the user and administrative accounts are separated. Regular users can touch only the files they own, and similarly the programs users run can only junk user−owned areas, no matter how buggy or malicious a program might

be. The separation of "administrative" and "user" accounts adds to system complexity, yet it also makes the Linux system truly multi−user. This is unlike the old MS Windows approach. With the latest version of MS Windows, Microsoft moves more towards the UNIX approach. An old saying comes to mind: "those who do not know UNIX are destined to re−invent it".

Conclusion: do not use the root account for routine work. Add a regular−user account as your first administrative task and use this account for your experimenting with Linux. Here is how to do it (as root):

```
adduser johnbrown

passwd johnbrown

[enter a good long password with a number in the middle]

[reenter the same password without a mistake]

exit

[login as johnbrown]
```

The root account is for administration and setup only. As root, I would not do things/run programs that I do not exactly understand what they do. At least not on a computer that I plan to use for real work. Really.

My learning path was as follows:

1. Install Linux.

2. Play around, experiment (root and not root, do cool things, setup stuff. I want to be able to predict the behaviour of my system−−only then do I know that I understand it).

3. Re−install.

4. Unless I don't have any more time (I never have enough), goto 2.

Once properly installed on good hardware, command−line Linux is legendarly stable−−it can run for months or even years. As a newbie, you can almost bet that a funny system behaviour results from either your actions as root, or from flakey hardware.

1.17 Can I use Graphical User Interface (GUI) all the time?

I imagine it is possible to use GUI exclusively under Linux, but I don't think it would be very efficient for administration. The GUI under Linux is very nice, but it pales in comparison with the possibilities available under the command line. To make sure, I use the GUI every day both under Linux and MS Windows, and I find GUI great for program launching, and other routine tasks. Yet, for system housekeeping or automation, GUI is inflexible. The command line interface (CLI) is a richer interface to me, and it also gives me some understanding of the working of my system.

Two quotes to amplify this point: "Using a graphical interface is fairly easy because it limits you to a few basic operations that you can learn quickly. But if those basic operations aren't what you need, then you may have problems". "One characteristic of a user−friendly system is that it does what the user wants. In other words, the most user−friendly system isn't necessarily the simplest one." [http://linux.oreillynet.com/pub/a/linux/2001/11/15/learnunixos.html]

I find that a mix of GUI and command line is perfect to cover all my needs. If you are determined to use GUI only and are not willing to learn any command line options, you might consider delaying your Linux installation until the GUI tools are better developed, unless you have somebody to help you administer your computer for now.

On the other hand, if you have a computer−agnostic girlfriend or boyfriend, and all s/he does is execute perhaps half−a−dozen different programs, you can set up a nice GUI screen for him with the icons or buttons or menus he requires. This, after some intial encouragement, may make him accept Linux.

From this Guide's point of view, command line is certainly simpler for documentation. Icons and menus are meant to be customizable and therefore your icons and menus may be quite different from mine. Also, to set up an icon or command, I need to know the command that stands behind the icon. In short, understanding of the command line is indispensible for setup and any work beyond trivialities, even under GUI. We include no screenshots in this guide−−we found them ourselves useless in learning about computers.

1.18 How do I upgrade a Linux distribution?

For a full system upgrade (updated distribution CDs, .e.g., RH7.0−>RH7.2), "follow the time−honoured principle of upgrading Unix systems: do a fresh, 'clean' install and add back your data. Yes, we're talking about reformatting your partitions and installing from scratch." (the quote from: http://www.northernjourney.com/opensource/newbies/newb025.html).

Thoughtful partitioning of your hard drive will facilitate future upgrades (your data files can be preserved in your "home" directory).

A checklist I made for myself when upgrading:

− Log in as root.

− Make a copy of the `/etc` directory to some place where it won't be distroyed (e.g., zipdrive). This will let me have a look at your previous setup in case I encounter problems.

− Make a backup of any valuable data in the /home directory. This is just in case something went really wrong, for example, if I happened to format a wrong partition.

− Determine and write down the mountpoints on your filesystem using the "diskfree" command:

```
df
```

For example, on my system it shows that the `/home` my home directory is mounted on a separate hard drive partition called `/dev/hda9`. and the directory `/usr/local` is on `/dev/hda8`.

− Perform the installatation of Linux, but preserve and don't reformat the partitions `/home` and `/usr/local`. This is critical if you want to preserve your data.

− Boot the new installation and check that it works.

− For each user on the system, create a login with the old name and old user ID number, for example,

`ls /home/maria/* −l` (check the user ID for account "maria" in the file listing of her home directory),
`useradd maria −u 503` (create a user acount "maria" with with an example uid "503").

− From each user home directory, delete the potentialy troublesome setup files, e.g.: ".kde", ".ICE*", etc. Good oportunity to delete any junk too. The files may be troublesome because the new version of kde is likely to perform better with it most recent settings (it will create them on first startup). E.g.,

```
cd /home/maria
rm −fr .kde
```

− Make sure that each remaining file in the home directories belongs to the appropriate user. For example, I may do:

```
cd /home/maria
chown −R maria *
chgrp −R maria *
```

Go to part 2: Linux Resources, Help and Some Links
Back to Main Page

Part 2. Linux Resources, Help and Some Links

LINUX NEWBIE ADMINISTRATOR GUIDE

ver. 0.193 2002−12−14 by Stan, Peter and Marie Klimas
The latest version of this guide is available at http://sunsite.dk/linux−newbie.
Copyright (c) by Peter and Stan Klimas. Your feedback, comments, corrections, and improvements are appreciated. Send them to linux_nag@canada.com This material may be distributed only subject to the terms and conditions set forth in the Open Publication License, v1.0, 8 or later http://opencontent.org/openpub/ with the modification noted in lnag_licence.html.

Contents of this section (Linux Resources, Help and Some Links):

2.1 Any Linux reading materials?
2.2 Is there a help command?
2.3 Any dictionary of terms?
2.4 Web search
2.5 Newsgroups
2.6 Linux Internet links
2.7 Source code−−the ultimate resource

2.1 Any Linux reading materials?

This guide is not sufficient?

The RedHat Linux distribution CDs contain lots of documentation. Part of it is in html format and part in plain text format. You can read it all from under DOS or MS Windows before you install Linux. For example, a soft−copy of the RedHat manual can be viewed with any MS Windows−based html browser, e.g. Netscape for Windows or MS Internet Explorer. Just access the file `D:\doc\rhmanual\manual\index.html`. (assuming your CDROM is drive D under MS Windows).

Also, check the directory `\doc\LDP` for the excellent Linux Documentation Project manuals. For example, you can browse the Linux System Administrators' Guide by accessing the file `\doc\LDP\sag\sag.html` with your favorite html browser.

Also, check the directory `\doc\HOWTO` for the HOWTO documents, the directory `\doc\HOWTO\mini` for the MINIHOWTOs and the directory `\doc\FAQ` for a set of FAQs on different topics (FAQ="frequently asked questions"). For example, these commands will let you read the Linux−FAQ document (plain−text format) from under DOS:

```
D:
cd \doc\FAQ\txt
edit Linux−FAQ
```

Under Linux, you can read the same documentation from the CD using, for example, this command:

```
lynx /mnt/cdrom/doc/rhmanual/manual/index.html
```

This will start lynx, a simple text−mode html browser, to view the RedHat manual. Please note that under Linux, the CD must be mounted first, and the example above assumes that the mountpoint is the directory `/mnt/cdrom/`. You can also use Netscape for Linux, StarOffice or any other html browser to view the RedHat manual and other documentation in the html format. You can read plain−text documents from the CD under Linux using, for example, these commands:

```
cd /mnt/cdrom/doc/FAQ/txt/
less Linux−FAQ
```

(The `less` command lets you scroll through the contents of a text file.)

After installing linux, the documentation, whatever part of it you installed, is in the directory `/usr/doc/` or `/usr/share/doc`. If you didn't install the documentation, consider installing everything now, it may be worth it. For example, the directory `/usr/doc/LDP` contains the Linux Documentation Project manuals. These commands will let you browse the Linux System Administrators' Guide:

```
cd /usr/share/doc/LDP/sag
lynx sag.html
```

Also, check `/usr/share/doc/HOWTO` for the HOWTO documents, and `/usr/share/doc/HOWTO/mini` for the MINIHOWTOs.

The location of the documentation is sometimes `/usr/doc`.

For more or updated documentation, see http://www.ibiblio.org/mdw/index.html

2.2 Is there a help command?

Most Linux commands can be run with the "−−help" option. For example, this command will give you concise help on the Linux `cp` (copy) command:

```
cp −−help | less
```

More extensive info is accessed from the command line using the so−called manual pages `man` *`topic`*. For example:

```
man cp
```

will display the manual page for the "cp" (copy) command. The manual pages are the standard "help" system under Linux, and contain a wealth of detailed, very technical information, but typically require some effort to be understood by a newbie.

The `man` command uses a simple utility called `less` that lets you scroll through a text. Use arrow keys to scroll, press "q" to quit. Actually, `less` can do more than this. Press "h" for help when running `less`, or learn more about `less` using the command

```
man less
```

There is also the info command `info` *`topic`*. For example:

```
info cp
```

will give you the help for the "cp" (copy) command. Often `info` contains information similar to `man`, but more up−to−date. Unfortunately, the `info` navigating utility is not very intuitive, so I use man pages more often. There is also `pinfo` (a substitute for the `info` interface, perhaps easier to use than `info`).
If you don't remember exactly the name of the command that you need to use, try `apropos`. For example, to obtain a list of commands which have something to do with "copy", I execute this from the command line:

```
apropos copy
```

The command `whatis` is similar to `apropos`, but matches only keywords, whereas `apropos` searches the complete database (keywords and their description). As a result, `whatis` tends to produce a shorter (perhaps more relevant) output.

In some menu driven programs, for example when configuring your system services using `ntsysv` (or `setup`, or `linuxconf`), you may press F1 for info about what the particular service does.
The list of bash built−in commands can be obtained by typing `help` on the command line. Then help on any specific bash built−in command can be obtained by issuing, for example:

```
help cd
```

Bash is the standard command line "shell", i.e., the Linux equivalent of the DOS command−line processor "COMMAND.COM".

The KDE environment includes a GUI−based "help browser", which can be started by clicking the appropriate icon on the "Kpanel" (the system bar, normally at the bottom of the screen). This browser can be used to access the KDE−specific help as well as the system manual pages. The Gnome desktop contains a similar help system.

If you want to learn about the many packages that come on your CDs in rpm format, you may want to use the GUI−based kpackage (type `kpackage` in an X−terminal) to browse through the packages, display the info that they contain, and install them if you wish (the installation has to be done as root). In place of `kpackage`, older distributions use `glint` (RH5.2) or `gnorpm` (RH6.0), which are slower and less convenient.

2.3 Any dictionary of terms?

This one is a rather maximalist one : "The New Hackers Dictionary" aka "Jargon file": http://www.tuxedo.org/~esr/jargon/jargon.html. It is not only an excellent resource, but also highly entertaining reading. Recommended.

To add entertainment to entertainment, here is another link I like: "A Girl's Guide to Geek Guys": http://www.lairgauche.com/geekguy.html. If you are of the other sex, you might prefer: "A Guy's Guide to Geek Girls": http://www.eecis.udel.edu/~masterma/GuideToGeekGirls.html.

A rather complete list of computer−related abbreviations and acronyms is found at http://www.geocities.com/ikind_babel/babel/babel.html

2.4 Web Search

Currently, the best websearch engine is Google, amazing what you can find with it. Google is *wow* fast, because it runs on Linux, no kidding. Try: http://www.google.com/. For a test, do an egosurf (type in the search box: your last name and a word of your choice). Google can be used to find almost anything relevant to Linux (or anything else) on the net. Just type−in a few keywords to find the Linux documentation you need.

2.5 Newsgroups

This can be an intimidating place to be−−the world's strangest wackos seem to be all represented in the newsgroups. I just choose to ignore the stupid or offensive postings or e−mails. For the malicious ones, I make an exception and inform the system administrator at their originating e−mail provider. Advertisements which I receive after posting to a newsgroup get deleted before reading−−I know I am not the only one doing this, so please mark your subject line clearly if you want your e−mail to be read, particularly if your e−mail address contains the string "aol".

Despite their drawbacks, newsgroups can be a very efficient way of finding the information you need.

Before going to the newsgroups, I would highly recommend the Google news archives (http://groups.google.com/ , once known as DejaNews). This is a huge archive of newsgroup postings and you can search it using nice search tools. This way, you can often find an answer to your question without going through tons of trash, and without exposing yourself to anger after posting a question which "was already asked ten times this week". You may be surprised by the amount of information available through the google archive.

There are several newsgroups devoted to Linux and they seem much better than other newsgroups (maybe they are better policed by the Linux experts?). Here is a short list:

news:comp.os.linux.announce (moderated−−the postings are done by a moderator, who reviews them prior to the posting. Inspect the footer of any message for info on how to post.)
news:comp.os.linux.setup
news:comp.os.linux.hardware
news:comp.os.linux.security
news:comp.os.linux.misc (miscellaneous)
news:comp.os.linux.advocacy (Use this one for discussion of pros and cons of Linux and perhaps a comparison of Linux with other operating systems. This is an excellent newsgroup if you like getting into endless arguments).
news:alt.linux.sux (Here you can read/write really all opinions on Linux.)
news:comp.os.linux.networking
news:comp.os.linux.x (X−windows)
news:comp.os.unix (general UNIX newsgroup)

Please note that there is a newsgroup etiquette ("netiquette"), and you risk rejection and perhaps expose yourself to flames if you choose to break it. The major points:

- Don't post on a topic that is unrelated to the subject of the newsgroup;
- Don't post to many newsgroups at the same time (cross−post);
- Use plain ASCII, don't post attachments, pictures, html, etc.;
- Don't advertise (particularly commercial products);
- Don't write UPPER CASE ONLY − THAT GETS YOU KILL FILED.
- Use a simple descriptive subject ("HELP" isn't going to work) and briefly explain your problem. Include distribution and version number, and identify the hardware (USR 56K modem is wrong − list the model number).

To read newsgroups (also called usenet), you have to configure your access to a newsserver. The simplest may be to use mozilla ("edit"−"preferences"−"mail and newsgroups") and specifying the news server (your Internet Service Provider, ISP, should have given the name of the server) and then add the appropriate newsgroup to your list of local "mailboxes". If you don't know the name of the news server, try: "news.my_isp_provider_name.and_domain", or perhaps just "my_isp_provider_name.and_domain".

For news reading, I prefer `knode&` (type in the X terminal). Installation and learning newsgroups was certainly worth my effort.

2.6 Any Linux Internet links?

There are surely thousands of Internet sites devoted to Linux. Here are some Linux links which I like, in no particular order. If you need something else, you should find a useful pointer on one of these pages.

http://sunsite.dk/linux−newbie/	Master site for this document (LNAG). Bookmark it.
http://www.linuxdoc.org/docs.html	Linux Documentation Project−−Home for the many FAQs, Howtos, Minihowtos and Guides. Always up−to−date.
http://www.kalug.lug.net/linux−admin−FAQ/	Linux Admin FAQ (the non−Newbie).
http://members.aa.net/~swear/pedia/learning−linux.html	Gary's Encyclopedia−−Learning Linux. Bookmark it.
http://jgo.local.net/LinuxGuide/	Josh homepage. Good resource for learning Linux.
http://www.control−escape.com/	This site seems good for newbies!
http://www.linuxninja.com/linux−admin/	Linux administration made easy (LAME). Recommended.
http://metalab.unc.edu/mdw/index.html#guide	Lots of Linux documentation. Bookmark it.
http://www.frankenlinux.com	Another help site for newbies
http://www.easyfeed.com/~jgo/LinuxGuide/	Yet another newbie guide
http://www.slashdot.org/	Discussions for nerds, hackers, gurus, etc. (= /.)
http://www.freshmeat.org/	Update on today's releases of Linux software
http://linuxtoday.com/	Linux news−−excellent daily reading. Bookmark it.
http://www.llp.fu−berlin.de/	"The Linux Lab Project." Data acquisition and other interesting material for those in science.
http://www.linuxberg.com/	Linuxberg. Big portal. They have everything there. I like their rating of Linux software and am installing only packages that received 5 penguins ;−) . Bookmark it.
http://counter.li.org/linuxcounter_eng.html	The Linux counter. Register yourself as a linuxer!
http://www.ap.univie.ac.at/users/havlik/Album/Linux−Counter/	See Dennis Havlik's impressive maps on Linux growth and geographical distribution.
http://www.cl.cam.ac.uk/users/iwj10/linux−faq/index.html	Linux FAQ.
ftp://sunsite.unc.edu/pub/Linux/	Tons of Linux software at the Sunsite archive. Bookmark it.
http://stommel.tamu.edu/~baum/linuxlist/linuxlist/linuxlist.html	Linux applications.
http://www.boutell.com/lsm/	Linux applications.
http://www.linuxlinks.com/Software/	Linux applications.
http://directorysearch.mozilla.org/Computers/Operating_Systems/Linux/	Great new portal (better than yahoo) with excellent links for Linux newbies.
http://dir.yahoo.com/.../Unix/Linux/	Yahoo's entries for Linux. Looks very corporate−they refuse to add this guide!
http://www.debian.org/	Debian Linux site.
http://hardware.redhat.com/hcl/genpage2.cgi	Linux hardware compatibility list.
http://metalab.unc.edu/mdw/links.html	Lots of useful Linux links
http://www.cse.unsw.edu.au/~conradp/linux/	Scores of excellent links.
http://www.linuxstart.com/documentation/	More links to Linux documentation.
http://www.linuxlinks.com/	Even more Linux links.
http://www.gnu.org/	Master GNU site (GNU's−Not−Unix. This is a recursive definition).
http://www.redhat.com/	The Red Hat site. It is typically too busy to bother.
http://www.cs.Helsinki.FI/u/torvalds/	Linus Torvalds home page.

2.7 Source code−−the ultimate resource

The ultimate reference under Linux is the source code. If you installed it (comes with standard distributions; we really recommend its installation if you have enough hard drive space), it is in `/usr/src/linux`(the kernel source) and `/usr/src/RPM/sources`(the source code for the balance of the rpm packages). How can the source code be of use to a newbie? Well, it contains all the comments and documentation down to the smallest detail. For example, later in this guide, we show how to read/set up some of the kernel runtime parameters via the /proc filesystem. You can read the complete documentation for all the available parameters using:

```
less /usr/src/linux/Documentation/proc.txt
```

To install kernel sources, I would select the appropriate rpm package during my main installation. To install sources for other packages that came with my distribution, I would put the "Source CD" into the cd drive and do something like (as root, with RedHat CD):

[install the source code for the gnumeric spreadsheet from the cd to the harddrive]
```
su
cd /mnt/cdrom/SRPMS/
rpm -ivh gnume<Tab>
```
[unzip the sourcecode which I just installed]
```
cd /usr/src/RPM/SOUR<Tab>
tar -xvzf gnumer<Tab>
```
[read the code for statistical functions in gnumeric]
```
cd gnumeric/src/functions
less fn-stat.c
```

This is truly the ultimate reference on how a particular spreadsheet function works, no kidding.

Go to part 3: Basic Operations FAQ
Back to the main page

Part 3: Basic Operations FAQ

LINUX NEWBIE ADMINISTRATOR GUIDE

ver. 0.193 2002−12−14 by Stan, Peter and Marie Klimas

The latest version of this guide is available at http://sunsite.dk/linux−newbie.

Copyright (c) by Peter and Stan Klimas. Your feedback, comments, corrections, and improvements are appreciated. Send them to linux_nag@canada.com This material may be distributed only subject to the terms and conditions set forth in the Open Publication License, v1.0, 8 or later http://opencontent.org/openpub/ with the modification noted in lnag_licence.html.

Contents of this section:

3.1 Basics
- 3.1.1 Filenames
- 3.1.2 What are the different directories for?
- 3.1.3 How do I run a program?
- 3.1.4 How can I change the PATH?
- 3.1.5 How can I shutdown my computer?
- 3.1.6 How do I deal with a hanged program?
- 3.1.7 Command options

3.2 Users, passwords, file permissions, and security
- 3.2.1 Home directories, root, adding user
- 3.2.2 About password security
- 3.2.3 I forgot the root password
- 3.2.4 I forgot my user password
- 3.2.5 Disabling or removing a user account
- 3.2.6 I have file permission problems. How do file ownership and permissions work?
- 3.2.7 My mp3 player chokes. The sound is kind of interrupted (how to set suid)

3.3 Job scheduling with "at", "batch", and cron
- 3.3.1 How do I execute a command in the "background"?
- 3.3.2 How do I execute a command at a specified time (using "at"or "batch")?
- 3.3.3 How do I set up cron?

3.4 Shell
- 3.4.1 What's a shell and do I want to use a different one?
- 3.4.2 How do I customize my shell prompt?
- 3.4.3 Colour in text terminal
- 3.4.4 How do I print symbols on the console, in a text mode application, or in X?
- 3.4.5 How do I write a simple shell script?
- 3.4.6 Meaning of quotes
- 3.4.7 Input/output redirection
- 3.4.8 Shell special characters (metacharacters)

3.5 Package installation and rpm package manager
- 3.5.1 How do I install a program I downloaded from the Internet?

3.1 Basics

3.1.1 Filenames

Linux is case−sensitive. For example: `myfile`, `Myfile`, and `myFILE` are three different files. Your password and login name are also case−sensitive. (This follows tradition since both UNIX and the "c" programming language are case−sensitive.) Naming conventions for files and directories are identical. All the files and directories which I create (for myself, as a user) are lower−case, unless there is a very special reason to make it different. Most of Linux commands are also all lower case.

Filenames under Linux can be up to 256 characters long and they normally contain letters, numbers, "." (dots), "_" (underscores) and "−" (dashes). Other characters are possible but not recommended. In particular, it is not recommended to use special metacharacters: "*" (asterisk), "?" (question mark), " " (space), "$" (dollar sign), "&" (ampersand), any brackets, etc. This is because metacharacters have special meaning to the Linux shell (shell is something like `COMMAND.COM`, the command processor under DOS). It is possible to have a space in the filename, but we don't recommend it either−−we use underscore "_" instead.

It is not possible at all to have '/' (slash) as a part of the filename because '/' is used to represent the top of the directory tree, and as a separator in the pathnames (the same as '\' is in DOS).

Like in DOS, I cannot have a file called `.` or a file called `..` (dot or two dots)−−they mean "current" and "parent to the current" directory respectively, exactly like in DOS.

Here is the meaning of some metacharacters:
* = Matches any sequence of zero or more characters, except for "." (a dot) at the beginning of a filename.
? = Matches any single character.
[abC1] = Matches a single character in the enumerated set. In this example the set contains: 'a', 'b', 'C', and '1'.
[a−z] = Matches any lower−case letter.
[A−F] = Matches any upper−case letter from A to F.
[0−9] = Matches any single digit.
[a−zA−Z0−9] = Matches any letter (lower or upper case) or any digit.

The character \ (backslash) is also special. It makes the subsequent special character aquire literal meaning (read on).

Examples. This command will list any filename in the current directory, with the exception of filenames starting with "." (dot):

```
ls *
```

An equivalent to this command is to type just `ls` or `dir` (without the "*"). Files with names starting with "." are not shown because "." as the first character of a filename is not matched by "*". Think of files with names starting with "." as an equivalent of DOS hidden files. Use `ls −a` (list with the option "all") or `ls .*` to see these "dot" files. The "dot−files" are common in the user home directories and they typically contain user−level configurations.
This command will list any file (in the current directory) that contains a dot (except files starting with a dot):

```
ls *.*
```

This command will list any filename that contains two dots (except those starting with a dot):

```
ls *.*.*
```

Please note that Linux does not have "filename extensions" the way DOS does, but you can still use them. For example, I can have a file my_text.txt.zip. Some other DOS−kind file−naming features are completely absent ("Micros~1.doc" comes to mind).
This command will find (on the whole filesystem) any file with the extension "htm" optionally followed by any one more character:

```
locate *.htm?
```

This command will show all filenames in the current directory that start with "a" or "b", or any capital letter:

```
ls [abA−Z]*
```

This command will list any file starting with "a" and ending with "n"

```
ls a*n
```

Command line autocompletion. This is a great command line feature−−I use the [Tab] key a lot to save on typing. It makes it brisk to deal with long and complicated filenames. For example using such a filename on the command line is really not a problems, if I use autocompletion:

```
dir Eurosong\ 2000\ Olson\ Brothers\ −\ Fly\ on\ the\ wings\ of\ love\ \(denmark\).mp3
```

I just type

```
dir Eu<Tab>
```

and if there are no other files starting with "Eu", the rest of the filename is automatically typed for me. Otherwise, I would have to look at my choices (which are printed for me) and type one or two more characters to make the filename unambiguous. The backslashes in the name of the example song above show that the spaces are "literal", i.e., they spaces are part of the filename.

Problems with weird filenames. Most of these problems can be solved using autocompletion. Additionally, to manipulate files with names that do contain metacharacters, I may use a pair of '' (two apostrophes), so that the metacharacters are quoted and therefore the shell does not interpret their meaning. For example, to rename a file `my file*` (contains space and asterisk), I would issue:

```
mv 'my file*' filename_without_weird_characters.txt
```

Please note that I use a pair of ' (apostrophes) for quoting. Quoting with a pair of " " (quotation marks) is generally weaker than quoting with ' ' . If you use " (quotation marks) some metacharacters may get interpreted by the shell (altering their meaning).

Following UNIX tradition, on Linux, one may create files with names contaning almost any character, including non−printable (control) characters. Those are very infrequent, but if you encounter such a file, it can make you feel really weird. I would rename such a file using a carefully positioned metacharacter. I would use ls first to try if my action indeed targets the desired file, and then rename the file (using the move "mv" command):

```
ls −l myfile*y.html
mv myfile*y.html myfile.html
```

(I assume that the non−standard character(s) are between the letters e and y.)

As an example of the perhaps weirdest problems that you might face when using non−recommended characters in a filename, try creating a file with a name starting with a dash and then remove it−−there seems to be no way to do it (because a dash normally introduces command options). E.g., the command

```
dir > −junk
```

will create such a funny file (like in DOS, the symbol ">" redirects the output from the `dir` command to a file named "`−junk`"). Since the regular way of removing the file `−junk` does not work, I use:

```
rm ./−junk
```

The "dot slash" at the beginning means "the current directory" and here serves just the purpose of hiding the leading dash so it is not interpreted as introducing an option to the `rm` command. The point here is that I would rather stick to traditional naming conventions than face the occasional complications.

Besides using autocompletion, apostrophes and quotes, I can manipulate files with weird names using \ (backslash). Backslash hides the special meaning of the subsequent character. For example, i can create a weird file with the name `*?[` using the following command:

```
touch \*\?\[
```

(The `touch` command creates an empty file or, if the file exists, updates its date/time of last modification.)

3.1.2 What are the different directories for?

Linux filesystem tree is large and complicated. It will vastly improve your skills if you familiarize yourself with it.

Briefly, typical Linux contains five filesystems. These filesystems can reside on a single or different physical hard drives and/or hard drive partitions, depending on the size and need of your system. (A single filesystem can also be distributed between different physical devices, if needed.)

The root "`/`" filesystem contains basic operating system and maintenance tools. The content of this filesystem should be sufficient to start up the system and perform emergency maintenance and repairs if they were necessary.
`/usr` filesystem contains all commands, libraries, documentation, and other files that do not change during normal operation. This will also contain major applications that come with your Linux distribution, for example Netscape.
`/var` filesystem contains files that change: spool directories, log files, lock files, temporary files, and formatted (on use) manual pages.
`/home` filesystem contains user files (users' own settings, customization files, documents, data, mail, caches, etc). The contents of this directory should be preserved on an operating system upgrade.
`/proc` filesystem contains entirely illusionary files. They don't really exist on the disk and don't take up any space there (although `ls −l` will show their size). When viewing them, you really access information stored in the memory. It is used to access information about the system.

The parts of the **root filesystem** are:
`/bin`−−executables (binaries) needed during bootup that might be used by normal users.
`/sbin`−−executables (system binaries) not intended for use by general users (users may still use them, but this directory is not on their PATH).
`/etc`−−system−wide configuration files for your operating system.
`/root`−−the home directory of the system administrator (called super−user or root).
`/dev`−−device files. Devices appear on Linux as files so that hardware is abstracted and it is easy to write to them or read from them.

`/mnt`−−mount points for removable media (floppy, cdrom, zipdrive), partitions of other operating systems (e.g. MS Windows), network shares, and anything else that is mounted on the file system temporarily. It normally contains a separate subdirectory for each mounting share. The contents of these drives/shares appear in these subdirectories−−there are no drive letters on Linux.
`/lib`−−shared libraries for programs that reside on the root filesystem and kernel modules.
`/boot`−−files used by the bootstrap loader (LILO or GRUB), the thing that loads first when the computer is booted and perhaps gives you the option of which operating system to boot, if you have more than one OS on your computer). It typically also contains the Linux kernel (compressed, file `vmlinuz)`, but this can be stored somewhere else, if only LILO is configured to know where it is.
`/opt`−−optional large applications, for example kde under RedHat 5.2 (under RedHat 6.0, kde is distributed as any other X−windows distribution, main executables are in the `/usr/bin` directory).
`/tmp`−−temporary files. This directory may clean up automatically.
`/lost+found`−−files recovered during the filesystem repair.

The most interesting parts of the **/usr filesystem** are:
`/usr/X11R6`−−X−windows system (version 11, release 6).
`/usr/X11`−−the same as /usr/X11R6 (it is a symbolic link to /usr/X11R6).
`/usr/X11R6/bin` −−lots of small X−windows apps, and perhaps symbolic links to the executables of some larger X−windows applications that reside in their own subdirectories somewhere else).
`/usr/doc`−−Linux documentation (on newer systems, this moved to /usr/share/doc).
`/usr/share` −−Data independent from your computer architecture, e.g., dictionary words.
`/usr/bin` and `/usr/sbin`−−similar to their equivalents on the root filesystem (`/bin` and `/sbin`), but not needed for basic bootup (e.g. during emergency maintenance). Most commands will reside here.
`/usr/local`−−the applications installed by the local administrator (perhaps each application in a separate subdirectory). After the "main" installation, this directory is empty. The contents of this directory should survive normal re−installation or upgrade of the operating system.
`/usr/local/bin`−−perhaps smaller "user"−installed executables, plus symbolic links to the larger executables contained in separate subdirectories under `/usr/local` .

It is important to understand that all directories appear in a single directory tree, even if the directories are contained on different partitions, physical drives (including floppies, etc), or even if they are distributed over the network. Therefore, there are no DOS−type "drive letters" under Linux. What would be a "drive" under DOS or MS Windows, appears on Linux as a subdirectory in a special "mounting" location.

The directory system is well−established and standard on most Linux distributions (the small differences are being currently addressed by the Linux Standard Base). It is also quite similar to that found on typical commercial UNIX systems.

To summarize:

- Users always save their files to the directory `/home/user_login_name` (and its subdirctories)
- The local administrator most likely installs the "additional" software under the directory `/usr/local` and makes a link to the main executable in /usr/local/bin.
- System settings are all in the directory /etc.
- It is not a good idea to temper with the content of the root directory ("/") or of the directory `/usr`, unless I really know what I want. These directories are best left as they came with my Linux distribution.
- Most tools and applications installed on my system are in the directories: /bin, /sbin, /usr/bin, /sbin, /usr/X11/bin, /usr/local/bin.
- All the files are in single directory tree. There are no drive letters.

More about the `/proc` filesystem (only for really curious).

The `/proc` "pseudo" file system is a real−time, memory−resident file system that tracks the state of the operating system kernel and the processes running on your computer. The `/proc` file system is totally virtual, i.e., it is not written on any particular disk or other persistent media, it exists only in the computer memory, and it is constantly updated to reflect any changes to your system. The size of the `/proc` directory is always zero and the last modification time is the current date. In some cases, it is possible to change your system settings by manually changing the contents of files in the `/proc` filesystem. Many Linux utilities use the `/proc` filesystem as the source of their information, e.g., `dmesg`, `ps`, `top`.

Contents of the `/proc` filesystem.

Directories with numerical names like "1" "170" "4908" are IDs of the processes running on your computer. Each directory contains several files, e.g.,: `cmdline` (contains the entire command line that was used to envoke the process), `cwd` (symbolic link to the cwd of the process), `environ` (the environment variables defined for this particular process in the form VARIABLE=value), `exe` (a symbolic link to the executable file that the current process is linked to), `fd` (a list of the file descriptors opened by the process),`maps` (a named pipe that can be used to access the process memory), `root` (a symbolic link to the directory which is the root file system for the particular process), `stat` (info on the status of the process).

Other files in the `/proc` filesystem:
`/proc/cpuinfo` −−information about the processor, such as its type, make, model, and performance.
`/proc/devices` −−list of device drivers configured into the currently running kernel.

`/proc/dma` −−DMA channels being used at the moment.
`/proc/filesystems` −−filesystem types configured into the kernel.
`/proc/interrupts` −−interrupts in use, and how many of each there have been.
`/proc/ioports` −−I/O ports in use at the moment.

For example, I can read the cpu info on my system using the following command:
`cat /proc/cpuinfo`

3.1.3 How do I run a program?

Typing the name of the executable on the command line doesn't help? There are three possibilities.

The first possibility: I did not type the name of the executable correctly. Check the case−−Linux is case sensitive! For example, typing "Pico" or "PICO" will not start the `pico` editor.

The second possibility: maybe the program is not on my PATH? Under Linux (or any UNIX), an executable must be on your PATH to run it, and the current directory is not on my PATH. Type the full path to the executable in front of the executable name, or do:

```
cd the_program_directory
./program_name
```

I must put the dot and slash in front of the program name or the program will not execute. (This is a security feature not to put one's current directory on the path. It makes "trojan horses" more difficult. A "trojan horse" is a malicious program that pretends to be something different than it really is.) The dot means "the current directory", and the slash "/" is a separator between the directory name and the filename (exactly as "\" in DOS).
I may check my path using:

```
echo $PATH
```

To learn how to change your PATH, or add your current directory to it, see the next answer.
If my executable is lost somewhere in the directory tree, I may want to find it using (for example):

```
find / −name "netscape"
```

to find a file named "netscape", starting the search from the root directory "/". You may be able to achieve the same result faster using:

```
locate netscape
```

(Locate runs faster because it relies on a pre−built database of files on your system. This database is updated by a background "cron" process that normally runs at night, so don't count on `locate` to find a file if you regularly switch off your computer for the night, or you are searching for a file that you have just installed.)
Please note that the PATH is normally different for root than for the regular users (root's PATH includes `/sbin` and `/usr/sbin` whereas users' don't). Therefore users cannot execute commands located in the "sbin" directories unless they specify the full path to the command. Also, if you become a superuser by executing the `su` command, you inherit the user's PATH, and to execute the command located in sbin, you need to specify the full path.
Conversely, if I need to learn where an executable which is on my PATH is located on your system (i.e., the executable runs by typing its name anywhere in the system, but I would like to know where it is located), I may use something like this:

```
which netscape
```

which will show the full PATH to the executable program called "netscape" (if one exists).

The third possibility: maybe the file is not executable. If it should be, change the permissions to make it executable. E.g. (as root or the user who owns the file):

```
chmod a+x my_file
```

will make the file "my_file" executable for all users. Check if it worked using:

```
ls −l  my_file
```

Read here if you don't understand the output of this command or the whole "third possibility".

Please note that under Linux (or UNIX), the file extension (for example .exe or .com or .bat) does not make the file executable. The file needs an "executable file access mode" which is not unlike a "file attribute" under DOS.

3.1.4 How can I change the PATH?

Typically, you don't have to change your PATH, but it very useful to understand what PATH is.

The PATH is the list of directories which are searched when you request the execution of a program. You can check your PATH using this command:

```
echo $PATH
```

which, on my system , shows the PATH for the user "yogin" to be:

```
/opt/kde/bin:/usr/local/bin:/bin:/usr/bin:/usr/X11R6/bin:/home/yogin/bin
```

The ":" is a separator, therefore the above PATH represents a list of directories as follows:

```
/opt/kde/bin
/usr/local/bin
/bin
/usr/bin
/usr/X11R6/bin
/home/yogin/bin
```

Here is the output from the command "echo $PATH" run on my system on the account "root":

```
/opt/kde/bin:/sbin:/bin:/usr/sbin:/usr/bin:/usr/X11R6/bin:/root/bin
```

You can change the PATH for all users on the system by editing the file `/etc/profile` and adjusting (as root) the line starting with "PATH=". I do it using the `pico` editor (as root):

```
pico −w /etc/profile
```

(The option −w turns off the wrap of long lines.)
Re−login for the change to take effect. To set up the PATH for an individual user only, edit the file `/home/user_login_name/.bash_profile` (please note the dot in front of the filename−−files starting with a dot are normally invisible, you have to use `ls −a` to see them).
If you really want to have the current directory on your PATH, add "." (dot) to your PATH. When used in the place when directory name is expected, a dot means "the current directory". The specification for the path in `/etc/.bash_profile` may then look like this:

```
PATH="$PATH:$HOME/bin:."
export PATH
```

This command takes the contents of the environmental variable called PATH (as set for all users in `/etc/profile`), and appends to it the name of your home directory as set by the variable HOME with an attached "/bin" and then a dot. Finally, the command assigns the resulting string back to the variable called PATH. It is necessary to use the command "export" after modifying PATH or any other user−environment variable, so that the variable is visible outside of the script that sets it.

3.1.5 How can I shutdown my computer?

Close all your programs saving the data as desired. From your GUI main menu (e.g., "K"), select "Logout". Then, from the logon screen, select: "System"−"Shutdown".

Alternatively, from a text terminal, press `<Ctrl><Alt><Del>` (the "three−finger salute", you press the three keys simultaneously), wait for the shutdown process to complete, and turn off your machine only after it starts rebooting again. If you are in X−windows, first switch to a text terminal by pressing `<Ctr><Alt><F1>` (three keys simultaneously).

Never turn off your machine without the proper shutdown or else you may have disk error messages next time you boot. (Typically, the errors resulting from improper shutdown will be repaired automatically during the next boot, but occasionally more serious problem may result, and then you may need to repair the files manually or re−install!)

If you prefer your computer to go to a halt after you press <Ctrl><Alt><Del> (instead of the default reboot), you can set this up by editing the file /etc/inittab. This file specifies something like this:

```
# Trap CTRL−ALT−DELETE
ca::ctrlaltdel:/sbin/shutdown −t3 −r now
```

As root, replace the option "−r" to "−h" so that the same fragment reads:

```
# Trap CTRL−ALT−DELETE
ca::ctrlaltdel:/sbin/shutdown −t3 −h now
```

The line starting with "#" is just a comment (it is for humans, it does not have any effect on the computer). The option "−t3" tells the shutdown command to wait 3 seconds before it starts killing processes. The options "−r" and "−h" stand for "reboot" and "halt" respectively, so they perform a shutdown to reboot or a shutdown to a system halt.

Root can also use the shutdown command directly. This command can be used for either local or remote shutdown of your computer, but is used mostly for remote shutdown when the local keyboard is not available so you cannot use <Ctrl><Alt><Del>. It can also be very useful if a program hangs so that the keyboard is no longer functional. For example:

```
telnet name_of_machine_with_no_operable_keyboard
```
[login as a user]
```
su
```
[give password]
Now either execute `ps axu |more,` find the process id of the offending command in the ps output and do

```
kill pid_of_offending_process
```

or reboot your machine with:

```
/sbin/shutdown −rn now
```

This command will shutdown really fast, bypassing standard (longer) shutdown procedure−−useful when the system becomes really buggy (the option −n will make "shutdown" kill all the processes before rebooting).

Please note that for security reasons, you cannot login to a remote machine as root (e.g., over the telnet). You have to login as a user and then execute su and give a password to become a super user (root).

The shutdown command may also be used to execute a shutdown later. E.g. (as root):

```
/sbin/shutdown −r 23:59
```

will reboot the system 1 minute before midnight. I could also use:
```
/sbin/shutdown −r +1
```

to shutdown 1 minute from now. I can cancel a scheduled shutdown with:

```
/sbin/shutdown −c
```

If the shutdown command is too long for you, you may want to try these two commands, which do exactly what their names suggest (as root):

```
reboot
halt
```

A fancy way to shut down your computer is to switch your system to the runlevel 0 (for halt) or runlevel 6 (for reboot). Try it using (as root):

```
init 0
```

The meaning of the different runlevels is explained in the file /etc/inittab and here.

3.1.6 How do I deal with a hanged program?

Buggy programs do hang under Linux. A crash of an application should not, however, affect the operating system itself so it should not be too often that you have to reboot your computer. Linux servers are known to run for more than a year without a reboot. In our experience, a misbehaving operating system may be a sign of hardware or configuration problems: we repeatedly encountered problems with the Pentium processor overheating (the fan on the Pentium did not turn as fast as it should or it stopped altogether, the heat sink on the Pentium was plugged with dirt), bad memory chips, different timing of different memory chips (you may try re−arranging the order of the chips, it might help), wrong BIOS setup (you should probably turn off all the "advanced" options, Linux takes care of things by itself). The "signal 11" error message is typically (99%) associated with hardware problems and is most likely to manifest itself when you perform computing−intensive tasks: Linux setup, kernel compilation, etc. If your Pentium has the tendency to overheat (very common for early Pentiums), here are some tips to keep it cool, particulary during hot weather: clean the processor heat sink, replace the processor fan, operate the computer with the cover off and aim an extra fan inside, increase the processor "wait−state" in the computer BIOS, don't overclock, decrease useless load, e.g., replace this super−fancy screen saver with a blank screen.
Not really hanged. Some programs might give the uninitiated impression of hanging, although in reality they just wait for user input. Typically, this happens if a program expects an input filename as a command line argument and no input filename is given by the user, so the program defaults to the standard input (which is console). For example, this command

```
cat
```

may look like it's hanged but it waits for keyboard input. Try pressing <Ctrl>d (which means "end−of−file") to see that this will satisfy the `cat` command. Another example: I have seen many questions on the newsgroups about the "buggy" `tar` command that "hangs" when trying to uncompress a downloaded file, for example:

`tar −zxv my_tar_file` [wrong!]

This waits for user input too, since no option "−f filename" was specified so "my_tar_file" was not recognized as a filename. The correct command is:

```
tar −zxvf my_tar_filename
```

Please note that the filename must follow immediately after the option "f" (which stands for "filename). This WILL NOT work (very common mistake):
`tar −zxfv my_tar_file` [wrong!]

Any program (hanged or not) can be killed.

A text−mode program in the foreground can often be killed by pressing <Ctrl>c. This will not work for larger applications which block the <Ctrl>c, so it is not used on them accidentally. Still you can get back in control either by sending the program to the background by pressing <Ctrl>z (no guarantee this will work) or switching to a different terminal, for example using <Ctrl><Alt><F2> and login as the same user that hanged the program (this should always work). Once you are back in control, find the program you want to terminate, for example:

```
ps
```

This command stands for "print status" and shows the list of programs that are currently being run by the current user. In the ps output, I find the process id (PID) of the program that hanged, and now I can kill it. For example:

```
kill 123
```

will kill the program with the process id (PID) of "123".
As user, I can only kill the processes I own (this is, the ones which I started). The root can kill any process. To see the complete list of all processes running on the system issue:

```
ps axu | more
```

This lists all the processes currently running (option "a"), even those without the controlling terminal (option "x"), and together with the login name of the user that owns each process ("u"). Since the display is likely to be longer than one screen, I used the "more" pipe so that the display stops after each screenful.

The `kill` command has a shortcut `killall` to kill programs by name, for example:

```
killall netscape
```

will kill any program with "netscape" in its name, while

```
killall pppd
```

will surely disconnect any dial−up connection by killing the ppp daemon.

X−windows−based programs have no control terminals and may be easiest to kill using this (typed in an X−terminal):

```
xkill
```

to which the cursor changes into something looking like a death sentence; you point onto the window of the program to kill and press the left mouse button; the window disappears for good, and the associated program is terminated.

A shortcut to the last command is to press <Ctrl><Alt><Esc>, to which the cursor changes into something looking like a death sentence−−you point at the window of the offending program, click your mouse, and the window closes and the program is gone.

If your X−windows system crashes so that it cannot recover, or you just get stuck, it may be the easiest to kill the X−server by pressing <Ctrl><Alt><BkSpace>. After that, it might be a good idea to run `ps axu`, find any possible X−programs that might still be running, and kill them. If you don't do this, and there really is a misbehaving program that caused your X−windows to crash, it might cause trouble again.
If you have programs in the background, the operating systems will object your logging out, and issue a message like "There are stopped jobs". To override and logout anyway, just repeat the `logout` (or `exit`) command −−the background program(s) will be automatically terminated and you will be logged out.
Core files. When a program crashes, it often dumps a "core" into your home directory. This is accompanied by an appropriate message. A core is a memory image (plus debugging info) and is meant to be a debugging tool. If you are a user who does not intend to debug the program, you may simply delete the core:

```
rm core
```

or do nothing (the core will be overwritten when another core is ever dumped). You can also disable dumping the core using the command:

```
ulimit −c 0
```

Checked if it worked using:

```
ulimit −a
```

(This shows "user limits", the option "−a" stands for "all".) To make the option of disabling core dumps permanent for all users, edit the file `/etc/profile` (as root), where `ulimit` is set, and adjust the setting. Re−login for the changes to `/etc/profile` to take effect.
If you would like to see how a core file can be used, try (in the directory where you have a core file):

```
gdb −c core
```

This launches GNU debugger (gdb) on the core file "core" and displays the name of the program that created the core, signal on which the program was terminated, etc. Type "quit" to exit the debugger. To learn the meaning of different signals, try:

```
cat /usr/include/bits/signum.h |more
```

3.1.7 Command options

Most commands accept numerous "options". An option can be introduced with an "−" (dash). For example:

```
dir −l
```

shows me the listing of the current directory but in a long format (the default format is "short"). Multiple options can be introduced in two, equivalent ways:

```
dir −l −a
```

or

```
dir −la
```

Either of the above commands will show me the listing of the current directory in the long file format (option −l), and include all files in the listing, i.e., also the hidden files (option −a).

Most popular options are marked with one letter. This follows the traditional UNIX way of invoking options. There is also a new style, which looks like this:

```
dir −−help
```

Here, a single option is more than one character long, and it must be introduced with two dashes. The above command displays a brief help for the dir command, including the listing of all options. Because there are so many of those (more than a screenful), I would probably do:

```
dir −−help | more
```

3.2 Users, passwords, file permissions, and security

3.2.1 Home directories, root, adding users

The (almost) only place on the harddrive that normal users (non−root) can write to is their home directory, which is `/home/user_login_name`.

This "home" directory is for all user files: settings, program configuration files, documents, data, netscape cache, mail, etc. As a user, you can create subdirectories under your home directory to keep yourself organized. Other users cannot read your files or write to your home directory unless you give them permission to do so.
Normal users can also see, read and execute many other files on the system (besides their home directory), but normally they cannot modify or remove (delete) them.

The "root" (also called "super user") is a special administrative account that has the power to modify any file on the system. It is not a good idea to habitually work on your system as root−−if you do so, your mistakes can cost you dearly. Set up and use a normal user account for everyday work for yourself, another user account for your son, and yet another for your wife. The root account is typically the only account that exists on Linux after the initial installation. Thus you have to explicitly create "user" accounts for normal work for you Linux system.

A user account can be created by "root" using, for example:

```
adduser joe
passwd joe
[type the password for the user joe]
[retype the password for the user joe so as to avoid mistakes]
```

In the example above, first I logged in as root. Then, on the command line, I issued the command "adduser" with the parameter (argument) "joe". This created the account "joe" on my Linux computer. Then, I issued the command "passwd joe" to change the password for the user "joe" to something fairly secure. Now, I can tell "joe" what her initial password is, and she can login and change the password to her liking. Please note that the account name (user login name, "joe") and the password are case−sensitive.

Root can change any user's password, although s/he cannot read it. [Passwords are encrypted using a one−way encryption algorithm and only this encrypted version is stored on the system, in the file `/etc/passwd` (older systems) or `/etc/shadow` (newer systems), and the "open" version of the password is never stored. When you login, the password you type is encrypted again using the same one−way algorithm and compared with the already encrypted version stored in `/etc/passwd` or `/etc/shadow`.]

The separation of the administrator and user makes Linux systems secure and robust−−it even makes viruses under Linux difficult (the programs that a user runs can write only to his/her own directories, and therefore cannot affect the vital parts of the operating system). It is customary that the user changes his/her password immediately after the first login, for example:

```
passwd
(current) UNIX password: pass_OLD
New UNIX password: pass_NEW
Retype New UNIX password: pass_NEW
```

In reality, the password will not appear on the screen as you type it (for security reasons). Take your time if you are changing the password for the very first time−−it can be difficult to type "blind".

On the Linux system, the same password is used to:

- login on the text terminal,
- login from a graphical (GUI) screen into your desktop (KDE or GNOME),
- unlock a locked text terminal,
- unlock a password−protected screen saver on a GUI (for example, KDE or GNOME).

3.2.2 About password security

Weak passwords are probably the most common source of security problems. Even at home, you may expose yourself to serious trouble because somebody may be able to hack your computer when you browse the Internet and read/delete your files, or use your computer to do something really nasty to the local police computer network. Therefore, keep all your login names/passwords secure, even at home. Once somebody logs into your computer (even as an ordinary user), he may find it quite easy to gain root access (depending on how well−maintained/up−to−date your system is vs. how good a hacker s/he is).

Here are some examples of hazardous passwords:
− No password (possible!).
− The word "password" (wow, this one is really weak!).
− Your login name (The login and the password the same? Hmm.).
− Your first name or the first name of your daughter, son, husband, wife, girlfriend, or any other first name. The number of first names in use is quite limited−−just check the paperback book "what to name your baby". Don't assume that a first name you think of is secure because you are from India−−Canada is really a multinational society and the typical namelist seems to cover all kinds of first names.
− Your last name or any other last name. The number of last names is surprisingly limited! Just check the US census data to see that your "rare" last name from the abamamahaba island is very well represented in the US 89,000 of the most frequent last names (e.g., http://www.census.gov/genealogy/www/freqnames.html). Or just check the Toronto telephone book. Another proof that we are all one family :))
− The nickname of your dog, wife, canary or computer. (Very few nick names humans use, much fewer than last names!)
− Name of your favourite sports team, celebrity, toothpaste, or detergent. Avoid names of popular soccer teams like fire. Same with rock bands (music).
− Date of your birth, social security number, etc; Sequences of digits can be easily probed.
− Name of your company, department, workgroup, etc.
− Password written in the calendar on your desk or on the side of your computer.
− A password which you also use in an insecure public place, for example an Internet store or a mailing list. In general, you should use different passwords for places controlled by different organizations.
− Any word which is in the English dictionary. The English dictionary does not contain as many words as it might seem. A not−so−skillful hacker can easily set a program to encrypt all dictionary words (100,000? that's under 1 MB!) and then compare all the encrypted strings to your encrypted password. As a matter of fact, tools for the "dictionary attack" are readily available on the Internet. Try the program `crack` yourself to find how easy it is. Swear words or "cool" (colloquial) expressions make the password particularly vulnerable for cracking.
− Any other word, last name, first name, pet or swear word, no matter in what language. For a cracker, to cover most languages is only a small overhead if he already covered one. How many significant languages are out there? 40? The cracker just grabs a few more files and appends it to his cracking list. The point here is that the subset of words that humans normally use if far far below the theoretical limit of the random combination of characters.
− Any of the above with an addition of a number/letter at the beginning or the end. "yuoping1" is really a very weak password.

A good password is relatively long (minimum 6 characters, some experts even recommend minimum 10 characters), contains a mixture of letters (upper and lower case, if possible), numbers and special characters, and is changed quite regularly (8−16 weeks?).

Unfortunately, the better the password, the harder it is to remember. I solved this problem for myself by taking 10 minutes to invent my personal password "scheme". Say, I always start and end with the monkey (@) sign, and use two words connected with an exclamation mark, the last letter of each word is capitalized, e.g., "@whitE!housE@". Seems like an adequate password, and it is easy to remember once I know what my password rule is. If you are a memory genius, you may consider truly excellent passwords generated with `mkpasswd` :))

The system administrator can set the password policy (minimum length, requirement of special characters, password expiry) through the utility included in this configuration program (run as root):

```
linuxconf
```

under the menu "user account"−"policies"−"password & account policies". Normal users won't be able to set a password which is too short, is a dictionary word, or does not contain the prescribed number of non−alphanumeric characters (but root can change any password to anything s/he likes, s/he will only be given a warning).
Also make sure that any file that contains any password of yours (e.g., `/root/.kde/share/config/kppprc`) has proper,

secure permissions so that it cannot be read by anybody. For example, most likely you want:

```
chmod 600 kppprc
```

If you use an "over the phone" Internet connection for just a couple of hours a week, you may be fine even with a relatively weak password on your system. But please really reconsider your system security if you use a cable modem, or are otherwise connected to the Internet for a significant amount of time.

Most computer semi−literate use amazingly weak passwords. "Around 50 percent of computer users ***base*** passwords on the name of a family member, partner or a pet. Thirty percent look to a pop idol or sporting hero," reports CNN (http://www.cnn.com/2002/TECH/ptech/03/13/dangerous.passwords/index.htmll). Please note the underlined ***base***. Appending a digit to an obvious word hardly makes the password more secure.

3.2.3 I forgot the root password

Even if I never forget any passwords, I would still study this issue in detail because it can give me a hint on how my mother might be reading my ICQ chats history :−)
First method. The easiest way to solve your "forgotten root password" problem is to boot your Linux in the single−user mode, namely at the "lilo"prompt (during bootup) type:

```
linux single
```

This will make you "root" without asking for a password. Now, being root, you may change the root password using this command (no knowledge of the old password required):

```
passwd
```

If it strikes you as insecure, that's because no computer system is secure if other people have physical access to your hardware. Nevertheless, I did not like the "linux single" hole on my home computer and plugged it by adding the following lines to my `/etc/lilo.conf` file (at the end of the "image=" section):

```
password="my_password"
restricted
```

[This "lilo" password is required when, at the LILO prompt during bootup, somebody enters the word "linux" with any parameter (normal bootup without any parameters will still be possible without a password).] For the changes to `/etc/lilo.conf` to take effect, I must re−run the command `lilo` . Since my lilo password is not encrypted, I must make `/etc/lilo.conf` readable only for root:

```
chmod 600 /etc/lilo.conf
```

Second Method. Another way to solve the "lost−root−password" problem is to boot your computer from the Linux boot diskette or the CD. Then find your Linux root partition on the hard drive, mount it, and edit the file `/etc/shadow`. (I can do it because after booting from the floppy, I become root without being asked for a password.) In the password file, I erase the encrypted password for root (for example, using the pico editor), so it is empty.

> Information about a user account is kept in plain−text files: `/etc/passwd` and `/etc/shadow`.
>
> The file `/etc/passwd` contains the "world−readable" information about all accounts on my computer Each line in this file contains information about one account. Each line has 7 colon−delimited fields (this means 8 entries separated by colons): login name, the letter "x", the numerical user ID, the numerical primary group ID for the user, a comment field (for example, the full name of the user), the user's $HOME directory, the name of the shell (meaning the program that is run at login).
>
> The balance of information about accounts on my computer is stored in the file `/etc/shadow`. This file is more secure because normally only root can read it. In this file, each line describes "shadow" information about one account, and has 9 colon−delimited fields: login name, encrypted password, days since Jan 1 1970 that password was last changed, days before password may be changed, number of days after which the password must be changed, number of days before password expiration to warn the user, number of days after password expiry that account is disabled, number of days since Jan 1 1970 that account is disabled, and a reserved field.
>
> Some (older) UNIX or Linux systems do not contain the file `/etc/shadow` and store the encrypted user password in the second field of each line of the file `/etc/passwd` (the field which on newer systems contains

just the letter x).

For example, my `/etc/shadow` entry for "root" account may look like this:

```
root:$1$BuPbmLAz$1G7.evIChyqaEI0TlZp0F.:11071:0:99999:7:−1:−1:134540356
```

and after the password is erased, it looks like this:

```
root::11071:0:99999:7:−1:−1:134540356
```

Now, the root account has no password, so I can reboot the computer and, at the login prompt, type "root" and for password just press ENTER (empty, no password). After a successful login, I immediately set the password for root using the command:

```
passwd
```

> Apparently, despite deleting the password from `/etc/shadow` , the Debian distribution will not let you log in "passwordless" (enhanced security?). In such a case, what needs to be done is to replace the password in /etc/shadow with an encrypted password from another account, where you know the password. After that, you can login since you know the password.
>
> E−mailing an encrypted password may be also a secure way to set up an account for somebody remote: "I am setting up an ftp account for you on my server. Email me your encrypted password." After you receive the encrypted password, you insert it into the appropriate field in `/etc/shadow`. Now, the user can log in, since she knows the password, but nobody else can.

To make the "floppy access" to my system a little bit more difficult, I considered running a computer without a floppy drive :−) Unfortunately, Linux CDs are bootable these days. I set up my boot sequence (in the BIOS setup) so that the system boots from the hard drive before floppy and CDROM are tried, and added an "administrative" password on changes to the BIOS settings. Still, I worry that these BIOS passwords are very easily crackable, or that one could remove the small battery that sustains the BIOS setting. One could also remove my harddrive and connect it to another computer for reading :−) . I am thinking about installing an "encrypted file system" which is now available on Linux, but considering all the trouble associated with it, perhaps I will settle on locking my room :−) . If all this sounds paranoid to you, it probably is−−it just illustrates the point there is little computer security, even under Linux, if the potential cracker has a physical access to your hardware.

3.2.4 I forgot my user password

If a regular (non−root) user forgets his/her password, this is not a problem since root can change any password. For example (as root):

```
passwd barbara
```

will prompt for a new password for the user "barbara" (no knowledge of the old password required by root). If a regular user (non−root) wants to change his/her password, s/he will be asked for the old password first. (This is a security feature so nobody changes your password if you have left your terminal unattended.)

3.2.5 Disabling or removing a user account

A user account can be temporarily disabled or permanently removed.

To temporarily disable (lock) a user account, there is no need to change his/her password. Just put an asterisk "*" at the beginning of the second field (before the encrypted password) in the file `/etc/shadow` . The "*" means that no login is permitted for this account. When you want to restore the account, you just erase the asterisk and the user account is back in operation, with its old password.

> Here is an example entry from the file `/etc/shadow` with the password disabled for user "peter":
>
> ```
> peter:*$1$narMEFm6$fhA1puOU422HiSL5aggLI/:11193:0:99999:7:−1:−1:134539228
> ```

I could also lock a user account with the following command:

```
passwd peter −l
```

and unlock it with

```
passwd peter −u
```

To irreversibly remove a user account from my home computer, I do the following:
− login as root

− change my identity to the user to be removed, to check if there is any new important mail:

```
su doomed_user_login_name
mail
logout
```

− delete the user account and group

```
userdel doomed_user_login_name
groupdel doomed_user_login_name
```

Remove the user affiliation to any supplementary groups:

```
usermod −G doomed_user_login_name doomed_user_login_name
```

− force−delete the user home directory with all its contents including any subdirectories:

```
rm −fr /home/doomed_user_login_name
```

3.2.6 I have file permission problems. How do file ownership and permissions work?

Linux (the same as any UNIX) is a secure, multiuser operating system, and this creates a level a complexity with "files permissions". Trouble with file permissions can lead to unexpected and nasty problems. Understanding file permissions is of uttermost importance to be able to administer any multiuser operating system (be it UNIX, WinNT, or Linux). My advice would be: learn the system of Linux (or any UNIX) file permission conventions; you will not regret it.

File owners. Each file (or directory) belongs to an owner (normally a login name) and to a group. The owner is typically the person who created (or copied) the file. The group often consists of one person−−the owner, and has a name identical to that of the owner, but it does not need to be so. A file can be removed (erased) only by the owner of the file, or a member of the group that owns the file, or the root. Other users, however, may be able to modify or erase the contents of the file if they are given permission to do so−−read on. The owner and group that owns the file will be shown in the output from the `ls −l` command (="list in the long format"). For example, the command:

```
ls −l junk
```

produced this output on my screen:

```
−rwx------   1 yogin    inca           27 Apr 24 14:12 junk
```

This shows the file "junk", belonging to the owner "yogin" and to the group "inca".

The ownership of a file can be changed using the commands `chown` (change owner) and `chgrp` (change group), which are normally executed by root:

```
chown peter junk
chgrp peter junk
ls −l junk
```

After executing the above 3 lines, the command `ls−l junk` produces this output on my screen:

```
−rwx------   1 peter    peter           27 Apr 25 20:27 junk
```

Changing file ownership comes handy if you move/copy files around as root for use by other users. At the end of your housekeeping you typically want to hand the file ownership over to the proper user.

File permissions. Now, an owner of a file can make the file accessible in three modes: read (r), write (w) and execute (x) to three classes of users: owner (u), members of a group (g), others on the system (o). You can check the current access permissions using:

`ls −l filename`
If the file is accessible to all users (owner, group, others) in all three modes (read, write, execute) it will show:

```
−rwxrwxrwx
```

Skip the first "−" (it shows the type of file, and is "−" for normal files, "d" for directories, "l" for links, "c" for character devices, "b" for block devices, "p" for named pipes i.e. FIFO files, "f" for stacks i.e. LIFO files). After the initial "−" character, the first triplet shows the file permission for the owner of the file, the second triplet shows the permissions for the group that owns the file, the third triplet shows the permissions for other users. A "no" permission is shown as "−". Here is an output from the `ls −l` command on a file that is owned by root, for which the owner (root) has all permissions, but the group and others can only read and execute:

```
drwxr−xr−x   2 root     root        21504 Apr 24 19:27 dev
```

The first letter "d" shows that the file is actually a directory.

You can change the permissions on a file which you own using the command `chmod` (="change mode"). For example, this command will add the permission to read the file "junk" to all (=user+group+others):

```
chmod a+r junk
```

In the command above, instead of "a" (="all"), I could have used "u", "g" or "o" (="user", "group" or "others"). Instead of "+" (="add the permission"), I could have used "−" or "=" ("remove the permission" or "set the permission"). Instead of "r" (="read permission"), I could have used "w" or "x" ("write permission" or "execute permission").
Second example. This command will remove the permission to execute the file "junk" from others:

```
chmod o−x junk
```

Instead of letters, one can also use numbers to specify the permissions. To understand how it works, look at this:

```
execute=1
write=2
read=4
```

The total permission for a class of users is the sum of the three. Thus:

```
0 = no permissions at all(neither to write, nor to read nor to execute)(common)
1 = execute only (seems unusual)
2 = write only  (seems unusual)
3 = write and execute (seems unusual)
4 = read only (common)
5 = read and execute (common)
6 = read and write (common)
7 = read, write and execute (common).
```

The permission for all three classes of users (owner, group, others) is obtained by gluing the three digits together one by one. For example, the command:

```
chmod 770 junk
```

will give the owner and the group the completto of permissions, but no permissions to others. The command:

```
chmod 666 junk
```

gives all three classes of users (owner, group, others) the permissions to read and write (but not execute) the example file named "junk". Please note the "666". It is quite often used and, for at least one person I know, it is proof that Linux (any UNIX for that matter) is the work of the devil >:−0.
This command:

```
chmod 411 junk
```

would give the owner the permission to read only, and the group and others to execute only. This one does not seem useful, but might be funny, at least for those North American Linux users who dial 411 (telephone number) for directory assistance. Mail me if you can think of any other funny permissions (perhaps 007?).
The numerical way of representing file permissions is called "octal" because the numbers have the base 8 (the decimal system's base is 10). The highest digit in the octal system is 7 (the octal system has eight digits: 0 to 7, analogous to the decimal system having ten digits: 0 to 9). The octal representation is really a convenient notation for the binary representation of file permissions, where each permission is flagged as "set" or "denied" with a one or zero and the total is represented as a string of zeroes and ones, as in this diagram:

```
user class:                                    owner  group others
example permissions:                            rwx    rw-    r--
absent permissions:                             ---    --x    -wx
binary representation of the permissions:       111    110    100
octal representation  of the binary:              7      6      4
```

Permissions for directories.
The meaning of the permissions is different for directories than it is for "normal" files. For normal files: r=permission to read the contents of the file, w=permission to modify the contents of the file, and x=permission to execute the file.
For directories: r=permission to list the filenames in the directory, w=permission to create or delete files in the directory, and x=permission to access the directory. Otherwise, the permissions are set the same way for directories as they are for normal files.

Default file permissions with umask. When a new file is created, it is given default permissions. On my system, these are:

```
-rw-r--r--
```

This means that files created by a user can be read and written by this user; the group and the others can only read the file. Still, on my default RedHat system, users cannot read the files in the other users' home directories because the permissions on the home directories are:

```
drwx------
```

I can check the default file permissions given to my newly created files using:

```
umask -S
```

(The option "-S" stands for "symbolic" and tells `umask` to display the permissions in an easy-to-read form, instead of the default numeric mode.)
I can change the default file permissions for newly created files using a command like:

```
umask u=rw,g=,o=
```

which will give the owner the read and write permissions on newly created files (r+w), and no permission to the group and others.
Using numbers to set default permissions with umask is more tricky. The number shows the permissions that you take away for users (opposite to `chmod`). Thus:

```
umask 000
```

will give full permissions to everybody on newly created files. The next example gives read and write permissions to the owner, and zero permissions for everybody else (seems that's what one might want):

```
umask 177
```

To make the settings permanent for all users on the system, adjust the appropriate line(s) in the file `/etc/profile` .

3.2.7 My mp3 player chokes. The sound is kind of interrupted (how to set suid).

The MP3 player might not be given enough processor power (it requires a lot of it). It could be that your computer is lousy. Or you might be running too many cpu-intensive programs at the same time. Or, most likely, you may need to run the player with a higher priority. (The priority of a program can be set with the command `nice` -- see `man nice` or `info nice`). Try to run the player as root--programs run by root are given higher priority than those run by normal users. If this solves the "interrupted music" problem, set the "suid" on the executable so all users are given the "effective user id" of the file ower (normally root) when running it, for example:

```
chmod a+s /usr/bin/xmms
```

will do the trick for the xmms program. The output from

```
ls -l /usr/bin/xmms
```

on my computer is now:

```
-rwsr-sr-x  1 root  root  908k Feb 22  2000 /usr/bin/xmms
```

The first "s" indicates that the substitute−user−id (suid) bit is set. The second "s" indicates that the substitute−group−id (sgid) is also set. Thus anybody who executes `xmms` is given the effective user id of the program owner and effective group id of the owner group, which in the example above is the user "root" and the group "root".

Setting the suid for a program could possibly become a security hole in your system. This is unlikely the case on a closed home network and when setting suid for a program of which the origin is well traceable. However, even at home, I wouldn't suid a piece of code of which the origin is uncertain, even if the setup instructions urged me to do so. Also, it is definitely a very bad idea to suid too many executables on your system−−it defies the whole idea of UNIX security.
Some programs do, however, require suid for proper functioning, for example `kppp` (the popular modem "ppp" connection utility under the KDE graphical−user−interface desktop). This is because they require direct access to the hardware−−something only root is allowed to.

If you have constant problems with a smooth performance of your system, or some "real time hardware" (e.g., CD writer) tends to crash, try to reduce the number of daemons on your Linux system. Run (as root) `setup` (RH specific command) and disable all the "services" that you don't really require. Ultimately, you can switch to the command line, shut down the GUI (command `init 3` as root), and then the performance should surely be better even.

For those who need (like) their Linux to be a "universal" operating system (workstation, server, office computer, game box, mulimedia, etc, everything at the same time), there are dedicated Linux kernel patches: "low latancy patch" and "pre−emptive kernel patch" which aggresively atack the "latency" problem that overloaded systems exhibit.

3.3 Job scheduling with "&", "at", "batch", and cron

3.3.1 How do I execute a command in the "background"?

Using the "&" at the end of the command. For example, this will start `licq` (an icq client) in the x−terminal in the background, so that after issuing the command, my x−terminal is not blocked:

```
licq &
```

The process identification number, *job_number*, is printed on the screen, so you can use it with related commands. The related commands are `fg` *`job_number`* (="foreground", bring the background process back to my immediate view/control, restart it if it was stopped), `bg` *`job_number`* (="background", send the process to the background, restart if it was stopped, exactly as if it was started using &), `<Ctrl>z` (send the current foreground process to the background and stop it), `jobs` (list the active jobs), `kill` *`process_ID`* (terminate the process, use the conmmand `ps` to find the *process_ID* of the process to kill).

To make a background process keep running after you disconnect, you may use the nohup (="no hungup"), for example:

```
nohup make &
```

that maybe compiling a large program.

3.3.2 How do I execute a command at specified time (using "at" or "batch")?

The `at` command will execute the command(s) you specify at the date and time of your choice. For example, I could start playing music from my CDROM at 7 o'clock in the morning:

```
at 7:00
cdplay<Ctrl>d
```

In the example above, I entered the first line "at 7:00" on the command line and then pressed ENTER. To this, the `at` command displayed a prompt "at>". At this prompt, I entered my command "cdplay" and then pressed the control key and "d" simultaneously to finish the input. If instead of pressing <Ctrl>d , I pressed "ENTER", the next "at>" prompt would appear, at which I would be able to enter the next command to be executed right after "cdplay", also at 7:00. And so on, I could have had many commands scheduled for execution one by one starting at 7:00. After typing the last command, I would finish the input with <Ctrl>d. Think of the <Ctrl>d as sending "end−of−file" to the current input. Don't press <Ctrl>d twice because this will log you out−−that's what <Ctrl>d does when entered straight on the Linux command line.

You can list the job you scheduled for execution using:

```
at −l
```

which will give you the numbered list of the jobs waiting.

If you changed your mind, you can remove a job from this list. For example:

```
atrm 8
```

will remove the job with the number eight on the list.
I could also schedule a job for execution much later, for example:

```
at 23:55 12/31/00
startx
```

would start my X−windowing system right on time for the new millennium (5 minutes before midnight on 31 of December 2000). If you cannot execute the `at` command, check if the at daemon ("atd") is loaded (as root, use ntsysv). If you cannot execute the `at` command as a regular user although it works for root, check if the empty file `/etc/at.deny` exists and there is no file `/etc/at.allow`. This should be the default setup and it permits all the users to execute `at`. If you want only certain users to use `at`, create a file `/etc/at.allow` and list these users there.
For other options, check:

```
man at
```

If you wish to perform a processor−intensive job in the background when the system load is low, you may choose to use the `batch` command. For example, I could run setiathome (a program crunching data to help in search of extraterrestrial intelligence, SETI) using:

```
batch
at>setiathome<Ctrl>d
```

In this example, I entered the command `batch` and then, at the "at>" prompt, I entered the command which I wanted to be executed in the background. The job tries to start immediately, but goes ahead only when the system load is under 0.8 You can check the system load by inspecting the contents of the (virtual) file `/proc/loadavg` . For example:

```
cat /proc/loadavg
```

When a batch job finishes, the output is sent to me via e−mail.

3.3.3 How do I set up cron?

Cron (a Linux process that performs background work, often at night) is set up by default on your RedHat system. So you don't have to do anything about it unless you would like to add some tasks to be performed on your system on a regular basis or change the time at which cron performs its duties.

Please note that some of the cron work might be essential for your system functioning properly over a long period of time. Among other things cron may:
− rebuild the database of files which is used when you search for files with the `locate` command,
− clean the /tmp directory,
− rebuild the manual pages,
− "rotate" the log files, i.e. discard the oldest log files, rename the intermediate logs, and create new logs,
− perform some other checkups, e.g. adding fonts that you recently copied to your system.

Therefore, it may not be the best idea to always switch your Linux machine off for the night−−in such a case cron will never have a chance to do its job. If you do like switching off your computer for the night, you may want to adjust cron so it performs its duties at some other time.

To find out when cron wakes up to perform its duties, have a look at the file `/etc/crontab`, for example:

```
cat /etc/crontab
```

It may contain something like this:

```
# run−parts
01 * * * * root run−parts /etc/cron.hourly
02 4 * * * root run−parts /etc/cron.daily
```

```
22 4 * * 0 root run-parts /etc/cron.weekly
42 4 1 * * root run-parts /etc/cron.monthly
```

You can see that there are four categories of cron jobs: performed hourly, daily, weekly and monthly. You can modify those or add your own category. Here is how it works.

The columns in the entries show: minute (0−59), hour (0−23), day of month (1−31), month of year (1−12), day of week (0−6−−Sunday to Saturday). The "*" means "any valid value".

Thus, in the example quoted, the hourly jobs are performed every time the computer clock shows "and one minute", which happens every hour, at one minute past the hour. The daily jobs are performed every time the clock shows 2 minutes past 4 o'clock, which happens once a day. The weekly jobs are performed at 22 minutes past four o'clock in the morning on Sundays. The monthly jobs are performed 42 minutes past four o'clock on the first day of every month. The directory with the script file that contain the command(s) to be executed is shown as the last entry on each line.

If you wanted your jobs to be performed at noon instead of 4 in the morning, just change the 4s to 12s. Cron wakes up every minute and examines if the `/etc/crontab` has changed so there is no need to re−start anything after you make your changes.
If you wanted to add a job to your cron, place a script which runs your job (or a link to your script) in the directory `/etc/cron.hourly` or `cron.daily` or `/etc/cron.weekly`, or `/etc/cron.monthly` .

Here is an example of an entry in `/etc/crontab` which causes a job to be performed three times a week (Mon, Wed, Fri):

```
02 4 * * 1,3,5 root run-parts/etc/cron.weekly
```

An example seen on usenet showing how to automatically email a log file (edited for space):

> Re: help in crontab
> From: Dean Thompson <Dean.Thompson@csse.monash.edu.au> Date: 2001−03−03 16:35
> Newsgroups: comp.os.linux.admin,comp.os.linux.networking,comp.os.linux.security
> > How can I set the job mail abc@abc.com < /var/log
> > every day in the /etc/crontab −e file ?
> You could try the following entry and see if you meet with any success:
> 0 0 * * * (/bin/mail abc@abc.com < /var/log/messages) > /dev/null 2>&1

3.4 Shell

3.4.1 What is a shell and do I want to use a different one?

A shell is the program that interprets what you type on the command line and decides what to do with it. A shell can also be invoked in a non−interactive way, for example to execute a pre−typed list of commands contained in a text file (a "shell script"). Think of a shell as the equivalent of the DOS "command.com" (command−line interpreter) and the shell script files as the equivalent of the DOS batch files (*.bat). In comparison with their DOS cousins, the Linux shell and scripting are on steroids.

There are several shells available on the Linux system (if you installed them): `bash` ("Bourne Again" shell), `sh` (Bourne shell, standard on many UNIX systems), `csh` (C shell, with a syntax akin to the "c" programming language, available on most UNIX systems), `pdksh` (public domain Korn shell), `tcsh` (tiny C shell, often used on small systems), `sash` (stand−alone shell, could be used when libraries are not available), `ash`, `zsh`, and perhaps a couple more.

The default shell on my system (and most probably on yours too) is `bash` , which is an excellent and standard shell, and I really cannot see a reason why a newbie like myself would want to change it. `bash` is fully backwards−compatible with the Bourne shell (the most popular shell on UNIX) and incorporates many enhancements and best features from other shells. From a newbie perspective, the different shells are included with Linux for historical reasons and backwards−compatibility of shell scripts that may require a particular shell to run. [Some shells may be useful if you write programs targeted for specialized "embedded" devices, that might run a "tiny" shell.]

You can determine the shell you are running using:

```
echo $SHELL
```

If you wanted to try another shell, type, for example:
`tcsh`
which will start the tiny c shell. When done, type
`exit`

which will return you to the previous shell (using `exit` on your first shell will log you out). You can find out how many shells you stacked on each other by displaying the "shell level" environmental variable:

```
echo $SHLVL
```

In the above command, the "$" means "expand the value of a shell environment variable", "SHLVL" is the variable name, and "echo" is a command that prints things.

The shell for each user is specified as the last field in the password file `/etc/passwd` . If you really wanted to change it, edit (as root) this file and replace the "/bin/bash" with the shell of your choice.

3.4.2 How do I customize my shell prompt?

On my machine, the prompt may look like this:

```
[stan@marie stan]$ _
```

Here "stan" is my login name, "marie" is the name of the computer, the second "stan" is the name of my current working directory, and "_" represents the cursor.

The prompt is set by the environmental variable called PS1. To display the current setting, I can use:

```
echo $PS1
```

The system−wide setting of the prompt (for all users on the system) is in the file `/etc/bashrc` which on my system contains such a line:

```
PS1="[\u@\h \W]\$ "
```

To customize the prompt, I can edit the file `/etc/bashrc` (as root) and insert almost any text inside the quotation marks. Here is the meaning of some special codes I may also choose to use:

```
\u   −    username of the current user (= $LOGNAME),
\h   −    the name of the computer running the shell (hostname),
\H   −    entire hostname,
\W   −    the base of the name of the current working directory,
\w   −    the full name of the current working directory,
\$   −    display "$" for normal users and "#" for the root,
\!   −    history number of the current command,
\#   −    number of the current command (as executed in the current shell),
\d   −    current date,
\t   −    current time (24−hr),
\T   −    current time (12−hr) − bash 2.0 only,
\@   −    current time (AM/PM format) − bash 2.0 only,
\s   −    name of the shell,
\a   −    sound alarm (beep),
\j   −    number of jobs the user has,
\n   −    new line,
\\   −    backslash,
\[   −    begin a sequence of non−printable characters,
\]   −    end a sequence of non−printable characters,
\nnn −    the ASCII character corresponding to the octal number nnn.
$(date) − output from the date command (or any other command for that matter),
```

Here is an example on how to add colour. See the next chapter for details about colour:

```
PS1="\[\033[1;32m\][\u@\h \W]\$\[\033[0m\] "
```

There is also the second−level prompt, set by a variable called PS2. The shell uses the second level prompt when it expects additional input, and on my system the secondary prompt is "> ". I don't worry too much about PS2, but if I did I could set it the same way as PS1. There are even PS3 and PS4, but these are rarely seen.

3.4.3 Colour on text terminal

Colour on the text terminal can be produced using the "ANSI escape sequences". For example:

```
echo −e "\033[44;37;5m ME \033[0m COOL"
```

The above sets the background to blue, foreground white, blinking video, and prints " ME ", then resets the terminal back to defaults and prints " COOL". The "−e" is an option specific to the `echo` command−−it enables the interpretations of the special characters. The "\033[" introduces the escape sequence. The "m" means "set attribute" and thus finishes the sequence. The actual codes in the example above are "44;37;5" and "0".

Change the "44;37;5" to produce different colour combinations−−the number/order of codes do not matter. The codes to choose from are listed below:

```
Code  Action/Color
−−−−−−−−−−−−−−−−−−−−−−−−−−
 0    reset all attributes to their defaults
 1    set bold
 2    set half−bright (simulated with color on a color display)
 4    set underscore (simulated with color on a color display)
 5    set blink
 7    set reverse video
22    set normal intensity
24    underline off
25    blink off
27    reverse video off
30    set black foreground
31    set red foreground
32    set green foreground
33    set brown foreground
34    set blue foreground
35    set magenta foreground
36    set cyan foreground
37    set white foreground
38    set underscore on, set default foreground color
39    set underscore off, set default foreground color
40    set black background
41    set red background
42    set green background
43    set brown background
44    set blue background
45    set magenta background
46    set cyan background
47    set white background
49    set default background color
```

Other interesting codes:

```
\033[2J      clear screen
\033[0q      clear all keyboard LEDs (won't work from Xterm)
\033[1q      set "Scroll Lock" LED
\033[2q      set "Num Lock" LED
\033[3q      set Caps Lock LED
\033[15;40H  move the cursor to line 15, column 40
\007         bell (beep)
```

LEDs (="Light Emitting Diods) are the lights on the keyboard which indicate if <CapsLock>, <NumLock> and <ScrollLock> are engaged.

See `man console_codes` for more.

3.4.4 How do I print symbols on the console, in a text mode application, and in X?

The procedure described here may give me fast access to the PC extended character set (codes 128−255) and is quite portable in the PC world: it works in MS Windows, DOS (if you have an ANSI driver installed), and inside any text mode Linux application (including right on the shell command line). I found it was worth my time to memorize the codes for the few characters I tend to use the most.

It works like this. Make sure that <NumLock> is on. Then press <Alt> and hold it. While <Alt> is pressed, key in on the numeric keypad these four digits: 0181. Now release <Alt> and the Greek letter mu "µ" appears. I find quite useful the following characters from the PC character set encoding: 176 ° (degree), 177 ± (plus minus), 178 ² (square), 179 ³ (power 3), 181 µ (Greek mu), 0183 · (multiplication sign), 232 è (French accent agrave), 233 é (French accent aigu) 228 ä (German a−umlaut), 243 ó (Polish u−zamkniete), 248 ø (Scandinavian o−bar) 252 ü (German u−umlaut). Some other characters are also possible, here is the full listing:

```
128 ? 147 ? 166 ¦ 185 ¹ 204 Ì 223 ß 242 ò
129 ? 148 ? 167 § 186 º 205 Í 224 à 243 ó
130 , 149 * 168 ¨ 187 » 206 Î 225 á 244 ô
131 f 150 − 169 © 188 ¼ 207 Ï 226 â 245 õ
132 ? 151 − 170 ª 189 ½ 208 Ð 227 ã 246 ö
133 ? 152 ~ 171 « 190 ¾ 209 Ñ 228 ä 247 ÷
134 ? 153 ? 172 ¬ 191 ¿ 210 Ò 229 å 248 ø
135 ? 154 s 173 - 192 À 211 Ó 230 æ 249 ù
136 ^ 155 > 174 ® 193 Á 212 Ô 231 ç 250 ú
137 ? 156 ? 175 ¯ 194 Â 213 Õ 232 è 251 û
138 S 157 ? 176 ° 195 Ã 214 Ö 233 é 252 ü
139 < 158 ? 177 ± 196 Ä 215 × 234 ê 253 ý
140 ? 159 Y 178 ² 197 Å 216 Ø 235 ë 254 þ
141 ? 160   179 ³ 198 Æ 217 Ù 236 ì 255 ÿ
142 ? 161 ¡ 180 ´ 199 Ç 218 Ú 237 í
143 ? 162 ¢ 181 µ 200 È 219 Û 238 î
144 ? 163 £ 182 ¶ 201 É 220 Ü 239 ï
145 ? 164 ¤ 183 · 202 Ê 221 Ý 240 ð
146 ? 165 ¥ 184 ¸ 203 Ë 222 Þ 241 ñ
```

Now, if I really want to, I can have a file with a name `µm·°C±b³.` MS Windows, DOS ANSI, and Unicode differ slightly in some of the above characters, but the useful "core" remains the same. See http://www.hclrss.demon.co.uk/demos/ansi.html if you want to know the details of the differences. Linux uses the Unicode standard.

Under X, the above key combinations will not work. But I may use:

```
kcharselect&
```

or

```
gcharmap&
```

to select a Unicode character and copy it into my application. Not all unicode characters are available yet, but many are. From Unicode pages other than page 0, the characters may display or not, depending on your application and the availability of the glyph in your font. For example, I can surely use the following characters in most KDE applications (if they display on your browser, depends on your browser AND the availability of a suitable Unicode font):

Greek (Unicode page 3, char 913 to 969): ???????????? ???????????????? ???????????????? ??????????
Russian: (Unicode page 4, chars 1040 to 1103): ????????? ????????????? ????????????? ????????????? ?????????????? ??
and many others. You can find common Unicode codes (numerical) and their html symbolic ("character entity") references at http://www.hclrss.demon.co.uk/demos/ent4_frame.html.

3.4.5 How do I write a simple shell script?

Create a text (ASCII) file which will contain the shell script. For example, I would use the pico editor to write a script that runs the program `tar` with all the parameters usually necessary to uncompress tarballs downloaded from the Internet (I never seem to remember the `tar` options). I decided to call my script "untar":

```
pico untar
```

Since the file "untar" did not exist in my current directory, it was created by the `pico` text editor. Now, I type in the content of my script:

```
#!/bin/bash
echo this is the script file $0
echo untarring the file $1
# this calls tar with options −xvzf (extract,
#    verbose, filter through gzip, input filename)
tar −xvzf $1
```

I save the file with <Ctrl>o and exit with <Ctrl>x

The first line of the script, starting with "#!" (called pound−bang), is special−−it tells the shell what program should be used to interpret my script. In this example, the script is to be interpreted by the bash shell `/bin/bash`. The first line must start with #! or the script will never run (the file will be interpreted as just a text file). Other lines starting with # are comments for the author (readers, users) of the shell and are totally ignored by the computer.

The $0, $1, $2 ... in my script are the parameters passed to my script. For example, if I ran a script called "myscript" with seven parameters like this:

```
myscript a b c d e f g
```

then $0 would be seen inside "myscript" as having the value "myscript", $1 would have the value "a", $2 would be "b", $3 would be "c", etc.
On the second and third line of my example script, the command `echo` prints on the screen everything that follows on the same line, expanding $0 and $1 to the values of the parameters passed to the script. The fourth and fifth line contains a comment I wrote to myself to remind myself what I was trying to achieve, just in case I ever had to modify my script. The last line performs the actual work.
Once the script is written, I make the file executable to the file owner ("u"=user):

```
chmod u+x untar
```

and my script is ready to run like this:

```
./untar my_tar.tar.gz
```

Linux scripting is definitely rich, flexible, powerful and can be complex. However, it does not require special knowledge to write simple scripts for automation of common tasks. You just put together a group of often used commands, one by one, into a file. I use scripting because I am too lazy to type the same groups of commands over and over again.

A really simple sequence of commands can also be typed into a text file and passed to shell for straight execution using:

```
source my_file
```

[No need for the initial "pound bang" or executable permission.]

3.4.6 Meaning of quotes

Normally, these characters are special to the shell:

```
\ ' " ` < > [ ] ? | ; #  $  ^ & * ( ) = <Space> <Tab> <Newline>
```

There are four different types of quotes: backslash (\), single quotes (apostrophes, '), double quotes (quotation marks, "), and backquotes (`).

The backslash \ means: disable the special meaning of the subsequent character.

Quoting with " (two apostrophes) means: quote exactly, disabling any special characters inside the quotes.

Quoting with "" means: disable the special characters inside the quotes except for `$ ` \`

The pair `` (two backquotes) means: do a command substitution inside the backquotes first. So what is inside the backquotes is executed by the shell first, and then the output is passed to the command outside the quotes. The same can also be acomplished with

`$(command)` which nests better than backquotes.

Examples. I can create a funny directory called "*" by either \ quoting or " quoting:

```
mkdir \*
mkdir '*'
```

This hides the special meaning of the "*" from the shell (without the quote it would mean "all files in the current directory").

3.4.7 Input/output redirection

There are three important input−output streams: standard input ("stdin"), standard output ("stdout"), and standard error output ("stderr"). They all default to the console ("console" means the keyboard for the input and the screen for the output), but they can be redirected.

To redirect the standard output I use ">". For example:

```
dir my_dir > filelisting.txt
```

will redirect the standard output of the `dir` command into the textfile `filelisting.txt` and nothing should appear on my screen. The file can be subsequently edited (e.g. with `pico filelisting.txt`) or embedded into a document.

To redirect the standard error, I need to use the construct "2>". For example:

```
dir my_dir 2> errorlisting.txt
```

The above will send the normal output onto the screen and nothing to the file unless `dir` produces an error. On error, nothing may go to my screen, and the file `errorlisting.txt` will contain the error message, which might be something like:

```
dir: my_dir: Permission denied
```

Finally, I can redirect both standard output and standard error to a file using:

```
dir my_dir > file_and_error_listing.txt 2>&1
```

which first redirects the standard output to a textfile, and then redirects the standard error to the same location as the standard output. A bit twisted, how it works, but it works.

In the examples above, if the file (to which to redirect) already existed, it will be overwritten. To append to an existing file, I use ">>" as in these examples:

```
dir my_dir >> filelisting.txt
dir my_dir 2>> errorlisting.txt
dir my_file >>file_and_error_listing.txt 2>&1
```

If you are puzzled by the "2>" symbol, here, briefly, is how to rationalize it. The standard streams have standard descriptors. "0" is standard input, "1" standard output and "2" is standard error.

```
dir my_dir > file.txt
```

is short for

```
dir my_dir 1> file.txt
```

and therefore the example below redirects the standard error:

```
dir my_dir 2> file.txt
```

One can also use the symbol "|" to send ("pipe") the output from one command as input for another command. In this popular example, the output from `dir` is piped to `more` (`more` pauses the display after each screenful):

```
dir | more
```

One can also split the output so it goes both to a file and the screen using "tee":

```
dir | tee filelisting.txt
```

It is called "tee" by the analogy to the "T"−letter−shape fitting that pipefitters use, and which divides flow.

This section so far dealt with redirecting standard output. Redirecting standard input is not nearly as useful as redirecting the output, but it can be done using a construct like this:

```
cat < my_file
```

There is also something called in−line redirection of the standard output, realized with "<<". Forget about it, seems of no use to me. Yet, here is an example if you really ever needed it (here, the ">" stands for the secondary prompt):

```
cat << my_marker
> my_line_from_the_keyboard
> another line_from_the_keyboard
> my_marker  [the marker of my choice ends the in−line redirection].
```

Apart from redirection to regular files and "filters" (as shown in the examples above), one can redirect to/from devices and other special files. Some examples follow.

An example of redirection to a device file. The following command displays the listing of files on the fourth text terminal:

```
dir > /dev/tty4
```

An example of redirection to a special "FIFO" file. This command sends the message "you are lucky" to the lucky ICQ user UIN 77777777 (assuming you are connected to the icq server with your licq program):

```
echo message 77777777 "you are lucky" > ~/.licq/licq_fifo
```

> The above works because the file "licq_fifo" in your licq directory is a special "fifo" (first−in−first−out) queue file. How could the above ever be more useful than sending a message using the pretty licq GUI front−end? For example, you could write a short script to impress fellow icq users with multiple (identitcal) messages:
>
> ```
> #!/bin/bash
> echo Messaging UIN:  $1  Message:  $2   Times: $3
> # The next command puts puts your licq in the status "on−line, invisible".
> echo 'status *online' > ~/.licq/licq_fifo
> c=0
> while  [ $c −le $3]
> do
> echo message $1 $2 > ~/.licq/licq_fifo
> c=`expr $c + 1`
> echo $c " "
> done
> echo 'status offline' > ~/.licq/licq_fifo
> echo "all done"
> ```
>
> The above example may give you an idea how one can use licq for automation, owing to the smart licq communication model (the fifo file) and simple file redirection.

3.4.8 Shell special characters (metacharacters)

Normally, these characters have special meaning to the shell:

```
\ ' " ` < > | ; <Space> <Tab> <Newline> ( ) [ ] ? # $ ^ & * =
```

Here is the meaning of some of them:

`\` `' "` and `'` are used for quoting and were described before.

`<` and `>` are used for input/output redirection and were described before.

`|` pipes the output of the command to the left of the pipe symbol "|" to the input of the command on the right of the pipe symbol.

`;` separates multiple commands written on a single line.

`<Space>` and `<Tab>` separate the command words.

`<Newline>` completes a command or set of commands.

`( )` enclose command(s) to be launched in a separate shell (subshell). E.g. `( dir )`.

`{ }` enclose a group of commands to be launched by the current shell. E.g. `{ dir }`. It needs the spaces.

`&` causes the preceding command to execute in the background (i.e., asynchronously, as its own separate process) so that the next command does not wait for its completion.

`*` when a filename is expected, it matches any filename except those starting with a dot (or any part of a filename, except the initial dot).

`?` when a filename is expected, it matches any single character.

`[ ]` when a filename is expected, it maches any single character enclosed inside the pair of [ ].

`&&` is an "AND" connecting two commands.

> *`command1 && command2`* will execute *command2* only if *command1* exits with the exit status 0 (no error). For example: `cat file1 && cat file2` will display file2 only if displaying file1 succeeded.

`||` is an "OR" connecting two commands.

> *`command1 || command2`* will execute *command2* only if *command1* exits with the exit status of non−zero (with an error). For example: `cat file1 || cat file2` will display file2 only if displaying file1 didn't succeed.

`=` assigns a value to a variable.

> Example. This command:
> `me=blahblah`
> assigns the value "blahblah" to the variable called "me". I can print the name of the variable using:
> `echo $me`

`$` preceeds the name of a variable to be expanded.

> The variables are either assigned using "=" or are one of the pre−defined variables (which cannot be assigned to):
> $0 name of the shell or the shell script being executed.
> $# number of the positional parameters to the command
> $1 the value of the first positional parameter passed to the command. $2 is the second positional parameter passed to the command. etc. up to $9.
> $* expands to all positional parameters passed to the command
> $@ expands to all positional parameters passed to the command, but individually quoted when "$@" is used.
> See `man bash` if you really need more.

3.5 Package installation and rpm package manager

3.5.1 How do I install a program I downloaded from the Internet?

The answer depends on what kind of package you downloaded. You can avoid many installation headaches if you download programs in the form of Red Hat binary packages *.rpm (that's the format I select if given a choice).

INSTALLATION OF REDHAT BINARY PACKAGES

o If the program I want to install is a RedHat binary package (*.rpm), I can use either a command line, or a GUI utility. I like to use the command−line utility because it is fast and trouble−free. The RedHat package manager installation utility is called `rpm` . First I read the info on the package content (optional):

```
rpm −qpi my_new_file.rpm
```

This queries (mode "q", must be the first letter after the dash) the yet uninstalled package (option "p") so that it displays the info (option "i") which the package contains. If I want to install the program, I run (as root):

```
rpm −ihv my_new_file.rpm
```

The above command does the installation job. It runs `rpm` telling it to install the package (mode "i", must be the first letter after the dash) while printing to the screen more information than usual (option "h"=display "hashes" to show the unpacking progress, option "v" = be verbose). The contents of the package are distributed to the directories where they belong (rpm knows where they belong). After this installation, the program is ready to run, I just have to know the executable name and its location. If I have trouble finding the executable, this lists all the files that the package contains together with their destination directories:

```
rpm −qpl my_new_file.rpm
```

This queries (option "q") the yet uninstalled package (option"p") so that it displays the listing (option "l") of all the files the package contains.

The GUI front−ends to `rpm` are: `gnopro` (the old version, that comes with RH6.0 is confusing, but newer versions are much improved), `kpackage` (available only with the more recent distributions), and the old `glint` (very slow, comes with RH5.2).

Troubleshooting. rpm is supposed to be an intelligent software package manager. If the installation fails I read the error message and may be able to figure what to do:
(1) Installation failed because I have an earlier version of the same package and the versions confilict. Solution: don't install, but "upgrade" the package.

```
rpm −Uvh my_new_file.rpm
```

(2) Installation failed because another package is needed first. I have to find the missing package and install it first, and then retry the installation. In extreme cases, I may choose to ignore the missing dependencies (I really should know what I am doing here else the software may malfunction):

```
rpm −ivh −−nodeps my_new_file.rpm
```

or perhaps even:

```
rmp −ivh −−nodeps −−force my_new_file.rpm
```

INSTALLATION FROM A SOURCE−CODE TARBALL

o If what I downloaded from the net is a Linux source code in the form of a compressed tarball (*.tar.gz or *.tgz), the installation procedure is longer and more troublesome than with the binary−only rpm. I typically install the program as root.

First, I change my current working directory to `/usr/local` :

```
cd /usr/local
```

Second, I decompress the tarball that I downloaded from the net:

```
tar −xvzf /home/the_dir_where_the_tarball_is/my_tarball.tar.gz
```

This extracts (option "x") the contents of the *.tar.gz (or *.tgz) tarball, unzips it (option "z"), while talking to me more than usual (option "v" = verbose). Please note that the option "f" means "file", so the filename must immediately follow the letter "f". The contents of the tarball are extracted into a subdirectory which `tar` creates under my current working directory, which in the typical case is `/usr/local/` . The tarball knows what the new subdirectory should be called.

If the tarball is not compressed (e.g., *.tar), I may use:

```
tar −xvf /home/the_dir_where_the_tarball_is/my_tarball.tar
```

Third, I have to figure how the new directory is called, then I `cd` into it:

```
dir
cd the_new_program_subdir
```

Since some of the directories have long names, I use the great autocompletion option to save on typing−−I just type the first few letters and then press <TAB> .

Fourth, most programs are compiled by executing these three commands:

```
./configure
make
make install
```

The above commands can take some time to complete (1 min? 0.5 h?). If any of them fail, it might be an idea to read the README or INSTALL or whatever info is provided with the new program. Some programs may require customization of the environment (e.g. definition of their path) or installation of an additional library, or yet something else. It can sometimes be a pain. Very simple programs might not need the "./configure" or/and "make install" step, in which case "make" alone will do.

Fifth, if everything goes well, I find the new executable which I just compiled. The names of executables display in green when running this command:

```
ls −−color
```

Now, I can run the executable, for example:

```
./the_executable
```

Some programs automatically install the executable to /usr/local/bin, so I may want to try:

```
/usr/local/bin/the_executable
```

Sixth, if I plan to run the program more often, I create a symbolic link to the executable from the directory `/usr/local/bin` :

```
cd /usr/local/bin
ln −s /usr/local/the_new_program_subdir/the_executable .
```

This way, the executable (actually, a symbolic link to it) is on my PATH and it can be run by simply typing its name (no need to type the full path to the executable any more). Some programs will install the executable (or a link to it) in a "bin" directory in which case you skip the last step.

INSTALLATION FROM SOURCE CODE RPM PACKAGE

o There are also programs distributed as "source code rpm" packages. They require installation of the *.rpm package with the "rpm" utility as described in the first part of this chapter. But since the "rpm" installs the source code (typically in the C language source code), I then have to compile the source code by executing the same: "./configure ; make ; make install" sequence as for the sourcecode distributed as tarballs (see the previous answer).

Go to Part 4.1: Startup Issues (LILO and GRUB)
Back to the Main Index