

Linux Newbie Guide

Table of Contents

Linux Newbie Guide	1
Linux Newbie Guide: Linux Benefits	4
0.1 Fundamentally, why Linux?	4
0.2 Is Linux for me?	5
0.3 Linux is difficult for newbies.	5
0.4 What are the Linux benefits?	5
0.5 What are the differences between Linux and UNIX?	7
0.6 What are the differences between Linux and MS Windows?	8
0.7 I don't believe in free software, etc.	8
0.8 "There ain't no such thing as a free lunch"	8
0.9 I need high security. With commercial software, I can sue them if things go wrong.	9
0.10 I need standards. Big software corporations (Microsoft) provide standards.	9
0.11 MS Windows popularity insures that it is "here to stay".	10
0.12 But LINUX may fork into many different systems	10
0.13 Linux sux etc.	11
Linux Newbie Guide: Before Installation	12
1.1 Which Linux distribution should I use?	12
1.2 What are the Linux hardware requirements?	14
1.3 Will my hardware work under Linux?	16
1.4 How do I download Linux?	17
1.5 How do I get a Linux CD?	17
1.6 I have Linux Installation CDs but no install floppy. What do I do?	17
1.7 What do I need to read before installation?	19
1.8 Can I have MS Windows and Linux installed on the same computer?	19
1.9 How do I partition my hard drive?	20
1.10 The MS Windows partition occupies my whole harddrive. Can I shrink/split it without a re-install?	24
1.11 How do I start the installation?	24
1.12 Is the Linux installation difficult?	24
1.13 Which packages should I install?	25
1.14 Which GUI desktop should I install, KDE or GNOME?	25
1.15 I finished the installation. How do I log-in for the very first time?	26
1.16 Can I use Graphical User Interface (GUI) all the time?	26
Linux Newbie Guide: Resources, Help And Links	28
2.1 Any Linux reading materials?	28
2.2 Is there a help command?	29
2.3 Any dictionary of terms?	30
2.4 Web Search	31
2.5 Newsgroups	31
2.6 Any Linux Internet links?	32
2.7 Source code--the ultimate resource	34
Linux Newbie Guide: Basic Operation FAQ	35
3.1 Basics	36
3.1.1 Filenames	36
3.1.2 What are the different directories for?	37
3.1.3 How do I run a program?	40

3.1.4 How can I change the PATH?	41
3.1.5 How can I shutdown my computer?	42
3.1.6 How do I deal with a hanged program?	43
3.2 Users, passwords, file permissions, and security	46
3.2.1 Home directories, root, adding users	46
3.2.2 About password security	47
3.2.3 I forgot the root password	49
3.2.4 I forgot my user password.	50
3.2.5 Disabling or removing a user account	50
3.2.6 I have file permission problems. How do file ownership and permissions work?	51
3.2.7 My mp3 player chokes. The sound is kind of interrupted (how to set suid).	55
3.3 Job scheduling with "&", "at", "batch", and cron	55
3.3.1 How do I execute a command in the "background"?	56
3.3.2 How do I execute a command at specified time (using "at" or "batch")?	56
3.3.3 How do I set up cron?	57
3.4 Swap space	59
3.4.1 Swap partitions	59
3.4.2 Swap files	59
3.5 Shell	60
3.5.1 What is a shell and do I want to use a different one?	60
3.5.2 How do I customize my shell prompt?	61
3.5.3 Colour on text terminal	62
3.5.4 How do I print symbols on the console or in a text mode application?	63
3.5.5 How do I write a simple shell script?	64
3.5.6 Meaning of quotes	65
3.5.7 Input/output redirection	66
3.5.8 Shell special characters (metacharacters)	68
3.6 Package installation and rpm package manager	69
3.6.1 How do I install a program I downloaded from the Internet?	69
Linux Newbie Guide: Administrator FAQ	72
4.1 Startup issues (LILO and GRUB)	72
4.1.0 LILO and GRUB	72
4.1.1 Linux cannot detect all my memory	73
4.1.2 LILO displays only LI (or LIL) and hangs	74
4.1.3 How can I change the operating system that LILO boots on default?	75
4.1.4 The LILO prompt stays too short (or too long) on the screen during the bootup	75
4.1.5 Uninstalling Linux	76
Linux Newbie Guide: Administrator FAQ	78
4.2 Accessing my drives	78
4.2.1 Where are my drives?	78
4.2.2 How can I access my CDROM?	79
4.2.3 How to mount a floppy, zip drive, DOS/Windows partition, or a network drive?	80
4.2.4 How to mount a remote MS Windows filesystem through Samba?	82
4.2.5 Any quick way to access a file on a DOS/Windows floppy?	82
4.2.6 Mounting works when I am root. Can a normal user mount?	83
4.2.7 Mounting command is too long, how can I simplify it with an alias?	84
4.2.8 Can I automount?	85
4.2.9 How do I get my parallel-port (external) Zip drive recognized?	86
4.2.10 Can I set 32-bit hard drive I/O?	87
4.2.11 I reached the limit on the number of opened files (error message)	88

Linux Newbie Guide: Administrator FAQ	89
4.3 Working with X-windows	89
4.3.1 How to switch between text and graphical consoles?	89
4.3.2 How do I setup video card, monitor and mouse for the X-server?	90
4.3.3 Can I have a GUI login prompt?	91
4.3.4 How do I install kde (e.g., on RedHat 5.2)?	91
4.3.5 How can I change my default desktop to KDE (or Gnome or yet another)	93
4.3.6 Can I have multiple sessions of X running at the same time?	94
4.3.7 Can my sister have second GUI login prompt so she does not have to kill my X-session to start hers?	94
4.3.8 How to X-window remotely?	95
4.3.9 How do I install TrueType fonts from my MS Windows partition?	96
4.3.10 How do I copy-paste?	97
Linux Newbie Guide: Administrator FAQ	99
4.4 Printer and soundcard	99
4.4.1 How to setup my soundcard?	99
4.4.2 How do I setup my printer?	100
4.4.3 Word Perfect 8 does not have a driver for my printer	102
4.4.4 Where are the setup and configuration files?	102
4.4.5 What are all the device files?	103
Some Linux daemons	103
Linux Newbie Guide: Administrator FAQ	106
4.5 Setting up a network	106
4.5.1 Would it be worth it to set up my home network?	106
4.5.2 How to set up my home network?	106
4.5.3 I have problems configuring my ppp dial out	109
4.5.4 How to browse the net from my networked computer without a modem?	112
4.5.5 How to use Samba?	113
4.5.6 Sendmail	115
4.5.7 Simple web server (running Apache)	116
4.5.8 Simple ftp server	117
4.5.9 How can one access my computer from the outside world when I am on the net using phone connection?	118
4.5.10 Can my home computer get hacked?	119
Linux Newbie Guide: Shortcuts And Commands	122
5.1 Linux essential shortcuts and sanity commands	123
5.2 Help commands	126
5.3 System info	127
5.4 Basic operations	130
5.5 File management	131
5.6 Viewing and editing files	132
5.7 Finding files	133
5.8 Basics of X-windows	134
5.9 Network apps	135
5.10 File (de)compression	137
5.11 Process control	138
5.12 Basic administration commands	139
5.13 Program installation	145
5.14 Accessing drives/partitions	146
5.15 Network administration tools	147

5.16 Music-related commands	148
5.17 Graphics-related commands	150
5.18 Small games	152
Linux Newbie Guide: Linux Applications	153
6.1 Word processing	153
6.1.1 StarOffice Suite	153
6.1.2 abiword	154
6.1.3 Word Perfect 8 for Linux	154
6.1.4 Ted	155
6.1.6 lyx and latex	155
6.2 Spreadsheet	158
6.2.1 gnumeric	158
6.2.2 kspread	158
6.3 Databases	158
6.4 CAD	158
6.5 Web browsers: Netscape and Lynx	159
6.6 Writing CD-Rs: cdrecord and cdparanoia	160
Linux Newbie Guide: Learning with Linux	166
7.1 Linux Advanced Text Processing Tools	166
7.2 Simple Programming under Linux	174
7.3 Math Tools	178
7.4 Miscellaneous	180
How do I run an MS Windows Application (using "wine")?	180
Can I have a RAID if my computer has two or more IDE (or other) harddrives?	181
Network traffic shaping using <code>shapcfig</code>	181
Linux Newbie Guide: Kernel Upgrade	182
A.1 Introduction	182
A.2 What is kernel?	183
A.3 Why upgrade?	183
A.4 Where can I get the new kernel?	183
A.5 Should I download full source or the patch only?	188
A.6 Unpacking the download file	188
A.6.1 Unpacking the patch	188
A.6.2 Unpacking the full source	189
A.7 Compiling new kernel	189
A.7.1 Using the patch	189
A.7.2 Using the full source	190
A.8 Installing the new kernel	190
A.9 What about next kernel upgrade?	193
A.10 Notice to Red Hat users	193
A.11 About the author	193
A.12 Changes	193
Linux Newbie Guide: Acknowledgments	194
Acknowledgments	194
Other Matters	194
Our Rewards and Testimonials	195
Changes	197
LNAG Sources	197
Linux Newbie Guide: LICENCE	198
LICENCE	198

LINUX NEWBIE ADMINISTRATOR GUIDE

ver. 0.120 2001-03-16 by Stan and Peter Klimas

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

Copyright (c) <1999,2000,2001> 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](#).

Quick navigation:

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). 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 wishing to set up and maintain their own Linux desktop computer, and/or administer their home or small office network. The answers are meant to be simple, with just sufficient detail, and always supported with readily applicable examples. The work is in progress--the Guide is incomplete and requires reorganizations. We welcome your corrections, advice, criticism, links, translations, and CONTRIBUTIONS. Pls note that there are no ad banners on this pages.

Part 0: For the Undecided (Linux Benefits)

If you wonder what are Linux pros and cons, 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, set up the swap space, 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, uninstall Linux, solve some memory problems ...

Part 4.2: Accessing my drives

Where are my drives, how to mount them, configure user access, get the zip drive recognized, set 32-bit hard drive IO, increase the limit on the number of opened files ...

Part 4.3: Working with 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.

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 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 that should have come first. A practical selection of Linux shortcuts and commands in a concise form. Perhaps this is everything what 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 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 learn about computers. Under development so perhaps not so good: grep, regular expressions, 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: How to upgrade the kernel (by Alesh Mustar)

All you need to know to upgrade the Linux kernel.

Acknowledgments (who helped us + why we are writing this page).

The master copy of this page: <http://sunsite.auc.dk/linux-newbie/> (Denmark, Europe)

Mirror: <http://www.linsup.com/newbie/> (Australia)

We are currently looking for a reliable mirror in North America.

A 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/>

A Chinese translation of the "Linux Shortcuts and Commands" (ver. 0.32) is available (locally) here.

The Spanish translation is available on the Spanish Esware Linux distribution CD, we were informed by <hfl@hardlogic.es>--please let us know if you see it on line. We were also informed

that the translations into the following languages may be under way: Brazilian Portuguese "Andre' Franciosi" <franciosi@disinfo.net>, German "Kaleb Yilma" <KalebY@gmx.de>, Swedish "JonasÖgren" <jonas.ogren@home.se>.

Trust automatic translations? Try this cool tool (expect problems though):

http://translator.go.com/search_trans?url=http%3A//sunsite.auc.dk/linux-newbie/&lk=noframes

The following (automaticallly generated) files are available for downloading:

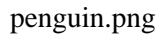
*The **pdf** version of the Linux Newbie Guide is [here](#) (~600 kB, best for printing and off-line reading).*

*The **pdf zip** version of the Linux Newbie Guide is [here](#), (slightly smaller, ~400 kB).*

*The **postscript** version of the Linux Newbie Guide is [here](#). (big, ~ 1 MB).*

*The **postscript zip** version of the Linux Newbie Guide is [here](#) (~200 kB)*

*The **html zip** version of the Linux Newbie Guide is [here](#) (~300 kB, includes all the local html files).*



Penguin from Deep River, Ontario :-)

We are proud Canadians of Silesian background.

LINUX NEWBIE ADMINISTRATOR GUIDE

ver. 0.120 2001-03-16 by Stan and Peter Klimas

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

Copyright (c) <1999,2000,2001> 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](#).

Quick navigation:

Part 0: For the Undecided (Linux Benefits)

Contents:

- 0.1 Fundamentally, why Linux? [p 4]
 - 0.2 Is Linux for me? [p 5]
 - 0.3 Linux is difficult for newbies [p 5]
 - 0.4 What are the Linux benefits? [p 5]
 - 0.5 What are the differences between Linux and UNIX? [p 7]
 - 0.6 What are the differences between Linux and MS Windows? [p 8]
 - 0.7 I don't believe in free software, etc. [p 8]
 - 0.8 "There ain't no such thing as a free lunch" [p 8]
 - 0.9 I need high security. With commercial software, I can sue them if things go wrong. [p 9]
 - 0.10 I need standards. Big software corporations (Microsoft) provide standards [p 9]
 - 0.11 MS Windows popularity insures that it is "here to stay" [p 10]
 - 0.12 But LINUX may fork into many different systems ... [p 10]
 - 0.13 Linux sux etc. [p 11]
-

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 proprietary science and unreviewed design, a plane at the internals of which nobody but the manufacturer could look? Then, why would you trust a closed, unreviewed, proprietary operating system? Linux is ideally suited for a mission-critical application.

Making horseshoes was once a closely guarded trade secret. Science and technology exploded 500 years ago thanks to sharing the 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 :-).

0.2 Is Linux for me?

Only you can answer this question. Linux is a mature, powerful 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 than MS Windows, so do not expect that if you can get around MS Windows, Linux will be obvious to 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 for you.

0.3 Linux is difficult for newbies.

This may be true. But the question is: do you really want to learn it?? None of the authors has any computing science background, yet we use Linux every day and we love it.

0.4 What are the Linux benefits?

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 Unsurpassed computing power, portability, and flexibility. A Linux cluster recently (April 1999) beat a Cray supercomputer in a standard benchmark. 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 is most popular on Intel-based PCs (price), but it runs very well on numerous other hardware platforms, from toy-like to mainframes.
- 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 installation of Linux at your school.
- 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! What 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.? What about browsing the net on two or more machines at the same time using a single Internet connection? What 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 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 a second-gear 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 Dozens of excellent and free, general-interest applications.

- 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. 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 Thousands of free applets, tools, and smaller programs. "Small is beautiful" goes well with Linux philosophy.

- o Scores of top-of-the line commercial programs including Word Perfect and 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 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, if you require it, is also free.

- o Freedom from viruses, software manufactures "features", invasion of privacy, forced upgrades, licensing and marketing schemes, 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.

- 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 and support will be provided as long as there are Linux users.

- 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.

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>

The license under which Linux is distributed is probably the most important part of it. Other important open-source projects include science and law, no kidding :), although the lawyers seem to be working to change this :(

To get a flavour of 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.

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 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. Same about 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 normal PC).
- 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 80s/early 90s, but is much sturdier and 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 an average Joe who wants to run his own small 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.
- 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).
- Windows is based on DOS, Linux is based on UNIX. Windows Graphical User Interface (GUI) is based on Microsoft-own specifications. Linux GUI is based on industry-standard X-Windows.
- Linux beats Windows hands down on network features, and as a developers platform. MS Windows desktop has a more polished appearance.
- 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 the 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. Internet works fine and is expanding at a rapid pace. So does Linux.

Here is an opinion of 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).

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

Brett Bazant <bbazant@shaw.wave.ca> wrote in LinuxToday
(<http://linuxtoday.com/cgi-bin/showtb.pl?tbsn=12450&sn=5418>):

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 opportunity cost for me, but I get to keep it even after I've given it to you. It is a free lunch only rivalled in history by the loaves and the fishes.

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.

Here is a quote from David Kastrup, Research Engineer, Bochum, Germany (after the "Internet Week", <http://www.techweb.com/se/directlink.cgi?INW19990329S0050>):

"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."

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.

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 (a MS "fast-save" anybody?). It is also completely closed--Microsoft does not publish the specifications. How can the user benefit from this in a longer term?

Xavier Basora (<http://www.osopinion.com/Opinions/XavierBasora/XavierBasora47.html>):
"... 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."

Wesley Parish (<http://www.osopinion.com/Opinions/WesleyParish/WesleyParish10.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."

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

Ted Clark (http://linuxtoday.com/news_story.php3?ltsn=2000-09-29-004-06-OP-MR-0010):
"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."

There are a few text/document oriented file formats that are quite definitely more standard than MS Word file format: ASCII, XML, HTML, SGML, LaTeX, TEX, PostScript, pdf, dvi ... and all of them have excellent support under Linux. The MS Word file format can be also read/written very well under Linux by Word Perfect, Star Office, Applixware, etc. to cover your today's needs. Advanced, "universal", open-source document formats (XML-based) are rapidly developing so watch us :) The story is similar 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).

0.11 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 so 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 Linux community.

0.12 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].

There is very little (if any) evidence of harmful forking of any software that constitute the typical Linux distribution. Where forking did occur, it always have turned beneficial. It is likely 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 a German philosopher Hegel 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."

0.13 Linux sux etc.

Then do not use Linux. You don't do 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.

In this context, it maybe 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 quite a bit 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, 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

LINUX NEWBIE ADMINISTRATOR GUIDE

ver. 0.120 2001-03-16 by Stan and Peter Klimas

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

Copyright (c) <1999,2000,2001> 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](#).

Quick navigation:

Part 1. Before Linux Installation

Contents:

- 1.1 Which Linux distribution should I use? [p 12]
 - 1.2 What are the Linux hardware requirements? [p 14]
 - 1.3 Will my hardware work under Linux? [p 16]
 - 1.4 How do I download Linux? [p 17]
 - 1.5 How do I get a Linux CD? [p 17]
 - 1.6 I have RedHat CD but no install floppy. What do I do? [p 17]
 - 1.7 What do I need to read before installation? [p 19]
 - 1.8 Can I have MS Windows and Linux installed on the same computer? [p 19]
 - 1.9 How to I partition my hard drive? [p 20]
 - 1.10 The MS Windows partition occupies my whole harddrive. Can I shrink/split it without a re-install? [p 24]
 - 1.11 How do I start the installation? [p 24]
 - 1.12 Is the Linux installation difficult? [p 24]
 - 1.13 Which packages should I install? [p 25]
 - 1.14 Which GUI desktop should I install, KDE or GNOME? [p 25]
 - 1.15 I finished the installation. How do I login for the very first time? [p 26]
 - 1.16 Can I use Graphical User Interface (GUI) all the time? [p 26]
-

1.1 Which Linux distribution should I use?

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 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 any problems. However, Mandrake makes a much nicer desktop than RedHat.

In short, as a newbie, you can safely bet on "Mandrake" or "RedHat" unless you like something else or have specialized needs. The most recent Mandrake distribution (December 2000) is Mandrake 7.2. This is an excellent distribution, much better than the most recent RedHat offering (RedHat 7.0). 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 old version (this applies not only to Linux). The authors have no connection to RedHat, Mandrake (or any other Linux distributor) whatsoever.

Our recommending 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 an 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 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.

Corel has been 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 is based on Debian distribution. So if you think about using Corel in the future, picking Debian may be your smart choice now. There is an uncertainty about commercial viability of the Corel corporation, so I wouldn't put all my eggs in the Corel basket. RedHat, Mandrake or Debian seem rock solid in comparison with the young and shaky Corel offering.

Slackware seems to be favorite among "cutting-edge hackers" who like being close to the operating system and perhaps upgrade their kernel every other day--we did not use it so this is hearsay. We would have trouble recommending Slackware for Linux newbies.

S.u.S.E distribution is very popular in Europe. It surely looks German--a solid, general-purpose distro with an easy setup and an excellent reputation. We couldn't find cheap Suse CDs though.

Caldera is another, well-known and respected distribution. It is said to be aiming at corporate users--it has the most fancy, perhaps most easy installation program. Recently (August 2000) Caldera purchased SCO Unix (the original UNIX, including the UNIX trademark) which gives them even more "corporate" look in my eyes. Perhaps one can now say that Linux is UNIX.

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 probably want to see.

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 for a small "special purpose" device, which could be good for the next-generation stereo, MP3 player, palm computer, or 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, 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 distribution for other hardware platforms too: PowerPC, Alpha, Apple, IBM mainframe, "Intel ArmStrong", 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" processor inside.

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.

1.2 What are the Linux hardware requirements?

"Out-of-box" Linux will 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 will just do for work under X-windows but don't expect it to fly. My 586-90 MHz with 64 MB of memory flies under Linux. My Pentium-2 233 MHz with 64 MB of memory is a complete pleasure with an almost instantaneous response even when running many large applications concurrently. 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 useful in my home network environment (a 486-class machine performs just adequately also stand-alone if it has at least 16 MB of memory). 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 Debian Linux on this Toshiba too, and it runs fine in text mode.

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 16 MB of memory, and if possible 32 or 64 MB --more memory can make a difference in performance when running concurrently several large GUI applications. Memory is cheap these days.

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 you will use if you ever have a permanent connection to the outside world.

Here is another suggestion on setting up a different kind of network, older type, which uses coaxial cables. 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 metres 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 TV & video connector but with a bayonet that secures the two sockets together.

For more detail 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 maybe \$20 less expensive than full modems, but they are crippled (some processing is done by the main computer CPU instead by the modem) and there is little chance they will ever work under a fully multi-user, multi-tasking operating system (they may not even work with the next version of MS Windows). 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).

Another area of potential problem 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://www.redhat.com/support/docs/hardware.html> 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-preinstalled from a smaller vendor.

If you are adventurous person, as I am, I would pay no attention to the remarks above, chances are 90-10 that my 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, get an installation CD instead. Linux can be downloaded completely from the Internet, but it is a large and sophisticated operating system. The download will take hours or days of download time and you will probably encounter problems when trying to stitch the downloaded pieces together.

If you do have a speedy Internet connection (definitely not a 56k-modem) and you are not a complete newbie, Linux download may be an option to you after all. Try:
`ftp://ftp.redhat.com/pub/` if the site is not too busy today.

1.5 How do I get a Linux CD?

Many possible ways. (1) Buy the "Mandrake" or "RedHat" CD 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?). (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! It is definitely a good idea to have a nice Linux handbook. With Linux countless utilities, I need a handbook all the time. (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 1 or more 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. (6) If you are really short on cash or are a Linux pioneer in one of these places in the world where US\$3 goes a long way, try "linux exchange" or "giveaway"--people loan or donate Linux CDs (e.g. try: <http://visar.csustan.edu/giveaway.html>). Once again, if you have a Linux CD, don't be shy to loan it to your neighbor.

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 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. 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 necessarily reflect the capability of your machine. If you can boot from 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 key. Here is a list of key combinations used by popular BIOSes: Acer notebooks: <F2> during Power-On Self-Test (POST). American Megatrends (AMI): during Power-On Self-Test. Award: , 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>, 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 got 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 that the DOS `FORMAT` utility will find 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?

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.

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 basic Unix commands will run fine under Linux. Manuals for MS Windows are 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 are your DOS/Windows partitions, 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). 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 the the monitor specification--this may damage your monitor.

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

Yes, you can. Many Linuxers use dual boots. 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, which will let you choose at the boot time the operating system you boot. Install MS Windows first and Linux only afterwards or else 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 get back to business.

From under Linux, you will be able to read from and write to your MS Windows drive partitions so that data exchange between MS Windows- and 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. 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 partition to dual boot. This means: MS Windows is on one partition, Linux 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 MS Windows partition, but it is not recommended by us.

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 a part of the hard drive(s) unpartitioned for Linux. You will make and format the Linux partitions during your RedHat (or Mandrake) installation. 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 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 you can keep users' data separate from the rest of the operating system. This way, if something ever goes wrong and you have to reformat and re-install the operating system, you don't lose the users' data. (You can however do a 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.)

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 that I dedicated for Linux):

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 give 1200 MB for the mount point that will be visible on my filesystem as the /usr directory and will contain 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 give 120 MB for 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). 120 MB is quite a bit of swap and it is unlikely you should need more. If you think you do need more (e.g. 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 swap a file(s) later.

2 GB is a lot of disk space and should be sufficient even 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 for Linux on the Internet,

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 for Linux):

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

In this example, I dedicate 600 MB to contain the base of the operating system, applications, and user documents/data, and give 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 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 come on Red Hat CD--it will give you a short description 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 even more selective as to what you install.

For a larger available hard drive space, I may consider the following setup (for a comfortable total of 8 GB dedicated for Linux):

mount point	type	size
/	ext2	800 MB
/usr	ext2	3600 MB
/usr/local	ext2	2500 MB
/home	ext2	800 MB
swap	swap	300 MB

Please note that the the mount points can reside on different physical hard drives.

Another consideration when setting up the partitions. 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 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 a SCSI device "sda4" (zip drives work in a SCSI-emulation mode).

The listing of partitions, that your Linux setup program shows you during installation, will include any MS Windows partitions which you may 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 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 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 andman mount., with some additions after I had a look at the source code files at /usr/src/linux/fs. The underline 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.

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 an 8 character name 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 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 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 provides 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 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 late beta version., i.e., not recommended for use with critical data, unless read-only.

ntfs--MS Windows NT filesystem. It is still "experimental" under Linux, i.e. not recommended for production machines, unless read-only (Feb.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 your 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 the partition. 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 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.

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 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), 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 for making 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 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--no panic. It can be easily installed later. Read on.

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

Space permitting, definitely both. You can later decide if you prefer GNOME or KDE, 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 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 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 power hungry. On older hardware (e.g., 133 MHz Pentium) I prefer GNOME to KDE2.0. Other windows managers are lighter than either KDE or GNOME. Therefore, on really modest hardware, I chose 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 GUI login screen, the login procedure looks similar, but is in an X-window screen. 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 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 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 it is inflexible and clumsy. The command line is a far richer interface to me, and it also gives me some understanding of the working of my system.

Thus, 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 waiting with Linux installation for another year or two, unless you have somebody to administer your computer.

From this Guide point of view, command line is certainly simpler for documentation of commands. 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, command line is indispensable for any work beyond trivialities.

On the other hand, if you have a computer-agnostic girlfriend or boyfriend, and all s/he does is executing perhaps half-a-dozen different programs, you can set up a nice GUI screen for him/her with the icons or buttons or menus s/he requires to launch them. This, after some initial encouragement, may make him accept Linux with no problems.

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

Back to Main Page

LINUX NEWBIE ADMINISTRATOR GUIDE

ver. 0.120 2001-03-16 by Stan and Peter Klimas

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

Copyright (c) <1999,2000,2001> 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](#).

Quick navigation:

Part 2. Linux Resources, Help and Some Links

Contents:

- 2.1 Any Linux reading materials? [p 28]
 - 2.2 Is there a help command? [p 29]
 - 2.3 Any dictionary of terms? [p 30]
 - 2.4 Web search [p 31]
 - 2.5 Newsgroups [p 31]
 - 2.6 Linux Internet links [p 32]
 - 2.7 Source code--the ultimate resource [p 34]
-

2.1 Any Linux reading materials?

The RedHat Linux distribution CD contains lots of documentation. Part of it is in the 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.htm`. (assuming your CDROM is the 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 MINHOWTOs 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.htm
```

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/doc/LDP/sag  
lynx sag.html
```

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

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

For more or updated documentation, see <http://metalab.unc.edu/mdw/index.html#guide>.

2.2 Is there a help command?

Most Linux commands can be run with the `--help` option. For example, this command will give you a 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 wealth of detailed, very technical information, but typically require an 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` command, 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
```

In some menu driven programs, for example when configuring your system using `linuxconf`, or under `setup` when browsing the system services, you may press F1 for info 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 windows manager includes a GUI-based "help browser", which can be started by clicking the "book with a light bulb" icon on the Kpanel. 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://college.antioch.edu/~totally/geek.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>.

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

These can be a 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 accessing the information you need.

Before going to the newsgroups, I would highly recommend DejaNews (http://www.dejanews.com/home_ps.shtml). 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 Dejanews.

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 how to post.)

news:comp.os.linux.help

news:comp.os.linux.setup

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).

Just for my reference, here are some useful abbreviations/acronyms not found in a standard Webster dictionary:

BTW = "By the way, "

AFAIK = "As far as I know," (S/he may actually know the best, but is being cautious)

RTFD = "Read the 'fine' documentation!" (Impatient and probably rude since it contains the f-word)

ASAP = "as soon as possible"

FYI = "For your information," (Perhaps implying everybody else knows about it.)

IMHO = "In my humble opinion," (Very polite or perhaps showing extreme self confidence.)

c.o.l.a. = The newsgroup com.os.linux.announce (or perhaps comp.os.linux.advocacy?)

tia = "thanks in advance!"

ty = "thank you"

iirc = "if I remember correctly," (cannot give the proper source, eh?)

fwiw = "For what it's worth" (The responder feels that the answer may not be a definitive one.)

lol = "laugh out loud" (showing really good mood or happiness from what you said)

bs = "bull shit". Same as male-bovine excrement (mbe). (The responder feels that what you say is an absolute rubbish.).

:-) = a smile to you. Many variants are possible, also showing other faces to express moods

:-D = a big smile to you

: (= sad

|-o = yawning because of what you say

:O = surprised

~:-| = steamed

:'(= crying

*:-o) = clown

0:-) = angel

;-) = a wink to you ("you know what I mean, don't you?")

:-P = raspberries (not very respectful vibration of the tongue, expression of contempt)

:-@ = screaming

:-& = tongue-tied ("cannot tell")

<:-| = dunce (a hat put on the head of a stupid person)

/. = a website called "slashdot" (<http://www.slashdot.org/>) ("news for nerds, stuff that matters")

oic = oh I see

k = ok

bf = "boyfriend", gf = "girlfriend"

fu = this couldn't mean "fuck you", could it?

S = "smiles"

brb = "will be right back"

Please mail me if you know of other interesting abbreviations that a Linux administrator ought to know :-)

To read newsgroups (also called usenet), you have to configure your access to a newsserver. The simplest may be to configure your netscape ("edit"- "preferences"- "mail and newsgroups") by 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 for clumpsy netscape. 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.auc.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.slashdot.org/	Discussions for nerd, 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 adding this guide!
http://www.debian.org/	Debian Linux site.
http://www.redhat.com/support/docs/hardware.html	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

LINUXNEWBIEADMINISTRATORGUIDE

ver. 0.120 2001-03-16 by Stan and Peter Klimas

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

Copyright (c) <1999,2000,2001> 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](#).

Quick navigation:

Part 3: Basic Operations FAQ

Contents:

3.1 Basics

- 3.1.1 Filenames [p 36]
- 3.1.2 What are the different directories for? [p 37]
- 3.1.3 How do I run a program? [p 40]
- 3.1.4 How can I change the PATH? [p 41]
- 3.1.5 How can I shutdown my computer? [p 42]
- 3.1.6 How do I deal with a hanged program? [p 43]

3.2 Users, passwords, file permissions, and security

- 3.2.1 Home directories, root, adding user [p 46]
- 3.2.2 About password security [p 47]
- 3.2.3 I forgot the root password [p 49]
- 3.2.4 I forgot my user password [p 50]
- 3.2.5 Disabling or removing a user account [p 50]
- 3.2.6 I have file permission problems. How do file ownership and permissions work? [p 51]
- 3.2.7 My mp3 player chokes. The sound is kind of interrupted (how to set suid) [p 55]

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

- 3.3.1 How do I execute a command in the "background"? [p 56]
- 3.3.2 How do I execute a command at a specified time (using "at" or "batch")? [p 56]
- 3.3.3 How do I set up cron? [p 57]

3.4 Swap space [p 59]

- 3.4.1 Swap partitions [p 59]
- 3.4.2 Swap files [p 59]

3.5 Shell

- 3.5.1 What's a shell and do I want to use a different one? [p 60]
- 3.5.2 How do I customize my shell prompt? [p 61]
- 3.5.3 Colour in text terminal [p 62]
- 3.5.4 How do I print symbols on the console or in a text mode application? [p 63]
- 3.5.5 How do I write a simple shell script? [p 64]
- 3.5.6 Meaning of quotes [p 65]
- 3.5.7 Input/output redirection [p 66]
- 3.5.8 Shell special characters (metacharacters) [p 68]

3.6 Package installation and rpm package manager

- 3.6.1 How do I install a program I downloaded from the Internet? [p 69]
-

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 the 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 space inside 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 directory respectively, exactly like in DOS.

To manipulate files with names that do contain metacharacters, I use a pair of ' (apostrophes), so that the metacharacters are quoted and therefore the shell does not interpret their meaning. For example, to remove a file `my file*` (contains space and asterisk), I would issue:

```
rm 'my file*'
```

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.

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.

Examples. This command will show 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 are typically used to contain user-level configurations.

This command will show any filename that contains a dot:

```
ls *.*.
```

This command will show any filename that contains two dots:

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

```
ls [abA-Z]*
```

As an example of 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 (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 just serves 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 rather stick to traditional naming conventions than face the occasional complications.

3.1.2 What are the different directories for?

This is explained very nicely in the Linux System Administrator Guide (SAG), which should be available on your system. Try:

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

This will start a text-mode browser "lynx" to view this html book. You can also use any other browser, e.g. Netscape for Linux or Windows, to view this book. This book and other LDP books are actually quite easy to read.

Briefly, 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 distribution, for example Netscape.

/var filesystem contains files that change: spool directories, log files, lock files, temporary files, and formatted 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 operating system upgrade.

/proc filesystem contains entirely illusionary files. They don't really exist on the disk and don't take any space there (although `ls -l` will show their size). When viewing them, you really access information stored in 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 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 appears 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 an option 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 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 many commercial UNIX systems.

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

The `/proc` "psuedo" file system is a real-time, memory-resident file system that tracks state of operating system kernel and the 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 presistent media, it exists only in the computer memory, and it is constantly update 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 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 ID 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: you don't 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 your PATH. Under Linux (or UNIX), an executable must be on your PATH to run it, and the current directory is NOT on your PATH. Type the full path to the executable before the executable name, or execute:

```
cd the_program_directory
./program_name
```

You 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).

You may check your path using:

```
echo $PATH
```

To learn how to change your PATH, or add your current directory to it, see the next answer [p 41] .

If your executable is lost somewhere in your directory tree, you may want to find it using (for example):

```
find / -name "netscape"
```

to find a file named "netscape" searching 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 search for a file that you just installed.)

Please note that the PATH is normally different for root than regular users (root's PATH includes `/sbin` and `/usr/sbin` whereas users' don't). Therefore users cannot execute command 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 you need to learn where an executable which is on your PATH is located on your system (i.e., the executable runs by typing its name anywhere in the system, but you would like to know where it is), you may use something like this:

```
which netscape
```

which will show a full PATH to the executable program called "netscape".

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 [p 51] if you don't understand the output of this command or the whole "third possibility".

Please note that under Linux (or UNIX), it is not the file extension (for example .exe or .com or .bat) that makes a 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 for the program the execution of which you request. 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?

In a text terminal, press `<Ctrl><Alt>` (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 `<Ctrl><Alt><F1>` (three keys simultaneously). Do not 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>` (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 performing the shutdown. 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 a 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>`. It can be also 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.

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 associated with hardware problems and it most likely to manifest itself when you perform computing-intensive tasks: Linux setup, kernel compilation, etc.

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 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.

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 immediately --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
```

3.2 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]
```

So 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 on to a graphical (GUI) screen,
- 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 a 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 "how 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 a 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 30,000 or so most frequent last names. Or just check the Toronto telephone book. Another proof that we are all one family :))

- The nickname of your dog, wife, cannary or computer (very few nick names the humans use).
- 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.
- 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. Try the program `crack` yourself to find how easy it is.
- 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? Just grap a few more files and append it to your cracking list. The point here is that the subset of words that the humans normally use if far far below the theoretical limit of 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), contains a mixture of letters (upper and lower case, if possible), numbers and special characters, and is changed quite regularly (8-16 weeks?).

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 the 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 "over the phone" Internet connection for just a couple of hours a week, you may be fine even with a relatively weak passwords 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.

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 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, 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.

The information about a user account is kept in plain-text files: `/etc/passwd` and `/etc/shadow`.

The file `/etc/passwd` contains "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, 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 the 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
```

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 so easily crackable, or one could remove the small battery that sustains 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 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 old password required). If a 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 left your terminal unattended.)

3.2.5 Disabling or removing a user account

A user account can be disabled or removed.

To temporarily disable 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 star 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$fhAlpuOU422HiSL5aggLI/:11193:0:99999:7:-1:-1:134539228
```

To irreversibly remove a user account on 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

```
userdel 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 the name identical to that of the owner, but it does not need to be so. A file can be removed (erased) only 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 content of the file if they are given the 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 the 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-in the file ownership 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 "d" for directories, "-" for normal files, "l" for links, "c" for character devices, "b" for block devices, "p" for named pipes like FIFO files). After this 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 other 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 complete set 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 a proof that Linux (any UNIX for that matter) is a 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 base 8 (the decimal system's base is 10). The highest digit in the octal system is 7 (octal system has eight digits: 0 to 7, analogous to 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=permissions 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 to 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 a 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 system 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 owner (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 origin of which is well traceable. However, even at home, I wouldn't `suid` a piece of code origin of which 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.

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 command `ps` to find the *process_ID* of the process to kill).

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 `setup-"system services"`). 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 a 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,
- rebuilding 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 nights, 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 `/etc/cron.daily` or `/etc/cron.weekly`, or `/etc/cron.monthly`.

Here is an example of 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 (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

> in 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 Swap space

Swap is an extension of the physical memory of the computer. Most likely, you created a swap partition during the initial RedHat setup. You can verify the amount of swap space available on your system using:

```
cat /proc/meminfo
```

The general recommendation is that one should have: at least 4 MB swap space, at least 32 MB total (physical+swap) memory for a system running command-line-only, at least 64 MB of total (physical+swap) memory for a system running X-windows, and swap space at least 1.5 times the amount of the physical memory on the system.

If this is too complicated, you might want to have a swap twice as large as your physical (silicon) memory, but not less than 64 MB.

If you ever need to change your swap, here are some basics.

3.4.1 Swap partitions

You can have several swap partitions. [Older Linux kernels limit the size of each swap partition to up to approximately 124 MB, but the linux kernels 2.2.x up do not have this restriction.] Here are the steps to create and enable a swap partition:

- Create the partition of the proper size using `fdisk` (partition type 82, "Linux swap").
- Format the partition checking for bad blocks, for example:

```
mkswap -c /dev/hda4
```

You have to substitute `/dev/hda4` with your partition name. Since I did not specify the partition size, it will be automatically detected.

- Enable the swap, for example:

```
swapon /dev/hda4
```

To have the swap enabled automatically at bootup, you have to include the appropriate entry into the file `/etc/fstab`, for example:

```
/dev/hda4 swap swap defaults 0 0
```

If you ever need to disable the swap, you can do it with (as root):

```
swapoff /dev/hda4
```

3.4.2 Swap files

Swapping to files is usually slower than swapping to a raw partition, so this is not the recommended permanent swapping technique. Creating a swap file, however, can be a quick fix if you temporarily need more swap space. You can have up to 8 swap files, each with size of up to 16 MB. Here are the steps for making a swap file:

- Create a file with the size of your swap file:

```
dd if=/dev/zero of=/swapfile bs=1024 count=8192
```

This physically creates the swap file `/swapfile`, the block size is 1024 bytes, the file contains 8192 blocks, the total size is about 8 MB. [The `dd` command copies files. In the example above, the input file (`if`) was `/dev/zero`, the output file (`of`) was `/swapfile`. You cannot use the `cp` (copy) command for creating a swap file because the swap file must be physically continuous on the hard drive.]

- Set up the file with the command:

```
mkswap /swapfile 8192
```

- Force writing the buffer cache to disk by issuing the command:

```
sync
```

- Enable the swap with the command:

```
swapon /swapfile
```

When you are done using the swap file, you can turn it off and remove:

```
swapoff /swapfile  
rm /swapfile
```

You may also want to see the nice info written by Linus Torvalds himself:

```
man mkswap
```

3.5 Shell

3.5.1 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 an equivalent of the DOS "command.com" (command-line interpreter) and the shell script files as an 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), `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

prospective, 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
```

The shell for each user is specified as the last field in the 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.

3.5.2 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 quotes. Here is the meaning of some special codes I may also choose to use:

<code>\u</code>	-	username of the current user,
<code>\h</code>	-	the name of the computer running the shell (hostname),
<code>\W</code>	-	the base of the name of the current working directory,
<code>\w</code>	-	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,
\d - current date,
\t - current time,
\s - name of the shell,
\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 those are really rarely seen.

3.5.3 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 the line 15, column 40
\007        bell (beep)

```

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

See man console_codes for more.

3.5.4 How do I print symbols on the console or in a text mode application?

The procedure described here gives me a fast access to the PC extended characters set (codes 128-255) and it quite portable in the PC world: it work in MS Windows, DOS (if you have an ANSI driver installed), and inside any text mode Linux application (including on the command line), but it does not work in X-windows (don't ask me why). 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 these 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	¹	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	º	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		

3.5.5 How do I write a simple shell script?

Create a text (ASCII) file that is to 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 me 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.]

3.5.6 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 be also accomplished 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").

3.5.7 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 redirect 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 be ever 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 (identical) 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.

3.5.8 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 [p 65] .

< and > are used for input/output redirection and were describe before [p 66] .

| 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 matches 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
```

\$ precedes 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.

3.6 Package installation and rpm package manager

3.6.1 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 what I do 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 -qp 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 is distributed to the directories where it belongs (rpm knows where it belongs). 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).

INSTALLATION FROM A SOURCE-CODE TARBALL

o If what I downloaded from the net is a Linux source code in a 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 is 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 a look how the new directory is called, then I cd into it:

```
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 (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. addition of their directory to the 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 that 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
```

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 put 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), 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: Linux Newbie Administrator FAQ

Back to the Main Index

LINUXNEWBIEADMINISTRATORGUIDE

ver. 0.120 2001-03-16 by Stan and Peter Klimas

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

Copyright (c) <1999,2000,2001> 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](#).

Quick navigation:

Part 4.1: Linux Newbie Administrator FAQ - LILO issues

Contents:

4.1 Startup issues (LILO and GRUB)

4.1.0 LILO and GRUB [p 72]

4.1.1 Linux cannot detect all my memory [p 73]

4.1.2 LILO displays only LI (or LIL) and hangs [p 74]

4.1.3 How can I change the operating system that LILO boots on default? [p 75]

4.1.4 The LILO prompt stays too short (or too long) on the screen during the bootup [p 75]

4.1.5 Uninstalling Linux [p 76]

4.1 Startup issues (LILO and GRUB)

4.1.0 LILO and GRUB

Both the newer GRUB and the older LILO are boot loaders. They make it possible for you to select the operating system to boot at the boot time. Most (all?) of the booting problems described in this section can likely be overcome by installing the most recent Linux kernel and the latest GRUB boot loader. GRUB is better than LILO because LILO relies on the absolute hard drive addresses to find the boot image, while GRUB understands the filesystems and looks for a file containing the boot image. We recommend Mandrake 7.2--it has a nice 2.2 kernel and the GRUB boot loader. RedHat 7.0 still relies on the older LILO.

The main GRUB configuration file is `/boot/grub/menu.lst`. Here are comments on the items found in this file:

Lines starting with the # mark are comments.

`timeout 5`

the above setting starts booting the default operating system after 5 seconds unless a key is pressed

`default 0`

the above setting makes the default operating system to be the first found in the menu list
#

`title linux`

`kernel (hd0,2)/boot/vmlinuz root=/dev/hda3 mem=64M`

The above two lines define a boottime menu item, and set the boot action for this item.

The first line names the menu item "linux".

The second line specifies that the kernel is located on the first physical hard

```
# drive (hd0), the second partition (2), the boot image is the file /boot/vmlinux
# Also on the second line, the following options are passed to the kernel:
# root=/dev/hda3 (i.e., make the root partition the 3 partition on the first hard drive (hda) ),
# mem=64M (i.e., use 64 megabytes of physical memory).
```

Good documentation for GRUB is available using `info grub`

4.1.1 Linux cannot detect all my memory

If you have more than 64 megabytes of physical memory, Linux kernel ver. 2.0.36 or lower will use, by default, only the first 64 MB. To see how much memory Linux uses on your system, type:

```
cat /proc/meminfo
```

or

```
free
```

You can check your version of Linux kernel with:

```
uname -a
```

The last popular kernel with the "memory problem", 2.0.36, comes with RedHat 5.2. My RedHat 6.0 came with kernel 2.2.5-15 so it does not have the "memory problem" any more.

To get more than 64 MB memory recognized on RH5.2, you have to edit (as root) the file `/etc/lilo.conf`, and add a line like this just before your first "image=" statement:

```
append="mem=80M"
```

If you have an amount of memory different than 80 MB, adjust the above line. For any changes in `/etc/lilo.conf` to take effect, you *must* re-run the program

```
lilo
```

(watch if it runs without errors) and reboot. After the reboot, you can check if your adjustment worked using either of these two commands:

```
cat /proc/meminfo
free
```

For testing purposes, or if you are having problems, the option of specifying the amount of memory at the LILO prompt is useful:

```
[type at LILO prompt during bootup] linux "mem=16M"
```

Occasionally, I hear the advice to skip the upper few megabytes if you have problems enabling all your memory, or the machine locks up. E.g., enable only 78 out of your 80 MB. This is apparently the case for some SCSI controllers that use the very upper chunk of the main memory. Take it for what it's worth.

Occasionally on some systems, Linux recognizes only 16 MB of memory. This is usually linked to the setting "memory hole at 15-16 MB" enabled in the BIOS setup (the solution is to disable this BIOS setting). It is probably a good idea to disable all "advanced" features in your BIOS setup anyway (for example, the BIOS virus detection seems to be a common source of problems).

Mixture of memory chips with different timings can also lead to memory recognition problems or to system crashes (the solution is to replace the memory chips so that the timing of all memory chips is the same).

4.1.2 LILO displays only LI (or LIL) and hangs

I quote from my good handbook "Red Hat Linux Unleashed" by Kamran Hussain, Timothy Parker, et al., published by SAMS Publishing:

"When LILO loads itself, it displays the word LILO. Each letter is printed before or after performing some specific action. If LILO fails at some point, the letters printed so far can be used to identify the problem. [...]"

LI [...] This is caused either by geometry mismatch or by moving `/etc/lilo/boot.b` without running the map installer.

LIL [...] This is typically caused by media failure or geometry mismatch."

The geometry means the number of sectors/heads/cylinders used in the hard drive configuration of your BIOS. Hope this helps!

It is a very good idea to have a handbook for Linux or at least a general UNIX handbook. Handbooks for Windows are useless, handbooks for Linux are great! "Red Hat Linux Unleashed" is a very good handbook but I am sure there are many other equally good ones.

With a LILO error like above, you can boot your machine using a Linux or DOS boot floppy. There seems to be several general possibilities to correct such a LILO error, depending on what is wrong:

1. If LILO simply got corrupted (does not seem very common), you can remove and re-install it. You can remove LILO by running under Linux:

```
lilo -u /dev/hda
```

or, under DOS:

```
FDISK/MBR
```

which rewrites the hard drive master boot record (MBR), in which LILO resides, and replaces it with "clean" DOS stuff. You will lose access to Linux if you rebooted your computer after removing LILO (if this happens, you can boot Linux from the floppy and re-install LILO on top of the DOS MBR).

To re-install LILO, simply re-run the command `lilo` (as root).

2. Specify the option

`linear`

in your `/etc/lilo.conf` file. This is particularly useful for large drives (>8 GB). See `man lilo.conf` for details. The option "linear" is safe--it should not affect a properly working system, so you can specify this option in any case.

You may also want to play with the method by which BIOS accesses your harddrive. For example, turn the LBA ("linear or large block access") mode on/off in your BIOS to see if this helps.

3. Look into your BIOS setup and figure out how the specified hard drive geometry does not match your hard drive. From under Linux, you can display the hard drive geometry using (for example, for the first IDE hard drive):

```
hdparm -g /dev/hda
```

You can typically easily find the recommended manufacturer geometry on the web using Google to search for your harddrive model number.

4. Put LILO on another partition (different hard drive) and, using `fdisk`, make this partition bootable (if your system supports booting from another drive). Or swap your harddrives so that the one that is better supported by your old BIOS comes as the first hard drive on your first IDE interface (DOS drive "C").

5. Maybe you prefer to use "loadlin" instead of "lilo". From under DOS, locate your CDROM and see the program `/dosutils/loadlin.exe`. It boots Linux from DOS.

6. Get rid of LILO and use GRUB. Mandrake 7.2 contains GRUB as the default boot loader.

4.1.3 How can I change the operating system that LILO boots on default?

There are typically two sections in the file `/etc/lilo.conf`. One starts with "image=" and it is relevant to Linux, the other starts with "other=" and this is the dos/win section. The first to appear in `/etc/lilo.conf` is booted by default. So to change the system that boots on default, swap the sections. Don't forget to re-run the command

```
lilo
```

after any changes to the `/etc/lilo.conf` file.

4.1.4 The LILO prompt stays too short (or too long) on the screen during the bootstrap

Add or adjust the line

```
delay=100
```

right before the first "image=" or "append=" statement in your `/etc/lilo.conf` file. The number is the time of delay in tenths of a second (0.1 s), so in the example above the delay will be 10 seconds. Don't forget to re-run `lilo` after making the changes to the `/etc/lilo.conf` file, or the delay will not change.

4.1.5 Uninstalling Linux

If you really wanted to "uninstall" Linux, you can run the following two commands (from under DOS or MS Windows):

```
LOCK C:
FDISK/MBR
```

which will get rid of LILO--it overwrites the master boot record (MBR) of your first hard drive, where LILO resides. The "lock" command allows "raw" writing to disk, which is normally disallowed on more recent DOS versions as an antiviral measure. The problem with FDISK/MBR is that it does not report back any success or failure, so it is better to proceed it with the "lock" command. After this you can remove the Linux partitions using the DOS "FDISK" utility to re-claim the hard drive space.

Apparently, MS FDISK does not always cope with removing the Linux partitions. In this case, I may use `linux fdisk`. The simplest may be to boot from Linux installation floppy/CD, and remove the partition using the Linux partitioning tool when it pops up during the "installation" procedure.

If you still have problems, here are the ultimate solutions for zeroing the MBR (after: <http://www.linuxgazette.com/issue63/okopnik.html>, edited out for space):

```
*****
```

Note: The following advice will completely wipe your Master Boot Record (MBR), which contains all your partition information. DO NOT DO THIS unless you know that this is exactly the result you want - it will leave your hard drive in an unbootable state, in effect bringing it back to "factory-fresh", i.e., empty of data and requiring partitioning and formatting.

```
*****
```

Linux-based solution. If you can boot Linux - say via boot floppy - you can simply invoke "dd":

```
dd if=/dev/zero of=/dev/hda bs=512 count=1
```

This fills up the MBR with zeros. Obviously, you have to be root to do this.

DOS-based solution . Boot with a DOS floppy that has "debug" on it; run

```
debug
```

At the '-' prompt, "block-fill" a 512-byte chunk of memory with zeroes:

```
f 9000:0 200 0
```


Start assembly mode with the 'a' command, and enter the following code:

```
mov dx,9000
mov es,dx
xor bx,bx
mov cx,0001
mov dx,0080
mov ax,0301
int 13
int 20
```

Press <Enter> to exit assembly mode, take a deep breath - and press "g" to execute, then "q" to quit "debug". Your HD is now in a virgin state, and ready for partitioning and installation.

Go to Part: 4.2 - Accessing my drives

[Back to Main Page](#)

LINUX NEWBIE ADMINISTRATOR GUIDE

ver. 0.120 2001-03-16 by Stan and Peter Klimas

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

Copyright (c) <1999,2000,2001> 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](#).

Quick navigation:

Part 4.2: Linux Newbie Administrator FAQ - Accessing my drives

Contents:

4.2 Accessing my drives

- 4.2.1 Where are my drives? [p 78]
 - 4.2.2 How can I access my CDROM? [p 79]
 - 4.2.3 How to mount a floppy, zip drive, dos partition, or a network drive? [p 80]
 - 4.2.4 How to mount a remote MS Windows filesystem through Samba? [p 82]
 - 4.2.5 Any quick way to access a file on a DOS/Windows floppy? [p 82]
 - 4.2.6 Mounting works when I am root. Can a normal user mount? [p 83]
 - 4.2.7 Mounting command is too long, how can I simplify it with an alias? [p 84]
 - 4.2.8 Can I automount? [p 85]
 - 4.2.9 How do I get my zip drive recognized? [p 86]
 - 4.2.10 Can I set 32-bit hard drive IO? [p 87] I
 - 4.2.11 I reached the limit on the number of opened files (error message) [p 88]
-

4.2 Accessing my drives

4.2.1 Where are my drives?

Linux shows all the directories in one directory tree, irrespectively of what drives/hardware they are found on. Generally, this is a much better solution than the traditional DOS/Windows model--it completely abstracts the file system from the underlying hardware. You will appreciate this if you ever have to re-arrange or expand your hardware or add network resources. But for the users who are accustomed to the DOS way of dealing with drives, it adds some extra complexity.

In short, don't search for drive letters. There are none under Linux; the content of your disks appears as subdirectories on your single Linux filesystem (directory tree). On default, the content of removable media does not appear automatically in these subdirectories--you have to "mount" your drives. See the next answers for details. You should also unmount a drive before ejecting the media.

You can access (read and write) a variety of drives and file systems from under Linux. This includes native Linux partitions, DOS and MS Windows partitions (on hard drives or floppies), ZIP and Jazz drives, and CDROM disks. Many less common file system types are also supported. This means that you can download your Linux software using Netscape for Windows, save the downloaded file on you MS Windows hard drive partition, and then boot Linux and copy the downloaded software from the Windows partition on you harddrive to the Linux partition, and

finally install the software under Linux.

4.2.2 How can I access my CDROM?

Mount it. The mounting adds all the directories and files from your CD to your Linux directory tree so you can easily access them without the drive letter.

As root, you can mount the CDROM with a command like this:

```
mount -t auto /dev/cdrom /mnt/cdrom
```

If this works, the content of your CD appears in the directory /mnt/cdrom

Chances are this command will not work for you right away--you may have to customize it. Here is how it works.

The command tells the operating system to mount a filesystem autodetecting the filesystem type ("-t auto"). The device is /dev/cdrom. The mountpoint (the directory where to which "mounting" takes place) is /mnt/cdrom. This directory must exist and be empty. If it does not exist, create it with:

```
mkdir /mnt/cdrom
```

If the mounting command fails, make sure that the device /dev/cdrom exists. If it doesn't, where is your CDROM? Chances are it is something like /dev/hdb if you have an IDE CDROM. Try /dev/hdb instead of /dev/cdrom in the mount example above. If this fails, you can try /dev/hdc or /dev/hdd, if your cd is an IDE CDROM. If none of them is your CDROM, maybe you don't have IDE but a SCSI CDROM? Then try /dev/sda1, /dev/sda2, etc. ["hda" is the the primary IDE master drive, "hdb" is the primary IDE slave drive, "hdc" is the secondary IDE master (if you have two IDE interfaces on your computer), hdd is the secondary IDE slave, "sda" is the first SCSI interface and the number is the SCSI device id number.]

It is a good idea to have a device /dev/cdrom anyway because some programs assume that it exists. If it does not exist on your system, you may create it as a symbolic link using, for example:

```
ln -s /dev/hdb /dev/cdrom
```

if your cdrom is the /dev/hdb drive.

If you cannot mount because "the device is already mounted or directory busy", perhaps the mountpoint /mnt/cdrom is your current directory. You have to change the directory to somewhere else in order to be able to mount to it; for example change the current directory to the root directory by issuing this command:

```
cd /
```

To unmount a mounted CD, exit the directory /mnt/cdrom and type as a root:

```
umount /mnt/cdrom
```

Your CDROM may refuse to eject the media if it is not unmounted. Also, you may have problems mounting the next CD if the previous one was not unmounted. If you cannot unmount because "the device is busy", perhaps /mnt/cdrom (or any subdirectory underneath) is your current directory? You need to change your current directory to somewhere else out of the mountpoint in order to unmount the device.

4.2.3 How to mount a floppy, zip drive, DOS/Windows partition, or a network drive?

Very much the same as CDROM--see the previous answer if you did not read it.

Floppy. I can mount my floppy (as root) with:

```
mount -t auto /dev/fd0 /mnt/floppy
```

Again, make sure that the directory /mnt/floppy exists and is empty. Also, /mnt/floppy/ cannot be your current directory.

After a successful mount, the files from the floppy appear in the directory /mnt/floppy/. All the users will be able to read the files, but only root will be able to modify/delete the files. Please read further if you wanted the users to be able to write to the floppy.

To unmount a floppy (you **must** do this before ejecting the disk!) use:

```
umount /mnt/floppy
```

If you cannot unmount because "the device is busy", perhaps the /mnt/floppy/ directory is your current directory. Exit it by typing (for example):

```
cd
```

which will change your current directory to your home directory.

Zipdrive. I mount the parallel port external zipdrive (scsi emulation) with:

```
mount -t vfat /dev/sda4 /mnt/zipdrive
```

The "-t vfat" is used here because zip disks come preformatted in the vfat filesystem, which is the filesystem of MS Windows with the long filename support. You won't be able to eject the disk without unmounting it.

All zipdrives (internal SCSI and IDE, external SCSI and parallel port) but the USB are supported under Linux (April 1999).

DOS/Windows partition. I use a dual boot system with both Linux and MS Windows on the same computer. I can access files on the DOS/Windows partition after mounting it with the following command:

```
mount -t vfat /dev/hda1 /mnt/dosdrive
```

Again, you may have to customize this command depending on what partition your DOS filesystem is. The "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; "hdb1"--second IDE hard drive, first partition (or just "hdb" if it is the CDROM installed as a slave on your first IDE interface). "hdc" is the third IDE drive, hdd is the fourth IDE drive. SCSI drives have analogous names but start with letters "sd", followed by the letter indicating the SCSI interface, followed by the number indicating the SCSI device id . For example sda4 means "first SCSI interface, id number 4".

To mount so that all the users can read and write, you may want to try:

```
mount -t vfat -o user,rw,exec,umask=000 /dev/hda1 /mnt/dosdrive
```

This uses options (-o user,rw,exec,umask=000) to give absolutely everybody all the permission to all files on your DOS /dev/hda1 partition (you should ask yourself if this is really safe on your system). If users still can't write to the DOS partitions, perhaps the permissions on your mountpoint need to be set. This command (executed by root) will set up the permissions on the mountpoint /mnt/dosdrive so that all users will be given rights to read, write and execute:

```
chmod a=rwx /mnt/dosdrive
```

Network File System (NFS). This is great for direct access to files that reside on another Linux computer. For mounting of a remote filesystem as NFS, first check if the NFS service is enabled (use the program `setup`). NFS also requires permission from the other computer. To configure the permissions on the server machine, run as root:

```
netconf
```

and adjust the setting under "Exported File Systems" menu.

If you prefer to do it manually, the permissions are set in the file `/etc/exports` . My `/etc/exports` looks like this:

```
/usr hacker(ro) mars(ro)
/home hacker(rw) mars(rw)
/mnt hacker(rw) mars(rw)
```

This gives the machines called hacker and mars the permission to mount the directories `/usr/` (read-only access), `/home` and `/mnt` (read-write).

If you set up your NFS properly, you should now be able to mount a network directory using a command like this:

```
mount -t nfs mars:/home /mnt/mars_home
```

This mounts the contents of the directory `/home/` on a machine called "mars" into the directory `/mnt/mars_home/` (which must exist and be empty).

Many operating systems know NFS, but MS Windows doesn't. Therefore MS Windows remote shares have to be dealt with differently. See the next answer for details.

4.2.4 How to mount a remote MS Windows filesystem through Samba?

A remote MS Windows filesystem can be mounted onto a Linux filesystem through the Samba protocol (Samba must be installed, go here [p ??] if it isn't). Type a command like this (as root):

```
smbmount //mars/windows /mnt/mars_windows -c marie
```

This mounts the MS Windows resource called windows from the MS Windows machine called mars. The mountpoint on the client computer is /mnt/mars_windows/. The option "-c" specifies that the samba server is a machine called marie (this should not be necessary, but it is on my system).

For the above to work, the permission must be given on the MS Windows machine for sharing the directory or drive as a resource. To do this, on the MS Windows machine, enable the filesharing using the "control panel-network", then launch the "Windows Explorer", click the right mouse button on the drive or directory to share, click on properties, switch to the page "sharing", give yourself the permission and give the resource a name.

To unmount an MS Windows directory use:

```
smbumount /mnt/mars_windows
```

If you have problems, see:

```
man smbmount
```

4.2.5 Any quick way to access a file on a DOS/Windows floppy?

Use "mtools", no mounting required. For example, I can use the `mdir` command to quickly inspect the content of the root directory on my DOS floppy:

```
mdir a:\
```

I can also use `mcopy` to copy the file "autoexec.bat" from the root directory on the floppy to my current directory on Linux:

```
mcopy a:\autoexec.bat .
```

You have to be root to be able to write to a floppy.

Type "mtools" to see the supported commands in the rich mtools set, which parallel the most popular DOS commands (for example: `mformat`, `mtype`, `mren`, `mmove`, `mdel`, `mrd`, `mattib`, ...), and use manual pages if you have problems using them. For example:

```
man mtype
```

will show me how to display a content of a text file on a DOS partition.

To access DOS drives other than a: or b:, you have to configure mtools so as to indicate which devices are associated with other DOS "drive letters". This is quite easy--you just edit and modify the file `/etc/mtools.conf`. I typically use `pico` to do it (as root):

```
pico /etc/mtools.conf
```

For example, my `/etc/mtools` contains a line like this:

```
drive c: file="/dev/hda1"
```

which instructs the `mtools` that the partition `/dev/hda1` will be called `"c:"`. The setup of `/etc/mtools.conf` requires just uncommenting (removing the `"#"` at the beginning of the line) and adjusting the appropriate entry.

4.2.6 Mounting works when I am root. Can a normal user mount?

You have to edit the file `/etc/fstab` as root to give the normal users the permission to mount a particular drive. For example I can use the `pico` text editor to do this:

```
pico -w /etc/fstab
```

The option `"-w"` turns off the long line wrap.

Here is the content of my `/etc/fstab`:

```
/dev/hda2 / ext2 defaults 1 1
/dev/hdc3 /home ext2 defaults 1 2
/dev/hdc2 /usr ext2 defaults 1 2
/dev/hdc4 swap swap defaults 0 0
/dev/fd0 /mnt/floppy auto noauto,users,rw 0 0
/dev/cdrom /mnt/cdrom auto noauto,user,ro 0 0
/dev/sda4 /mnt/zipdrive vfat noauto,user,rw,exec 0 0
/dev/hda1 /mnt/dosdrive vfat noauto,user,rw 0 0
none /proc proc defaults 0 0
hacker:/mnt/cdrom /mnt/hacker_cdrom nfs noauto,user,ro 0 0
hacker:/mnt/floppy /mnt/hacker_floppy nfs noauto,user,rw 0 0
hacker:/home /mnt/hacker_home nfs noauto,user,rw 0 0
hacker:/usr /mnt/hacker_usr nfs noauto,user,rw 0 0
```

Each line contains six space-delimited fields (this means that each line has six entries separated by white space). The first field is the name of the device. The second field is the mountpoint (an existing directory on your Linux system to which the resource will be mounted). The third is filesystem type. For removable media that may contain filesystems of several types, I use the option `"auto"` to let Linux probe which filesystem is currently present there. (The order in which they are probed is determined by the content of the file `/etc/filesystems`. You may want to make sure that it specifies `"vfat"` before `"msdos"` or the long DOS filenames may be cut short.) The fourth field contains options: `"auto"` = mount the filesystem on the system startup; `"rw"` = read and write allowed; `"ro"` = read only, `"user"` = users have the permission to mount this filesystem (one can also use `"users"` to allow a user to mount and another user to unmount--otherwise only the user that mounted the filesystem can unmount it), `"exec"` execution of programs is permitted from this filesystem. The number in the field 5 specifies if the filesystem is to be backed up during a system backup, the number in the field 6 determines if to check up the filesystem integrity during bootup. The hacker stuff in my `/etc/fstab` are filesystems on another computer (called `"hacker"`) on my home network and it serves here as an example of how to mount network resources. Check `man fstab` for more info.

For example, if regular (non-root) users have the permission to mount the cdrom (the "user" option is specified), they can mount it using a command like this:

```
mount /mnt/cdrom
```

The command which the root uses for mounting (see here [p 80]) will not work for a regular user because the regular user is restricted by the options in `/etc/fstab` and therefore s/he cannot specify simultaneously both the device and the mountpoint.

For a regular user to be able to write to a disk or execute a program on it, s/he must also be given the appropriate permission on the "mountpoint" directory. For example, this will give all the users all the permissions (read, write, execute) on the directory `/mnt/floppy`:

```
chmod a+rwX /mnt/floppy
```

Now (also the "rw" option is specified for the floppy in the `/etc/fstab`) the user will be able to write to a floppy. If the "exec" option was enabled in the `/etc/fstab`, the user would also be able to execute programs from the floppy.

Please note that the DOS vfat file system doesn't know about the file permissions the way Linux does. Linux manages this during mounting by giving the default file permissions on the mounted filesystem: the user who mounted the filesystem will be the owner of all files and will be given the right to write to the filesystem (if "rw" was specified in `fstab`) but other users can only read. If you wanted to change this behaviour, you could use the "umask=" option so that the appropriate line in your `/etc/fstab` may look like this example:

```
/dev/sda4 /mnt/zipdrive vfat noauto,users,rw,exec,umask=000 0 0
```

This gives absolutely everybody all the permissions on your zipdrive (mounting, unmounting, read, write, execute).

To summarize, the file `/etc/fstab` is the place to keep your defaults on how to mount filesystems and what kind of access is allowed for users. You really want to customize it to simplify mounting on your system. Linux default mounting scheme is restrictive so as to be secure, you may want to remove some restrictions when setting up Linux at home.

4.2.7 Mounting command is too long, how can I simplify it with an alias?

An alias is an abbreviation of a more complex or often used command. For creating aliases, I edit, as root, the file `/etc/bashrc`. This way the aliases are available for all the users on the system. (For creating user-specific aliases, I edit the file `.bashrc` in the user home directory.) The relevant part of my `/etc/bashrc` looks like this:

```
alias cdrom="mount -v /mnt/cdrom"
alias ucdrom="umount -v /mnt/cdrom"
alias dosdrive="mount -v /mnt/dosdrive"
alias udosdrive="umount -v /mnt/dosdrive"
alias zipdrive="mount -v /mnt/zipdrive"
alias uzipdrive="umount -v /mnt/zipdrive"
alias floppy="mount -v /mnt/floppy"
alias ufloppy="umount -v /mnt/floppy"
```


The option "-v" stands for "verbose", i.e., it tells Linux to talk to me a lot during mounting. For the aliases to take effect, the user has to re-login. Now the user can mount the floppy using this simple command:

```
floppy
```

and s/he can unmount it using

```
ufloppy
```

4.2.8 Can I automount?

You can. Automount will automatically mount a filesystem as you access it and unmount when you stop using it.

To set up automount, I first run the programs `setup` (as root). Go to the option "netsysv" (RH5.2) or "System Services" (RH6.0) and make sure that automount service ("autofs") is enabled.

Then, I configure automount by editing the files `/etc/auto.master` and `/etc/auto.misc`, e.g. (as root):

```
pico /etc/auto.master
```

My `/etc/auto.master` looks like that:

```
/misc /etc/auto.misc --timeout 1
```

This says that my automount devices will be mounted in the directory `/misc` (which must exist and be empty). My automount drives will automatically unmount one second after I stop using them (for example, after I exit the directory). This is a short time--you may choose a longer one. The detailed config file is `/etc/auto.misc`. Here is mine:

```
kernel -ro,soft,intr ftp.kernel.org:/pub/linux
cdrom -fstype=auto,ro :/dev/cdrom
floppy -fstype=auto,rw :/dev/fd0
zipdrive -fstype=vfat,rw :/dev/sda4
dosdrive -fstype=vfat,ro :/dev/hda1
hacker_cdrom -fstype=nfs,ro hacker:/mnt/cdrom
hacker_floppy -fstype=nfs,rw hacker:/mnt/floppy
hacker_usr -fstype=nfs,ro hacker:/usr
```

Each line consists of 3 space delimited fields. The first field is the "key" which will be the name of the subdirectory (under `/misc`) to which the device will be mounted. This directory must NOT exist. It will not be visible when I use the command `ls`, but I can "cd" to it and my device will then mount. Don't ask me why it is so, and how to use this automount in GUI. I don't know. The hacker stuff in my `auto.misc` is the `cdrom` and `floppy` from another computer in my home network.

I automount to the directory `/misc` (not `/mnt`) so that I can also mount filesystems manually, without using automount, to the directory `/mnt`.

4.2.9 How do I get my parallel-port (external) Zip drive recognized?

RedHat 5.2 During the initial RH5.2 installation, answer "yes" to the question "do you have scsi devices" Then pick up the parallel port zip drive (ppa) from the list. During the disk partitioning later in the installation process, either have a disk in your zip drive or tell the install process to "skip" when it attempts to access your zip drive.

If you didn't setup your Zip drive during the initial RedHat installation, you may also add the Zip support later in a way similar to adding the network card or any other module to the kernel:

As root, start an X-terminal, and run

```
/usr/bin/kernelcfg
```

to insert the ppa module into the kernel. (The "ppa" stands for, I guess, the "parallel port first interface 'a' ".)

After everything is done, inspect `/proc/modules` to see if the ppa module is loaded (or run as root `lsmod` to list the loaded modules). The information from `kernelcfg` goes to the file `/etc/conf.modules` so if you have difficulty removing modules (e.g., inserted by mistake)--as I did--just edit and adjust this file manually.

Now your zipdrive should be recognized. Put a zipdisk into it and try mounting using the previously described command (as root):

```
mount -t vfat /dev/sda4 /mnt/zipdrive
```

Using the Linux kernel ver.2.0.36 and below, you will not be able to use a printer which is connected through the parallel port zip drive (this works with the kernel 2.2.x). Use the command

```
uname -a
```

to see which version of kernel you are running.

RedHat 6.0 and 6.1 The zipdrive (zip100 drive) installation did not work during my upgrade to RedHat 6.0 (the installation program said that it couldn't find the zipdrive). So, after the installation was completed, I issued the following commands to insert the modules for parallel port zip drive into the kernel (as root):

```
/sbin/insmod parport  
/sbin/insmod ppa
```

To have these two lines executed automatically after each bootup, I added them at the end of the file `/etc/rc.d/rc.local` (this file is something like `autoexec.bat` on DOS).

If this still does not work for you, you may also want to edit the file `/etc/conf.modules`. Mine contains such a line:

```
alias parport_lowlevel parport_pc
```

and there is no line mentioning the "ppa" module.

For the newer Zip250 drive, I have the following two lines executed from my `/etc/rc.d/rc.local` file:

```
/sbin/insmod parport
/sbin/insmod imm
```

4.2.10 Can I set 32-bit hard drive I/O?

Newer Linux distributions (e.g., Mandrake 7.0) can automatically turn on the hard drive optimization: 32 bit input/output (I/O) and direct memory access (DMA). Here is how to turn on the hard drive optimization manually. (Based on <http://hardwarezone.community.com.sg/main.htm> by Edward Choh.)

This procedure worked fine for me, yet be warned that it can possibly harm the content of your harddrive, so do not do it if you are a real PC newbie, don't feel geeky today, or have a weak heart--I can't guarantee it will work for you.

The turning on of the 32-bit I/O and DMA has to be done by root, and I did it in a single-user mode (to minimize the damage to the file system if something went wrong and I had to reboot). I definitely would not do it on a system currently running many programs or X-windows, and would have a current backup of any precious data.

To boot your computer in a single-user mode, I type this at the lilo prompt (during bootup):

```
linux single
```

Say, I would like enable the 32-bit I/O on my first IDE harddrive, which is "hda". First, I time the harddrive current performance, and note the score:

```
hdparm -t /dev/hda
```

Now, I display my current I/O and DMA settings:

```
hdparm -c /dev/hda
```

[my system showed 0, meaning that the 32-bit I/O is turned off and the default 16-bit access is used]

```
hdparm -d /dev/hda
```

[my system showed 0 again, meaning that the harddrive DMA access is turned off].

Now, I turn on the 32 bit IO and DMA:

```
hdparm -c 1 /dev/hda
hdparm -d 1 /dev/hda
```

Now, I can time the performance of the harddrive again to compare the score with the original one:

```
hdparm -t /dev/hda
```

If everything worked ok, and the performance has improved, I can "commit" the new settings, so they can survive a soft reset:

```
hdparm -k 1 /dev/hda
```

To have the new settings in effect every time you reboot the machine, you may add a line at the end of the file `/etc/rc.d/rc.local` (this file is something like `AUTOEXEC.BAT` in DOS):

```
hdparm -c 1 -d 1 -k 1 /dev/hda
```

If something did not work as expected, or the performance did not really improve, I can reboot at any time and the old settings will be in effect as long as I did not perform the last operation.

I performed this tune-up on 4 hard drives on my home network. It was a success on 3 newer harddrives: the performance improved by 30-300% and at least one computer "feels" faster than before. One harddrive (which is always flaky) hanged the computer hard during the performance test and I had to reset the machine (no damage done).

4.2.11 I reached the limit on the number of opened files (error message)

You can increase the limit via the `/proc` file system. This file system is entirely virtual--it is just a "window" to see or set some parts of the Linux kernel. To read the maximum number of simultaneously opened files on my system, I use the following command:

```
echo /proc/sys/fs/file-max
```

On my system (Mandrake 7.2), the limit is 8192. To increase it, I use (as root):

```
echo 16000 > /proc/sys/fs/file-max
```

You may also want to increase the limit on a related kernel variable:

```
echo 30000> /proc/sys/fs/inode-max
```

To make the changes permanent, add the relevant lines at the end of your startup script `/etc/rc.d/rc.local`

To learn more about the `/proc` Linux kernel interface, the meaning of the variables it contains, and their recommended values, you may wish to read (if you installed the Linux kernel source codes, which is a great resource even for a newbie):

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

Go to Part: 4.3 - Working with X-windows

LINUX NEWBIE ADMINISTRATOR GUIDE

ver. 0.120 2001-03-16 by Stan and Peter Klimas

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

Copyright (c) <1999,2000,2001> 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](#).

Quick navigation:

Part 4.3: Linux Newbie Administrator FAQ - Working with X-windows

Contents:

4.3 Working with X-windows

- 4.3.1 How to switch between text and graphical consoles? [p 89]
 - 4.3.2 How do I set up my video card, monitor and mouse for the X-server? [p 90]
 - 4.3.3 Can I have a GUI login prompt? [p 91]
 - 4.3.4 How do I install KDE [p 91]
 - 4.3.5 How can I change my default desktop to KDE (or Gnome or yet another)? [p 93]
 - 4.3.6 Can I have multiple sessions of X running at the same time? [p 94]
 - 4.3.7 Can my sister have second GUI login prompt so she does not have to kill my X-session to start hers? [p 94]
 - 4.3.8 How to X-window remotely? [p 95]
 - 4.3.9 How do I install TrueType fonts from my MS Windows partition? [p 96]
 - 4.3.10 How do I copy-paste? [p 97]
-

4.3 Working with X-windows

4.3.1 How to switch between text and graphical consoles?

Pressing the key combination <Ctrl><Alt><F1> will switch you to the first text console at any time. <Ctrl><Alt><F2> will switch you to the second text console, <Ctrl><Alt><F3> to the third text console, etc, up to <Ctrl><Alt><F6>, for the total of 6 text consoles. <Ctrl><Alt><F7> will switch you to the first graphical user interface (GUI) console if one is running. <Ctrl><Alt><F8> to the second GUI console, etc., up to <Ctrl><Alt><F11> for the total of 5 GUI consoles. The 12th console is either used as the 6th GUI (RedHat 6.1) or a place to which kernel messages are continually displayed (Mandrake 7.0, really cool feature). Typically none or only the first GUI console is running.

<Ctrl><Alt><F1> means: "Press the left <Ctrl> and <Alt> keys and hold them. Now press <F1>. Release <F1>. Release <Ctrl> and <Alt> keys."

Briefly, sitting at a Linux computer you can have many consoles opened at the same time, and switch between them using the hot keys described above. You have to log in on each to be able to use it--you may log in as the same user (multiple times), or different users. Each login session is quite separate, and you they should not interfere with each other. The first 6 consoles are text-based terminals and are named tty1, tty2 ... tty6 (historical name, "tty" stands for "teletypewriter"). The subsequent consoles are graphical (GUI). These are all **local** consoles.

You can also connect to a Linux computer over a network, have your work done on your Linux server and the resulting display sent to your remote terminal on your remote console (which can be Linux- or another operating system-based).

The full-screen text consoles are terminals by themselves. Under the GUI consoles, you can create "slave" pseudo-terminals (in a window) on demand under each GUI console--they will be called pts0, pts1,

If you are sitting at a text terminal, you can determine the name of your terminal using the following command:

```
tty
```

You can close any text terminal by typing "exit" inside it.

The Linux GUI console is quite similar to other GUI you might have used, e.g., MS Windows, but there are also numerous important differences. I love multiple "desktops" to stay organized--the default setup in KDE offers 4 desktops (it can be customized to between 1 and 16). I run many programs and never close them, so to stay organized I keep connectivity application (netscape with all its windows, knode newsreader, and licq) together on Desktop2, abiword, staroffice and a text editor on Desktop3, games and konqueror with helpfiles on Desktop4, and leave Desktop1 for the more occasional chores. One can switch between desktops by clicking on the "desktop pager" on the "K-panel" or using <Ctrl><TAB>. To toggle between applications on a same desktop, I may click the application window, or on its representation on the "application panel", or use <Alt><TAB>.

4.3.2 How do I setup video card, monitor and mouse for the X-server?

This should be set-up during your Linux initial installation unless you skipped the step. To set it up now, you may try, as root, one of these configurators:

```
Xconfigurator  
XF86Setup  
xconf
```

The simplest way is to run the command `setup` (under RedHat, as root) and access Xconfigurator from there.

To setup X-windows under Linux, you may need to know your hardware. You may want to dust your monitor manual to see what max synchronization frequencies (vertical and horizontal) your monitor supports. The message when the computer boots may give you a clue about what type of video card you have and with how much memory. Read the label underneath your mouse to find out about the mouse. Next time you buy a mouse, get a 3-button "Linux-ready" Logitech or similar (Linux makes a good use of all three mouse buttons).

When testing the X-server, if the screen goes blank, displays funny lines, or otherwise obviously does not function as designed, kill it fast with <Ctrl><Alt><BkSpace> and re-check your monitor sync frequencies. Running wrong frequencies can be bad for your monitor.

If you really have problems, set up a plain vga (640x480) X server first. You can fine-tune it later, after you get some understanding of how things work on your system.

4.3.3 Can I have a GUI login prompt?

To start your X-server automatically on the system start-up and display a graphical login prompt, you have to change (as root) just one character in the file `/etc/inittab`. This file specifies something like:

```
id:3:initdefault:
```

Change it to

```
id:5:initdefault:
```

This sets up the default runlevel to 5, which is X-Windows. The meaning of the different runlevels is explained in the same `/etc/inittab` file :

```
0 - halt (Do NOT set initdefault to this)
1 - Single user mode
2 - Multiuser, without NFS (The same as 3, if you do not have
networking)
3 - Full multiuser mode
4 - unused
5 - X11
6 - reboot (Do NOT set initdefault to this)
```

You can change the runlevel from the command line. E.g., this command (has to be executed as root):

```
init 6
```

will reboot your computer, while the following command would switch your computer to a single-user mode:

```
init 1
```

4.3.4 How do I install kde (e.g., on RedHat 5.2)?

RedHat 5.2 does not install kde by default because at the time of the RH5.2 release there was a problem with the license for a library that the kde uses. Otherwise, kde is a very good GUI, the library license problem is now solved, and the RedHats 6.0 up include kde (alongside the more ornamental GNOME). Still, the kde binaries are on your RedHat 5.2 CDs, you just have to install them yourself. (If you don't have the CDs, you can download the binaries from the kde site on the Internet.)

First check if your X-server works by executing:

```
xinit
```

[The X-server is a bare-bone X-windows system, without a "Windows Manager." You can execute your X-windows programs from here by typing the program name (with leading `./` or full path) in the X-terminal window, but you will not be able to move or resize the windows, add icons, etc.]

You can exit your X-server by typing exit in the X-terminal window, or pressing <Ctrl><Alt><Bksp> to kill the X-windows server. (The last solution is perfect should your windows ever hang--don't reboot in such a case.) If your X-server does not work, see the next answer.

Now, login as root. Mount the RH5.2 CD:

```
mount /mnt/cdrom
```

Go to the proper directory:

```
cd /mnt/cdrom/kde/distribution/RedHat/i386/binary
```

It is useful to use the command line autocompletion (press Tab) when typing long paths or filenames.

Read the README file:

```
cat README-2rh51-rpms | more
```

Use the rpm "RedHat Package Manager" to install the necessary packages

```
rpm -iv packagename
```

The packages have filenames ending with .rpm. First install the qt libraries, then kde support, then kde libs, then kde base. If you choose the wrong order, the dependency check will fail and the package will not install (rpm will issue a message). This is not serious, just re-install the required package first, and then try the next package again. After you are done with the base, you may want to install all other packages for kde (utils, admin, network, games, graphics, multimedia)--they are not big, so you may consider installing them all. Finally, just to make sure that you installed everything type:

```
rpm -ivh *.rpm
```

The options "vh" print some extra info. You will get some messages like "the package is already installed" If there is more than a screenful of them, you can scroll back using <Shift><PgUp>. If you really don't like the command-line-based rpm package manager, you may install the same packages using a GUI front to rpm called glint (available only in RH5.2). Just type "glint" in the X-windows terminal.

Now tell your system that kde is to be your default X-windows manager. In user home directory, create an .Xclients file:

```
pico .Xclients
```

which contains just one line:

```
/opt/kde/bin/startkde
```

Type in the line and save the file. (Adjust the line as required so the location of the startkde file is correct.) Now, make the file executable to all users:


```
chmod a+x .Xclients
```

Check if the permissions were changed:

```
ls -l .Xclients
```

[Files with a dot at the beginning are not displayed by a regular `ls` command, there are something like hidden files under DOS. You must use its name or `ls -a` .]

If you created the file as root not the user, change the owner and the group of the file to the proper user:

```
chown user_name .Xclients
chgrp user_name .Xclients
```

That's it. Now typing `startx` should start your X-server with the kde as the windows manager.

4.3.5 How can I change my default desktop to KDE (or Gnome or yet another)

In my home directory, I create (or edit if it exists) file `.xsession` using my favourite pico editor:

```
pico .xsession
```

[Pls note the dot at the beginning of the filename, files with names starting with dots are normally "invisible".]

On my RedHat 6.2 system, the file contains just one line:

```
exec startkde
```

KDE clearly works best for me, although it feels heavy on older hardware or under load. Here is my list of windows managers available on RedHat or Mandrake installation CD:

<code>startkde</code>	(to run kde. on some systems, the command may be <code>kde</code>)
<code>gnome-session</code>	(to run Gnome)
<code>xfce</code>	(to run XFce, my favourite "lightweight" desktop)
<code>afterstep</code>	(to run afterstep)
<code>AnotherLevel</code>	(to run AnotherLevel)
<code>fvwm2</code>	(to run fvwm2)
<code>fvwm</code>	(to run fvwm)

Of course the alternative windows manager will run only if it is installed on your system. The above windows managers are available on RH/Mandrake CDs for you to decide if you want to install them. I use almost exclusively KDE, although the other managers may be smaller and faster. Gnome is a famous X-windows project which is said to be more advanced and is prettier than KDE, but it is still quite buggy, so perhaps not recommended unless you don't mind occasional trouble. RH6.x contains both major X-windows systems, Gnome and KDE.

4.3.6 Can I have multiple sessions of X running at the same time?

Yes, you can. When you issue the first `startx` command on your system, it opens the first X-session on the default display 0. The second X-session must be opened on a different display. For example, this will open a second X-session on the display 1:

```
startx -- :1
```

You can have up to 6 concurrent X sessions. Use `<Ctrl><Alt><F7>` to switch to display 0, `<Ctrl><Alt><F8>` to second screen, etc. up to `<Ctrl><Alt><F12>`. Try `man startx` if you need more info.

In a similar fashion, you can open another bare X-server session without a window manager. This will open one on display 2:

```
xinit -- :2
```

On this bare-bone X-display I can run a different windows manager (so as to have two different ones running at the same time) by typing in the X-terminal window one of these (see the previous answer for more details):

```
startkde
gnome-session
xfce
afterstep
AnotherLevel
fvwm2
fvwm
```

4.3.7 Can my sister have second GUI login prompt so she does not have to kill my X-session to start hers?

To enable several concurrent GUI logins on different local consoles under RedHat (RedHat uses program `gdm` for graphical logins), I had to edit the file: `/etc/X11/gdm/gdm.conf`. I have the following entry at the end of this file to enable 4 login terminals `<Ctrl><Alt><F7>` to `<Ctrl><Alt><F10>`:

```
[servers]
3=/usr/bin/X11/X vt10
2=/usr/bin/X11/X vt9
1=/usr/bin/X11/X vt8
0=/usr/bin/X11/X vt7
```

Having four GUI lets me run KDE and GNOME at the same time on one computer with two GUI terminals spare, so my sister can login despite my having screensavers with password-protection.

To enable several concurrent GUI logins on different local consoles under Mandrake (Mandrake uses `kdm` for graphical logins), I had to modify two files: `/etc/X11/xdm/Xservers` to include something like:

```
:0 local /usr/X11R6/bin/X :0
:1 local /usr/X11R6/bin/X :1
:2 local /usr/X11R6/bin/X :2
```

and `/etc/X11/xdm/xdm-config` to copy all the settings for display 0 to display 1 and 2 so that it includes this:

```
DisplayManager._0.authorize: true
DisplayManager._1.authorize: true
DisplayManager._2.authorize: true
DisplayManager._0.setup: /etc/X11/xdm/Xsetup_0
DisplayManager._0.startup: /etc/X11/xdm/GiveConsole
DisplayManager._0.reset: /etc/X11/xdm/TakeConsole
DisplayManager._1.setup: /etc/X11/xdm/Xsetup_0
DisplayManager._1.startup: /etc/X11/xdm/GiveConsole
DisplayManager._1.reset: /etc/X11/xdm/TakeConsole
DisplayManager._2.setup: /etc/X11/xdm/Xsetup_0
DisplayManager._2.startup: /etc/X11/xdm/GiveConsole
DisplayManager._2.reset: /etc/X11/xdm/TakeConsole
```

This enables me to run gnome and kde at the same time on a single computer with third GUI terminal spare.

4.3.8 How to X-window remotely?

- Start X-server on the local machine, e.g.

```
xinit
```

- From the x-terminal give the remote machine the permission to display on your local screen:

```
xhost name_of_the_remote_server
```

In the really secure environment of my house, I could even give all servers the permission to display on my screen using (don't do it when connected to the Internet):

```
xhost +
```

- Telnet the remote server.

- Start an X-program on the remote server directing the display on your local screen, for example, you may start a window manager:

```
startkde -display local_machine_name:0.0 &
```

The symbol "&" puts the command in the background, so that your telnet window is still available to you.

The 0.0 means "display zero, screen 0", which is your first screen on the first display and makes sense since you can have many concurrent sessions of X running on your computer with Linux.

You don't have to specify the "-display" option if your environment variable DISPLAY specifies the correct location on your current terminal, which is the case on my systems by default, but not on everybody else's as I am told. You can check your DISPLAY setting using:

```
echo $DISPLAY
```

- After I finish my remote X session, I restore the access control to my X-server using:

```
xhost -name_of_the_remote_server
```

or

```
xhost -
```

Example. This sequence of commands will run Netscape on the remote machine called marie, directing the display to the X-server with X-windows manager which runs on the local machine hacker:

```
startx
xhost marie
telnet marie
[login]
netscape -display hacker:0.0 &
[do my stuff]
[logout]
xhost -marie
```

In principle, you can run a program on any computer on the network, and display the output on any other (not necessarily the one you are sitting at).

I use remote X-windowing a lot to run fat programs (kde, Word Perfect 8, and Netscape) on a slim machine (486-33, 8 MB mem) which would not be able to run those by itself. It is also a convenient and fast way to work with files on a remote system for which the nfs mount is not set up.

X-windows was designed to run remotely over the network. Remote X-windowing is a very powerful tool, on top of being quite a pleasant experience. Try it out.

You can even run a program on a remote Linux (or any Unix) computer and redirect the display to a local MS Windows machine if you install an X-windowing program for MS Windows. For a good overview of choices, see:

http://www.linuxworld.com/linuxworld/lw-2000-09/lw-09-legacy_1.html

4.3.9 How do I install TrueType fonts from my MS Windows partition?

Red Hat 6.2 comes with TrueType font server but no TrueType fonts. You can install your own TrueType fonts though. Here is how I did it.

0. From under K-menu (KDE), select "System"-"Font Manager" (or equivalent) and note what fonts you have installed.

1. Check if the "free type" font server is installed:

```
rpm -q freetype
```

This queries (option "q") the rpm package manager for the package "freetype". If the package is installed, go to next step. If "freetype" is not installed, install it now from your distribution CD. "freetype" was installed on my system after a "full" RH installation.

2. As root, make a directory that is to hold your TrueType fonts:

```
cd /usr/X11R6/lib/X11/fonts
mkdir TrueType
```

This directory is referred to in the configuration file `/etc/X11/XF86Config` so make sure that the name of the directory exactly as shown. If you would like to name the directory differently, you have to edit `/etc/X11/XF86Config` and make appropriate adjustments. My "default installation" RedHat contained such a line:

```
FontPath "/usr/X11R6/lib/X11/fonts/TrueType"
```

3. As root, copy your *.ttf files from the original location to the TrueType font directory that you just created. I took some TrueType from my MS Windows partition, you may need to use different source location:

```
cd /usr/X11R6/lib/X11/fonts/TrueType
cp /mnt/dos_hda1/windows/fonts/my_private_fonts/*.ttf .
```

Before copying any fonts, make sure that it does not violate your licence agreement.

4. As root, run the following commands:

```
cd /usr/X11R6/lib/X11/fonts/TrueType
ttmkfdir> fonts.dir
cp fonts.dir fonts.scale
```

5. Close all X-windows applications and log out from X-windows.

6. As root, restart your X-font server (or reboot your computer:

```
/etc/rc.d/init.d/xf86 stop
/etc/rc.d/init.d/xf86 start
```

7. Log back to your KDE, and from under K-menu, select "System"->"Font Manager" to see if the fonts installed correctly.

This parts is based on: <http://www.computerbits.com/archive/20001000/linux0010.htm>

4.3.10 How do I copy-paste?

Under X-windows, use mouse: highlight the text to be copied, switch to the location where to copy, and press the middle mouse button to paste. This works nice and fast, as long as during the switching you don't disselect the original text.

Some GUI applications do support MS-Windows-style "copy-paste": Select the text. Use the menu item "edit"->"copy". Switch to the location where to copy. Use the menu item "edit"->"paste".

You can also use the cut-paste history. Try running `klipper` (in X-terminal, KDE).

To capture the content of a window or the entire screen to a graphics file, I use `knapshot`.

To catch contents of a text console (outside of the GUI console), I could use in X terminal:

```
cat /dev/vcs1
```

and then copy and paste whatever I need from the X terminal with a mouse.

The text-based consoles support mouse if you run the `gpm` daemon. Type `gpm` to test it--it will run fine if your mouse is appropriately configured. You may want to run `mouseconfig` to configure your mouse first. To have `gpm` run automatically on system startup and stay running, select the `gpm` daemon using the `ntsysv` utility. Use `gpm` exactly the same as the GUI cut-paste: highlight the text to be copied, move the mouse cursor to the "copy to" location, and press the middle mouse button to paste.

Go to Part: 4.4 - Basic Configurations (Pinter, soundcard...)

LINUX NEWBIE ADMINISTRATOR GUIDE

ver. 0.120 2001-03-16 by Stan and Peter Klimas

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

Copyright (c) <1999,2000,2001> 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](#).

Quick navigation:

Part 4.4: Linux Newbie Administrator FAQ - Basic Configurations

Contents:

4.5 Basic Configurations

- 4.4.1 How to setup my soundcard? [p 99]
 - 4.4.2 How do I setup my printer? [p 100]
 - 4.4.3 Word Perfect 8 does not have a driver for my printer [p 102]
 - 4.4.4 Configuration files [p 102]
 - 4.4.5 Device files [p 103]
 - 4.4.6 Some daemons [p 103]
-

4.4 Printer and soundcard

4.4.1 How to setup my soundcard?

Try to run (as root)

```
setup
```

The third button from the top ("sndconfig") sets up the sound card. Unless you have a very fancy sound card, this will work for you. At the end of the setup, Linus says how he pronounces "Linux". ("sndconfig" can be also run directly from the command line--type `sndconfig --` useful under Mandrake, which does not have "setup".)

You may want to try your soundcard and cdrom using a command line cdplayer. Put a music CD to your CDROM and type:

```
cdplay
```

If this does not work, maybe you don't have `/dev/cdrom`? Check if you can mount a data CD as a root (look here [p ??]) and create the device `/dev/cdrom` by linking it to the appropriate drive (most likely `/dev/hdb`), for example:

```
ln -s /dev/hdb /dev/cdrom
```

If `cdplay` works for root, but does not work for a regular user, maybe you need to give (as root) everybody the permissions to read and write to the the file `/dev/cdrom` :

```
chmod 666 /dev/cdrom
```

(The directory `/dev` is where all your devices appear as files.)

Now, that you got hooked on `cdplay`, you can play third song, try:

```
cdplay play 3
```

You can also use the command `cdp` for rudimentary command line interface to `cdplay`, but perhaps you prefer the interfaces available from under X-windows (e.g., from KDE "K" menu, choose: Multimedia-"CD Player").

To stop the music either press the button on you CDROM or issue one of these commands:

```
eject  
cdplay stop
```

4.4.2 How do I setup my printer?

From the GUI login screen, logon as root. If X-windows is not running, start an X-windows session by typing (as root):

```
startx
```

Read here [p ??] if your X-windows is not set up properly yet.

Start an X-terminal (the "two-computer" button on your K-bar) and type in it:

```
printtool
```

This program does a complete printer setup, you just have to fill up the information about your type of printer and where it is hooked up.

Specifying the proper printer port is the most important part. If you don't know which one is yours try:

on RedHat 5.2: `lp1` (this is the first parallel port on RH5.2) or `lp2` (this is the second parallel port on RH5.2) or `lp3` (this is the third parallel port on RH5.2); **on RedHat 6.0 (or later):** `lp0` (this is the first parallel port on RH6.x) or `lp1` (this is the second parallel port on RH6.x) or `lp2` (this is the third parallel port on RH6.x). After upgrading from RH5.2 to 6.0, the printing stopped working because the name of the parallel ports changed. I had to re-run the `printool` and adjust the port. The numbering of ports changed to bring it in line with numbering of other devices, which always starts from 0.

Try printing an ASCII test-page straight to the port. Only when this works set up the bells and whistles.

If you are setting up a remote printer, make sure that your machine has the permission to use the remote printer. The permissions are set in the file `/etc/hosts.lpd` (more secure) or `/etc/hosts.equiv` (less secure) on the machine to which the printer is attached. These files simply list the names of the remote computers that can use a local printer, one computer name per line. Mine looks like this:


```
hacker
mars
```

The file `/etc/hosts.lpd` did not exist on my system, so I created it.

For quick information about the printers on your machine, you may want to view the file `/etc/printcap`:

```
cd /etc/
cat printcap
```

Here is the meaning of some codes that I see in my `/etc/printcap`:

:	Field separator (separates the entries in the file).
\	(at the end of line) Continuation on the next line.
lp	Name of the printer. "lp" is the name of the default printer on your machine. Subsequent printers are often, by default, given the names lp0 or lp1, ... (or whatever you like) but this should not be confused with the name of the devices (parallel ports) to which they are connected.
sd=/var/spool/lpd/lp	My spool directory (sd).
mx#0	Maximum size of print jobs (mx) in blocks. "0" means no limit.
sh	I want headers to be suppressed (sh). Header is the page with your name that prints before your printing job (waste of paper if you print at home).
rm=mars	Name of the remote machine (rm), which on my system is called "mars" (my printer is connected to a different computer).
rp=lp or lp=/dev/lp0	Name of the remote printer (rp), which is the name of the printer on the remote machine ("lp" on "mars" on my home network) or the name of the device on the local machine. "/dev/lp0" is the first parallel port on RH6.x (it used to be /dev/lp1 on RH5.2, the numbering of parallel ports changed).
if=/var/spool/lpd/lp/filter	Input filter (if). Your printing job will be formatted by this "filter" before it is sent to the printer.
sf	Suppress the form feed (sf) that is normally sent when printing is completed (use it if your printer keeps printing an empty page at the end of each jobs).

The printer is controlled using the command `lpc` (as root). Type "?" to see the options. This program is notorious for its peculiarities, so don't get discouraged easily. The printer queue can be viewed with `lpq` and cleaned up with `lprm`, both of which work for a user (not only root). You can print from the command line using the command `lpr`. Under KDE, you can control the printer queue from the program available under the "K-button"-"Utilities"-"Printer Queue".

4.4.3 Word Perfect 8 does not have a driver for my printer

If you installed your printer in KDE using the printtool and it had a driver which works fine, set up Word Perfect to print using the "passthru postscript" driver.

4.4 4 Where are the setup and configuration files?

System-wide settings are stored in the `/etc` directory. User-specific settings are stored in the user home directory `/home/user_login_name`.

Here is a listing of some system-wide configuration files that you I use most often:

SHELL DEFAULTS

`/etc/bashrc` - system-wide default functions and aliases for the bash shell

`/etc/profile` - system-wide defaults for bash shell, including system-wide environment variables.

ADMINISTRATIVE SETTINGS

`/etc/passwd` - contains passwords and other information concerning users who are registered to use the system. It can be modified by root directly, but it is preferable to use a configuration utility such as `passwd` to make the changes. A corrupt `/etc/passwd` file can easily render a Linux box unusable.

`/etc/group` - similar to `/etc/passwd` but for groups

`/etc/crontab` - setup for "cron", which runs commands periodically (hourly, daily, weekly, monthly, etc.).

`/etc/inittab` - runs different programs and processes on star.

`/etc/issue` - message that accompanies login prompt. This is often overwritten by the `rc.local` script.

`/etc/issue.net` - same as above, but used when login is attempted over the network.

`/etc/motd` - "message of the day" file, displayed after a user logs in.

NETWORK CONFIGURATION

`/etc/hosts` - contains a list of host names and absolute IP addresses.

`/etc/hosts.allow` - hosts allowed to access Internet services

`/etc/hosts.deny` - hosts forbidden to access Internet services

`/etc/resolv.conf` - setups for a list of domain name servers used by the local machine

`/etc/inetd.conf` - configures the `inetd` daemon to tell it what TCP/IP services your machine should run.

`/etc/exports` - specifies hosts to which file systems can be exported using NFS (network file system). `man exports` contains information on how to set up this file for remote users.

HARDWARE CONFIGURATION

`/etc/conf.modules` - setup for the linux kernel modules. Modules are like "device drivers" under MS Windows or DOS.

`/etc/fstab` - contains information on partitions and filesystems used by system to mount different partitions and devices on the directory tree.

`/etc/mtab` - shows currently mounted devices and partitions and their status.

`/etc/lilo.conf` - configuration file for lilo boot loader.

`/etc/printcap` - setup for printers.

`/etc/termcap` - ASCII database defining the capabilities and characteristics of different consoles, terminals, and printers. You typically don't want to change those.

`/etc/X11/XF86Config` - X configuration file.

4.4.5 What are all the device files?

Devices appear as files in the directory `/dev`. They can be read, or written to, if you have the permission to do so. The listing of the file reveals some important details about the device, for example:

```
ls -l /dev/ttyS3
```

on my system produces the following output:

```
crwxr-xr-x    1 root      tty          4,  67 Mar 13 22:59 ttyS3
```

The initial "c" indicates a character device. "b" would mean "block device", "p"=FIFO device, "u"=unbuffered character device, "d"=directory, "l"=symbolic link. The numbers "4, 67" mean that the device major number is 4 and the minor number is 67.

Here is a list of some common devices:

- `/dev/modem` - the serial modem. In the typical case, a symbolic link to `/dev/ttyS1`, `/dev/ttyS2`, `/dev/ttyS3` or `/dev/ttyS0`, depending to which serial port your modem is connected.
- `/dev/mouse` - mouse. In the typical case, a symbolic link to `/dev/ttyS0` or similar (see above), depending to which serial port your mouse is connected.
- `/dev/lp0` - printer on the first parallel port
- `/dev/lp1` - printer on the second parallel port
- `/dev/fd0` - first floppy disk drive
- `/dev/fd0H1440` - driver for the first floppy drive in high density mode. Generally, this is invoked when formatting a floppy drive for a particular density. Slackware also comes with drivers that allow for formatting a 3.5" diskette with up to 1.7MB of space. Red Hat and Mandrake do not contain these device driver files by default.
- `/dev/fd1` - second floppy disk drive.
- `/dev/hda` - first IDE hard drive .
- `/dev/hdb` - second IDE hard drive.
- `/dev/hdc` - third IDE drive. On many machines, the IDE cdrom drive is attached here.
- `/dev/cdrom` - typically, a symbolic link to the appropriate hard drive, e.g. `/dev/hdc`.
- `/dev/tty1` - the first text console
- `/dev/dsp` - digital audio, i.e., the sound card. "dps" stands for "digital signal processing".
- `/dev/sndstat` - do `cat /dev/sndstat` to learn about the status of your sound devices.
- `/dev/null` - used when you want to send output into oblivion.
- `/dev/random` - used to read pseudo-random numbers.

For more info try:

```
less /usr/src/linux/Documentation/devices.txt
man MAKEDEV
```

Some Linux daemons

Daemons are a "resident" programs that periodically wake up, check your system and may perform certain functions. Your Linux system is likely "factory" set to run quite a number of daemons. Most of them can be (dis)selected by running the program `ntsysv` (RedHat) as root and checking the appropriate box. The short description of each daemon is available from `ntsysv` by pressing <F1>. If the daemon you need is not listed in `ntsysv`, you need to insert your RedHat/Mandrake installation CD and install the appropriate package. The alternative to `ntsysv` may be `tksysv` (type

as root, in X terminal), which is perhaps more flexible, but way more complicated (it lets you set up the list of daemons to run in each runlevel). Another, simpler and even more powerful+flexible+difficult-to-use tool is `/sbin/chkconfig`.

Here is a short list of popular daemons with a short description:

`anacron` - checks 'cron' jobs that were left out due to down time and executes them. Useful if you have cron jobs scheduled but don't run your machine all the time--anacron will detect that during bootup.

`amd` - automount daemon (automatically mounts removable media).

`apmd` - Advanced Power Management BIOS daemon. For use on machines, especially laptops, that support apm.

`arpwatch` - keeps watch for ethernet/ip address pairings.

`atd` - runs jobs queued by the "at" command.

`autofs` - control the operation of automount daemons (competition to amd).

`bootparamd` - server process that provides information to diskless clients necessary for booting.

`crond` - automatic task scheduler. Manages the execution of tasks that are executed at regular but infrequent intervals, such as rotating log files, cleaning up /tmp directories, etc.

`cupsd` - the Common UNIX Printing System (CUPS) daemon. CUPS is an advanced printer spooling system which allows setting of printer options and automatic availability of a printer configured on one server in the whole network. The default printing system of Linux Mandrake.

`dhcpcd` - implements the Dynamic Host Configuration Protocol (DHCP) and the Internet Bootstrap Protocol (BOOTP).

`gated` - routing daemon that handles multiple routing protocols and replaces routed and egppup.

`gpm` - useful mouse server for applications running on the Linux text console.

`httpd` - daemon for the Apache webserver.

`inetd` - listens for service requests on network connections, particularly dial-in services. This daemon can automatically load and unload other daemons (ftpd, telnetd, etc.), thereby economizing on system resources.

`isd4linux` - for users of ISDN cards.

`kernelld` - automatically loads and unloads kernel modules.

`kudzu` - detects and configures new or changed hardware during boot.

`keytable` - loads selected keyboard map.

`linuxconf` - the linuxconf configuration tool. The automated part of is run if you want linuxconf to perform various tasks at boot-time to maintain the system configuration.

`lpd` - printing daemon.

`mcserv` - server program for the Midnight Commander networking file system. It provides access to the host file system to clients running the Midnight file system (currently, only the Midnight Commander file manager). If the program is run as root the program will try to get a reserved port otherwise it will use 9876 as the port. If the system has a portmapper running, then the port will be registered with the portmapper and thus clients will automatically connect to the right port. If the system does not have a portmapper, then a port should be manually specified with the -p option (see below).

`named` - the Internet Domain Name Server (DNS) daemon.

`netfs` - network filesystem mounter. Used for mounting nfs, smb and ncp shares on boot.

`network` - activates all network interfaces at boot time by calling scripts in `/etc/sysconfig/network-scripts`

`nfsd` - used for exporting nfs shares when requested by remote systems

`nfslock` - starts and stops nfs file locking service

`numlock` - locks numlock key at init runlevel change.

`pcmcia` - generic services for pcmcia cards in laptops.

portmap - needed for Remote Procedure Calls
postfix - mail transport agent which is a replacement for sendmail. Now the default on desktop installations of Mandrake (RedHat uses sendmail instead).
random - saves and restores the "entropy" pool for higher quality random number generation.
routed - manages routing tables
rstatd - kernel statistics server
rusersd, rwalld - identification of users and "wall" messaging services for remote users.
rwhod - server which maintains the database used by the rwho(1) and ruptime(1) programs. Its operation is predicated on the ability to broadcast messages on a network.
sendmail - mail transfer agent. This is the agent that comes with Red Hat.
smbd - the SAMBA (or smb) daemon, a network connectivity services to MS Windows computers on your network (hard drive sharing, printers, etc).
squid - An http proxy with caching. Proxies relay requests from clients to the outside world, and return the results. You would use this particular proxy if you wanted to use your linux computer as a gateway to the Internet for other computer on your network. Another (and probably safer at home) way to do it, is to set up masquerading.
syslogd - manages system activity logging.
smtpd - Simple Mail Transfer Protocol, designed for the exchange of electronic mail messages. Several daemons that support SMTP are available, including sendmail, smtpd, rsmtpd, qmail, zmail, etc.
usb - daemon for devices on Universal Serial Bus .
xfs - X font server
xntpd - finds the server for a NIS domain and stores the information about it in a binding file
ypbind - NIS binder. Needed if computer is part of Network Information Service domain.

Go to Part: 4.5 - Networking

LINUX NEWBIE ADMINISTRATOR GUIDE

ver. 0.120 2001-03-16 by Stan and Peter Klimas

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

Copyright (c) <1999,2000,2001> 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](#).

Quick navigation:

Part 4.5: Linux Newbie Administrator FAQ - Networking

Contents:

4.5 Networking

- 4.5.1 Would it be worth it to set up my home network? [p 106]
 - 4.5.2 How to set up my home network? [p 106]
 - 4.5.3 I have problems configuring my ppp dial out [p 109]
 - 4.5.4 How to browse the net from my networked computer without a modem? [p 112]
 - 4.5.5 How do I use Samba? [p 113]
 - 4.5.6 Sendmail [p 115]
 - 4.5.7 Simple web server (running Apache) [p 116]
 - 4.5.8 Simple ftp server [p 117]
 - 4.5.9 How can one access my computer from the outside world when I am on the net using phone connection? [p 118]
 - 4.5.10 Can my home computer get hacked? [p 119]
-

4.5 Setting up a network

4.5.1 Would it be worth it to set up my home network?

This is an excellent idea. It will let you use the machines that are in your closet now because they were not powerful enough to run standalone. The benefits include sharing hard drives, zipdrives, CDROMs, modem, printers, even soundcards, running programs remotely (the text or graphics mode), browsing the Internet on all computers at the same time over one phone connection. If you ever lose control of your machine, you may also be able to shut it down remotely and thus avoid possible problems--see the answer on shutting down for details.

4.5.2 How to set up my home network?

Hardware. Your hardware must be set up properly. Your network card should have been set up during the initial RedHat installation. If it wasn't, you may want to set up your network card now from X-windows. If you have more than one network card on your computer, you must also set up the second cards manually, since Linux will autodetect only 1 network card.

Start an X-terminal, execute "su" (to make yourself a root) and then run:

`/usr/bin/kernelcfg`

Under Linux, most drivers for network cards are implemented as modules. So setting up a card involves just inserting the proper module with its parameters. You don't need to recompile the kernel, nor even reboot your computer (unless you have an uncommon card). To figure out what module(s) and parameters you need, see the Linux Network Administrator Guide (file `/usr/doc/LDP/nag` on your system or check <http://metalab.unc.edu/mdw/index.html#guide>). This excellent guide is known as nag.

The parameters (IRQ and address) were a problem during my setup of two network cards on one computer--you must make sure that you don't have any hardware conflicts. A common source of problems is that the card wants to configure on IRQ 5, which is occupied by the SoundBlaster, or IRQ 3 which conflicts with the second serial port (COM2, cua1, ttyS1). Inspecting the files `/proc/interrupts`, `/proc/ioport` and reading bootup messages may help.

For example, my WD8013 card (same as SMC Elite and SMC Elite plus, according to nag) runs under IRQ 10 (set by a jumper on the card and I specified the IRQ in the kernel module setup), under the address 0x300 and uses the "WD" module. My "SMC EtherEZ" card (no jumper settings on the card) runs under IRQ 9, address 0x240 and uses "SMC EtherEZ" module. Please make sure you don't omit the leading "0x" in the address--it means "hexadecimal" and must be there else the number will be interpreted as decimal.

After the module is added from `kernelcfg` and the kernel is re-run (`kernelcfg` provides this, see the menus), inspect the file `/proc/modules` to see if the module is indeed loaded (or run `lsmod` as root). Information from `kernelcfg` goes to the file `/etc/conf.modules` so if you encounter difficulties (for example, I had difficulty removing modules inserted by mistake), just edit and adjust this file manually, e.g., using `pico`.

Network. After setting up the network cards and connecting the cables, set up the network by running (as root):

`netconf`

This program has help! `netconf` can also be run under GUI, but I did not really try it. If you need more understanding of how networking works, you may want to read the previously mentioned nag. Also, reading this material may be of help:
<http://ieee.uow.edu.au/~mjp16/wylug-netlinux/notes.html>.

It also contains very clear examples on how to set up a small office network that, like our home setup, has only part-time connection to the outside world.

How and what to fill up in `netconf` depends on your network. You surely want to fill up "Basic Host Information" (enable the first ethernet interface, `eth0`, fill in the name, aliases and IP number of your local computer) and the "Information About Other Hosts" (names, aliases and IPs of other computers on your home network). This information goes to the files: `/etc/hosts` and `/etc/sysconfig/network`, so you may want to inspect these files and adjust them manually.

When setting up the network, don't mess up with the "loopback driver" which has the IP 127.0.0.1. It is always there--it is the IP through which the computer talks to itself.

If you don't have an IP address (as will typically be the case for a home network with no permanent connection to the outside world), you may want to invent one. It does not matter very much what it is since when connecting to the outside world by your modem, you will be dynamically allocated an IP address (a machine can have many IP addresses at the same time). Your invented IP must be formally correct and the net mask must match the class of the network (class A, B or C). See the chapter on IP addresses in the already mentioned Network Administrator Guide (nag) (`/usr/doc/LPD/nag` on your system, or if you don't have it, download it now from <http://metalab.unc.edu/mdw/index.html#guide>).

For a home network, you might want to invent a class C network (up to 254 machines, the smallest) IP number which has the first three digits between 192 to 223. The last three digits identify the machine on your class C network and must be between 1 and 254 (don't use 0, it means "whole network" or 255 which is the "broadcast address"). The middle two sets of digits can be anything from 0 to 255. Thus 223.223.223.1 is the first machine on the class C network 223.223.223.0 (the last zero signifies the whole network), with the broadcast address 223.223.223.255. The network mask for a class C network is always 255.255.255.0 (unless you subdivide your network into smaller "subnet", which is not discussed here).

Rather than completely inventing an IP number, it might be safer/better to use one of the numbers reserved for "private networks". For me, the IP number 192.168.1.1 works just fine.

Typically, the first machine on a network is the one that is expected to have the connection to the outside world (since it was connected first, but there is no standard for that). So, I filled up the GATEWAY to xxx.xxx.xxx.1 (my first machine) on all machines, except for the machine xxx.xxx.xxx.1, where I left this field blank. Actually, although I left the field blank, `netconf` inserted the gateway 0.0.0.0 into the `/etc/sysconf` file. This was a source of an annoying error message during the loading of the network card on the bootup. To get rid of the message, I edited `/etc/sysconfig/network` and set it to something like this:

```
GATWAYDEV=  
GATEWAY=
```

[You don't want a gateway on your ethernet interface on this machine if it does not lead to the outside world. The `ppp` interface on this first machine will be set up as a default gateway once you connect through your modem, e.g. using `kppp`.]

The name of a computer is entirely arbitrary--the main user normally chooses a short word s/he likes. The domain name of the home network is also entirely arbitrary, unless you have a permanent connection in which case a domain name is registered to you. Try to invent something that does not exist yet--it could make your life easier once you have a permanent connection.

As for the DNS server (also called "named"), RedHat 5.2 and 6.0 comes with DNS preinstalled as "caching-only" so it is easy to configure as such. You may also choose not to use DNS at all--if DNS is looking up an "outside" server and can't find it, it can be a real show-stopper (the machine can appear to be hanged for up to a few minutes). To use local DNS, the "named" service must be enabled-- check this by running `setup`. To set up the caching-only DNS, fill up the appropriate boxes in `netconf`. E.g., I filled "nameserver 1" to my first machine on all computers (I entered the loopback address 127.0.0.1 on the machine xxx.xxx.xxx.1, and the proper xxx.xxx.xxx.1 address on all other machines).

No routes to other networks and hosts were required in my network, since I don't have other local networks. So I left this field blank.

Other than setting up the hardware correctly and filling up the info under `netconf` on each computer, as described above, I did not have to do anything on the standard RedHat to get my network working.

Reboot all computers one by one in any order (this is not necessary, but won't hurt you) and watch the boot messages (if they scroll too fast off screen, use <Shift><PgUp> to scroll up, or use `dmesg` from the command line to view them later). Did your cards configure correctly? Use the command `route` (as root) to see if the `eth0` interface is running. Use the `ping` command to test the connections between individual machines. Try to `telnet` your local computer to see if the loopback-only (`lo`) interface works:

```
telnet name_of_the_machine_you_are_sitting_at
```

After a successful login, you can exit the telnet session by typing

```
exit
```

Finally, try to telnet another computer on your network:

```
telnet name_of_a_remote_machine
```

If this works on all machines, your `eth0` network interface is set up.

After setting up your ppp and connecting to your Internet Service Provider (ISP), you will have another network interface (`ppp0`) and then will be able to telnet any machine in the world.

4.5.3 I have problems configuring my ppp dial out

GUI. If I were you, I would install `kde` and use `kppp`. (If you don't have `kde` installed, look here [p ??] for information on how to install it.) `kppp` is really easy to configure and run. To set up your ppp dial out, run `kppp` (it is under the "K" menu "Internet, or you can run it from the xterm by typing `kppp`), press the "setup" button, create an account, and fill out the information required: telephone number, authentication protocol, and your Internet Service Provider's (ISP) domain name and the Domain Name Server (DNS) number of your ISP. All this information should have been provided to you by your ISP.

Next, I created a device called `/dev/modem`, by symbolically linking it to the device `/dev/ttyS1` (as root)

```
ln -s /dev/ttyS1 /dev/modem
```

`ttyS1` should work if your modem is on the port that DOS calls COM2. Use `ttyS0` for COM1 and `ttyS2` for COM3. The modem will not dial at all if a wrong port is chosen.

Finally, I tried to run the `kppp` but got an error message complaining about a "lock". So, I made sure that the file `/etc/ppp/options` is empty by editing it (as root):

```
pico /etc/ppp/options
```

and deleting the word "lock", then saving the file. This problem does not exist in the more recent versions of kppp.

After these simple steps, the kppp could connect to my ISP with no problems on RedHat 5.2.

RedHat 6.0 required one additional step: setting the "suid" ("substitute user id") so that "kppp" runs with the effective user id of root (because it needs to access hardware directly). Without it, kppp complains that "it was not properly set up" and "can't create lock file". This has to be done as root:

```
cd /usr/bin/  
chmod a+s kppp
```

Troubleshooting. If your modem refuses to dial on the port that you are positive is chosen properly, maybe the modem is not set up properly (or maybe it is a "winmodem"? Then throw it away and buy a proper modem). In one instance, I had to run kppp setup, edit the "modem commands" and input ATZ1 as the initialization string (instead of the default ATZ for a standard Hayes-compatible modem).

If your telephone line requires pulse dialing (instead of the default touch-tone dialing), you may need to change the modem dialup command from "ATDT" ("ATantion Dial Touchtone") to "ATDP" ("ATantion Dial Pulse"). These commands work with any standard Hayes-compatible modem.

If your modem dials correctly and you are able to connect, but your authentication fails, perhaps your Internet Service Provider (ISP) uses a different authentication protocol. Call them and ask what authentication protocol they use. Or try pap, terminal-based or chap (in your kppp setup) until you find one which works with your ISP.

In one instance, I had a problem with the reliability of establishing a connection (the error would pop up saying something like: "time-out for the pppd startup", and the connection would establish only once every few trials). The problem was solved by changing the "flow control" option (in the kppp "setup" under "device") from "CRTSCTS" to XON/XOFF. (Still CRTSCTS is the recommended flow control method in most cases.)

The performance (speed) and the reliability of the connection with my kppp was much improved after I upgraded from RH5.2 to RH6.0. So if you have problems with these, maybe it is time to upgrade.

If you keep having problems setting up ppp, you may want to try minicom to see if you can get your modem working from there. minicom is something like PROCOMM for Linux. It should be present on your system if you chose to install it during your RedHat initial setup. Here is a post from a newsgroup comp.os.linux.help which explains how to start ppp manually using minicom (edited for space):

From: mark <balthazaar@one.net.au> Subject: Re: pppd problem with kppp
BachuZ wrote:

>>Also, for an experiment, try using minicom to connect to your ISP, start ppp manually ... this can prove buggy scripts. >how would u do that?

Easy!! If your ISP doesn't allow a manual logon then you might be in trouble. Every ISP I've ever used does allow this, so.. 1. Start minicom. 2. dial your ISP. 3. Log in. 4. After your ISP starts PPP, quit minicom with ALT-Q (or whatever the sequence is to 'quit without reset'). 5. start pppd, eg:

```
pppd -d -detach /dev/modem 115200 &
```

OK, PPP will be running. Try pinging your ISP or another known IP address. That will test everything is OK. BTW, this is all in the PPP-HOWTO. If you can get PPP running this way, then you have a scripting problem. If PPP doesn't work, you have a PPP configuration problem. Cheers.

Command line. If you would like to start your ppp from the command line, run `netconf` (as root) to configure your first ppp interface (ppp0). The information you must enter is similar to what you entered when setting up kppp (have a look above!): the proper device for the modem port, modem initialization and dialup strings, the telephone number of your Internet Service Provider (ISP), the proper authentication protocol (by entering the login name and password into the right slot). In older versions, the `netconf` utility lacked a place to enter the IP addresses of my ISP DNS server, so I edited the file `/etc/ppp/pap-secrets` (I use PAP authentication protocol) and added the two DNS IP addresses at the end of the setup line which was created by `netconf` so it looked like this:

```
# added by linuxconf
my_login_name ppp0 my_password 111.111.111.111 222.222.222.222
```

When done with the settings, I could start my ppp0 interface using the command (as root, unless I specified in `netconf` that normal users can start the interface):

```
ifup ppp0
```

and shut it down with

```
ifdown ppp0
```

Setting up the command line ppp was not more difficult on my machine than running kppp and the connection is more reliable for me. There is lots of command line scripts to start/stop ppp, but they apparently are not so easy to set up and use, and many newbies seem to have problems with them.

If this minimalistic setup of ppp does not work for you, here are some useful links:

Roderick A. Anderson <raanders@altoplanos.net> wrote:

I have a web page on setting up diald to work with RedHat Linux 5.x that works for me every time. It is at <http://home.altoplanos.net/~raanders/diald.html>

Bill Unruh <unruh@physics.ubc.ca> wrote:

I just wanted to bring your attention to the page <http://axion.physics.ubc.ca/ppp-linux.html> for detailed instructions for setting up ppp. This is especially for cases in which the remote side uses (perhaps without the ISP even telling you) PAP or CHAP. While kppp is useful, there are a number of situations where it can fail.

4.5.4 How to browse the net from my networked computer without a modem?

Another computer on your network must have a modem (or another Internet connection) though :-). Set up IP masquerading. This way, all requests going from your network to your Internet Service Provider (ISP) appear to have originated from a single computer, and your ISP will let them through.

ON REDHAT 5.2, simple masquerading required just one command (on the computer with the modem):

```
/sbin/ipfwadm -F -p m
```

This sets up masquerading as your default forwarding policy of your IP firewall, and therefore is insecure but probably ok for a home user. (The danger is that if somebody hacked your computer, s/he can use it as an anonymous forwarder to hide his identity. Whatever malicious the hacker does to anybody, you take the blame. The hacker can even set his route to "tunnel" back to your network thus concealing his identity from you.) For more info, please check the file `/usr/doc/HOWTO/mini/IP-Masquerade`. A more secure setup is shown here:

```
ipfwadm -F -p deny
ipfwadm -F -a m -S 192.168.1.1/32 -D 0.0.0.0/0
ipfwadm -F -a m -S 192.168.1.3/32 -D 0.0.0.0/0
ipfwadm -F -a m -S 192.168.2.0/24 -D 0.0.0.0/0
```

This sets up the default policy to "deny" and explicitly masquerades two machines with IPs 192.168.1.1 and 192.168.1.3. It also masquerades any machine from the network 192.168.2.0. The number /32 stands for point-to-point networking (this means "machine-to-machine"), the option /24 identifies a class C network. The -D 0.0.0.0/0 identifies the default route that the machines to be masqueraded use to go out to the Internet.

ON REDHAT 6.0 THE NAME OF THE COMMAND is `ipfwadm-wrapper` (instead of `ipfwadm`) and I had to use the second, more secure method (setting up masquerading as the default policy does not seem to work on my system any more). Also, with newer kernels (2.0.34 and later) forwarding is disabled by default and must be turned on using:

```
echo "1"> /proc/sys/net/ipv4/ip_forward
```

Actually, `ipfwadm-wrapper` is a wrapper because it lets me use the old rules of setting up the firewall policies using the brand new firewalling kernel code. Doing something like this may work better for you:

```
ipchains -P forward DENY
ipchains -A forward -s 192.168.1.0/24 -j MASQ
```

If you would like to have this command(s) always executed on your system startup, add it as the last line(s) to the file `/etc/rc.d/rc.local`. This file is something like AUTOEXEC.BAT in DOS. As always, it is recommended to read the manual page and other documentation to see what the command(s) does and what are the other options:

```
less /usr/doc/HOWTO/mini/IP-Masquerade
man ipfwadm
man ipchains
```

4.5.5 How to use Samba?

Samba (smb) is for Linux-MS Windows networking. It is a program that makes a Linux computer pretend to be a MS Windows NT server, and thus lets your MS Windows 3.1/95/98-based computers connect to the network. Samba not only replaces WinNT--it is acclaimed to do a much better job than WinNT!

One of the three machines on my home network is a dual Linux/Win95 boot. I configured my network so that if I boot Win95, another Linux machine acts as a Samba server. This way, the Win95 machine has access to the network printer, shared directories on Linux, can telnet, browse the Internet through a modem on Linux, etc. I can also access the files on the remote Win95 machine from Linux computers.

It is necessary to configure Samba only on one Linux machine.

First, I ran `setup` as root, choose "System Services" (RH6.0) or "ntsysv" (RH5.2) from the menu, and make sure that the "smb" service is enabled. (The program `setup` has help if you press F1. Just in case you were curious what the different services are for.) If smb is not listed there, maybe you did not install it during your RedHat setup? You may want to put your RedHat CD into the CDRom, mount the CD, start `glint` (RH5.2) or `gnorpm` (RH6.0) installation utility (as root) and add Samba to your system.

The second part is to configure Samba. This is relatively simple since the Samba configuration is done through a single, well commented file: `/etc/smb.conf`. The minimum setup includes specifying the workgroup name. Note that if you don't fill up the "host allow" option, all hosts are allowed, which is probably ok for the home network. The other options in the example `/etc/smb.conf` supplied with your RedHat CD may require adjustment to suit your particular needs, but they didn't have to be changed to get a Windows machine connected to my Samba server.

Check that the options

```
domain master = yes
domain logons = yes
```

are enabled in `/etc/smb.conf`. You probably want them.

Under RH6.0, you may also perform a basic samba setup using (as root):

```
netconf
```

The third step is to configure your MS Windows 95 (or whatever) for networking. This is done exactly as if you were connecting to a WinNT server. Make sure to enable the networking ("client for Microsoft Network") and fill up the workgroup name. To login on the network when booting MS Windows, use your Linux user ID and password. To see if it worked, click on the icon "Network Neighborhood"--your Linux server should be listed there, and underneath you should see the shared directories and printers that you chose to share in the file `/etc/smb.conf`.

If you can't see or use the public directories, make sure that you created them and set the proper read/write permissions for all users.

MS Windows 95b and above (95c, 98, and newer NTs) may use password encryption. This will make your logins from the Window's machine fail, and you may need to enable this option in the `/etc/smb.conf` file:

```
encrypt passwords = yes
```

On the Linux server, you can start, stop, restart and check the Samba status using these commands (as root):

```
samba start
samba status
samba restart
samba stop
```

You need to re-start samba after making changes to your `/etc/smb.conf` file.

You can browse the net using your Netscape for Windows if you are connected to your Linux computer through Samba and the Linux machine is currently connected to the Internet. To do this, the IP masquerading must be set up on the Linux machine with the modem (described here [p 112]) and you must enable the Samba dns name resolution in `/etc/smb.conf`:

```
dns proxy = yes
```

and then tell Windows to enable the dns server, specify the Linux server name and IP address (in ControlPanel-Network-TCP/IP).

You can also mount a remote Windows directory onto your Linux filesystem. Look here [p ??] to see how.

The above described just a minimal Samba setup. You can get more information from:

```
man samba; man smb.conf; man smbclient; man smbmount
less /usr/doc/HOWTO/SMB-HOWTO (under RH5.2)
documentation in /usr/doc/samba-2.0.3/doc (under RH6.0)
http://www.sfu.ca/~yzhang/linux/samba/ (samba minihowto)
http://www.germanynet.de/teilnehmer/101,69082/samba.html
```

and also by studying the file `/etc/smb.conf`.

4.5.6 Sendmail

Sendmail is the Internet standard mail-transport system and the default mail-transport on RedHat Linux (Mandrake uses PostFix instead). As a user, you don't use sendmail directly--sendmail is the underlying engine that manages the mail on your machine in the background. To read/send mail in the text mode, you probably want to use pine or elm (choose one and stay with it--it can be inconvenient to manage two separate mail boxes). In KDE, you may choose to use the "mail client" available from the K-menu. To communicate with the outside world through your ppp dial-out, you probably would like to use the Mozilla mailer that is built into your Netscape and which communicates directly with your remote Internet-Service-Provider-based mailbox.

Sendmail is very flexible and robust, but also notoriously difficult to manage if you needed to customize it to your specific needs. Luckily, Red Hat (5.2 or 6.x) comes with the sendmail that runs out-of-box (with some limitations though).

On my home system (default Red Hat setup), I can send mail to another user on the same machine with no problem (e.g. using pine). I can send mail locally or anywhere in the world from any mailer once I am connected to my Internet Service Provider through the modem (IP masquerading enabled [p 112]). But when I am not connected, the mail sits in the queue and waits for the Internet connection, even if the mail is to be sent to another computer on my home network (a minor annoyance). It gets sent once I connect. (It happens because sendmail is trying to do a DNS lookup and this is not available on my system--RedHat default DNS is cache-only.) If you really want to avoid this on RH6.x, you may use `netconf` (as root) and specify that sendmail is not to use DNS at all (Linuxconf that comes with RH5.2 does not give you the choice). I do use DNS.

Under RedHat 6.x, you may also want to use `netconf` to configure other sendmail options using (as root). I entered the name of my Internet Service Provider and the names of other machines on my home network under "mail deliver system"- "relay to hosts". I also added the names of my home network machines under "relay for by name".

You should also be aware of the limitation of your simple setup that arises if you send e-mail from your home network (for example using pine), to the world beyond your network. If you invented the IP address and your domain name is unregistered, there is no chance you will receive a reply. Your outgoing mail is given a "reply" address in the form: "user_login_name@machine.domain". This is NOT ok since your domain name does not exist, according to any DNS in the outside world, hence no way for any reply to ever get to you. To overcome this, you may use the Netscape mailer to communicate to the outside of your home network. Netscape uses the settings you enter in its "edit-preferences" to communicate directly to your ISP-based mailbox (which resides on a registered server of your ISP) and thus bypasses your unregistered-home-network-based e-mail system. Another possibility is to specify the correct address in the "reply to" field. If you do it in Netscape, you can setup your NetscapeMail to use your local Linux computer as the mail server, and this way be able to send e-mail from your NetscapeMail also to the computers on your home network (not only in the "outside world"). The option "reply-to" can also be set in the KDE "mail client" setting, if you use the KDE mailer. Also in "pine" you can specify the reply-to address under "setup-configure", "customized-hdrs" with something like:

Reply-to: joe@joe_net.net

where "joe@joe_net.net" is your good reply-to e-mail address.

If you do wish to fetch your mail from your Internet provider-based mailbox onto your account without the help of Netscape, you may consider installing `fetchmail` (see `man fetchmail`).

4.5.7 Simple web server (running Apache)

To set up a simple apache web server was extremely easy under RedHat 6.0. First of all, I made sure to install the apache web server rpm package during my RedHat installation. If you didn't do it, you can put the RH CD in your CDROM now, mount it, and install the package `apache-*.rpm` (instead of the "*" the name also contains the version number and the platform). If I know what I want to install, it is simple do it (as root):

```
cd /mnt/cdrom/RedHat/RPMS
rpm -ivh apache*.rpm
```

Then, I run "setup" (as root) and make sure that the httpd daemon (under "system services") is enabled. [A daemon is a program that sits in the background and wakes up when it is needed. In the case of httpd, it gets awakened when somebody calls on your http server.]

Now, the httpd will start automatically every time I boot the computer. I can also start it manually using (as root):

```
/etc/rc.d/init.d/httpd start
```

and shut it down using:

```
/etc/rc.d/init.d/httpd stop
```

I should already have the directory: `/home/httpd` (check if it exists), and under it, the subdirectory `html`--this is the "root directory" for people accessing my computer from the web ("their root directory" means that they will not be able to access any directories above the directory `/home/httpd/html/` on your system). On default, this directory contains some html files and manual that the apache installation program put there. So I should now be able to connect to my web server from another machine on my home network. For example I would type on the Netscape "Location" line:

```
http://my_http_server_name
```

and be able to browse the "apache" manual.

To put my own content on my web server, I move the "apache" sample files somewhere else, and copy or link my html files (the ones which I want to display to the public) to the directory `/home/httpd/html/` (don't forget to include the file `index.html`, this is the one that appears first when somebody connects to your server).

The apache configuration files and log files are in the directory `/etc/html/` if you wanted to view/customize them. The log file can be viewed in "real time" using this simple command (as root):


```
tail -f /etc/httpd/logs/access_log
```

[The `tail` command normally displays the end ("tail") of a text file. With the option "-f", "tail" keeps displaying the end of the log file as it grows--really handy to monitor the log file and see who logs onto your server.]

For graphical setup of the Apache server, try this (in an X-window, as root, if you installed "Comanche" from your RedHat CD):

```
comanche
```

4.5.8 Simple ftp server

With my RedHat Linux distribution, setting up an ftp server could not be simpler--it just works out-of-box. This is because the ftp service is enabled on default as one of the standard services (as is telnet and gopher), in the file `/etc/inetd.conf`. Here is the relevant part of my `/etc/inetd.conf`:

```
ftp      stream  tcp      nowait  root    /usr/sbin/tcpd  in.ftpd
-l -a
telnet   stream  tcp      nowait  root    /usr/sbin/tcpd
in.telnetd
gopher   stream  tcp      nowait  root    /usr/sbin/tcpd  gn
```

The second part of my ftp server setup is in the file `/etc/passwd` which defines the ftp account:

```
ftp:*:14:50:FTP User:/home/ftp:
```

The ":" is a field separator. The first field is the account (user) name "ftp", the "*" in the second field indicates that the password is disabled (nobody can login under the "ftp" user name), the user id is 14, group id is 50, "FTP User" is a comment, the home directory is `/home/ftp`, the last field is empty (for "normal" user accounts, it specifies the name of the shell for the user).

Because this setup was already done for me by RedHat, anybody can ftp my computer and either login as a user (will be prompted for password and directed to his/her home directory), or login as "anonymous" and give his/her e-mail address as a password. Any user can also enter something like this on the Netscape "location" line:

```
ftp://my_computer_name
```

and connect automatically (Netscape will take care of sending the "anonymous" user name and the e-mail address as password).

The "anonymous" ftp users are directed to the directory `/home/ftp`, which appears to be a root directory to them (they cannot access any directory above it). I put the files I want to serve in the subdirectory `/home/ftp/pub`.

The directory `/home/ftp/bin` contains the commands that the remote users are able to execute. On my system these are: `compress`, `cpio`, `gzip`, `ls`, `sh`, `tar`, `zcat`; all with execute-only (111) permissions.

The directory `/home/ftp/etc` contains the setup files necessary for the anonymous account to function (edited `passwd`, `group`, `ld.so.cache`). The directory `/home/ftp/lib` contains the libraries (I guess these libraries are used by the commands that the anonymous ftp users are allowed to run).

4.5.9 How can one access my computer from the outside world when I am on the net using phone connection?

The only difficulty is that your IP address is dynamically allocated to you by your Internet Service Provider (ISP) from their IP address pool, and therefore the IP address is not the same every time you connect (unless you made specific arrangements with your ISP). To telnet, ftp, or access your web pages (served by your apache web server) from the outside world, one has to know your current IP address. To find out my current IP address, I use this "interface configuration" command which, when run without any parameters, just displays info on all active network interfaces present on your machine:

```
/sbin/ifconfig
```

On my machine this displays three paragraphs of information on: `eth0` (the first ethernet network interface that leads to other computers on my home network), `lo` (the loopback-only interface, the one with IP 127.0.0.1, this one must be present on every machine), and `ppp0` (the first point-to-point protocol interface). My current IP address, assigned to me by my ISP, is displayed under the `ppp0` heading. (Your Linux machine can have multiple IP addresses assigned at the same time, so if you have a "static" IP that you use on your home network, it is still valid but visible only on your home network.)

Once I know the IP address, I can send it through ICQ or e-mail to a friend, who can then, for example, telnet or ftp my computer (s/he must have an account on my machine) and run a program on my linux machine, or enter `http://my_ip_address` on the "location" line in the browser to browse my home web pages, etc. If the friend has Xwindows on his/her local machine, s/he can even run a GUI program on a my server and direct the display on his computer.

I can also write a short script that will automatically notify my friend when I am connecting to the Internet and enter the name of the script under `kppp-setup-account-edit-dial`-"execute program upon connect". Here is my script which notifies me at work when somebody in my house is going on-line (I entered the text into a text file and made the file executable using `chmod o+x file_name`):

```
#!/bin/bash
sleep 15
/sbin/ifconfig | mail -s notification my_email_address
```

The first line of this script tells my computer to interpret this text file as a bash shell script. The second line makes the script wait 15 seconds (just to make sure that the e-mail is not sent before the ppp connection is fully established). The third line executes the `ifconfig` command and pipes the output to the mail utility that sends it to `my_email_address` under the subject "notification".

A more flexible way to access your home computer remotely is to configure it as a dial-up ppp server (as opposed to the dial-in client that you use when you connect to your ISP). If somebody has a simple recipe how to do it, please drop a line.

To summarize, unlike MS Windows 3.x/95/98 which severely restricts traffic to your computer, Linux is very network oriented and it is easy to make all kinds of network connections both FROM and TO your Linux computer. The powerful networking features are generally considered a Linux strength but, from a real newbie point of view, they can be a problem (see the next question).

4.5.10 Can my home computer get hacked?

Unfortunately, this is perfectly possible and attempts to do so are quite common. Every time you are connected to your Internet Service Provider (ISP) you are at risk. Read the previous answer if you would like to know how it is possible. Obviously, the risk is much higher if you have a permanent Internet connection (e.g. cable modem), and it is lower if your connections are more transient (as typical with short-duration, over-the phone modem connections).

The real danger is that the intruder, if s/he is able to login onto your machine on any account, may find another security hole and get root access. This is particularly possible if you are a real newbie administrator and/or your machine is not really security oriented (you are at home, aren't you--who would care about security, you think!).

To protect yourself, just never let a stranger log in onto your computer. Have fairly long passwords that contain both numbers and letters for all accounts on your computer. Change the passwords occasionally. The best way to enforce the password policies on all users of your computer is to run (as root, available on RH6.0) `linuxconf` and under "password and account policies" change the minimum password length to 6 or more characters, the minimum number of non-alpha characters (i.e., not-letters) to 1 or 2, the number of days after which the password must be changed to something like 90 or less, and set a warning about password expiry to something like 7 days before the expiration. Check here for more info on weak passwords. Absolutely NEVER create an account with no password, or with a silly weak password.

It is also an excellent idea to occasionally screen the files that contain a record of all the logins onto your computer: `/var/log/secure` (the most recent log) `/var/log/secure.1` (older log) `/var/log/secure.2` (yet older log), etc. There are also other useful log files in the directory `/var/log` that you might want to view, check them out from time to time. The most typical "warning" sign is a scanning of the ports on your computer: there are repeated entries on connection request from the same IP number to your system telnet, ftp, finger and other ports--somebody tried to learn more about your system.

If you never use remote connections to your home Linux machine, you might want to restrict the rights to use telnet, ftp, etc. (all the network services are listed in the file `/etc/inetd.conf`) to the machines on your home network. The access is controlled by two files: `/etc/hosts.allow` and `/etc/hosts.deny`. These access-control files work as follows. When an outside connection is requested, the file `/etc/hosts.allow` is scanned first and if the name of the machine from which the connection is requested is matched, the access is granted (irrespective of any entry in `/etc/hosts.deny`). Otherwise, the file `/etc/hosts.deny` is scanned, and if the name of the machine from which the connection is requested is matched, the connection is closed. If no matches are found in either file, the permission is granted. As an example, you can deny access to telnet and ftp your home server from any machine from outside your home network by inserting the following entry in the file `/etc/hosts.deny`:

```
in.telnetd, in.ftpd: ALL EXCEPT LOCAL, .your_home_domain.name
```

For more info, check the excellent "Linux Network Administrator Guide" which is surely present on your RedHat (or whatever) distribution CD. I printed this book and had it hardcovered.

For security reasons, it is also a good idea not to advertise the of the OS/version/kernel version that you use. I replaced the contents of the file `/etc/issue` and `/etc/issue.net` which on my computer read:

```
Red Hat Linux release 6.2 (Zoot)
Kernel 2.2.14-5.0 on an i586
```

with something like this:

```
WARNING: THIS IS A PRIVATE NETWORK
UNAUTHORIZED USE IS PROHIBITED AND ALL ACTIVITIES ARE LOGGED
IBM S/390 LINUX
```

This blends a joke with a little bit more security (I hope).

The contents of the files `/etc/issue` and `/etc/issue.net` are recreated at every reboot (when the script `/etc/rc.local` is run). So, to make the changes permanent, I can make these files read-only for all users (as root):

```
chmod a=r /etc/issue*
```

Instead of the last command, I could have edited (as root), the script `/etc/rc.d/rc.local` and commented out 5 lines with `###` so that the relevant part reads:

```
# This will overwrite /etc/issue at every boot. So, make any
changes
# want to make to /etc/issue here or you will lose them when you
reboot
### echo "> /etc/issue
### echo "$R">> /etc/issue
### echo "Kernel $(uname -r) on $a $SMP$(uname -m)">> /etc/issue
### cp -f /etc/issue /etc/issue.net
### echo>> /etc/issue
```

Another good security measure is to disable ping. Ping is a sonar-like response that your computer sends back when inquired. It is mostly useful for setup and debugging, to probe whether your machine is available on the network. It can also be used for probing your machine and/or attacking it by flooding with ping requests ("ping of death"). To disable my machine response to pinging from the net, I use the IP masquerading. I took and slightly modified the following command and explanation from

<http://www.securityfocus.com/focus/linux/articles/linux-securing2.html>:

```
ipchains -A input -p icmp --icmp-type echo-request -i ppp0 -j
REJECT -l
          (1) (2)      (3)                (4)                (5)
(6)      (7)
```

The ipchains flags explained:

1. (A)ppend a new rule.
2. The chain to apply the rule to, in this case the rule will apply to ingress (input) packets.
3. (P)rotocol to apply the rule to. In this case, it is icmp.
4. ICMP type, in this case all icmp echo requests will be blocked. "ICMP echo" means ping.
5. Interface name. In this case, it is the first over-the-phone connection, ppp0.
6. Target, or what should actually be done with the packet in question.
7. Log all packets matching the rules criteria to system log file.

IP masquerading was described in more detail in the chapter on masquerading [p 112] of this Guide.

Go to Part: 5 - Linux Shortcuts and Commands

LINUX NEWBIE ADMINISTRATOR GUIDE

ver. 0.120 2001-03-16 by Stan and Peter Klimas

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

Copyright (c) <1999,2000,2001> 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](#).

Quick navigation:

Part 5: Linux Shortcuts and Commands

Contents:

- 5.1 Linux essential shortcuts and sanity commands [p 123]
 - 5.2 Help commands [p 126]
 - 5.3 System info [p 127]
 - 5.4 Basic operations [p 130]
 - 5.5 File management [p 131]
 - 5.6 Viewing and editing files [p 132]
 - 5.7 Finding files [p 133]
 - 5.8 Basics of X-windows [p 134]
 - 5.9 Network apps [p 135]
 - 5.10 File (de)compression [p 137]
 - 5.11 Process control [p 138]
 - 5.12 Basic administration commands [p 139]
 - 5.13 Program installation [p 145]
 - 5.14 Accessing drives/partitions [p 146]
 - 5.15 Network administration tools [p 135]
 - 5.16 Sound-related commands [p 148]
 - 5.17 Graphics-related commands [p 150]
 - 5.18 Small games [p 152]
-

Intro. This is a practical selection of the commands we use most often. Press <Tab> on the empty command line to see the listing of all available commands (on your PATH). On my small home system, it says there are 3786 executables on my PATH. Many of these "commands" can be accessed from your favourite GUI front-end (probably KDE or Gnome) by clicking on the right menu or button. They can all be run from the command line (unless you didn't install one, but they all came on our CDs). Programs that require GUI have to be run from under the GUI, for example from a terminal opened in kde or gnome (xterm). Some more advanced tools are described in the Part Learning with Linux of this Guide.

Legend:

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

Notes for the UNIX Clueless:

1. LINUX IS CASE-SENSITIVE. For example: Netscape, NETSCAPE and nEtscape would be three different commands (but of the three, only netscape is available on my system). Also `my_file`, `my_file`, and `my_FILE` are three different files. Your user login name and password are also case sensitive. (This goes with the tradition of UNIX and the "c" programming language being case sensitive.)
2. Filenames can be up to 256 characters long and can contain letters, numbers, "." (dot), "_" (underscore), "-" (dash), plus some other non-recommended characters.
3. Files with names starting with "." are normally not shown by the `ls` (list) or `dir` command. Think of these files as "hidden". Use `ls -a` (list with the option "all") to see these files.
4. "/" is an equivalent to DOS "\" (root directory, meaning the parent of all other directories, or a separator between a directory name and a subdirectory or filename). For example, try `cd /usr/doc`
5. Under Linux, all directories appear under a single directory tree (there are no DOS-style drive letters). This means directories and files from all physical devices are merged into this single file system.
6. In a configuration file, a line starting with # is a comment. When changing a configuration file, don't delete old settings--comment out the original lines. Always insert a short comment describing what you have done.
7. Your personal settings are in your home directory which is `/home/your_user_login_name`. Many settings are kept in files with names starting with a dot "." so as to keep them out of your way (see point 3 above).
8. System-wide settings are kept in the directory `/etc`.
9. Under Linux, as in any multiuser operating system, directories and files have an owner and set of permissions. You will be typically allowed to write only to your home directory which is `/home/your_user_login_name`. Learn to use the file permissions else you will be constantly annoyed with Linux.
10. Command options are introduced by a dash, "-", followed by a single letter (or -- when the option is more than one letter). Thus "-" is an equivalent of DOS's switch "/". For example, try `rm --help`.
11. Type `command&` (the command name followed by the & sign) to start a command in the background. This is usually the preferred way of starting a program from the X-window terminal.

5.1 Linux essential shortcuts and sanity commands

`<Ctrl><Alt><F1>`

Switch to the first text terminal. Under Linux you can have several (6 in standard setup) terminals opened at the same time.

`<Ctrl><Alt><Fn>` (n=1..6)

Switch to the nth text terminal. (The same could be accomplished with the rarely used command `chvt n`. "chvt" stands for "change virtual terminal").

`tty`

Print the name of the terminal in which you are typing this command. The number of the active terminal can be printed using the command `fgconsole`.

`<Ctrl><Alt><F7>`

Switch to the first GUI terminal (if X-windows is running on this terminal).

<Ctrl><Alt><Fn> (n=7..12)

Switch to the nth GUI terminal (if a GUI terminal is running on screen n-1). On default, nothing is running on terminals # 8 to 12, but you can run another server there.

<Tab>

(In a text terminal) Autocomplete the command if there is only one option, or else show all the available options. THIS SHORTCUT IS GREAT! It even works at LILO prompt!

<ArrowUp>

Scroll and edit the command history. Press <Enter> to execute.

<Shift><PgUp>

Scroll terminal output up. Work also at the login prompt, so you can scroll through your bootup messages.

<Shift><PgDown>

Scroll terminal output down.

<Ctrl><Alt><+>

(in X-windows) Change to the next X-server resolution (if you set up the X-server to more than one resolution). For multiple resolutions on my standard SVGA card/monitor, I have the following line in the file /etc/X11/XF86Config (the first resolution starts on default, the largest determines the size of the "virtual screen"):

```
Modes "1024x768" "800x600" "640x480" "512x384" "480x300" "400x300"
"1152x864"
```

Of course, first I had to configure the X server, either by using Xconfigurator, xf86config, or manually by editing the file /etc/X11/XF86Config, so that it supports the above resolutions (mostly the matter of uncommenting the line that defines my video chipset, and specifying the synchronization frequencies my monitor supports).

<Ctrl><Alt><->

(in X-windows) Change to the previous X-server resolution.

<Ctrl><Alt><Esc>

(in X-windows, KDE) Kill the window I am going to click with my mouse pointer (the pointer changes to something like a death symbol). Similar result can be obtained with the command xkill (typed in X-terminal). Useful when an X-window program does not want to close (hangs?).

<Ctrl><Alt><BkSp>

(in X-windows) Kill the current X-windows server. Use if the X-windows server cannot be exited normally.

<Ctrl><Alt>

Shut down the system and reboot. This is the normal shutdown command for a user at the text-mode console. Don't just press the "reset" button for shutdown!

<Ctrl>c

Kill the current process (works mostly a small text-mode applications).

<Ctrl>d

(pressed at the beginning of an empty line) Log out from the current terminal. See also the next command.

`<Ctrl>d`

Send [End-of-File] to the current process. Don't press it twice else you also log out (see the previous command).

`<Ctrl>s`

Stop the transfer to the terminal.

`<Ctrl>q`

Resume the transfer to the terminal. Try if your terminal mysteriously stops responding.

`<Ctrl>z`

Send the current process to the background.

`exit`

Logout. I can also use `logout` for the same effect. (If you have started a second shell, e.g., using `bash`, the second shell will be exited and you will be back in the first shell, not logged out.)

`reset`

Restore a screwed-up terminal (a terminal showing funny characters) to default setting. Use if you tried to "cat" a binary file. You may not be able to see the command as you type it.

`<MiddleMouseButton>`

Paste the text which is currently highlighted somewhere else. This is the normal "copy-paste" operation in Linux. (It doesn't work with Netscape and WordPerfect which use the MS Windows-style "copy-paste". It does work in the text terminal if you enabled "gpm" service using "setup".) Best used with a Linux-ready 3-button mouse (Logitech or similar) or else set "3-mouse button emulation").

`~`

(tilde) My home directory (normally the directory `/home/my_login_name`). For example, the command `cd ~/my_dir` will change my working directory to the subdirectory "*my_dir*" under my home directory. Typing just "`cd`" alone is an equivalent of the command "`cd ~`".

`.`

(dot) Current directory. For example, `./my_program` will attempt to execute the file "*my_program*" located in your current working directory.

`..`

(two dots) Directory parent to the current one. For example, the command `cd ..` will change my current working directory one level up.

`<Alt><SysRq><command_key>`

(Non-essential.) This is a group of commands implemented at the Linux kernel level (low level). It means, chances are these key combinations will work most of the time. The combinations are meant for debugging purposes and in an emergency; you should try other, safer solutions first. The key `<SysRq>` is also known on PC as `<PrintScreen>`.

`<Alt><SysRq><k>` Kill all processes (including X) which are running on the currently active virtual console. This key combination is known as "secure access key" (SAK).

`<Alt><SysRq><e>` Send the TERM signal to all running processes except init, asking them to exit.

`<Alt><SysRq><i>` Send the KILL signal to all running processes except init. This may be more successful in killing runaway processes than the previous key combination, but it may cause some of

them to exit abnormally.

<Alt><SysRq><l> Send the KILL signal to all processes, including init. The system will not be functional.

<Alt><SysRq><s> Run an emergency sync (cache write) on all mounted filesystems. This can prevent data loss.

<Alt><SysRq><u> Remount all mounted filesystems as read-only. This has the same effect as the sync combination above, but with one important benefit: if the operation is successful, fsck won't have to check all filesystems after a computer hardware reset.

<Alt><SysRq><r> Turn off keyboard raw mode. This can be useful when your X session hangs. After issuing this command you may be able to use <CTRL><ALT>.

<Alt><SysRq> Reboot immediately without syncing or unmounting your disks.

<Alt><SysRq><o> Shut the system off (if configured and supported).

<Alt><SysRq><p> Dump the current registers and flags to your console.

<Alt><SysRq><t> Dump a list of current tasks and their information to your console.

<Alt><SysRq><m> Dump memory info to your console.

<Alt><SysRq><digit> The digit is '0' to '9'. Set the console log level, controlling which kernel messages will be printed to your console. For example, '0' will cause only emergency messages like PANICs or OOPSes displayed on your console.

<Alt><SysRq><h> Display help. Also, any other unsupported <Alt><SysRq><key> combination will display help.

5.2 Help commands

any_command --help |more

Display a brief help on a command (works with most commands). For example, try `cp --help |more`. "--help" works similar to DOS "/h" switch. The "more" pipe is needed when the output is longer than one screen.

man topic

Display the contents of the system manual pages (help) on the topic. Press "q" to quit the viewer. Try `man man` if you need any advanced options. The command `info topic` works similar to `man topic`, yet it may contain more up-to-date information. Manual pages can be hard to read--they were written for UNIX programmers. Try *any_command* --help for a brief, easier to digest help on a command. Some programs also come with README or other info files--have a look to the directory `/usr/doc`. To display manual page from a specific section, I may use something like: `man 3 exit` (this displays an info on the command `exit` from section 3 of the manual pages) or `man -a exit` (this displays man pages for `exit` from all sections). The man sections are: Section 1-User Commands, Section 2-System Calls, Section 3-Subroutines, Section 4-Devices, Section 5-File Formats, Section 6-Games, Section 7-Miscellaneous, Section 8-System Administration, Section 9, Section n-New. To print a manual page, I use: `man topic | col -b | lpr` (the option `col -b` removes any backspace or other characters that could make the printed manpage difficult to read).

info topic

Display the contents of the info on a particular command. `info` is a replacement for man pages so it contains the most recent updates to the system documentation. Use <Space> and <BkSpace> to move around or you may get confused. Press "q" to quit. A replacement for the for the bit confusing info browsing system might be `pinfo` - try if you like it any better.

`apropos topic`

Give me the list of the commands that have something to do with my topic.

`whatis topic`

Give me a short list of commands matching my topic. `whatis` is similar to `apropos` (see the command above)--they both use the same database. But `whatis` searches keywords, while `apropos` also searches the descriptions of the keywords.

`help command`

Display brief info on a bash (shell) built-in command. Using `help` with no *command* prints the list of bash built-in commands.

The shortest list of bash built-in commands would probably include: `alias`, `bg`, `cd`, `echo`, `exit`, `export`, `fg`, `help`, `history`, `jobs`, `kill`, `logout`, `pwd`, `set`, `source`, `ulimit`, `umask`, `unalias`, `unset`.

`kdehelp`

(in X-terminal). Browse the whole system help using the graphical KDE help navigator. Normally, KDE help is invoked by pressing the appropriate icon on the KDE control panel. Use `gnome-help-browser` for the GNOME equivalent.

5.3 System info

`pwd`

Print working directory, i.e., display the name of my current directory on the screen.

`hostname`

Print the name of the local host (the machine on which you are working). Use `netconf` (as root) to change the name of the machine.

`whoami`

Print my login name.

`id username`

Print user id (uid) and his/her group id (gid), effective id (if different than the real id) and the supplementary groups.

`date`

Print or change the operating system date and time. E.g., I could change the date and time to 2000-12-31 23:57 using this command:

`date 123123572000`

To set the hardware (BIOS) clock from the system (Linux) clock, use the command (as root)

`setclock`

`time`

Determine the amount of time that it takes for a process to complete + other info. Don't confuse it with the `date` command (see previous entry). E.g. I can find out how long it takes to display a directory content using:

`time ls`

`who`

Determine the users logged on the machine.

`w`

Determine who is logged on the system, find out what they are doing, their processor usage, etc.

Handy security command.

`rwho -a`

(=remote who) Determine users logged on other computers on your network. The `rwho` service must be enabled for this command to run. If it isn't, run `setup` (RedHat specific) as root to enable "`rwho`".

`finger user_name`

System info about a user. Try: `finger root`

`last`

Show listing of users last logged-in on your system. Really good idea to check it from time to time as a security measure on your system.

`lastb`

("=last bad") Show the last bad (unsuccessful) login attempts on your system.

`history | more`

Show the last (1000 or so) commands executed from the command line on the current account. The "`| more`" causes the display to stop after each screenful. To see what another user was doing on your system, login as "`root`" and inspect his/her "`history`". The history is kept in the file `.bash_history` in the user home directory (so yes, it can be modified or erased).

`uptime`

Show the amount of time since the last reboot.

`ps`

("=print status" or "process status") List the processes currently run by the current user.

`ps aux | more`

List all the processes currently running, even those without the controlling terminal, together with the name of the user that owns each process.

`top`

Keep listing the currently running processes, sorted by cpu usage (top users first). Press `<Ctrl>c` when done.

`gtop`

`ktop`

(in X terminal) Two GUI choices for `top`. My favourite is `gtop` (comes with `gnome`). In KDE, `ktop` is also available from the "K" menu under "System"->"Task Manager".

`uname -a`

("= "Unix name" with option "all") Info on your (local) server. I can also use `guname` (in X-window terminal) to display the info more nicely.

```
cat /etc/issue
```

Check what distribution you are using. You can put your own message in this text file--it's displayed on login. It is more common to put your site-specific login message to the file `/etc/motd` ("`motd`"="message of the day").

```
free
```

Memory info (in kilobytes). "Shared" memory is the memory that can be shared between processes (e.g., executable code is "shared"). "Buffered" and "cached" memory is the part that keeps parts of recently accessed files--it can be shrunk if more memory is needed by processes.

```
df -h
```

(=disk free) Print disk info about all the filesystems (in human-readable form).

```
du / -bh | more
```

(=disk usage) Print detailed disk usage for each subdirectory starting at the "/" (root) directory (in human legible form).

```
cat /proc/cpuinfo
```

Cpu info--it show the content of the file `cpuinfo`. Note that the files in the `/proc` directory are not real files--they are hooks to look at information available to the kernel.

```
cat /proc/interrupts
```

List the interrupts in use. May need to find out before setting up new hardware.

```
cat /proc/version
```

Linux version and other info.

```
cat /proc/filesystems
```

Show the types of filesystems currently in use.

```
cat /etc/printcap | more
```

Show the setup of printers.

```
lsmod
```

(= "list modules". As root. Use `/sbin/lsmod` to execute this command when you are a non-root user.) Show the kernel modules currently loaded.

```
set | more
```

Show the current user environment.

```
echo $PATH
```

Show the content of the environment variable "PATH". This command can be used to show other environment variables as well. Use "set" to see the full environment.

```
dmesg | less
```

Print kernel messages (the content of the so-called kernel ring buffer). Press "q" to quit "less". Use `less /var/log/dmesg` to see what "dmesg" dumped into this file right after the last system bootup.

```
chage -l my_login_name
```

See my password expiry information.

`sysctl -a |more`

Display all the configurable Linux kernel parameters.

`stat filename`

Print general info about a file (the contents of the so-called inode).

`runlevel`

Print the previous and current runlevel. The output "N5" means: "no previous runlevel" and "5 is the current runlevel".

Runlevel is the mode of operation of Linux. Runlevel can be switched "on the fly" using the command `init`. For example `init 3` (as root) will switch me to runlevel 3. The following runlevels are standard:

0 - halt (Do NOT set `initdefault` to this)

1 - Single user mode

2 - Multiuser, without NFS (The same as 3, if you do not have networking)

3 - Full multiuser mode

4 - unused

5 - X11

6 - reboot (Do NOT set `initdefault` to this)

The system default runlevel is set in the file: `/etc/inittab`.

5.4 Basic operations

`ls`

`dir`

List the content of the current directory. The command `dir` is an alias to `ls` so this two commands do exactly the same thing. The file listing is normally color-coded: dark blue= directories, light grey = regular files, green = executable files, magenta = graphics files, red = compressed (zipped) files, light blue = symbolic links, yellow = device files, brown = FIFO ("First-In First-Out" named pipes).

`ls -al |more`

List the content of the current directory, all files (also those starting with a dot), and in a long form. Pipe the output through the "more" command, so that the display pauses after each screenful. The `ls` command has several very useful options. Some of these may have shortcuts (aliases) to avoid clumsy typing. Try `ll` ("long ls", an alias to `ls -l`). Another option I use quite often is `ls -ad` (list all the subdirectories in my current directory, but don't list their contents).

`cd directory`

Change directory. Using "`cd`" without the directory name will take you to your home directory. "`cd -`" will take you to your previous directory and is a convenient way to toggle between two directories. "`cd ..`" will take you one directory up (very useful).

`./program_name`

Run an executable in the current directory. The `./` are needed when the executable is not on my `PATH`. Executables which are you your `PATH` are simply run using: `program_name`

`shutdown -h now`

(as root) Shut down the system to a halt. Mostly used for a remote shutdown. Use `<Ctrl><Alt>` for a shutdown at the console (which can be done by any user).

```
halt
reboot
init 6
```

(as root, three commands) Halt or reboot the machine. Used for remote shutdown, simpler to type than the previous command. Also great if the computer "hangs" (I lose control over the keyboard)--I telnet to it from another machine on the network and remotely reboot it. I use <Ctrl><Alt> for normal shutdown at the console of a local computer.

```
vlock
```

(Not present on older versions of RedHat.) Lock a local (text mode) terminal. I can also use `vlock -a` to lock all terminals (probably not a good idea). The best is probably to log out. You don't use `vlock` in GUI--the windows managers come with password-protected screensaver and a locking utility (e.g., the small icon with padlock in KDE).

5.5 File management

```
cp source destination
```

Copy files. E.g., `cp /home/stan/existing_file_name .` will copy a file to my current working directory. Use the "-r" option (for recursive) to copy the contents of whole directories, e.g., `cp -r my_existing/dir/ ~` will copy a subdirectory under my current working directory to my home directory.

```
mcopy source destination
```

Copy a file from/to a DOS filesystem (no mounting of the filesystem is necessary). E.g., `mcopy a:\autoexec.bat ~/junk`. See `man mtools` for other commands that can access DOS files without mounting: `mdir`, `mcd`, `mren`, `mmove`, `mdel`, `mmd`, `mrd`, `mformat` We don't use the `mtool` commands that often--operations on DOS/MS Windows files can be performed using regular Linux commands after you mount the DOS/MS Windows filesystem.

```
mv source destination
```

Move or rename files. The same command is used for moving and renaming files and directories.

```
rename string replacement_string filename
```

Flexible utility for changing parts of filenames. For example:

```
rename .htm .html *.htm
```

```
ln source destination
```

Create a hard link called *destination* to the file called *source*. The link appears as a copy of the original files, but in reality only one copy of the file is kept, just two (or more) directory entries point to it. Any changes to the file are automatically visible throughout. When one directory entry is removed, the other(s) stay(s) intact. The limitation of the hard links are: the files have to be on the same filesystem, hard links to directories or special files are impossible.

```
ln -s source destination
```

Create a symbolic (soft) link called "destination" to the file called "source". The symbolic link just specifies a path where to look for the "real" file. In contradistinction to hard links, the source and destination don't have to be on the same filesystem. In comparison to hard links, the drawback of symbolic links are: if the original file is removed, the link is "broken"--it points to nowhere; symbolic links can also create circular references (like circular references in spreadsheets or databases, e.g., "a" points to "b" and "b" points back to "a"). In short, symbolic links are a great tool and are very

often used, but they can create an extra level of complexity.

`rm files`

Remove (delete) files. You must own the file in order to be able to remove it (or be "root"). On many systems, you will be asked for confirmation of deletion, if you don't want this, use the "-f" (=force) option, e.g., `rm -f *` will remove all files in my current working directory, no questions asked.

`mkdir directory`

Make a new directory.

`rmdir directory`

Remove an empty directory.

`rm -r files`

(recursive remove) Remove files, directories, and their subdirectories. Careful with this command as root--you can easily remove all files on the system with such a command executed on the top of your directory tree, and there is no undelete in Linux (yet). But if you really wanted to do it (reconsider), here is how (as root):

`rm -rf /*`

`rm -rf files`

(recursive force remove). As above, but skip the prompt for confirmation, if one is set on your system. Careful with this command particularly as root--see the command above.

`mc`

Launch the "Midnight Commander" file manager (looks like "Norton Commander" for Linux).

`konqueror &`

Launch the KDE file manager. Perhaps this is the ultimate for file management. Much better than the MS "Windows Explorer".

5.6 Viewing and editing files

`cat filename | more`

View the content of a text file called "filename", one page at a time. The "|" is the "pipe" symbol (on many American keyboards it shares the key with "\"). `more` makes the output stop after each screenful. For long files, it is sometimes convenient to use the commands `head` and `tail` that display just the beginning and the end of the file. If you happened to use `cat` a binary file and your terminal displays funny characters afterwards, you can restore it with the command `reset`.

`cat filename | less`

`less filename`

(two commands, use either) Scroll through a content of a text file. Press q when done. "less" is roughly equivalent to "more", the command you know from DOS, but often "less" is more convenient than "more".

`head filename`

Print first 10 lines of the (long) text file.

`tail filename`

Print last 10 lines of a long or growing text file. Use `tail -f filename` for tail to follow the file as it grows--really handy for continuing inspection of log files.

`pico filename`

Edit a text file using the simple and standard text editor called `pico`. Use `<Ctrl>x` to exit. There are many text editors for Linux, including several GUI-based.

`pico -w filename`

Edit a text file, while disabling the long line wrap. Handy for editing configuration files, e.g. `/etc/fstab`.

`kwrite`

(in X terminal) Very nice, "advanced text editor". Supports veritcal text selection!

`kedit`

`gedit`

(in X terminal). Simple yet nice text editors (GUI based).

`gxedit`

(in X terminal) Another multi-purpose, feature packed text editor. This one even has timed backup.

`latte`

(in X termina) "Code" editor, i.e., plain text meant for writing programs.

`nedit`

(in X terminal) Another programmer editor. Very nice and loaded.

`bluefish`

(in X terminal) html editor (source with syntax highlighting and maaaany tools and options).

`ispell filename`

Spell check an ASCII text file. AbiWord, WordPerfect, StarOffice and other word processors come with "as-you-type" spellchecking, so you really don't have to worry about the simple `ispell` unless you need it. Newer Linux distributions (e.g., RH7.0) contain an improved spellchecking module called `aspell`, yet the above command will still works.

`touch filename`

Change the date/time stamp of the file `filename` to the current time. Create an empty file if the file does not exist.

5.7 Finding files

`find / -name "filename"`

Find the file called "filename" on your filesystem starting the search from the root directory `/`. The "filename" may contain wildcards (`*`, `?`).

`locate filename`

Find the file name of which contains the string "filename". Easier and faster than the previous command but depends on a database that normally rebuilds at night.

`which executable_name`

Show me the full path to the executable that would run if I just typed its name on the command line.

For example, this command:

`which netscape`

on my system produces:

`/usr/bin/netscape`

`whereis command`

Print the locations for the binary, source, and manual page files of the command "*command*".

`rgrep -r 'celeste' . |more`

Search all files in the current directory and all its subdirectories (the option "-r" stands for "recursive") for the string "celeste". Print the filename and the line in the file that contains the searched string.

`kfind &`

(in X terminal). A GUI front-end to `find` and `grep`. Very nice. The `&` at the end of the command makes `kfind` run in the background so that the console remains available.

5.8 Basics of X-windows

`xinit`

Start a barebone X-windows server (without a windows manager).

`startx`

Start an X-windows server and the default windows manager. Works like typing "win" under DOS with Win3.1

`startx -- :1`

Start another X-windows session on the display 1 (the default is opened on display 0). You can have several GUI terminals running concurrently. Switch between them using `<Ctrl><Alt><F7>`, `<Ctrl><Alt><F8>`, etc.

`xterm`

(in X terminal) Run a simple X-windows terminal. Typing `exit` will close it. There are other, more advanced "virtual" terminals for X-windows. I like the popular ones: `konsole` and `kvt` (both come with `kde`) and `gnome-terminal` (comes with `gnome`). If you need something really fancy-looking, try `Eterm`.

`startkde`

`gnome-session`

`xfce`

`afterstep`

`AnotherLevel`

`fvwm2`

`fvwm`

(in X terminal, 7 different commands, use the one which starts your fav windows manager) Start your favourite windows manager in an X terminal on bare X server.

5.9 Network apps

`netscape`

(in X terminal) Run netscape (requires a separate Netscape installation). The current versions of Netscape (4.x) are known to be big and buggy. They occasionally crash by vanishing (no other harm done). Also, when not connected to the network, Netscape likes to refuse to do anything (looks like it hanged)-it revives when you connect.

`netscape -display host:0.0`

(in X terminal) Run netscape on the current machine and direct the output to machine named "host" display 0 screen 0. Your current machine must have a permission to display on the machine "host" (typically given by executing the command `xhost current_machine_name` in the xterminal of the machine host. Other X-windows program can be run remotely the same way.

`lynx file.html`

View an html file or browse the net from the text mode. Although lynx's look or convenience of use is not as great as GUI-based broser, it is light-weight, almost always works, and does not require any configuration, as long as your networks is functional.

`pine`

A good text-mode mail reader. Another good and standard one is `elm`. Your Netscape mail will read the mail from your Internet account. `pine` will let you read the "local" mail, e.g. the mail your son or a cron process sends to you from a computer on your home network. The command `mail` could also be used for reading/composing mail, but it would be inconvenient--it is meant to be used in scripts for automation.

`elm`

A good tex-mode mail reader. See the previous command.

`mutt`

A really basic but extremally useful and fast mail reader.

`mail`

A basic operating system tool for e-mail. Look at the previous commands for a better e-mail reader. `mail` is good if you wanted to send an e-mail from a shell script.

`kmail`

(in X-terminal) Nice, GUI mail program.

`licq`

(in X terminal) An icq "instant messaging" client. Another good one is `kxicq`. Older distributions don't have an icq client installed, you have to do download one and install it.

`knode`

(in X terminal) Start my favourite newsgroup (usenet) reader. It is MUCH better than the netscape's built-in reader.

`talk username1`

Talk to another user currently logged on your machine (or use "`talk username1@machinename`" to talk to a user on a different computer). To accept the invitation to the conversation, type the command "`talk username2`". If somebody is trying to talk to you and it disrupts your work, you may use the command "`mesg n`" to refuse accepting messages. You may

want to use "who" or "rwho" to determine the users who are currently logged-in.

telnet server

Connect to another machine using the TELNET protocol. Use a remote machine name or IP address. You will be prompted for your login name and password--you must have an account on the remote machine to login. Telnet will connect you to another machine and let you operate on it as if you were sitting at its keyboard (almost). Telnet is not very secure--everything you type goes in open text, even your password!

rlogin server

(=remote login) Connect to another machine. The login name/password from your current session is used; if it fails you are prompted for a password.

rsh server

(=remote shell) Yet another way to connect to a remote machine. The login name/password from your current session is used; if it fails you are prompted for a password.

ssh server

(=secure shell) Connect to a server (remote login) using a secure connection. ssh is secure because it checks identity of the person connecting, and encrypts all the data transferred over the network using your pair of RSA "public-private" keys.

Both the client and the server must have ssh service (daemon) running. This is normally available on newer Linux distributions (e.g., RH7.0). Before using ssh, some setup is necessary. The user creates his/her RSA key pair by running the command `ssh-keygen`. This stores the private key in the file `$HOME/.ssh/identity` and the public key in `$HOME/.ssh/identity.pub` in the user's home directory. The user should then copy the `identity.pub` to `$HOME/.ssh/authorized_keys` in his/her home directory on the remote machine. After this, the user can log in without giving the password. The most convenient way to use RSA authentication may be with an authentication agent. See `man 1 ssh-agent` for more information. If automatic authentication methods fail, ssh prompts the user for a password. The password is sent to the remote host for checking; however, since all communications are encrypted, the password cannot be seen by someone listening on the network.

ftp server

Ftp another machine. (There is also `ncftp` which adds extra features and `gftp` for GUI.) Ftp is good for copying files to/from a remote machine. Try user "anonymous" if you don't have an account on the remote server. After connection, use "?" to see the list of available ftp commands. The essential ftp command are: `ls` (see the files on the remote system), `ASCII`, `binary` (set the file transfer mode to either text or binary, important that you select the proper one), `get` (copy a file from the remote system to the local system), `mget` (get many files at once), `put` (copy a file from the local system to the remote system), `mput` (put many files at once), `bye` (disconnect). For automation in a script, you may want to use `ncftpput` and `ncftpget`, for example:

```
ncftpput -u my_user_name -p my_password -a remote.host.domain
remote_dir *local.html
```

For keeping mirrors of ftp directories, one can use `fmirror`

```
wget -m http://sunsite.dk/linux-newbie
```

Copy files from web sites. The example above uses the option `-m` (=mirror) to retrieve a complete set of files from the master site of this guide.

`minicom`

Minicom program (looks like "Procomm for Linux").

`rx`

Receive files using the Zmodem, Ymodem, or Xmodem protocol. Xmodem requires a filename. Use

`rx --help` for more info.

5.10 File (de)compression

`tar -zxvf filename.tar.gz`

(=tape archiver) Untar a tarred and compressed tarball (*.tar.gz or *.tgz) that you downloaded from the Internet.

`tar -xvf filename.tar`

Untar a tarred but uncompressed tarball (*.tar).

`tar czvpf /var/backups/mybackup.tar.gz /home`

`cd /; tar xzvpf /var/backups/mybackup.tar.gz '*/myfile.rtf'`

(as root) Create a backup of /home to a compressed file. The second command shows how to restore a file from the backup.

`gunzip filename.gz`

Decompress a zipped file (*.gz" or *.z). Use `gzip` (also `zip` or `compress`) if you wanted to compress files to this file format.

`zcat filename.gz | more`

(=zip cat) Display the contents of a compress file.

`bunzip2 filename.bz2`

(=big unzip) Decompress a file (*.bz2) zipped with `bzip2` compression utility. Used for big files.

`unzip filename.zip`

Decompress a file (*.zip) zipped with a compression utility compatible with PKZIP for DOS.

`unarj e filename.arj`

Extract the content of an *.arj archive.

`uudecode -o outfile filename`

Decode a file encoded with `uuencode`. uu-encoded files are typically used for transfer of non-text files in e-mail (`uuencode` transforms any file into an ASCII file).

`cat filename | mimecode -o filename.mime`

`cat filename.mime |mimecode -u -o filename`

(2 commands.) Encode and then decode back a file to/from the mail-oriented Internet standard for 7-bit data transfer called "mime". On older distributions, the command that does the work (`mimecode`) is called `mmencode`. Usually, you don't have to bother with these commands, your mailer should do the mime encoding/decoding in a transparent way.

5.11 Process control

`ps`

(="print status" or "process status") Display the list of currently running processes with their process IDs (PID) numbers. Use `ps aux` to see all processes currently running on your system (also those of other users or without a controlling terminal), each with the name of the owner. Use "top" to keep listing the processes currently running.

`any_command &`

Run any command in the background (the symbol "&" means "run the proceeding command in the background"). The *job_number* is printed on the screen so you can bring the command in the foreground (see below) if you want.

`jobs`

List my background or stopped processes and show their job numbers.

`fg job_number`

Bring a background or stopped process to the foreground.

`bg job_number`

Place a process in the background, so it is exactly as if it had been started with &. This will restart a stopped background process. Current foreground process can often be stopped with <Ctrl>z. If you have stopped or background jobs, you have to type `exit` twice in row to log out.

`batch any_command`

Run any command (usually one that is going to take more time) when the system load is low. I can logout, and the process will keep running.

`at 17:00`

Execute a command at a specified time. You will be prompted for the command(s) to run, until you press <Ctrl>d. The associated commands are `atq` (display the queue of processes started with `at`) and `atrm` (remove a process from the "at queue").

`kill PID`

Force a process shutdown. First determine the PID of the process to kill using `ps`.

`killall program_name`

Kill program(s) by name.

`xkill`

(in X terminal) Kill a GUI-based program with mouse. (Point with your mouse cursor at the window of the process you want to kill and click.)

`kpm`

(in X terminal) KDE process manager.

`lpc`

(as root) Check and control the printer(s). Type "?" to see the list of available commands.

`lpq`

Show the content of the printer queue. Under KDE (X-Windows), you may use GUI-based "Printer Queue" available from "K"menu-Utilities.

`lprm job_number`

Remove a printing job "job_number" from the queue.

`nice program_name`

Run *program_name* adjusting its priority. Since the priority is not specified in this example, it will be increased by 10 (the process will run slower), from the default value (usually 0). The lower the number (of "niceness" to other users on the system), the higher the priority. The priority value may be in the range -20 to 19. Only root may specify negative values. Use `top` to display the priorities of the running processes.

`renice -18 PID`

(as root) Change the priority of a running process to minus 18. Normal users can only adjust processes they own, and only up from the current value (make them run slower). One could also `renice +10 -u peter` to make user peter use fewer cpu clicks so that other user don't suffer when he runs his computing-intensive tasks.

`<Ctrl>c`, `<Ctrl>z`, `<Ctrl>s`, and `<Ctrl>q` also belong to this chapter but they were described previously [p 123]. In short they mean: stop the current command, send the current command to the background, stop the data transfer, resume the data transfer.

`lsolf`

List the opened files. If I am a root, all files will be listed. I can limit myself to files opened by processes owned by the second console if I use `lsolf /dev/tty1`.

5.12 Basic administration commands

`printtool`

(as root in X-terminal) Configuration tool for your printer(s). Settings go to the file `/etc/printcap` and (strangely) `/var/spool/lpd`.

`setup`

(as root) Configure mouse, soundcard, keyboard, X-windows, and system services. There are many distribution-specific configuration utilities, `setup` is the default on RedHat. Mandrake 7.0 offers very nice `DrakConf`.

`linuxconfig`

(as root, either in text mode or in the X terminal). You can access and change hundreds of network setting from here. Very powerful--don't change too many things at the same time, and be careful with changing entries you don't understand. RedHats network configuration utility `netconf` is a subset of `linuxconfig`, therefore it is simpler and sometimes easier to use.

`timeconfig`

(as root) Set the timezone for your system.

`setclock`

(as root). Set your computer hardware clock from the current linux system time. Use the command "date" first to set up the linux system time. E.g., I could change the date and time to 2000-12-31 23:57 using this command:

`date 123123572000`

and then write the time to the hardware clock using:

`setclock`

`xvidtune`

(in X-terminal). Adjust the settings of the graphical display for all resolutions so as to eliminate black bands, shift the display right/left/up/down, etc. (First use the knobs on your monitor to fit your text mode correctly on the screen.) Then use `xvidtune` to adjust the monitor frequencies for each resolution so it fits well in your screen. To make the changes permanent, display the frequencies on the screen and then transfer them to the setup file `/etc/X11/XF86Config`.

`alias ls="ls --color=tty"`

Create an alias for the command "ls" to enhance its format with color. In this example, the alias is also called "ls" and the "color" option is only invoked when the output is done to a terminal (not to files). Put the alias into the file `/etc/bashrc` if you would like the alias to be always accessible to all users on the system. Aliases are a handy way to customize your system. Type "alias" alone to see the list of aliases on your system. Use `unalias alias_name` to remove an alias.

`adduser user_name`

`useradd user_name`

(Two commands doing the same. Use either.) Create a new account (you must be root). E.g., `adduser barbara` Don't forget to set up the password for the new user in the next step. The user home directory is `/home/user_name`.

`userdel user_name`

Remove an account (you must be a root). The user's home directory and the undelivered mail must be dealt with separately (manually because you have to decide what to do with the files).

`groupadd group_name`

(as root) Create a new group on your system. Non-essential on a home machine, but can be very handy even on a home machine with a small number of users.

For example, I could create a group "friends", then edit the file `/etc/group`, and add my login name and the names of my friends to the line that lists the group, so that the final line might look like this:

`friends:x:502:stan,pete,marie`

Then, I can change the permissions on a selected file so that the file belongs to me AND the group "friends".

`chgrp friends my_file`

Thus, the listed members of this group have special access to these files that the rest of the world might not have, for example read and write permissions:

`chmod g=rw,o= my_file`

The alternative would be to give write permission to everybody, which is definitely unsafe even on a home computer.

`groups`

List the groups to which the current user belongs.

`userconf`

(as root) Menu-driven user configuration tools (password policy, group modification, adding users, etc). Part of `linuxconf` package, but can be run separately.

`passwd`

Change the password on your current account. If you are root, you can change the password for any user using: `passwd user_name`

`chfn`

(="change full name"). Change the information about you (full name, office number, phone number, etc). This information is displayed when the `finger` command is run on your *login_name*.

`chage -M 100 login_name`

(="change age"). Set the password expiry to 100 days for the user named *login_name*.

`chmod perm filename`

(=change mode) Change the file access permission for the files you own (unless you are root in which case you can change any file). You can make a file accessible in three modes: read (r), write (w), execute (x) to three classes of users: owner (u), members of the group which owns the file (g), others on the system (o). Check the current access permissions using:

`ls -l filename`

If the file is accessible to all users in all modes it will show:

`rw-rw-rw-`

The first triplet shows the file permission for the owner of the file, the second for the group that owns the file, and the third for others ("the rest of the world"). A "no" permission is shown as "-".

When setting permissions, these symbols are used: "u"(=user or owner of the file), "g"(=group that owns the file), "o"(=others), "a" (=all, i.e., owner, group and others), "="(=set the permission to), "+"(=add the permission), "-"(=take away the permission), "r"(=permission to read the file), "w"(=write permission, meaning the permission to modify the file), "x"(=permission to execute the file).

For example, this command will **add** the permission to read the file `junk` to all (=user+group+others):

`chmod a+r junk`

This command will remove the permission to execute the file `junk` from others:

`chmod o-x junk`

Also try here for more info.

You can set the default file permissions for the news files that you create using the command `umask` (see `man umask`).

`chown new_ownershipname filename`

`chgrp new_groupname filename`

Change the file owner and group. You should use these two commands after you copy a file for use by somebody else. Only the owner of a file can delete it.

`lsattr files`

List attributes for the file(s). Not very often used because the most interesting attributes are still not implemented. The attributes can be changed using the `chattr` command. The attributes are: A (=don't update atime when the file is modified), S (=synchronous updates), a (=append only possible to this file), c (=file compressed on the kernel level, not implemented yet), i (=immutable file), d (=no dump), s (=secure deletion), and u (undeletable, not implemented yet). An interesting usage may be to make a file undeletable even by root (until s/he clears the attribute).

`su`

(=substitute user id) Assume the superuser (=root) identity (you will be prompted for the password). Type "exit" to return you to your previous login. Don't habitually work on your machine as root. The root account is for administration and the `su` command is to ease your access to the administration account when you require it. You can also use "su" to assume any other user identity, e.g. `su barbara` will make me "barbara" (password required unless I am the superuser).

```
cat /var/log/httpd/access_log
```

Show who connected to your http (apache) server since the last time the log file was "rotated" (normally rotated once a day, when cron runs). The previous log file is `access_log.1`, the yet previous `access_log.2`, etc.

```
cat /var/log/secure
```

(as root) Inspect the important system log. It is really a good idea to do it from time to time if you use Internet access.

```
ftpwho
```

(as root) Determine who is currently connected to your ftp server.

```
SVGATextMode 80x25x9
```

```
SVGATextMode 80x29x9
```

(as root) Change the text resolution in the text terminal. In the above example (second line) I changed the text screen to 80 columns x 29 lines with characters 9 pixels high. The first line defines a resolution that always works, so that if the second command did not work on my system, I can press <ArrowUP> twice and <Enter> to regain control over my screen. The possible modes depend on your video card and your monitor synchronization frequencies--I needed to edit (as root) the file `/etc/TextConfig` and (un)comment the proper lines to let `SVGATextMode` know what my system supports.

```
kernelcfg
```

(as root in X terminal). GUI to to add/remove kernel modules. Module is like a device driver--a piece of Linux kernel that provides support for a particular piece of hardware or functionality. You can do the same from the command line using the command `insmod`.

```
lsmod
```

(= list modules). List currently loaded kernel modules. A module is like a device driver--it provides operating system kernel support for a particular piece of hardware or feature.

```
modprobe -l |more
```

List all the modules available for your kernel. The available modules are determined by how your Linux kernel was compiled. Every possible module/feature can be compiled on linux as either "hard wired" (fast, non-removable), "module" (maybe slower, but loaded/removable on demand), or "no" (no support for this feature at all). The modules with which your kernel supports (with which it was compiled) are all as files under the directory `/lib/modules` so browsing it may give you a clue if you are lost.

```
modprobe sb
```

Load the soundblaster (sb) module.

```
insmod parport
```

```
insmod ppa
```

(as root) Insert modules into the kernel (a module is roughly an equivalent of a DOS device driver). This example shows how to insert the modules for support of the external parallel port zip drive (it appears to be a problem to get the external zip drive to work in any other way under RH6.0 and 6.1).

```
rmmod module_name
```

(as root, not essential). Remove the module *module_name* from the kernel.

```
depmod -a
```

(as root) Build the module dependency table for the kernel. Not essential unless you modified `/etc/modules` and don't wish to reboot.

```
setserial /dev/cua0 port 0x03f8 irq 4
```

(as root) Set a serial port to a non-standard setting. The example here shows the standard setting for the first serial port (`cua0` or `ttyS0`). The standard PC settings for the second serial port (`cua1` or `ttyS1`) are: address of i/o port `0x02f8`, irq 3. The third serial port (`cua2` or `ttyS2`): `0x03e8`, irq 4. The forth serial port (`cua3` or `ttyS3`): `0x02e8`, irq 3. Add your setting to `/etc/rc.d/rc.local` if you want it to be set at the boot time. See `man setserial` for good a overview.

```
tunelp
```

(as root, rarely needed) Tune up your parallel ports.

```
fdisk /dev/hda
```

(= "fixed disk". As root.) Linux hard drive partitioning utility (DOS has a utility with the same name). In the example above, I specified that I would like to partition the first harddrive on the first IDE interface, hence `hda`. If I were you, i would backup my data before using `fdisk` on any partition. I do not not know anybody who likes `fdisk` (either Linux or DOS edition)--I prefer easier to use `cfdisk`, see next command.

```
cfdisk /dev/hda
```

(as root) Hard drive partitioning utility, menu-based. Easier to use then the plain-vanilla `fdisk` (see ther previous command).

```
cd /usr/src/linux-2.2.14
```

```
make xconfig
```

(as root in X terminal). Nice GUI front-end for configuration of the kernel options in preparation for compilation of your customized kernel. (The directory name contains the version of your Linux kernel so you may need to modify the directory name if your Linux kernel version is different than 2.2.14 used in this example. Also, you need the "Tk" interpreter and the kernel source code installed.) The alternatives to "make xconfig" are: "make config" (runs a scripts that asks you questions in the text mode) and "make menuconfig" (runs a text-based menu-driven configuration utility). Try: `less /usr/doc/HOWTO/Kernel-HOWTO` for more information.

After configuring the options for the new kernel, you may choose to proceed with kernel compilation of the new kernel by issuing the following commands:

```
make dep
```

```
make bzImage
```

The last command will take some time to complete (maybe 0.5 h or 2 h, depending on your hardware). It produces the file "bzImage", which is your new Linux kernel. Next:

```
make modules
```

```
make modules_install
```

Now you the new kernel and modules so you can install them--installation involves copying the new files into the `/boot` directory and making changes to `/etc/lilo.conf` so you can select at the boot time which kernel (old or new) to boot. See this for details on kernel upgrade. It may also be helpful to read: `/usr/doc/HOWTO/Kernel-HOWTO` and perhaps `man depmod`. Configuration, compilation and installation of a new kernel is not difficult at all but it CAN lead to problems if you don't know what you are doing (tough luck, if you cannot boot and are really unable to fix things after booting from the floppy, you may need to re-install). Compilation of a kernel is also a good way to test your hardware, because it involves massive amount of computing. If your hardware is "flaky", you will most likely receive the "signal 11" error (read the beatiful

```
/usr/doc/FAQ/txt/GCC-SIG11-FAQ).
```

```
ldconfig
```

(as root) Re-create the bindings and the cache for the loader of dynamic libraries ("ld"). You may want to run `ldconfig` after an installation of new dynamically linked libraries on your system. (It is also re-run every time you boot the computer, so if you reboot you don't have to run it manually.)

```
mknod /dev/fd0 b 2 0
```

(=make node, as root) Manually create a device file. This example shows how to create a device file associated with your first floppy drive and could be useful if you happened to accidentally erase it. The options are: b=block mode device, c=character mode device, p=FIFO device, u=unbuffered character mode device. The two integers specify the major and the minor device number. I normally wouldn't know the parameters which `mknod` requires. So to make devices, I first read `man MAKEDEV` to figure the name of the device and then run the script `/dev/MAKEDEV` which knows about Linux devices by their names--see the next command. If the mentioned manual page does not help, I may refer to the ultimate documentation included with the kernel source code:

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

```
cd /dev
```

```
./MAKEDEV audio
```

(as root). Restore the "audio" device that I just somehow screwed up. Also see the previous command.

```
fdformat /dev/fd0H1440
```

```
mkfs -c -t ext2
```

(=floppy disk format, two commands, as root) Perform a low-level formatting of a floppy in the first floppy drive (`/dev/fd0`), high density (1440 kB). Then make a Linux filesystem (`-t ext2`), checking/marking bad blocks (`-c`). Making the filesystem is an equivalent to the high-level formatting.

```
badblocks /dev/fd01440 1440
```

(as root) Check a high-density floppy for bad blocks and display the results on the screen. The parameter "1440" specifies that 1440 blocks are to be checked. This command does not modify the floppy.

```
fsck -t ext2 /dev/hda2
```

(=file system check, as root) Check and repair a filesystem, e.g., after an "unclean" shutdown due to a power failure. The above example performs the check on the partition `hda2`, filesystem type `ext2`. You definitely want to boot Linux in the "single mode" to perform this (type "linux single" at the LILO prompt or use `init 1` as root to enter the single user mode). If errors are found during the filesystem checkup, I accept the defaults for repair.

```
dd if=/dev/fd0H1440 of=floppy_image
```

```
dd if=floppy_image of=/dev/fd0H1440
```

(two commands, `dd`="data duplicator") Create an image of a floppy to the file called "floppy_image" in the current directory. Then copy `floppy_image` (file) to another floppy disk. Works like DOS "DISKCOPY".

```
/sbin/chkconfig
```

A tool to check/enable/disable system services under different runlevels. Typically, I just use RedHat `setup` utility if I need to enable/disable a service, but `chkconfig` does give me extra flexibility should I ever need it.

SuperProbe

(as root). A utility to determine the type of the video card and the amount of its memory.

`symlinks -r -cds /`

(as root) Check and fix the symbolic links on my system. Start from / and progress through all the subdirectories (option `-r`="recurse") and change absolute/messy links to relative, delete dangling links, and shorten lengthy links (options `-cds`). If my filesystem spreads over different hard drive partitions, I need to re-run this command for each of them (e.g., `symlinks -r -cds /usr`).

5.13 Program installation

`rpm -ivh package_name-version.platform.rpm`

(as root) Install a package (option `"i"`, must be the first letter after the dash), while talking to me a lot (option `"v"`=verbose) and printing "hashes" to show installation progress (option `"h"`). rpm stands for "Redhat Package Manager".

`rpm -Uvh package_name-version.platform.rpm`

(as root) Upgrade (option `"U"`, must be the first letter after the dash) a package, while being verbose (option `"v"`) and displaying hashes (`"h"`).

`rpm -ivh --force --nodep package_name-version.platform.rpm`

(as root) Install the package ignoring any possible conflicts and package dependency problems.

`rpm -e package_name`

(as root) Uninstall (option `"e"`=erase) the package `package_name`. Please note the absence of `"-version.platform.rpm"` at the end of the package name (the package name is the same as the name of the `*.rpm` file from which the package was installed but without the dash, version, platform and `"rpm"`).

`rpm -qpi package_name-version.platform.rpm`

Query (option `"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.

`rpm -qpl package_name-version.platform.rpm`

Query (option `"q"`, must be the first letter after the dash) the yet uninstalled package (option `"p"`) so that it displays the listing (option `"l"`) of all the files the package contains.

`rpm -qf a_file`

Find the name of the installed package to which the file `"a_file"` belongs or belonged. Useful if I accidentally erased a file and now I need to find the right package and re-install it.

`rpm -qi package_name`

Query the already installed package so that it displays the info about itself. Please note the absence of `"-version.platform.rpm"` at the end of the package name.

`rpm -qai | more`

Query all the packages installed on my system so that they display their info. On my simple system, I have ~600 packages installed so obviously, I must have a lot of time to read all their info. To count your packages, try: `rpm -qa | grep -c ''`

```
rpm -Va
```

Verify (the option "V") all the packages (option "a") installed on my system. This lists files that were modified since the installation. Here is the legend for the output:

```
.      Test passed
c      This is a configuration file
5      MD5 checksum failed
S      File size is different
L      Symbolic link has changed
T      File modification time changed
D      Device file is modified
U      User that owns the file has changed
G      Group that owns the file has changed
M      File mode (permissions and/or file type) has been modified
```

kpackage

gnorpm

glint

(in X terminal, as root if you want to be able to install packages) GUI fronts to the Red Hat Package Manager (rpm). "glint" comes with RH5.2 and seems obsolete now. gnorpm is the "official" RedHat GUI package installer, older versions are very slow and confusing but the newer version (the one that comes with RH7.0) is vastly improved. kpackage is the "official" KDE program and has been pretty good all along. Use any of them to view which software packages are installed on your system and the what not-yet-installed packages are available on your RedHat CD, display the info about the packages, and install them if you want (installation must be done as root).

5.14 Accessing drives/partitions

mount

See here for details on mounting drives. Examples are shown in the next commands.

```
mount -t auto /dev/fd0 /mnt/floppy
```

(as root) Mount the floppy. The directory /mnt/floppy must exist, be empty and NOT be your current directory.

```
mount -t auto /dev/cdrom /mnt/cdrom
```

(as root) Mount the CD. You may need to create/modify the /dev/cdrom file depending where your CDROM is. The directory /mnt/cdrom must exist, be empty and NOT be your current directory.

```
mount /mnt/floppy
```

(as user or root) Mount a floppy as user. The file /etc/fstab must be set up to do this. The directory /mnt/floppy must not be your current directory.

```
mount /mnt/cdrom
```

(as user or root) Mount a CD as user. The file /etc/fstab must be set up to do this. The directory /mnt/cdrom must not be your current directory.

```
umount /mnt/floppy
```

Unmount the floppy. The directory /mnt/floppy must not be your (or anybody else's) current working directory. Depending on your setup, you might not be able to unmount a drive that was mount by somebody else.

5.15 Network administration tools

`netconf`

(as root) A very good menu-driven setup of your network.

`ping machine_name`

Check if you can contact another machine (give the machine's name or IP), press <Ctrl>C when done (without <Ctrl>c, the command keeps going). As all Linux commands, `ping` has options, including the "ping of death" attack, when it seems you can ping some servers so they die--try the the options -f and -s.

`route -n`

Show the kernel routing table.

`nslookup host_to_find`

Query your default domain name server (DNS) for an Internet name (or IP number) *host_to_find*. This way you can check if your DNS works. You can also find out the name of the host of which you only know the IP number.

`traceroute host_to_trace`

Have a look how your messages travel to *host_to_trace* (which is either a host name or IP number).

`mtr host_to_trace`

(as root) A powerful and nice tool that combines the functionality of the older `ping` and `traceroute` (RH7.0)

`nmblookup -A ip_address`

Status of a networked MS Windows machine (with an NetBIOS name). This command is an equivalent of Windows `nbtstat` command.

`ipfwadm -F -p m`

(for RH5.2, see the next command for RH6.0) Set up the firewall IP forwarding policy to masquerading. (Not very secure but simple.) Purpose: all computers from your home network will appear to the outside world as one very busy machine and, for example, you will be allowed to browse the Internet from all computers at once.

`echo 1 > /proc/sys/net/ipv4/ip_forward`

`ipfwadm-wrapper -F -p deny`

`ipfwadm-wrapper -F -a m -S xxx.xxx.xxx.0/24 -D 0.0.0.0/0`

(three commands, RH6.0). Does the same as the previous command. Substitute the "x"s with digits of your class "C" IP address that you assigned to your home network. See here for more details.

`ipchains -P forward DENY`

`ipchains -A forward -s xxx.xxx.xxx.0/24 -d 0.0.0.0/0 -j MASQ`

(two commands, RH7.0). Same as previous commands, but works under RH7.0.

`ipchains -L`

List all firewall rules. Use to check if your firewalling works.

`ifconfig`

(as root) Display info on the network interfaces currently active (ethernet, ppp, etc). Your first ethernet should show up as eth0, second as eth1, etc, first ppp over modem as ppp0, second as ppp1, etc. The "lo" is the "loopback only" interface which should be always active. Use the options (see `ifconfig --help`) to configure the interfaces.

`ifup interface_name`

(`/sbin/ifup` to run as a user) Startup a network interface. E.g.:

`ifup eth0`

`ifup ppp0`

`ifup ppp1`

Users can start up or shutdown the ppp interface only when the permission is given in the ppp setup (using `netconf`). To start a ppp interface (dial-up connection), I normally use `kppp` available under the KDE "K" menu (or by typing `kppp` in an X-terminal).

`ifdown interface_name`

(`/sbin/ifdown` to run it as a user). Shut down the network interface. E.g.: `ifdown ppp0` Also, see the previous command.

`netstat | more`

Displays a lot (too much?) information on the status of your network.

`/usr/sbin/mtr --gtk`

(as root, in X windows if you wish the nice gtk-based interface). Network diagnostic tool combining the capabilities of traceroute and ping. Comes with RH7.0.

`nmap ip_number`

Map the ports on the machine with `ip_number`. REALLY useful to establish the security of your network configuration as you can see the opened ports. `nmap` is included on the RH7.0 "Linux PowerTools" CD, as is a convenient GUI front end, "`nmapfe`". `nmap` can also do operating system "fingerprinting". Normally, people don't like their computer ports being scanned by `nmap` so they may complain.

`ethereal`

(as root, in Xterminal) Network analyzer--view the network traffic going through your computer. Included on the RH7.0 "Linux PowerTools" CD. Using `ethereal` may be unethical in some situations, and unauthorized use at workplace would probably be a fireable offence.

5.16 Music-related commands

`cdplay play 1`

Play the first track from a audio CD. Use `cdplay` to play the whole CD. Use `cdplay stop` when had enough.

`eject`

Get a free coffee cup holder :))). (Eject the CD ROM tray). This command defaults to the `cdrom`, but could be used to eject other removable media by specifying the mount point or device. E.g., I can eject the zipdisk from the zipdrive (as root) using: `eject /dev/sda4`


```
play my_file.wav
```

Play a wave file.

```
rec my_file.wav
```

Record a wave file from my microphone.

```
mpg123 my_file.mp3
```

Play an mp3 file.

```
mpg123 -w my_file.wav my_file.mp3
```

Create a wave audio file from an mp3 audio file. Useful if you wanted to write a regular audio CD from mp3s--you have to convert the mp3s to the *.wav format first. Don't be surprised the conversion is slow--decompressing mp3s is very processor intensive.

```
xmms
```

(in X terminal) Nice GUI mp3 player.

```
freeamp
```

(in X terminal) Another GUI mp3 player.

```
lame input_file output_file
```

MP3 encoder.

```
knapster
```

(in X terminal) Start the program to download mp3 files that other users of napster have displayed for downloading. You may share your mp3s too. Really cool, while it lasts. Gnutella and FreeNet will soon replace them->it gets even cooler.

```
cdparanoia -B "1-"
```

(CD ripper) Read the contents of an audio CD and save it into wavefiles in the current directories, one track per wavefile. The "1-" means "from track 1 to the last". -B forces putting each track into a separate file.

```
playmidi my_file.mid
```

Play a midi file. `playmidi -r my_file.mid` will display text mode effects on the screen.

```
sox audio_file another_format_audio_file
```

(="SOund eXchange") Convert from almost any audio file format to another (but not mp3s). See `man sox` for the list of supported audio file formats (many). `sox` also lets you add special effects to your sound file.

```
kscd
```

(in X terminal) CD player.

```
kmidi
```

(in X terminal) MIDI player.

```
kmid
```

(in X terminal) MIDI/caraoke player.

`kmix`

(in X terminal) Sound mixer.

`studio`

(in Xterminal) Sound Studio--edit sound files, add effects, etc. Available on the on the PowerTools CD that comes with RH7.0.

`festival --tts my_file.txt`

Say the content of the `my_file.txt` file (ascii text). "festival" is a speech synthesizer that comes on the RedHat 7.0 "Linux PowerTools" CD. To say something from the command line, you need to start up "festival" and then, at the "festival>" prompt, type the appropriate command (scheme interpreter), as in this example (**bold** represents the prompt):

`festival`

festival>(SayText "good dog, really good dog")

festival> (quit)

5.17 Graphics-related commands

`kghostview my_file.ps`

(in an X-terminal) Display a postscript file on screen. I can also use the older-looking `ghostview` or `gv` for the same end effect. I can print the postscript file from the viewer too.

`enscript my_file.txt -U 2`

Convert a text file to postscript and print it to the default printer. I could also send the output to a postscript file:

`enscript my_file.txt -U 2 -o my_file.ps`

The option `-U 2` makes `enscript` print 2 logical pages on one physical page which saves me paper, and creates more convenient, compact printouts. You may also select four pages per page, more makes the printout kind of difficult to read. `enscript` is really flexible, see `man enscript` to select from among the many formatting options.

`ps2pdf my_file.ps my_file.pdf`

Make a pdf (Adobe portable document format) file from a postscript file.

`mpage -2 my_file.ps > new_file.ps`

Print the postscript file `my_file.ps`, outputting two logical pages on one physical page. Save the output to the file `new_file.ps`.

`ps2ps file.ps new_file.ps`

`psnup -nup 2 -pletter new_file.ps new_file2.ps`

Another way of making a postscript file containing 2 logical pages on one physical page. First, I used the "postscript distiller" `ps2ps` to make the postscript file simpler (at the cost of it becoming much larger). Then, I used the `psnup` utility to make `new_file2.ps` which contains 2 logical pages per one physical page. I could have also put 4 or 8 logical pages per one physical page.

`gimp`

(in X terminal) A humble looking but very powerful image processor. Takes some learning to use, but it is great for artists, there is almost nothing you can't do with `gimp`. Use your mouse right button to get local menus, and learn how to use layers. Save your file in the native `gimp` file format `*.xcf` (to preserve layers for future editing) and only then flatten it and save as `png` (or whatever) for use.

"Learning how to make proper selection is the key."

`gphoto`

(in X terminal) Powerful photo editor and camera image acquisition program.

`kpaint`

(in X terminal) Simple bitmap paint program ("paintbrush"-type).

`xfig`

(in X terminal) A simple drawing program. Useful for making elementary sketches or diagrams.

`dia`

(in X terminal) A tool for drawing diagrams from pre-built components.

`display my_picture`

(in X terminal) Display a picture for viewing only. Part of ImageMagick package (together with several other utilities described here).

`giftopnm my_file.giff > my_file.pnm`

`pnmtopng my_file.pnm > my_file.png`

Convert the proprietary giff graphics into a raw, portable pnm file. Then convert the pnm into a png file, which is a newer and better standard for Internet pictures (better technically plus there is no danger of being sued by the owner of giff patents).

`xwd -out my_capture_screen_file.xwd`

(in X terminal) Capture the contents of X-windows screen into a graphics X-windows "dump" file (*.xwd). You can later convert the xwd file into your favourite format using the convert utility.

`convert my_capture_screen_file.xwd my_capture_screen.jpg`

Convert the X-windows screen dump file (*.xwd) into the *.jpg file format. The convert utility can convert graphics from/to many different file formats.

`import -display 192.5.100.10:0 -window root my_file.jpeg`

Capture the contents of the root screen from X-windows running on server 192.5.100.10 display 0. The output file is my_file.jpeg (change the file format by it giving an appropriate filename extension). You need to have the permission to write to the screen in order to be able to capture its content (the permission to everybody can be given by running `xhost +` in the X-terminal). See `man import` for options.

`ksnapshot`

(in X terminal) GUI-based utility to capture screen contents.

`identify -verbose my_picture`

Give me a description of an image file *my_picture*: format, type, class, size in pixels, number of colours, size in bytes, etc.

`animate -delay 6x5 pic1 pic2 pic3`

Keep showing pictures two or more pictures in sequence. In the example above, the picture files are named pic1, pic2 and pic3, the delay between pictures is 0.06 second, and the whole presentation sequence is repeated in 5 seconds.

5.18 Small games

Many small games are probably installed on your system. Here are a few I like--they installed from my standard Linux distribution CD.

kpat

(in X terminal) Patience card game. sol (fast) and pysol (slow but loaded) are two other choices.

xboing

(in X terminal). Very nice, pin-ball game.

xboard

(in X terminal) Chess.

konquest

(in X terminal) Compete with your son in a conquest of a galaxy. Nice board game.

kmines

(in X terminal) Minesweeper.

civserver

civclient

(in X terminal) Startup server for the FreeCivilization game (first command). Afterwards, when the server is already running, start up the client (second comamand). Somebody else starts another client--and you play. FreeCiv came on my RH7.0 CDs.

fgfs

"Flight Gear" flight simulator.

Go to Part 6: Essential Linux applications (proprietary or not)

Back to Top Page

LINUX NEWBIE ADMINISTRATOR GUIDE

ver. 0.120 2001-03-16 by Stan and Peter Klimas

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

Copyright (c) <1999,2000,2001> 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](#).

Quick navigation:

Part 6: Some Essential Linux Applications (proprietary or not)

Contents:

- 6.1. Word processing
 - 6.1.1 StarOffice Suite [p 153]
 - 6.1.2 abiword [p 154]
 - 6.1.3 Word Perfect 8 for Linux [p 154]
 - 6.1.4 Ted [p 155]
 - 6.1.5 klyx, lyx and latex [p 155]
 - 6.2 Spreadsheet [p 158]
 - 6.3 Databases [p 158]
 - 6.4 CAD [p 158]
 - 6.5 Web browsers: Netscape and Lynx [p 159]
 - 6.6 Writing CD-Rs: cdrecord and cdparanoia [p 160]
-

Intro. This part covers only application we use or like or both. There are thousands of Linux programs. If you are unsatisfied with our lean choice, try: <http://www.linuxapps.com/> or <http://stommel.tamu.edu/~baum/linuxlist/linuxlist/linuxlist.html> or <http://www.boutell.com/lsm/> or <http://www.linuxlinks.com/Software/>

6.1 Word processing

6.1.1 StarOffice Suite

Star Office is a very complete office suite: word processor, spreadsheet, presentation program, drawing program, html editor, all integrated with a monolithic "desktop" which some hate, but I don't mind. Full version of Star Office is available for free for both Linux and MS Windows--it can be downloaded over the Internet: try <http://www.stardivision.com/freeoffice/> (large, ~100 MB download, definitely not practical with a modem). I got a copy of StarOffice on a CD supplied with a Linux Magazine (check your newsstand). Star Office source code has recently been released under GPL (Aug.2000) so it is sure to be vigorously developed.

Star Office looks and acts very much like MS Office for Windows. This includes richness of features, large size, and slow startup (boy, slow). It may not be worth the trouble without at least 32 MB of physical memory, the more the better. It has a very good file-level compatibility with MS Office: read and write MS Word, MS Excel and MS PowerPoint file formats. The StarOffice word processor is faster and feels better than WordPerfect for Linux. In short, we highly recommend StarOffice to cover even quite demanding office needs.

The installation of StarOffice is confusing and it goes like this:

- Make sure you have enough hard drive space. To check the space use the `df` (="disk free") command:

```
df -h
```

This displays a report on the used and available hard drive space in a human-legible form (option `-h`). At minimum, you need some 350 MB of free disk space (of which, ~100 MB you can release after installation).

- Decompress the downloaded file. I did it as root in the `/usr/local` directory for "local server" installation, but you may choose `/home/your_login` for "personal" installation:

```
cd /usr/local
```

```
tar -xvf StarOffice5.2.tgz
```

- As root, run the setup program for a "local server" with the DOS-style `/net` switch:

```
cd Office52
./setup /net
```

[Without the `/net` switch, StarOffice will perform a personal installation (into your home directory), and then only one user will be able to run it.]

- After this "network" installation, each user has to perform her/his own installation to put some personal files into their "home" directories (by running `program/setup` as a user, without the `/net` switch).

- StarOffice will insert a menu item into your "K-menu"-"Personal"-"StarOffice 5.2". To run it from the command line, I use:

```
/home/my_login_name/office52/soffice
```

6.1.2 abiword

(type `abiword` in an X-terminal) AbiWord (<http://www.abisource.com>). It is a good light-weight wordprocessor. Really worth trying for simple word processing needs. Although still fairly incomplete, it is quite useful to me, e.g. it supports spelling-as-you-type without the overhead of StarOffice. It is under heavy development and both version for Linux and MS Windows are available. Included on Mandrake distribution CD.

6.1.3 Word Perfect 8 for Linux

This is a very good, powerful and standard word processor. A free version is available for personal use. Try: <http://linux.corel.com/linux8/download.htm> (free registration of the program is required). The free version lacks the equation editor, built-in graphics editor, chart editor and the "art text", but it is otherwise a fully functional version of the perhaps best word processor in the world. The full version costs about \$50.

The file format of the Word Perfect 8 for Linux is the same as WordPerfect 6, 7 or 8 for MS Windows. Also, the file-level compatibility with MS Word is very good--I had instances in which WP8 was actually be more compatible with the different "sub-versions" of MS Word file format than MS Word itself. More complex MS Word documents, however, tend to look ugly after translation.

On the down side, Word Perfect for Linux feels very slow with longer documents, particularly when scrolling larger documents--StarOffice beats WordPerfect easily on that (see the next section).

WP8 is normally launched through an icon or from the K-menu but you have to add this yourself after the installation. The main executable is `/usr/local/wp8/wpbin/xwp`, but the location may vary, depending where you installed it. Use the command

```
locate xwp
```

or

```
find / -name "xwp"
```

if you cannot find it.

To "manually" launch WP8, try in X-terminal:

```
/usr/local/wp8/wpbin/xwp
```

Corel just released (Nov. 1999) their own Linux distribution (based on Debian) , which apparently centers around their WordPerfect suite (WordPerfect word processor, Quattro Pro spreadsheet, etc.) different from their current offering on Linux in that it will be based on the GPLed "Wine" (=MS Windows-Application Programming Interface-Emulation) library.

6.1.4 Ted

(type Ted in an X-terminal). An excellent looking *.rtf text editor. The version I have likes crashing though. Ted is not included on RH or Mandrake CDs that I know of.

6.1.6 lyx and latex

(Type lyx in an X-windows terminal). Both lyx is a front end (WYSIWYG, running under X-Windows) for Latex. [There is also Klyx, which is a "K-desktop" variant of Lyx.] Latex has for years been the heavy-duty document preparation and typesetting program, particularly popular in academia (good with equations, etc.).

The good news is that even if you do not know what Latex is, you may still be able to use lyx. Think of lyx as a word processor, although its philosophy is different from that of other popular word processors, and therefore it may require an adjustment of your mindset. Latex (and lyx) philosophy is to type in the text, define the "styles" and leave the formatting to the typesetting program. This means you never adjust the spacing (between words, sentences, paragraphs, chapter, etc.) manually. When done with typing of your document, you "compile" your text to create a device independent file (*.dvi). The *.dvi file can be viewed using a dvi viewer and printed. The quality of the output is usually outstanding, but its creation process is typically

somewhat more frustrating than using a regular word processor.

The strength of Latex is the quality of the printouts, its capability to cope with long, complex documents (technical books, math, etc.), availability of all foreign characters and even rarely used symbols, its portability across many different platforms, and the popularity of the file format.

lyx is free and it is included on your Mandrake CD for you to try. As almost any piece of Linux software, you can also download it from Linuxberg:

http://idirect.linuxberg.com/kdehtml/off_word.html or any other fine Linux software depository on the Internet.

If instead of easier lyx, you wanted to try straight, hard-core Latex, here is some intro to get you started:

- * Use your favorite plain-text editor to create a Latex document, spell check it, etc., save the text file with the extension "*.tex". Read on to see my sample Latex document.

- * Invoke Latex to "compile" the text file into a "*.dvi" ("device independent") file by typing on the command line:

```
latex my_letex_file.tex
```

- * Print the "my_latex_file.dvi" file which was created by the previous command by invoking the dvi to postscript utility, that on default send the output to the lpr printer:

```
dvips my_file.dvi
```

You can also save the output to postscript file by typing:

```
dvips -o output_file.ps my_file.dvi
```

The option -o introduces the output file.

Here is my sample Latex file:

```
% Any line starting with "%" is a comment.
% "\" (backslash) is a special Latex character which introduces a Latex
% command.
```

```
\documentclass[10pt]{article}
\begin{document}
```

```
% Three commands are present in every Latex document. Two of them are
% above and one at the very end of this sample document.
```

```
This is a simple document to try \LaTeX. Use your favorite plain text
editor to type in your text. See how the command \LaTeX produces the
\LaTeX logo. Here is the end of the first paragraph.
```

```
Here starts the second paragraph (use one or more empty lines in your
input file to introduce a new paragraph).
```

```
The document class of this sample is ``article'' and it is defined at the
very beginning of the document. Other popular classes are ``report'',
``book'' and ``letter''.
```


Please note that the double quote is hardly ever used, utilize two ' to begin a quote and two ' to close it. This nicely formats the opening and closing quotes.

Here are different typefaces:

```
{\rm This is also roman typeface. It is the default typeface.}
{\bf This is bold typeface. }
{\em This is emphasize (italic) typeface.}
{\sl This is slanted typeface, which is different from the italic.}
{\tt This is typewriter typeface.}
{\sf This is sans serif typeface.}
{\sc This is small caps style.}
```

You can itemize things:

```
\begin{itemize}
\item one
\item two
\item three
\end{itemize}
```

You can also enumerate things:

```
\begin{enumerate}
\item one
\item two
\item three
\end{enumerate}
```

Try some foreign letters and symbols:

```
\aa \AA \o \O \l \L \ss \ae \AE \oe \OE \pounds \copyright \dag \ddag \S
\P. There are also three dashes of different length: - -- ---.
```

Try some accents over the letter 'a': '\{a} \' {a} \"{a} ^{a} ~{a}
\={a} \. {a} \b {a} \c {a} \d {a} \H {a} \t {a} \u {a} \v {a}. Other letters can
be accented in a similar way.

The pair of '\$' marks a math context. Many special characters are
available only in the 'math' context. For example, try the Greek
alphabet:

```
Small: $ \alpha \beta \gamma \delta \epsilon \varepsilon \zeta \eta
\theta \vartheta \iota \kappa \lambda \lambda \lambda \mu \nu \xi \omicron \pi \varpi
\rho \varrho \sigma \varsigma \tau \upsilon \phi \varphi
\chi \psi \omega $
```

```
Capital: $ A B \Gamma \Delta E Z H \Theta I K \Lambda M \Xi \Pi P
\Sigma T \Upsilon \Phi X \Psi \Omega $
```

Try some equations: $x^{y+1} + \sqrt{p \times q} = z_{\text{try_subscripts}}$ \$

```
\begin{center}
$ \frac{x \times y}{x/2+1} = \frac{1}{3} $
\end{center}
```

LaTeX math commands are very similar to those in the old 'Word Perfect'
equation editor.

Use the verbatim mode to print the 10 special symbols which normally have
special meaning in LaTeX: \verb|%_#\^~\|. The special symbols must be
contained between any two identical characters which in the example above
is |. Most of these special symbols can also be printed by preceding the
character with a backslash: \% \\$ \% \{ \% _ \% \# \% \& \% \^.

```
% This command ends the document (this is the third one that *must* be
% present in every document).
\end{document}
```

6.2 Spreadsheet

I currently use the good spreadsheet included in StarOffice. I am a very heavy spreadsheet user, so here are some other promising programs I keep my eyes on.

6.2.1 gnumeric

(in X terminal) Nice spreadsheet, part of GNOME, included with standard RH distributions (RH6.0 or higher). Although still fairly incomplete and sluggish, it is already quite usable. Gnumeric is under heavy development and definitely has the potential to become really great in the near future--it already has a lot of built-in functions, but its printing is unreliable--major pain.

6.2.2 kspread

Kspread is another highly promising spreadsheet. It is part of the KDE project to be integrated with the KDE2.0, most likely you don't have it on your system and I would not advise you to try to install it--it requires next-generation KDE libraries (on which KDE2.0 will be based) and you can screw up your KDE if you try to install these with your KDE1.x version). So far, kspread is not included with RH6.2 or Mandrake 7.0.

Both gnumeric's and kspread's file format is xml (the already standard, next-generation, enhanced html). This file format is definitely good news if you ever experienced problems with MS-Windows-based spreadsheet file formats.

6.3 Databases

If you are a database person, you will be pleased to see that Linux is very well covered in this area.

postgreSQL is a high-powered database available on Mandrake and RH CD (free,unrestrictive BSD license).

mySQL <http://www.MySQL.com/> GPL database, simpler and easier than PostgreSQL.

There are also commercial databases which are free for personal use, e.g. **Sybase** for Linux (http://www.sybase.com:80/sqlserver/linux/aselinux_install.html) and **Interbase** (<http://www.interbase.com/downloads/products.html>). There is also **Oracle** for Linux: http://platforms.oracle.com/linux/index_lin.htm. For an Oracle-Linux howto, see: <http://jordan.fortwayne.com/oracle/index.html>

6.4 CAD

QCAD (GPL): <http://www.qcad.org> (simple but very useful)

OCTREE (free for non-commercial): http://www.octree.de/html/frames/eng/f_octree.htm

VariCAD (proprietary commercial): <http://www.varicad.com/>

VARKON (LGPL): <http://www.varkon.com/>

Microstation (proprietary): <http://www.microstation.com/academic/products/linux.htm>--the academic edition of Microstation includes the Linux version of their excellent CAD system (better than AutoCad).

6.5 Web browsers: Netscape and Lynx

You probably have installed a 4.xx version of Netscape during your RedHat installation. To run it, try (in X-terminal):

```
netscape
```

If you didn't install Netscape, you may want to put your RedHat CD into the CDROM, mount it using, for example (as root):

```
mount -t auto /dev/cdrom /mnt/cdrom
```

and then start, from X-terminal: `glint` (for RH5.2) or `gnorpm` (for RH6.0) or `kpackage` (for RH6.1) to browse the available packages and perhaps install netscape from your RedHat CD. If you prefer to do the installation from the command line, try, after mounting the CDROM:

```
cd /mnt/cdrom
cd RedHat/RPMS
rpm -ivh netsca*
```

Netscape is a very good browser, with the same look, feel, and power as Netscape for MS Windows, so you will have no problems navigating it. On the dark side, Netscape sometimes crashes (just disappears from the screen, no damage done, you have to restart it). Also, in some configurations, Netscape does not like to be run without a connection to the Internet (depending on your configuration, Netscape can take it up to a couple of minutes to figure out that there is no connection). Netscape version 4.72 seems to be much better than previous versions, so you if having problems you may consider upgrading.

Take your heart, once tuned-up, Netscape runs quite well (I use it all the time). Mozilla (the already famous, revolutionary offspring of Netscape) is under way so hold your breath :-).

If your Netscape version lower than 4.72 is too buggy to you, you may want to disable Java ("Edit-Preferences-Advanced"). This improved the Netscape reliability tremendously on my system. Additionally, you may want to disable the automatic checkup for new mail (solved the "freezing" problem I used to experience when working off-line), empty your mail "trash" bin, compress the mail folders on regular basis, and disable Java scrip. After doing all of this, my previous versions of Netscape worked quite nicely.

If your Netscape crashes and on a subsequent re-start complains about the presence of a lock file, it might have left a lock file in your home directory. Just quit any instance of Netscape that you might be running and delete the "lock" file from the directory `/home/user_login_name/.netscape`:

```
cd ~
cd .netscape
rm lock
```

If your ISP connection is really slow, you may prefer a text-based browser:

lynx

which is a real piece of art and does not have any problems whatsoever. Don't expect it to look as fancy a GUI-based browser though--it is text-mode based.

Another choice of a Internet browser is your KDE help utility (click on the "book with the lightbulb" button on you K-bar). This is a simple html browser so if you are connected to the Internet (e.g. using your kppp), you can browse almost anything using this utility.

To compose html pages, I use Word Perfect, StarOffice or Netscape (WYSiWYG view) and WebMaker (code view).

6.6 Writing CD-Rs: cdrecord and cdparanoia

Disclaimer: Copying copyrighted material is illegal. Do NOT use the instructions below for anything illegal.

INTRO

Setting up a CD writer can be tricky. For good intro info see the page <http://www.guug.de/~winni/linux/cdr/html/CD-Writing-3.html> .

I used the above document to set up a cheap, no-name IDE-ATAPI CD writer on a PC running RH6.0. It works great, and no changes to my setup were required after the recent upgrade to RH6.1 or RH6.2. My experience is that writing CDs under RedHat is much more reliable than under MS Windows (estimated 95% success rate!). I had problems achieving reliable results with Mandrake though. Here are the steps I followed (almost everything has to be done as root):

SETUP

• In the file `/etc/lilo.conf` , add a line at the end of the Linux "image" section:

```
append="hdb=ide-scsi"
```

Adjust the line above if your CD writer is not "hdb" (second drive on the first IDE interface). It makes your IDE-ATAPI CD-W(R) to be seen on your Linux system as a SCSI device. (It is not really a SCSI device, it is an IDE device, it just pretends to be SCSI.) Run `lilo` after making any changes to `/etc/lilo.conf` . The above change to `/etc/lilo.conf` appears to be necessary to be able to emulate SCSI on IDE-ATAPI CD-R(W) if you use Linux kernel 2.2.x.

• Add the loop devices to the `/dev/` directory. This is not a obligatory, but a nice feature if you plan creating your own data CDs. The loop device will let you mount a CD image file (as if it was a already a filesystem) to inspect its content. The loop devices don't exist on my hard drive after Linux RedHat installation, so I create them using:

```
cd /dev/  
./MAKEDEV loop
```

o Add these two lines at the end of the file `/etc/rc.d/rc.local` so that the needed kernel modules are automatically loaded on system startup:

```
/sbin/insmod ide-scsi
/sbin/insmod loop
```

These two kernel modules are needed for SCSI emulation of IDE drives and to support the loop devices, respectively.

o Create or modify the device `/dev/cdrom` so it now points to the correct device , most likely:

```
ln -s /dev/sr0 /dev/cdrom
```

You need to do this because `/dev/cdrom` pointed to an IDE device (probably `/dev/hdb`) but now this changes since your CD-R is going to be in SCSI emulation mode.

o Reboot so the changes to `/etc/lilo.conf` can take effect. Check if your CD-R(W) still works properly for normal reading.

o Download the program "cdrecord" from your favorite Linux software repository (e.g., <http://idirect.linuxberg.com/>). Then install the source code, compile it, install the program, and make symbolic links so that the executable are easy to run (the installation would be much easier if you found a binary *.rpm file):

```
cd /usr/local
tar -xvzf /the_path_to_which_you_downloaded/cdrecord-1.6.1.tar.gz
ls
cd cdrecord-1.6.1
make
make install
ls /opt/schily/bin/
ln -s /opt/schily/bin/* /usr/local/
```

The program `cdrecord` is a spartan, command line utility for writing CD. There are several GUI front ends to it, but they will be useless if the underlying `cdrecord` does not work properly. My advice: use command line for some time--you get to understand how things work, get flexibility, and reliable results. Then you can install GUI front ends to make CD covers, and make things easier for Windows-educated users on your system.

o See if your cdwriter is recognized. If it is, it should now show in the output from this command:

```
cdrecord -scanbus
```

CREATING DATA CDs

o Create a CD image containing your data:

```
mkisofs -r -o cd_image input_data_directory
```

This makes an International Standard Organization (ISO) standard 9660-type filesystem containing the files from `input_data_directory`, but writes the filesystem to an ordinary file on the hard drive. This output file is an "image" of the new CD which I am creating. The option `"-o"` indicates that the parameter that follows is the output filename of this image. The

option "-r" enables "Rock Ridge" extensions to the ISO protocol so that file attributes are saved, and it sets the file permissions so all the files on the CD are publicly readable (can be read by all user, not only the file owner). The filenames are abbreviated to the "8.3" DOS-type length but, since Linux supports so called "Rock Ridge" extensions to ISO9660, it also writes the full names and all the file permissions as well--this way the new filesystem is portable across all popular operating systems (DOS, MS Windows, Linux, UNIX, etc)--really convenient to the user.

The input data directory can be assembled from different directories and files from all-over your filesystem using symbolic links (saves harddrive space because the data is not copied), but if you do it you probably want tell `mkisofs` to follow symbolic links using the option `-f`:

```
mkisofs -r -f -o cd_image
input_data_directory_containing_symlinks
```

o You may want to inspect the CD image file by mounting it through the loop device:

```
mount -t iso9660 /dev/loop0 cd_image /mnt/cdrom
[now the content of the file should appear in /mnt/cdrom]
cd /mnt/cdrom
[inspect the file mounted through the loop device]
```

When done with inspection, change your working directory away from the mountpoint and unmount the file:

```
cd
umount /mnt/cdrom
```

o If everything worked, you may burn your data CD:

```
cdrecord -v speed=2 dev=1,0,0 -data cd_image
```

The numbers in "dev=" stand for the scsi bus number (the first one is 0, second bus is 1, ...), device id on the scsi bus (between 0 and 7), and the scsi lun number (always 0) respectively. You must customize them: the first two number can be read in the output from `cdrecord -scanbus`, the third number is 0. Make sure to use the correct numbers or you may write to a wrong drive and corrupt your data.

The timing of writing to CD-Rs is very important, or an error may occur (the laser cannot be switched on and off at will). Therefore avoid doing intensive tasks during creating a CD, e.g. don't create or erase large files on the hard drive. My system will not permit me to start new tasks when using `cdrecord`.

CREATING AUDIO CDs

o Audio tracks have to be in files of *.cdr (I guess it is the same as *.cdda.raw), *.wav (wave), or *.au format before you can write them to a CD.

o The utility `sox` converts between the various audio file formats (`sox` understands quite a few of them). For example, this will convert a .wav file to a .cdr file:

```
sox my_file.wav my_file.cdr
```

You don't need to do the conversions manually - `cdrecord` supports *.wav and *.au directly (it does a conversion from *.wav or *.au to *.cdr "on the fly"). This is very convenient because audio files tend to be large.

- o Audio CDs don't contain a filesystem, they store "raw data". This means that you cannot mount an audio CD. Also, each track is written separately, i.e., as if it was a different "partition" on the CD.

- o To read audio tracks from an audio CD and write them to a suitable file on your hard drive (typical format is *.raw or *.wav) , you need a "cd ripper". A popular CD ripper is "cdparanoia". After downloading the source for cdparanoia, install it (use the autocompletion <Tab> shortcut when typing the long filenames):

```
cd /usr/local
tar -xvzf
/the_path_to_which_you_downloaded/cdparanoia-III-alpha9.6.src.tgz
ls
cd cdparanoia-III-alpha9.6
./configure
make
make install
```

The installation program places a proper link to the executable (`/usr/local/bin/cdparanoia`) so I don't have to create the link manually.

To rip the first track from an audio CD, I can use:

```
cdparanoia 1
```

which will put the first track from the CD into the wave file "cdda.wav" in the current directory.

To rip tracks 1 to 2 from an audio CD to a "raw" file format, I can use:

```
cdparanoia -B -p "1-2"
```

The option -B specifies to use a "batch" mode, so that each track is put into a separate file (this is probably what you want, otherwise all tracks would be placed in one output file). The "-p" option specifies output in raw format. The files are named `track1.cdda.raw` and `track2.cdda.raw`.

To rip all tracks from an audio CD, each track to a separate *.wav file, while forcing reading speed 4x, I can use:

```
cdparanoia -S 4 -B "1-"
```

Make sure you have sufficient free space on your hard drive. You can use the space on your DOS partition (if you have dual boot).

- o To write suitable audio files to a CD-R(W), I can use:

```
cdrecord -v speed=2 dev=1,0,0 -audio track*
```

• Older stereos often will not play burned CD-Rs (because of the size of the pits on the CDs). My home stereo cannot read re-writable CDs (CD-RW) at all, although it will read write-once disks (CD-Rs), so re-writables may be good to store data but are useless for audio (unless I plan to play them exclusively on my computer).

CREATING MIXED-MODE CDs

Mixed-mode CDs (meaning CDs which contain both data and audio, often game CDs) are not a problems, e.g.:

```
mount -t iso9660 /dev/cdrom /mnt/cdrom (mount the data part of the mixed-mode CD)
mkisofs -r -o cd_image /mnt/cdrom (make an ISO filesystem from the data on the CD).
umount /mnt/cdrom (unmount the CD)
cdparanoia -B "2-" (rip the content of all audio tracks on the CD, except the first track since it is data)
cdrecord -v speed=2 dev=1,0,0 -data cd_image -audio track* (write the data and audio files, piece by piece)
```

MAKING A COPY OF THE ENTIRE CD

Most CDs can be copied by first copying all data (for data CDs) or all tracks (for audio CDs) onto the hard drive as described before, but some CDs cannot.

For example, these kinds of data (not audio) CDs need to be treated differently: bootable CDs (like Linux installation CD), CDs that require the label, disk with errors, etc. For data CDs, I use these commands to make an exact copy:

```
dd if=/dev/cdrom of=cd_image
cdrecord -v speed=2 dev=1,0,0 -data cd_image
```

The `dd` command copies the input file (`if`), which in this case is the device `/dev/cdrom` to the output file (`of`) which in this example is a file called `cd_image` (on the hard drive in the current working directory). The second command copies the file `cd_image` that was created by the `dd` command onto an empty CD.

For data disk with error, you might want to try:

```
dd conv=noerror,notrunc if=/dev/cdrom of=cd_image
cdrecord -v speed=2 dev=1,0,0 -data cd_image
```

The option "`conv=noerror,notrunc`" specifies that the potential read errors are to be ignored, and files not truncated on error.

For audio CDs, I use these command to make a copy:

```
cdparanoia -B "1-" (rip the content of all audio tracks on the CD, from track 1 on. The tracks are saved into files in the current directory and named: track01.cdda.wav, track02.cdda.wav, etc.)
cdrecord -v speed=2 dev=1,0,0 -audio track* (write all the audio files to the
```


CD, one by one. The tracks are separated by a 2 s gap).

To make an exact copy of mixed mode CDs,

`dd if=/dev/cdrom of=cd_image` (The `dd` command will output an error message when the data has ended and audio started. This is expected and ok).

`cdparanoia -B "2-"` (rip the content of all audio tracks on the CD, except the first track since it is data)

`cdrecord -v speed=2 dev=1,0,0 -data cd_image -audio track*` (Write the data and subsequent audio files, piece by piece.)

RE-WRITABLE CDs

Re-writable CDs (CD-RW) are used the same way as regular write-once CDs (CD-R), but you have to blank re-writable disks before you will be able to re-use them, e.g.:

`cdrecord -v speed=2 dev=1,0,0 blank=fast`

To see other (more thorough and slower) options for blanking, use:

`cdrecord blank=help`

SIMPLIFYING LONG COMMANDS WITH AN ALIASES

To simplify writing long commands required by `cdrecord`, I may define a global alias by placing the following line in the file `/etc/bashrc`:

`alias cdrecord="cdrecord -v speed=2 dev=1,0,0"`

Re-login for the changes in `/etc/bashrc` to take effect. After creating this alias, a I can record a CD using the following shortened command (no need to specify the CD writer speed and device name all the time):

`cdrecord -audio track*`

Goto part 7: Learning with Linux

LINUX NEWBIE ADMINISTRATOR GUIDE

ver. 0.120 2001-03-16 by Stan and Peter Klimas

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

Copyright (c) <1999,2000,2001> 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](#).

Quick navigation:

Part 7: Learning with Linux

Contents :

7.1 Linux Advanced Text Processing Tools [p 166] : emacs, vi [p 166] , joe, diff, patch, grep [p 167] , regular expressions [p 168] , sed [p 169] , (g)awk [p 170] , cvs, sort [p 171] and ascii, gpg (pgp) [p 173]

7.2 Simple Programming under Linux [p 174] : perl [p 174] , python [p 175] , tcl [p 176] /tk/wish [p 176] , c/c++ [p 176] , kdevelop, glade [p 177]

7.3 Math tools [p 178] : dc [p 178] , bc and "regular" calculator, scilab , generation of random [p 180] characters on the command line

7.4 : Miscellaneous: wine [p 180] , RAID disk tools [p 181] , network traffic shaping.

Intro: This page contains commands that are not essential for newbies but can be fun and/or very productive if you take the time to learn them. Above all, they can be very educational--linux is a great learning platform.

7.1 Linux Advanced Text Processing Tools

`/usr/games/banner -w79 "Happy Birthday, Marie"> marie.txt`

Create an ascii "banner" with the width of 79 characters. The output is sent to file `marie.txt`.

Funny, old fashioned tool.

`script`

Log my current session in the text terminal into text file `typescript` . The log finishes when I type `exit` or press `<Ctrl>d`.

`emacs`

(in X-terminal) The emacs text editor. Advanced and sophisticated text editor. Seems for gurus only: "emacs is not just an editor, it is a way of living". Emacs surely seems rich or bloated, depending on your point of view. There are likely 3 versions of emacs installed on your system: (1) text-only: type `emacs` in a text (not X-windows) terminal (I avoid this like fire); (2) graphical-mode: type `emacs` in an X-windows terminal (fairly usable even for a newbie if you decide to take some time to learn it); and (3) X-windows mode: type `xemacs` in an X-windows terminal.

`vi`

The famous (notorius?) "vi" text editor (definitely not recommended for newbies). To exit "vi" (no changes saved) use these five characters: `<ESC>:q!<Enter>` I use the pico text editor and don't ever need vi (well, unless I have to unmount the `/usr` subsystem and modify/edit some configuration

files).

Experts do love `vi`, but `vi` is definitely difficult unless you use it every day. Here is a non-newbie opinion on `vi` (<http://linuxtoday.com/stories/16620.html>):

"I was first introduced to `vi` in 1988 and I hated it. I was a freshman in college... `VI` seemed archaic, complicated and unforgiving... It is now 12 years later and I love `vi`, in fact it is almost the only editor I use. Why the change? I actually learned to use `vi`... Now I see `vi` for what it really is, a powerful, full featured, and flexible editor..."

A short introduction to basic `vi` commands and modes can be viewed at <http://www.thelinuxgurus.org/vitut.shtml>.

For your entertainment, you may also want to try the even more ancient `ed` editor (just type `ed` on the command line).

`joe`

A nice and relatively newbie-friendly ascii-text editor. In my eyes, `joe` would come right after `pico` if I could remembered the WordStar or Borland-IDEs keystroke combinations.

`khexedit`

(in X terminal) Simple hexadecimal editor. Another hexadecimal editor is `hexedit` (text based, less user=friendly). Hex editors are used for editing binary (non-ASCII) files.

`diff file1 file2 > patchfile`

Compare contents of two files and list any differences. Save the output to the file `patchfile`.

`sdiff file1 file2`

Side-by-side comparison of two text files. Output goes to the "standard output" which normally is the screen.

`patch file_to_patch patchfile`

Apply the patch (a file produced by `diff`, which lists differences between two files) called `patchfile` to the file `file_to_patch`. If the patch was created using the previous command, I would use: `patch file1 patchfile` to change `file1` to `file2`.

`grep` filter

Search content of text files for matching patterns. Definitely worth to learn at least the basics of this command.

A simple example. The command:

`cat * | grep my_word | more`

will search all the files in the current working directory (except files starting with a dot) and print the lines which contain the string "my_word".

A shorter form to achieve the same may be:

`grep my_word * | more`

The patterns are specified using a powerful and standard notation called "regular expressions".

There is also a "recursive" version of `grep` called `rgrep`. This will search all the files in the current directory and all its subdirectories for `my_word` and print the names of the files and the matching line:

```
rgrep -r my_word . | more
```

Regular expressions (regexpr)

Regular expressions are used for "pattern" matching in search, replace, etc. They are often used with utilities (e.g., `grep`, `sed`) and programming languages (e.g., `perl`).

In regular expressions, most characters just match themselves. The exceptions are the "metacharacters" that have special meaning.

In `regexpr`, the special characters are: `"\"` (backslash), `"."` (dot), `"*"` (asterisk), `"["` (bracket), `"^"` (caret, special only at the beginning of a string), `"$"` (dollar sign, special only at the end of a string). A character terminating a pattern string is also special for this string.

The backslash, `"\"` is used as an "escape" character, i.e., to quote a subsequent special character. Thus, `"\""` searches for a backslash, `"\"."` searches for a dot, `"\"*"` searches for the asterisk, `"\"["` searches for the bracket, `"\"^"` searches for the caret even at the beginning of the string, `"\"$"` searches for the dollar sign even at the end of the string.

Backslash followed by a regular (non-special) character may gain a special meaning. Thus, the symbols `<` and `>` match an empty string at the beginning and the end of a word, respectively. The symbol `\b` matches the empty string at the edge of a word, and `\B` matches the empty string provided it's not at the edge of a word.

The dot, `"."`, matches any single character. [The `dir` command uses `"?"` in this place.] Thus, `"m.a"` matches `"mpa"` and `"mea"` but not `"ma"` or `"mppa"`.

Any string is matched by `".*"` (dot and asterisk). [The `dir` command uses `"*"` instead.] In general, any pattern followed by `"*"` matches zero or more occurrences of this pattern. Thus, `"m*"` matches zero or more occurrences of `"m"`. To search for one or more `"m"`, I could use `"mm*"`.

The `*` is a repetition operator. Other repetition operators are used less often--here is the full list:

<code>*</code>	the preceding item is to be matched zero or more times;
<code>\+</code>	the preceding item is to be matched one or more times);
<code>\?</code>	the preceding item is optional and matched at most once);
<code>\{n\}</code>	the preceding item is to be matched exactly n times;
<code>\{n, \}</code>	the preceding item is to be matched n or more times);
<code>\{n, m\}</code>	the preceding item is to be matched at least n times, but not more than m times.

The caret, `"^"`, means "the beginning of the line". So `"^a"` means "find a line starting with an `"a"`".

The dollar sign, `"$"`, means "the end of the line". So `"a$"` means "find a line ending with an `"a"`".

Example. This command searches the file `myfile` for lines starting with an "s" and ending with an "n", and prints them to the standard output (screen):

```
cat myfile | grep '^s.*n$'
```

Any character terminating the pattern string is special, precede it with a backslash if you want to use it within this string.

The bracket, "[" introduces a set. Thus `[abD]` means: either a or b or D. `[a-zA-C]` means any character from a to z or from A to C.

Attention with some characters inside sets. Within a set, the only special characters are "[", "]", "-", and "^", and the combinations "[:", "[=", and "[.". The slash is not special within a set.

Some useful categories of characters are: `[:upper:]` =upper-case letters, `[:lower:]` =lower-case letters, `[:alpha:]` =letters meaning upper+lower, `[:digit:]` =0 to 9, `[:alnum:]` = alpha and digits, `[:space:]` =whitespace meaning <Space>+<Tab>+<Newline> and similar, `[:graph:]` =graphically printable characters but space, `[:print:]` =printable characters including space, `[:punct:]` =punctuation characters meaning graphical characters minus alpha and digits, `[:cntrl:]` =control characters meaning non-printable characters.

Example. This command prints lines containing a capital letter followed by a digit:

```
dir -l | grep '[:upper:][:digit:]'
```

`tr`

(=translation). A filter useful to replace all instances of characters in a text file or "squeeze" the white space.

Example :

```
cat my_file | tr 1 2> new_file
```

This command takes the content of the file `my_file`, pipes it to the translation utility `tr`, the `tr` utility replaces all instances of the character "1" with "2", the output from the process is directed to the file `new_file`.

`sed`

(=stream editor) I use `sed` to filter text files. The pattern to match is typically included between a pair of slashes `//` and quoted.

For example, to print lines containing the string "1024", I may use:

```
cat filename | sed -n '/1024/p'
```

Here, "sed" filters the output from the `cat` command. The option "-n" tells `sed` to block all the incoming lines but those explicitly matching my expression. The `sed` action on a match is "p"=print.

Another example.

```
cat filename | sed '/.*o$/d'> new_file
```

In this example, lines ending the an "o" will be deleted. I used a regular expression for matching any string followed by an "o" and the end of the line. The output (i.e., all lines but those ending with "d") is directed to `new_file`.

To search and replace, I use the sed 's' action, which comes in front of two expressions:

```
cat filename | sed 's/string_old/string_new/'> newfile
```

A shorter form for the last command is:

```
sed 's/string_old/string_new/' filename> newfile
```

`gawk`

(=GNU awk. The awk command is a traditional UNIX tool.) A tool for processing text file, in many respects similar to sed. Perl can do all that gawk can, and more, so I don't bother with gawk. For simple tasks, I use sed, for more complicated tasks, I use perl.

`cvs`

Concurrent versions system. Try: `info cvs` for more information. Useful to keep the "source code repository" when several programmers are working on the same computer program.

`cervisia`

(in X-terminal). A GUI front-end to the cvs versioning system.

`file -z filename`

Determine the type of the file *filename*. The option -z makes `file` look also inside compressed files to determine what the compressed file is (instead of just telling you that this is a compressed file).

To determine the type of content, `file` looks inside the file to find particular pieces of contents ("magic numbers")--it does not just look at the filename extension like MS Windows does. The "magic numbers" are stored in the text file `/usr/share/magic`--really impressive database of filetypes.

```
strings filename | more
```

Display the strings contained in the binary file called *filename*. "strings" could, for example, be a useful first step to a close examination of an unknown executable.

`od`

(=octal dump). Display contents as octal numbers. This can be useful when the output contains non-printable characters. For example, a filename may contain non-printable characters and be a real pain.

Examples:

```
dir | od | more
cat my_file | od | more
od my_file
```

`wc`

(=word count) Print the number of lines, words, and bytes in the file.

Examples:

```
dir | wc
cat my_file | wc
wc myfile
```

`cksum filename`

Verify the integrity of file *filename* by computing the CRC (="cyclic redundancy check").

`sort -f filename`

Arrange the lines in *filename* according to the ascii order. The option `-f` tells `sort` to ignore the upper and lower character case. The ascii character set is (see `man ascii`):

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	00	NUL '\0'	32	20	SPACE	65	41	A	97	61	a
1	01	SOH	33	21	!	66	42	B	98	62	b
2	02	STX	34	22	"	67	43	C	99	63	c
3	03	ETX	35	23	#	68	44	D	100	64	d
4	04	EOT	36	24	\$	69	45	E	101	65	e
5	05	ENQ	37	25	%	70	46	F	102	66	f
6	06	ACK	38	26	&	71	47	G	103	67	g
7	07	BEL '\a'	39	27	'	72	48	H	104	68	h
8	08	BS '\b'	40	28	(73	49	I	105	69	i
9	09	HT '\t'	41	29)	74	4A	J	106	6A	j
10	0A	LF '\n'	42	2A	*	75	4B	K	107	6B	k
11	0B	VT '\v'	43	2B	+	76	4C	L	108	6C	l
12	0C	FF '\f'	44	2C	,	77	4D	M	109	6D	m
13	0D	CR '\r'	45	2D	-	78	4E	N	110	6E	n
14	0E	SO	46	2E	.	79	4F	O	111	6F	o
15	0F	SI	47	2F	/	80	50	P	112	70	p
16	10	DLE	48	30	0	81	51	Q	113	71	q
17	11	DC1	49	31	1	82	52	R	114	72	r
18	12	DC2	50	32	2	83	53	S	115	73	s
19	13	DC3	51	33	3	84	54	T	116	74	t
20	14	DC4	52	34	4	85	55	U	117	75	u
21	15	NAK	53	35	5	86	56	V	118	76	v
22	16	SYN	54	36	6	87	57	W	119	77	w
23	17	ETB	55	37	7	88	58	X	120	78	x
24	18	CAN	56	38	8	89	59	Y	121	79	y
25	19	EM	57	39	9	90	5A	Z	122	7A	z
26	1A	SUB	58	3A	:	91	5B	[123	7B	{
27	1B	ESC	59	3B	;	92	5C	\ '\\'	124	7C	
28	1C	FS	60	3C	<	93	5D]	125	7D	}
29	1D	GS	61	3D	=	94	5E	^	126	7E	~
30	1E	RS	62	3E	>	95	5F	_	127	7F	DEL
31	1F	US	63	3F	?	96	60	`			
			64	40	@						

If you wondered about the control characters, here is the meaning of some of them on the console (Source: `man console_codes`). Each line below gives the code mnemonics, its ASCII decimal number, the key combination to produce the code on the console, and a short description:

BEL (7, <Ctrl>G) bell (=alarm, beep).

BS (8, <Ctrl>H) backspaces one column (but not past the beginning of the line).

HT (9, <Ctrl>I) horizontal tab, goes to the next tab stop or to the end of the line if there is no earlier tab stop.

LF (10, <Ctrl>J), VT (11, <Ctrl>K) and FF (12, <Ctrl>L) all three give a linefeed.

CR (13, <Ctrl>M) gives a carriage return.

SO (14, <Ctrl>N) activates the G1 character set, and if LF/NL (new line mode) is set also a carriage return.

SI (15, <Ctrl>O) activates the G0 character set.

CAN (24, <Ctrl>X) and SUB (26, <Ctrl>Z) interrupt escape sequences.

ESC (27, <Ctrl>[) starts an escape sequence.

DEL (127) is ignored.

CSI (155) control sequence introducer.

`uniq`

(=unique) Eliminate duplicate lines in **sorted** input. Example: `sort myfile | uniq`

`fold -w 30 -s my_file.txt> new_file.txt`

Wrap the lines in the text file `my_file.txt` so that there is 30 characters per line. Break the lines on spaces. Output goes to `new_file.txt`.

`fmt -w 75 my_file.txt> new_file.txt`

Format the lines in the text file to the width of 75 characters. Break long lines and join short lines as required, but don't remove empty lines.

`nl myfile> myfile_lines_numbered`

Number the lines in the file `myfile`. Put the output to the file `myfile_lines_numbered`.

`rev filename> filename1`

Print the file `filename`, each line in reversed order. In the example above, the output is directed to the file `filename1`.

`shred filename`

Repeatedly overwrite the contents of the file `filename` with garbage, so that nobody will ever be able to read its original contents again.

`paste file1 file2> file3`

Merge two or more text files on lines using <Tab> as delimiter (use option "d=" to specify your own delimiter(s)).

Example. If the content of `file1` was:

1
2
3

and `file2` was

a
b
c
d

the resulting `file3` would be:

1 a
2 b
3 c
 d

`des -e plain_file encrypted_file`

(="Data Encryption Standard") Encrypt `plain_file`. You will be asked for a key that the program will use for encryption. Output goes to `encrypted_file`. To decrypt use

`des -d encrypted_file decrypted_file`.

gpg

"Gnu Privacy Guard"--a free equivalent of PGP ("Pretty Good Privacy"). gpg is more secure than PGP and does not use any patented algorithms. gpg is mostly used for signing your e-mail messages and checking signatures of others. You can also use it to encrypt/decrypt messages. <http://www.gnupg.org/> contains all the details, including a legible, detailed manual.

To start, I needed a pair of keys: private and public. The private key is used for signing my messages. The public key I give away so that other can use it to verify my signatures. [One can also use a public key to encrypt message so it can only be read using my private key.] I generated my keypair using this command:

```
gpg --gen-key
```

My keys are stored in the directory `~/.gnupg` (encrypted using a passphrase I supplied during the key generation). To list my public key in plain text file, I use:

```
gpg --armor --export my_email_address> public_key_stan.gpg
```

which created a file `public_key_stan.gpg` containing something like this:

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1.0.1 (GNU/Linux)
Comment: For info see http://www.gnupg.org

mQGibDmnzEYRBACoN438rxANaMfCy5bfj6KWM0/TR6x6HZ0gpmhGeuouM/SOR2IU
/G30NdCuzHeFs93BhtY0IdzoEMtMyZHnvdhZC2bx/jhgaaMbEaSSxwRhVB0xVPYx
rHbSgSULHYzRFF34MS3/Lse3QWfWxzA7I0lbXB7nLwZKZqaNONRFR42owCg60hV
TDPEB2N01lMyt12R4ZByFSsEAJ1te7pb9b6TP7cw21vkIjc+BI2uSzn/B4vNlCWK
TTuZHVv0w0jFcbdb8DB0/1tlZUOrIzLSqJyQDGiNn258+7LetQ+LKG/1YKbiAcosz
4QirBuLIeF2M9GuXYCwZypE3Dwv+4YupvybR3lCgLTJ8p4sKqC5n0eSr2oSrtDHZ
yuJtA/9v2HcebOncfCNOK+cVRmcTB1Fr1/Gh/vNCfeZyXaJxlqDfCU2vJHtBemiE
AtcfZHB/iHyODM68LfRJSaIFaa5um9iWHh5/vWCGZLqtpwZ7kyMw+2D6CFkWATsy
wQAgl1VcGkNc14Crrd36qf60bI+b8pn2zDhwZtLsELsXyXkNhbQmU3RhbiBKIEts
aWlhcyA8U3RhbktsaWlhC0B3ZWJoYXJ0Lm5ldD6IVgQTEQIAFgUCOafMRgQLCgQD
AxUDAGMWAgECF4AACgkQt+ZBooH8bHd2kwCghAt9aKIk0mRJv+g7YcRPotVtrwKA
nla4xEVEyaKgKoMaJnopf69K9+vouQENBDmnzH4QBADgFpLP+tWZPnVYg47cn+9b
XQRjdOtNsDE6BYH872/sRloCrdH6k+gXFOiZxRZ3PElK2/olo59kh5xa9aBxNdEC
FuXJN0UelmhOFbDtqVksIqVWYfXnLz+wtcXg0Q0L0q8vY4IuTzw2WkV6EkM+/x8
6Uha2XVaMJBkBRKFSvIlbwADBQP+JCzLj5HDgpRvf+KM72nzSg7sp8Tki7nF9wNA
PODK0SeQgI3dwXYyF6AVenLETE/3xRWoYQNlxbVZsOex9vzqPrQC3dR0NB1jd74r
kfXwUt12fNQX4N9iuVCo2gCGbi5+gfEk1GhsWDSq0z40f+18k+XBdWmY8sCNiolt
tnvmlQeIRgQYEQIABgUCOafMfgAKCRC35kGigfxsd9SGAJ9/FWSkEfgbE/Yc46d8
Ef1gYg3I1ACff3oLeAMeGGO79gW6UGp9RJ6mRao=
=X1k2
-----END PGP PUBLIC KEY BLOCK-----
```

Now, I can e-mail my public key to the people with whom I want to communicate securely.

They can store it on their pgp system using;

```
gpg --import public_key_stan.gpg
```

Now, I can start using gpg. To nicely sign a plain text file `my_message`, I use:

```
gpg --clearsign my_message
```

This created file `my_message.asc` which may contain something like:

```
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1
```

```
Hello World!
```

```
-----BEGIN PGP SIGNATURE-----
```

```
Version: GnuPG v1.0.1 (GNU/Linux)
```

```
Comment: For info see http://www.gnupg.org
```

```
iD8DBQE5p9+3t+ZBooH8bHcRApn/AJ9kx9+pU3GJBuvJN9Bo3bW3ku/5PwCgquht
```

```
mfrPrt7PQtdmGox72jkY0lo=
```

```
=rtK0
```

```
-----END PGP SIGNATURE-----
```

To verify a signed message, I could do:

```
gpg --verify my_message.asc
```

If the contents of the signed section in `my_message.asc` was even slightly modified, the signature will not check.

"docbook" tools

Docbook is the incoming standard for document depository. The docbooks tools are included with RH6.2 and include the converters: `db2ps`, `db2pdf`, `db2dvi`, `db2html`, `db2rtf` which convert a docbook files to: postscript (*.ps), Adobe Portable Document Format (*.pdf), device independent file format (*.dvi), HyperText Markup Language (*.html), and Rich Text Format (*.rtf), respectively.

"Document depository" means the document in a format that can automatically translated into other useful formats. Consider a developing document which may need to be published as a report, a journal paper, a newspaper article, a webpage, perhaps a book, I (the author) am still uncertain. Formatting the document using "hard codes" (fonts, font sizes, page breaks, line centering, etc.) is rather a waste of time--styles vary very much between the particular document types. The solution is to format the document using "logical" layout elements which may include the document title, chapter titles, subchapters, emphesise style, picture filenames, caption texts, tables, etc. Thats what "docbook" does--it is a description of a style (using xml, a superset of html, and a close relative of sgml)--a so-called stylesheet. The logical layout is rendered to a physical appearance when the document needs to be published.

This section will be expanded in the future as we learn to use docbook.

7.2 Simple Programming under Linux

perl

Powerful and widely used scripting language, very popular among gurus. Perl looks cryptic yet it is quite straight-forward if you need to achieve simple tasks. Think of perl as a swiss-army knife for simple programming.

How do I write a simple perl script?

Perl is a scripting language famous for its power, flexibility and perhaps cryptic syntax. It is also very popular among linux and unix (and not only) gurus. I may use `pico` (or any other text editor of my choice) to type in a simple perl script:

```
pico try_perl
```

The example script below does nothing useful, except illustrates some features of perl:

```
#!/usr/bin/perl -w
# a stupid example perl program
# the lines starting with # are comments except for the first
line
# names of scalar variables start with $
$a=2;
$b=3;
# each instruction ends with a semicolon, like in "c"
print $a**$b, "\n";
$hello_world='Hello World';
print $hello_world, "\n";
system "ls";
```

The first line tells the shell how to execute my text file. The option "-w" causes perl to print some additional warnings, etc. that may be useful for debugging your script. The next 3 lines (starting with #) are comments. The following lines are almost self explanatory: I assign some values to two variables (\$a and \$b), put \$a to power \$b and print the result. The "\n" prints a new line, just like in the "c" programming language. Then I define another variable to contain the string "Hello World" and, in the next line, I print it to the screen. Finally, I execute the local operating system command "ls", which on Linux prints the listing of the current directory content. Really stupid script.

After saving the file, I make it executable:

```
chmod a+x try_perl
```

Now, I can run the script by typing:

```
./try_perl
```

python

Modern and very elegant object oriented interpreter. Powerful and (arguably) more legible than perl. Very good (and large) free handbooks by G. van Rossum (the Python creator) are available on the net (try: <http://www.python.org/doc/> for browsing or <ftp://ftp.python.org> for downloading).

How do I write a simple Python program?

Edit a text file that will contain your Python program:

```
pico try_python
```

Type in some simple python code to see if it works:

```
#!/usr/bin/env python
print 2+2
```

The first line (starting with the "pound-bang") tells the shell how to execute this text file--it must be there (always as the first line) for Linux to know that this particular text file is a Python script. The second line is a simple Python expression.

After saving the file, I make it executable:

```
chmod a+x try_python
```

after which I can run it by typing:

```
./try_python
```

Python is an excellent, and very modern programming language. Give it a try if you like object oriented programming.

tcl

(Pronounced "tickle".) Popular scripting language.

A simple tcl program?

```
#!/usr/bin/tclsh
puts stdout {Hello World!}
```

wish

(type in X-terminal) A front-end to Tk, an X-windows extension of tcl. Often used for building front-ends of a program.

How do I write a simple GUI program (using Tk)?

Tk is a GUI extension of the easy yet powerful tcl programming language. For example, I may use pico to create a text file that will contain a simple tk program:

```
pico try_tk
```

and type in a simple example of tk code to see if it works:

```
#!/usr/bin/wish
button .my_button -text "Hello World" -command exit
pack .my_button
```

The first line (starting with the "#" pound-bang) tells the shell what utility to use to execute my text file. The next two lines are an example of a simple tk program. First, I created a button called "my_button" and placed it at the root of my class hierarchy (the dot in front of "my_button"). To the button, I tied the text "Hello World" and a command that exists the program (when the button is pressed). Last line makes my program's window adjust its size to just big enough to contain my button.

After saving the file, I make it executable:

```
chmod a+x try_tk
```

after which I can run it by typing (in the X-terminal, because it requires X-windows to run):

```
./try_tk
```

Tk is a very popular for building GUI front ends.

gcc filename.c

GNU C compiler. Quite straight-forward if you know C. Extensive free manuals are available on the net.

How do I compile a simple C program?

Start your favourite text editor and type in your source code. For example, I may use `pico`:

```
pico hello.c
```

and type in the Kerningham and Richie (the creators of "c") intro example of a C program:

```
#include <stdio.h>
void main(void) {
printf("hello world\n");
}
```

I save the file and then invoke the GNU C compiler to compile the file "hello.c":

```
gcc hello.c
```

The gcc compiler produces an executable binary file "a.out", which I can run:

```
./a.out
```

`g++ filename.C`

GNU C++ compiler. The capital "C" is often used for C++ sources. If you need an "integrated development environment" (IDE), `kdevelop` is really something you would probably like to look at.

`kdevelop`

(type in X-terminal) Intergrated development environment for K. Really worth downloading (if it does not come with your distribution).

`glade`

(type in X-terminal) A graphical builder of user interfaces.

"Glade is an interface builder developed by Damon Chaplin. It allows graphical and interactive construction of Gnome/Gtk graphical user interfaces. From Glade, the generated interface can be saved in a xml file or directly exported to C code to be included in a C source tree. Glade also allows to define the name of the handlers - functions - to be attached to the various event of the interface. For example the function (name) to be called when a specific menu item is pressed."
(From: http://linuxtoday.com/news_story.php3?ltsn=2000-07-16-013-04-PS-GN)

`guile`

An implementation of "Scheme" programming language. Scheme is a modern dialect of LISP.

Silly guile examples:

```
(+ 1 1)
(system "ls")
(display "hello\n")
(define a 7)
(exit)
```

`g77`

GNU FORTRAN. If you are into FORTRAN, you may want to check:

<http://studbolt.physast.uga.edu/templon/fortran.html> to find a FORTRAN compiler that suits your particular needs under Linux.

`expect`

Scripting language for "programmed dialog". See `man expect`.

kylix

This is a brand-new (Feb.2001) commercial offering from Borland (aka Inprise). In short, it is a Linux port of the famous object-oriented Pascal ("Delphi"). kylix is unlikely to be on your Linux distribution CD, you must pay for it, but if you want the best rapid application development (RAD) platform with a code portability between Linux and MS Windows, large number of pre-built components, etc., kylix is likely the best. In my opinion, Delphi is significantly better than MS Visual Basic.

make

Run the "make" utility to build (compile, link, etc) a project described in the `Makefile` found in the current directory.

yes

Generate a never-ending output of strings containing "yes" (it does end when <Ctrl><c> is pressed). Sounds like a silly utility, but it can be used to write simple program on the command line. This amusing example determines the frequency of digits in 100 000 random numbers (the whole command is a single line):

```
yes | sed '100000q' | awk  
'BEGIN{srand();u=6*log(10)}{printf"%e\n",rand()*exp(rand()*u)}'| cut  
-c1 | sort | uniq-c
```

Hope this example does not scare you too much, but it surely shows that old-fashioned UNIX can be as complicated as you want to make it. At the same time, it shows you the power of the humble command line. If you are interested why the frequency of digits varies, try the place from which I borrowed the example: <http://www.newscientist.com/ns/19990731/letters4.html>

7.3 Math Tools

dc

A command-line, arbitrary-precision "reverse Polish notation" (RPN) calculator.

dc is based on the concept of a stack, which is central to the operations of modern digital computer. A computer stack is not unlike a stack of kitchen plates, the last to come on stack, is the first to go out (this is also known as LIFO="last-in, first-out"). This contrasts with a queue (another important concept) where the first in is the first out (FIFO).

You can perform operations only on the number(s) which is on the top of the stack. The two basic operations are: push and pop (put on the top of stack, and retrieve from the top of stack). Unary operations pop one value off the stack ("unary" means "requiring one operand"). Binary operations pop two values off the stack ("binary" means "requiring two operands"). Tertiary operations pop three values off the stack ("tertiary" means "requiring three operands"). In all cases, the result is always pushed back onto the top of stack.

RPN calculators (regular, hand-held) are very popular among technically oriented people and in academia. The RPN notation never requires parentheses.

History. The parentheses-free logic was developed by Polish mathematician Jan Lukasiewicz (1878-1956) before the WWII. Originally, the operator preceded the values. For computer applications, it's been modified so that the operator goes after the values, hence "reversed" in the name "reversed Polish notation".

To exercise some operations on stack, try this:

```
dc [start the arbitrary precision reverse Polish notation calculator]
1 [push "1" on the stack]
2 [push another number on the stack]
3 [push yet another number on the stack]
4 [push yet another number on the stack]
f [print the entire stack; you should see 1 2 3 4]
p [print the number on the top of the stack without affecting the stack; you should see 4]
+ [perform addition (binary operation), therefore pop two last values off the stack (4,3), and
push the result (7) on the stack]
p [print the number on the top of the stack, i.e. the results of the last addition (7).]
p [print again the number on the top of the stack to see that the stack wasn't affected by
printing (7)]
* [perform multiplication (binary operation), therefore pop two last values, and push the
result (14)]
p [print the result of the multiplication (14)]
P [pop the last number off the stack (14)]
p [print the number on the top of the stack]
2000 [push a large integer on the stack]
k [set the precision to the value which is on the top of the stack, i.e. 2000]
1 [push another number on the stack]
f [print the content of the entire stack]
701 [push another number on the stack]
/ [divide last two numbers on the stack, i.e. "1/701" with 2000 decimal places of precision]
p [print the result of the last division]
q [quit the arbitrary precision reverse Polish notation calculator]
```

Please note that when using the reverse Polish notation (RPN) you never need parentheses. Try
man dc to read about other capabilities of dc.

bc

An interactive arbitrary-precision calculator. Type "quit" to exit bc. Type "scale=20" (or so) before
doing floating point divisions or you end up with the quotient instead of the real result of the division.

kcalc

xcalc

(in X terminal) Standard, GUI calculators.

scilab

(in X terminal) A large and sophisticated system for numerical computing, somewhat resembling
"matlab", and also with clampsy interface. Don't even try it unless you have rather sophisticated math
needs else you won't appreciate it. It is included on RH7.0 "powertool" CD.

A silly example session showing some matrix algebra. My input is shown **bold**.

```
-->a=[1 1;2 3 4]
```

```
a =
```

```
! 1.  1.  1. !
```

```
! 2.  3.  4. !
```

```

-->b=[1 1; 2 2;3 3]
b =
! 1.  1. !
! 2.  2. !
! 3.  3. !

-->c=a*b
c =
! 6.  6. !
! 20. 20. !

-->d=inv(c)
d =
1.0E+14 *

! 11.258999 - 3.3776997 !
! - 11.258999  3.3776997 !

-->

```

```

head -c 8 /dev/random
cat /dev/random | od
cat /dev/urandom | memencode

```

(3 commands.) Examples on how to generate random characters on the Linux command line by reading from the device "random" or "urandom". The first command produces approximately 8 characters by reading from the device "random", which generates high quality (difficult to predict) random numbers. It will become slow once the "entropy" on your computer is exhausted (e.g., when producing lots of random characters). The solution then is to wait or type something on the keyboard, move your mouse, switch the terminal, make your hard drive to read or write, etc., to generate more random noise ("entropy"). The second command keeps producing random characters, but displays them as octal numbers (using the "octal dump", od, filter). It has to be interrupted with <Ctrl><c>. The third command uses the device "urandom", which is faster than "random" when generating lots of random characters. But when the system entropy is low, the randomness of its output from "urandom" might be compromised, yet it probably is still good for all but most demanding applications. The output is filtered to the mime format (the Internet mail-oriented 7-bit encoding standard) so it is all printable ASCII. The detailed description of the theory and implementation of the Linux algorithm for generating the random numbers can be found in the source code file://usr/src/linux/drivers/char/random.c on your Linux system.

7.4 Miscellaneous

How do I run an MS Windows Application (using "wine")?

You don't want to install Linux to run MS Windows applications. Still, the Linux-based "wine" library lets me execute some MS Windows binaries, although with a rather severe speed penalty.

On my system (Wine installed), I can execute MS Solitaire by typing in the X-windows terminal:

```
wine /mnt/dos_hda1/windows/sol.exe
```

The /mnt/dos_hda1 is the mount point of the harddrive partition that contains MS Windows, and it is mounted.

If you don't have wine installed, put your Mandrake cd into the cdrom, mount it, and then do something like this (as root):

```
cd /mnt/cdrom/Mandrake/RPMS/  
rpm -ihv wine-991212-1mdk.i586.rpm
```

Mandrake's packages are RedHat compatible so you can use Mandrake CD to install software that RedHat lacks.

Can I have a RAID if my computer has two or more IDE (or other) harddrives?

RAID = "redundant array of inexpensive drives". RAID can be used for a "on-the-fly" mirroring of one drive to another so as to protect your data and keep your system functioning in case of a disk failure. Linux comes with a set of RAID tools that let you custom-design a RAID system to suit your particular needs. The pieces of RAID on Linux are:

```
mkraid    - initializes/upgrades RAID device arrays  
raid0run  - starts up old (superblock-less) RAID0/LINEAR arrays  
raidstart - command set to manage md devices  
raidstop  - command set to manage md devices  
raidtab   - configuration file for md (RAID) devices
```

RAID operates by joining two or more disks into a single logical device. There are several layers of RAID:

RAID 0 layer ("striping") just joins two or more disks into a single logical device, without giving any redundancy. It is often used to join RAID 1 or RAID 5 layers. RAID 0 + RAID 1 is called RAID 10. RAID 0 + RAID 5 is called RAID 50.

RAID 1 (mirroring) combines two disks, each containing the same data.

RAID 4 combines three or more disks, with one of the disks dedicated to parity. If any disk fails, the whole logical device remains available, but with degraded performance. It is not used very often because of the performance.

RAID 5 combines three or more disks, with parity distributed across the disks. Functionality similar to RAID 4 but apparently better performance.

Try http://www.osfaq.com/vol1/linux_softraid.htm if you would like set up a raid on your computer.

Network traffic shaping using `shapcfg`

Nice info can be found at: <http://oreilly.linux.com/pub/a/linux/2000/08/24/LinuxAdmin.html>
Unlikely I will really ever need traffic shaping on my home network, yet it makes an interesting exercise for the curious.

Go to Appendix: How to Upgrade the Kernel

Back to Main Menu

LINUX NEWBIE ADMINISTRATOR GUIDE

ver. 0.120 2001-03-16 by Stan and Peter Klimas

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

Copyright (c) <1999,2000,2001> 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](#).

Quick navigation:

Appendix.

Kernel Upgrade - version 1.1.0

October 11th 1999

All contents of this page were written by Alesh Mustar.

Distributed under the General Public Licence <http://www.gnu.org/copyleft/gpl.html>. Your feedback, comments, corrections, and improvements are appreciated. Comment specific to this page:

alesh@jpdesign.net

Contents:

- A.1 Introduction [p 182]
 - A.2 What is kernel? [p 183]
 - A.3 Why upgrade? [p 183]
 - A.4 Where can I get the new kernel? [p 183]
 - A.5 Should I download full source or the patch only? [p 188]
 - A.6 Unpacking the download file [p 188]
 - A.6.1 Unpacking the patch [p 188]
 - A.6.2 Unpacking the full source [p 189]
 - A.7 Compiling new kernel [p 189]
 - A.7.1 Using the patch [p 189]
 - A.7.2 Using the full source [p 190]
 - A.8 Installing the new kernel [p 190]
 - A.9 What about next kernel upgrade? [p 193]
 - A.10 Notice for Red Hat users [p 193]
 - A.11 About the author [p 193]
 - A.12 Changes [p ??]
-

A. How to upgrade your kernel?

A.1 Introduction

This document was written to help the new users of Linux upgrade their kernel.

I used various sources for writing this document. From the official HOW-TOs (

<http://metalab.unc.edu/LDP/HOWTO/>) to the Linux Knowledge Base and other sources.

A.2 What is kernel?

The kernel acts as a mediator for your programs and your hardware. First, it does (or arranges for) the memory management for all of the running programs (processes), and makes sure that they all get a fair (or unfair, if you please) share of the processor's cycles. In addition, it provides a nice, fairly portable interface for programs to talk to your hardware.

There is certainly more to the kernel's operation than this, but these basic functions are the most important to know.

A.3 Why upgrade?

There are several reasons to upgrade your kernel. Most of the reasons are that you perhaps need a new device driver for your new or old hardware (the previous device driver could have had a bug somewhere) or you need to upgrade due to a bug in the kernel (security one or a non-security one).

Usually new kernels run faster than the old ones and are more stable and reliable.

A.4 Where can I get the new kernel?

First you need to get the new kernel. Load a browser (any kind of) and point it to <http://www.kernel.org/>. There you can find a list of mirrors. Mirrors are sites with the (mostly) same information as the main site. Since the main kernel.org site is heavily overloaded with downloads choose a mirror site closest to you or even the one located in your country. For example: to get the new kernel from Norway point your browser to the .no kernel archive => <http://www.no.kernel.org/>. Notice the .no after the www string. Lots of countries have mirrors for the kernel archive.

Once the mirror page loads you will find information on that page which looks like this:

<= start of example page=>

The Linux Kernel Archives mirror at Sunsite.uio.no

All transfers are logged. If you don't like this policy please disconnect now.

Welcome to the Linux Kernel Archives. This is an official mirror site for the Linux kernel source. Please see <http://www.kernel.org/mirrors/> for information about how to connect to an other kernel-mirror site.

Protocol	URL
HTTP	http://linux-kernel.uio.no/pub/
FTP	ftp://linux-kernel.uio.no/pub

What is Linux?Linux is a Unix clone written from scratch by Linus Torvalds with assistance from a loosely-knit team of hackers across the Net. It aims towards POSIX compliance.

...

<= end of example page =>

Now choose a protocol which suits you better to transfer the files. Lets say we will choose HTTP protocol (same protocol as your WWW browser uses).

Your browser will display a page with a directory structure, perhaps similar to this:

<= start of example page =>

Index of /pub

Name	Last modified	Size	Description
------	---------------	------	-------------

[DIR] Parent Directory 28-Dec-98 12:06 -

[DIR] linux/ 31-Dec-98 09:57 -

[] ls-lR 29-Mar-99 14:56 262k

[DIR] software/ 31-Dec-98 18:25 -

<= end of example page =>

Note the *linux* directory. Point your browser at it. A page like this will show:

<= start of example page =>

Index of /pub/linux

Name	Last modified	Size	Description
------	---------------	------	-------------

[DIR] Parent Directory 29-Mar-99 14:56 -

[DIR] daemons/ 12-Feb-99 01:19 -

[DIR] devel/ 18-Dec-98 19:34 -

[DIR] distributions/ 18-Dec-98 19:34 -

[DIR] docs/ 31-Jan-99 04:01 -

[DIR] kernel/ 18-Mar-99 21:55 -

[DIR] libs/ 18-Dec-98 19:35 -

[DIR] utils/ 31-Jan-99 04:01 -

<= end of example page =>

This site contains lots of things to download but currently we are only interested in the kernel.

Point your browser at the *kernel* directory.

Something like this should show:

<= start of example page =>

Index of /pub/linux/kernel

Name Last modified Size Description

[DIR] Parent Directory 31-Dec-98 09:57 -
[] COPYING 13-Mar-94 00:00 18k
[] CREDITS 16-Sep-96 00:00 36k
[DIR] Historic/ 18-Dec-98 20:55 -
[DIR] SillySounds/ 18-Dec-96 09:45 -
[DIR] alan/ 18-Mar-99 21:55 -
[DIR] davem/ 29-Jan-99 04:05 -
[DIR] hpa/ 19-Dec-98 02:16 -
[DIR] people/ 18-Mar-99 21:55 -
[DIR] testing/ 27-Mar-99 00:55 -
[DIR] v1.0/ 19-Dec-98 02:19 -
[DIR] v1.1/ 19-Dec-98 03:51 -
[DIR] v1.2/ 19-Dec-98 04:36 -
[DIR] v1.3/ 19-Dec-98 13:21 -
[DIR] v2.0/ 11-Feb-99 16:48 -
[DIR] v2.1/ 11-Feb-99 16:45 -
[DIR] v2.2/ 24-Mar-99 00:27 -
[DIR] whawes/ 18-Dec-98 19:34 -

Linux kernel release 2.0.xx

These are the release notes for linux version 2.0. Read them carefully, as they tell you what this is all about, explain how to install the kernel, and what to do if something goes wrong.

...
....

<= end of example page =>

For our example we will take in consideration that while writting this the kernel version 2.2.x series is the stable version. That's why we choose the v2.2/ directory on the server.
How do you know which is the latest version? At the bottom of the <http://www.kernel.org/> page you have a text like this:

<= *start of example page* =>

The latest stable version of the Linux kernel is:	2.2.12
The latest beta version of the Linux kernel is:	2.3.20
The latest prepatch (alpha) version <i>appears</i> to be:	none

<= *end of example page* =>

That is your information to find out which is the latest stable kernel and which one is also recommended to use. Also as you will see you can usually see a file with the name like *LATEST-IS-2.2.12* at the top of the directory listing which tells you the latest version.

Once that we are in the *v2.2/* directory on our server we will see something like this:

<= start of example page =>

Index of /pub/linux/kernel/v2.2

Name Last modified Size Description

```
[DIR] Parent Directory 16-Aug-99 09:13 -
[ ] LATEST-IS-2.2.12 26-Aug-99 02:45 0k
[ ] linux-2.2.0.tar.bz2 26-Jan-99 02:41 10.1M
[ ] linux-2.2.0.tar.bz2.s..26-Jan-99 02:41 1k
[COMP] linux-2.2.0.tar.gz 26-Jan-99 02:41 12.5M
[ ] linux-2.2.0.tar.gz.sign26-Jan-99 02:41 1k
[ ] linux-2.2.1.tar.bz2 28-Jan-99 21:56 10.1M
[ ] linux-2.2.1.tar.bz2.s..28-Jan-99 21:56 1k
[COMP] linux-2.2.1.tar.gz 28-Jan-99 21:56 12.5M
[ ] linux-2.2.1.tar.gz.sign28-Jan-99 21:56 1k
[ ] linux-2.2.10.tar.bz2 14-Jun-99 07:33 10.8M
[ ] linux-2.2.10.tar.bz2...14-Jun-99 07:33 1k
[COMP] linux-2.2.10.tar.gz 14-Jun-99 07:33 13.3M
[ ] linux-2.2.10.tar.gz.s..14-Jun-99 07:33 1k
[ ] linux-2.2.11.tar.bz2 10-Aug-99 01:03 11.2M
[ ] linux-2.2.11.tar.bz2...10-Aug-99 01:03 1k
[COMP] linux-2.2.11.tar.gz 10-Aug-99 01:03 13.8M
[ ] linux-2.2.11.tar.gz.s..10-Aug-99 01:03 1k
[ ] linux-2.2.12.tar.bz2 26-Aug-99 02:45 11.5M
[ ] linux-2.2.12.tar.bz2...26-Aug-99 02:45 1k
[COMP] linux-2.2.12.tar.gz 26-Aug-99 02:45 14.2M
[ ] linux-2.2.12.tar.gz.s..26-Aug-99 02:45 1k
[ ] linux-2.2.2.tar.bz2 23-Feb-99 03:58 10.1M
[ ] linux-2.2.2.tar.bz2.s..23-Feb-99 03:58 1k
[COMP] linux-2.2.2.tar.gz 23-Feb-99 03:58 12.5M
[ ] linux-2.2.2.tar.gz.sign23-Feb-99 03:58 1k
[ ] linux-2.2.3.tar.bz2 09-Mar-99 01:42 10.2M
[ ] linux-2.2.3.tar.bz2.s..09-Mar-99 01:42 1k
[COMP] linux-2.2.3.tar.gz 09-Mar-99 01:42 12.6M
[ ] linux-2.2.3.tar.gz.sign09-Mar-99 01:42 1k
[ ] linux-2.2.4.tar.bz2 23-Mar-99 23:33 10.4M
[ ] linux-2.2.4.tar.bz2.s..23-Mar-99 23:33 1k
[COMP] linux-2.2.4.tar.gz 23-Mar-99 23:33 12.8M
[ ] linux-2.2.4.tar.gz.sign23-Mar-99 23:33 1k
[ ] linux-2.2.5.tar.bz2 29-Mar-99 08:54 10.4M
[ ] linux-2.2.5.tar.bz2.s..29-Mar-99 08:54 1k
[COMP] linux-2.2.5.tar.gz 29-Mar-99 08:54 12.9M
[ ] linux-2.2.5.tar.gz.sign29-Mar-99 08:54 1k
```

[] linux-2.2.6.tar.bz2 16-Apr-99 23:46 10.5M
 [] linux-2.2.6.tar.bz2.s..16-Apr-99 23:46 1k
 [CMP] linux-2.2.6.tar.gz 16-Apr-99 23:46 13.0M
 [] linux-2.2.6.tar.gz.sign16-Apr-99 23:46 1k
 [] linux-2.2.7.tar.bz2 28-Apr-99 20:42 10.6M
 [] linux-2.2.7.tar.bz2.s..28-Apr-99 20:42 1k
 [CMP] linux-2.2.7.tar.gz 28-Apr-99 20:42 13.0M
 [] linux-2.2.7.tar.gz.sign28-Apr-99 20:42 1k
 [] linux-2.2.8.tar.bz2 11-May-99 21:59 10.7M
 [] linux-2.2.8.tar.bz2.s..11-May-99 21:59 1k
 [CMP] linux-2.2.8.tar.gz 11-May-99 21:59 13.2M
 [] linux-2.2.8.tar.gz.sign11-May-99 21:59 1k
 [] linux-2.2.9.tar.bz2 14-May-99 01:54 10.7M
 [] linux-2.2.9.tar.bz2.s..14-May-99 01:54 1k
 [DIR] Parent Directory 16-Aug-99 09:13 -
 [] LATEST-IS-2.2.12 26-Aug-99 02:45 0k
 [] linux-2.2.0.tar.bz2 26-Jan-99 02:41 10.1M
 [] linux-2.2.0.tar.bz2.s..26-Jan-99 02:41 1k
 [CMP] linux-2.2.0.tar.gz 26-Jan-99 02:41 12.5M
 [] linux-2.2.0.tar.gz.sign26-Jan-99 02:41 1k
 [] linux-2.2.1.tar.bz2 28-Jan-99 21:56 10.1M
 [] linux-2.2.1.tar.bz2.s..28-Jan-99 21:56 1k
 [CMP] linux-2.2.1.tar.gz 28-Jan-99 21:56 12.5M
 [] linux-2.2.1.tar.gz.sign28-Jan-99 21:56 1k
 [] linux-2.2.10.tar.bz2 14-Jun-99 07:33 10.8M
 [] linux-2.2.10.tar.bz2...14-Jun-99 07:33 1k
 [CMP] linux-2.2.10.tar.gz 14-Jun-99 07:33 13.3M
 [] linux-2.2.10.tar.gz.s..14-Jun-99 07:33 1k
 [] linux-2.2.11.tar.bz2 10-Aug-99 01:03 11.2M
 [] linux-2.2.11.tar.bz2...10-Aug-99 01:03 1k
 [CMP] linux-2.2.11.tar.gz 10-Aug-99 01:03 13.8M
 [] linux-2.2.11.tar.gz.s..10-Aug-99 01:03 1k
 [] linux-2.2.12.tar.bz2 26-Aug-99 02:45 11.5M
 [] linux-2.2.12.tar.bz2...26-Aug-99 02:45 1k
 [CMP] linux-2.2.12.tar.gz 26-Aug-99 02:45 14.2M
 [] linux-2.2.12.tar.gz.s..26-Aug-99 02:45 1k
 [] linux-2.2.2.tar.bz2 23-Feb-99 03:58 10.1M
 [] linux-2.2.2.tar.bz2.s..23-Feb-99 03:58 1k
 [CMP] linux-2.2.2.tar.gz 23-Feb-99 03:58 12.5M
 [] linux-2.2.2.tar.gz.sign23-Feb-99 03:58 1k
 [] linux-2.2.3.tar.bz2 09-Mar-99 01:42 10.2M
 [] linux-2.2.3.tar.bz2.s..09-Mar-99 01:42 1k
 [CMP] linux-2.2.3.tar.gz 09-Mar-99 01:42 12.6M
 [] linux-2.2.3.tar.gz.sign09-Mar-99 01:42 1k
 [] linux-2.2.4.tar.bz2 23-Mar-99 23:33 10.4M
 [] linux-2.2.4.tar.bz2.s..23-Mar-99 23:33 1k
 [CMP] linux-2.2.4.tar.gz 23-Mar-99 23:33 12.8M
 [] linux-2.2.4.tar.gz.sign23-Mar-99 23:33 1k
 [] linux-2.2.5.tar.bz2 29-Mar-99 08:54 10.4M
 [] linux-2.2.5.tar.bz2.s..29-Mar-99 08:54 1k

```

[CMP] linux-2.2.5.tar.gz 29-Mar-99 08:54 12.9M
[ ] linux-2.2.5.tar.gz.sign29-Mar-99 08:54 1k
[ ] linux-2.2.6.tar.bz2 16-Apr-99 23:46 10.5M
[ ] linux-2.2.6.tar.bz2.s..16-Apr-99 23:46 1k
[CMP] linux-2.2.6.tar.gz 16-Apr-99 23:46 13.0M
[ ] linux-2.2.6.tar.gz.sign16-Apr-99 23:46 1k
[ ] linux-2.2.7.tar.bz2 28-Apr-99 20:42 10.6M
[ ] linux-2.2.7.tar.bz2.s..28-Apr-99 20:42 1k
[CMP] linux-2.2.7.tar.gz 28-Apr-99 20:42 13.0M
[ ] linux-2.2.7.tar.gz.sign28-Apr-99 20:42 1k
[ ] linux-2.2.8.tar.bz2 11-May-99 21:59 10.7M
[ ] linux-2.2.8.tar.bz2.s..11-May-99 21:59 1k
[CMP] linux-2.2.8.tar.gz 11-May-99 21:59 13.2M
[ ] linux-2.2.8.tar.gz.sign11-May-99 21:59 1k
[ ] linux-2.2.9.tar.bz2 14-May-99 01:54 10.7M
[ ] linux-2.2.9.tar.bz2.s..14-May-99 01:54 1k

```

<= end of example page =>

A.5 Should I download full source or the patch only?

Now you have a choice to download either the patch or the full source.

If you're upgrading from 2.0.x kernel series you will have to download the full source.

If you're upgrading a 2.2.x kernel series you can download only the patch, which is ofcourse much more smaller then the full source. The patch contains only the differences.

If you would like more detailed info on patches and diff take a look at this link:

http://www.linuxpower.org/display_item.phtml?id=101 .

Now grab *your choice* and download it. I highly suggest you download it to */usr/src/* directory.

I suggest downloading the file with the *.tar.gz* extension (if you're downloading the full source).

For patches I suggest *.bz2* extension. Please note that the examples which follow will be based on this decisions (about file types).

A.6 Unpacking the download file

A.6.1 Unpacking the patch

We are assuming that you downloaded the patch into */usr/src/* directory. For our example lets say the patch filename is *patch-2.2.12.bz2*. Before using it we must first unpack it. *.bz2* extension is bunzip2 format. By using the following command we unpack the file:

<= start of example command =>

```
bunzip2 patch-2.2.12.bz2
```

<= end of example command =>

No visual output will be shown. Check what files are located in the directory now. You will notice that the *patch-2.2.12.bz2* file is gone and *patch-2.2.12* is there of a much bigger size.

A.6.2 Unpacking the full source

Again we are assuming that you downloaded the full source into `/usr/src/` directory. For our example lets say the source filename is `linux-2.2.12.tar.gz`.

Before even unpacking it we must first do something else. In `/usr/src/` you have a symbolic link named `linux` which points to for example at `linux-2.0.36`. How can you check that out? Type `ls -al` in the `/usr/src/` directory. You'll see something like this:

<= start of example page =>

```
total 7295
drwxr-xr-x 5 root root 1024 Mar 19 15:01 .
drwxr-xr-x 20 root root 1024 Mar 13 22:52 ..
lrwxrwxrwx 1 root root 11 Mar 15 18:21 linux -> linux-2.0.36
drwxr-xr-x 17 root root 1024 Mar 13 22:52 linux-2.0.36
...
```

<= end of example page =>

Notice the `lrwxrwxrwx 1 root root 11 Mar 15 18:21 linux -> linux-2.0.36` which is the symbolic link pointing to `linux-2.0.36` directory.

If we would unpack the kernel source now, by default it would be unpacked into the `linux` directory overwriting our previous source.

Lets delete the link by typing `rm linux`. It only removes the link, not the directory!

Now lets unpack the source. Type `tar -xvfz linux-2.2.12.tar.gz`. Lots of text will fly by. If you do a `ls -al` now you'll notice the `linux` directory. That is our new kernel source directory.

A.7 Compiling new kernel

A.7.1 Using the patch

If you downloaded the patch you will need to go thru this section.

We are assuming right now that you are located in `/usr/src/` directory. We have unpacked the patch (Section 5.6.1). Now we will move to the kernel directory (assuming `linux`). Type `cd linux`.

Now we are located in the kernel directory. At this point we have a lot of options and questions. Is this your first patch of the kernel? No? Yes? We won't go into details and we will assume that this is your first kernel patch. Now we are ready to patch the kernel.

To do this we will use the patch utility. For more detail and wider information about this I suggest checking out the article from the [www.linuxpower.org](http://www.linuxpower.org/display_item.phtml?id=101) web site, located at this url: http://www.linuxpower.org/display_item.phtml?id=101.

Type `patch -E -p1 < ../patch-2.2.12`

Lots of text will fly by. If it doesn't stop anywhere but only at the end (otherwise it will be askign you a question) everything went ok. If not, then you might have a problem (you already patched a kernel once, or something went really really wrong).

If everything went ok read on. Now it's time to configure the kernel and then compile it. We now have 2 choices. One is to compile the new kernel totally from scratch, the other one is to just re-compile it using the old configuration. Since you used the patch we are assuming that you're gonna compile with the old configuration. If you don't wish to, read section 6.2 where we compile the kernel from scratch and use the steps there. The only difference is in

step 1, where you should type *make menuconfig*.

This is the re-compile of the kernel using the old configuration. Follow these steps:

1. type *make oldconfig*

At this point you could have been asked for new drivers and development ones. Unless you really need them answer no (*n*).

2. type *make dep*

3. type *make clean*

4. type *make zImage* (if this fails use *bzImage*, which stands for *big zImage*)

5. type *make modules* (if you enabled them)

6. type *make modules_install* (if you typed #5)

If everything went ok, we have a new kernel, ready to be installed and used on the system.

A.7.2 Using the full source

If you downloaded the full source you will need to go thru this section.

We are assuming right now that you are located in */usr/src/* directory. We have unpacked the new kernel (Section 5.6.2). Now we will move to the kernel directory (assuming *linux*).

Type *cd linux*.

Now we are located in the kernel directory. Now lets start our fresh compile.

1. type *make menuconfig*

Here you will have to configure your kernel. We won't go into details or explains here since the configuration is different from machine to machine. Go thru the configuration wisely and if you don't know what something is, rather leave it in then putting it out of the configuration.

2. type *make dep*

3. type *make clean*

4. type *make zImage* (if this fails use *bzImage*, which stands for *big zImage*)

5. type *make modules* (if you enabled them)

6. type *make modules_install* (if you typed #5)

If everything went ok, we have a new kernel, ready to be installed and used on the system.

A.8 Installing the new kernel

Installing the new kernel takes a knowledge of editing the *lilo.conf* file located in the */etc/* directory. I will state now that *lilo.conf* deferes from machine to machine and it might not look the same on your machine as it does on my machines.

First of all, we will assume that you are still in the */usr/src/linux/* directory. We will have to copy the *new kernel* and *System.map* to certain location which again is different on some distributions. Lets change directory to the new kernel.

Type *cd /arch/i386/boot/*

Type *ls -al* just ot see what's there. You'll see something like this:

<= start of example page =>

```
total 588
drwxr-xr-x  4 1046   1046    1024 Apr  5 20:54 .
drwxr-xr-x  7 1046   1046    1024 Mar 13 02:39 ..
-rw-r--r--  1 1046   1046   2633 Jan  2 19:27 Makefile
```

```

-rwxr-xr-x 1 root root 544 Apr 5 20:54 bootsect
-rw-r--r-- 1 1046 1046 9536 Jun 24 1998 bootsect.S
-rw-r--r-- 1 root root 1238 Apr 5 20:54 bootsect.o
-rw-r--r-- 1 root root 8293 Apr 5 20:54 bootsect.s
drwxr-xr-x 2 1046 1046 1024 Apr 5 20:54 compressed
-rw-r--r-- 1 1046 1046 904 Jan 3 1995 install.sh
-rwxr-xr-x 1 root root 1428 Apr 5 20:54 setup
-rw-r--r-- 1 1046 1046 20136 Nov 29 02:18 setup.S
-rw-r--r-- 1 root root 3053 Apr 5 20:54 setup.o
-rw-r--r-- 1 root root 24136 Apr 5 20:54 setup.s
drwxr-xr-x 2 1046 1046 1024 Apr 5 20:54 tools
-rw-r--r-- 1 1046 1046 36836 Sep 30 1998 video.S
-rw-r--r-- 1 root root 477285 Apr 5 20:54 zImage

```

<= end of example page =>

If you at section 5.7.x used *make zImage* you should copy the *zImage* file then, otherwise copy the *bzImage* file. We will assume *zImage*.

The option to which we encounter right now is where to copy the kernel. On Red Hat distributions the kernel is located in */boot/* on Slackware in */*. I'm not very familiar with other distributions (Debian, SuSe, etc.). Depending on your distribution make your choice where you will copy the new kernel.

We will make an example for Red Hat.

Type *cp zImage /boot/vmlinuz-2.2.12*

Now change the directory back to */usr/src/linux/*.

Type *cd /usr/src/linux/*.

Now lets copy the new *System.map*. Please note that *System.map* is always located in */boot/*.

Type *cp System.map /boot/System.map-2.2.12*

Now we have to make a new *symbolic link* which points to the new *System.map*.

First change the directory to */boot* by typing *cd /boot*.

Now delete the old symbolic link (note that deleting the symbolic link only deletes the link, not also the file or directory to which it points). Type *rm System.map*.

Now lets create a new symbolic link which points to our new *System.map-2.2.12*. Type *ln -s System.map-2.2.12 System.map*.

All we have to do now is to edit the *lilo.conf* file, add our new kernel in there and run *lilo*.

Lets change directory to */etc* by typing *cd /etc*. With your favourite editor (*vi*, *joe*, *pico*) edit the *lilo.conf* file. For our example we will use *joe*.

You will see something like this in your file :

= start of example page =>

```

boot=/dev/hda
map=/boot/map
install=/boot/boot.b
prompt
timeout=50
image=/boot/vmlinuz-2.0.36-0.7
    label=linux.old

```

```
root=/dev/hda1
read-only
```

<= end of example page =>

Please note that this can and probably is different on your system.

Now we will add the new kernel to our *lilo.conf* file. Copy the part from image to the end and paste it before the current image. You should get something like this:

= start of example page =>

```
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
prompt
timeout=50
image=/boot/vmlinuz-2.0.36-0.7
    label=linux.old
    root=/dev/hda1
    read-only
image=/boot/vmlinuz-2.0.36-0.7
    label=linux.old
    root=/dev/hda1
    read-only
```

<= end of example page =>

Now change the pasted part to the settings and version(s) of your new kernel. For our example (we use the 2.2.4 kernel) it looks like this:

= start of example page =>

```
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
prompt
timeout=50
image=/boot/vmlinuz-2.2.12
    label=linux
    root=/dev/hda1
    read-only
image=/boot/vmlinuz-2.0.36-0.7
    label=linux.old
    root=/dev/hda1
    read-only
```

<= end of example page =>

We changed the label part, the image part, everything else stayed that same.

I won't explain the basics of the *lilo.conf* file, since it's very well described in the *LILO How-To*

.

Now run *lilo* by simply typing that. You should see something like this (more or less exactly this):

= start of example page =>

Added linux *
Added linux.old

<= end of example page =>

You can now reboot your machine to the new kernel.

A.9 What about next kernel upgrade?

Well everything should be the same only one thing not. If you will be patching the kernel you will first have to remove the old patch before applying the new one. This is done by first typing *patch -R -p1 < ../patchfile* (where patchfile is the name of the old patch) and then applying the new patch with the -E switch. Everything else should be the same.

A.10 Notice to Red Hat users

Red Hat users should note this. If you're upgrading your kernel from the 2.0.xx series to 2.2.x series you must read this. The document states similar to this one, step by step how to upgrade certain system specific things like init, samba, etc.. which for such upgrade is necessary. If there are similar issues for other distributions, please let me know. It would be nice if you could point me to a certain document with information about it.

A.11 About the author

My name is Alesh Mustar, I'm 20 years old, living in Slovenia, Europe working as a programmer and system administrator. Beside spending lots of time behind the computer screen(s) I like spending time with my girlfriend Tanja, reading books and driving around with my car. I can be contacted at the email address which is at the top of the page. Comments, suggestions or anything else is more than just welcome.

A.12 Changes

Version 1.1.0 - minor changes, some typo errors fixed and some kernel numbers repairs

Go back to top: [Linux Newbie Administrator Guide \(LNAG\)](#)

Acknowledgments

The Linux Newbie Administrator Guide (LNAG) is hosted FREE OF CHARGE on the SunSite server at Aalborg University, Denmark (<http://sunsite.dk/linux-newbie>). Thanks to Esben Haabendal Soerensen <bart@sunsite.auc.dk>.

Thanks to "Linux Supplies" (<http://www.linsup.com/newbie/>) for hosting our free Australian mirror.

Thanks to Andamooka for hosting LNAG among their great free books.

Thanks to the UMBC (University of Maryland Baltimore County) Linux Users Group who hosted our free USA mirror for a while.

Major help and advice was received from (alphabetical order): Alan W. Irwin <irwin@beluga.phys.uvic.ca>, Benjamin Smith <bens@saber.net>, Barbara Klimas <bklimas@magma.ca>, Ben McCosh <ben@albagroup.com>, Bill Staehle <staehle@netvalue.net>, Bill Unruh <unruh@physics.ubc.ca>, Brian Kelsay <ripdrd6@kcinter.net>, Ding-Hou Lee <dl105@columbia.edu>, Gary <swear@aa.net>, Greg Mizell <GMizell@peek-traffic.com>, Jeff Greenlee <greenj@analogy.com>, jeff covey <jeff.covey@pobox.com>, Juhapekka Tolvanen <juhtolv@st.jyu.fi>, Ken Foskey <waratah@zip.com.au>, and Maria Klimas. Special thanks to the Portland Linux User Group!

Thanks to the several dozen others who sent comments or good word to us. We always appreciate any feedback. We received no flames so far :-). Sorry if we could not answer specific questions you sent us on your particularly annoying problem :((

Thanks to the few thousands who developed GNU/Linux. Should I mention Richard Stallman and Linus Torvalds by name?

Other Matters

You can see that this guide was written by newbies for newbies. It should never be considered an authoritative source on any subject--there are much more exhaustive docs, typically more difficult to read too :-), most of them available right on your Linux system in the directory `/usr/doc` . Please e-mail us immediately if you spot a mistake that can confuse or mislead a new Linux user--this work is in progress and the current version may contain such mistakes. Don't use this guide if your life was to depend on it!

If you wanted to contribute a part (on a useful Linux topic of your choice), we would like to hear from you--we will be happy to include a part with your name, etc. The only condition is that your advice be easy to follow by newbies like us.

If you create your own mirror, pls make sure to update it at least from time to time. LNAG is under development and inaccuracies are found and corrected on regular basis.

Hope this helps. Best regards, Stan and Peter Klimas.

Our Rewards and Testimonials

The Linux Newbie Administrator Guide (LNAG) has been on-line since February, 1999. Our web pages were visited by (estimated) 0.5 million readers, and linked to more than 3000 www sites (most perishable though). We write the "guide" for a joy of it, while we are learning Linux, which is reflected in the contents of the guide.

We are proud that our Guide was:

- o Awarded "Link van de dag" ("Link of the Day") by Dutch NL.linux.org in March 1999: "Alles wat je als newbie zou moeten weten." http://www.nl.linux.org/index/link_van_de_dag.html

- o Mentioned twice by "Linux Today", the best source of Linux News: Apr 14th 1999: <http://linuxtoday.com/stories/4985.html>, and Oct 25, 1999: <http://linuxtoday.com/stories/11525.html>.

- o Mentioned twice by the prestigious Linux Weekly News: <http://lwn.net/1999/0422/announce.phtml> and <http://lwn.net/2000/0803/a/lnag.php3>

- o Mentioned by LinuxWord Czech digest, May, 1999: http://online.idg.cz:8080/cw_news.nsf/4f682153745289cdc12566b80055b9df/999e7c8725ec0848c1256767004ed0dc
Přiručka administrátora pro začínající uživatele. Pro začínající uživatele Linuxu přibyl velice zajímavý dokument. Jedná se o příručku "Linux Newbie Administrator", která je psána linuxovým začátečníkem. Autor sám píše, že Linux používá pouze devět měsíců, ale to je možná v tomto případě výhoda - je jisti blíž ke potenciálnímu členovi než některý linuxový guru. Příručka se nachází na webové serveru (<http://sunsite.auc.dk/linux-newbie/>) a obsahuje poměrně jednoduché návody na činnosti, které jako správce vašeho linuxového systému budete nucen dělat. Související URL: sunsite.auc.dk/linux-newbie/

- o Awarded "Best of Linux Winner" by the popular portal "Dave Central" (subsidiary of Andover), August 20, 1999: http://linux.davecentral.com/bol_19990820.html: "Best of Linux Archive Getting Linux. Andover. Best of Linux Winner August 20, 1999. Newbie Administrator Guide Tutorials - Getting Started:

"There are two ways to approach Linux use. One is strictly as a user; you can work away, and leave it to a guru friend to fix problems and add new features to your system.

"The do-it-yourself types in the crowd may be satisfied with this for awhile, but eventually, they'll want to get under the hood themselves. The Newbie Administrator Guide is geared toward such folks, and provides a gentle introduction to the craft of maintaining a Linux system. There are other comprehensive works on the subject, such as The Linux System Administrators' Guide, but they're mostly geared toward professional sysadmins who deal with lots of machines with lots of users.

"The Newbie Admin Guide (NAG hereafter) is of more immediate use to the home user who has decided to take on the task of caring and nurturing for her machine herself. Like a lot of great tutorials, the NAG started with users, the Klimas family in this case, who went through the struggle to set things up and said, "Why should other people go through the same work when we've already done it?" So, they started making notes about what they'd done, and put them on the Web for everyone's benefit.

"The NAG assumes you're starting from ground zero, offering explanations of what Linux is, why you'd want to use it, and where to get it, but moves on to clear and detailed explanations of complex issues you'll face when you try to configure and troubleshoot your system. If you've flipped through book after book but found yourself out of your depth and longing for simple, direct explanations, give the NAG a try. "

o Recommended by popular <http://www.linuxlinks.com/Beginners/>: "Linux Newbie Administrator Guide. Simple answers to problems that Linux newbies frequently encounter while setting up/using/administrating their computer or home network."

o Linked by German <http://www.linuxstart.de/einsteiger/index.htm>: "Linux Newbie Administrator Guide: Von der Installation, Wartung, Tastaturbefehle bis zur Kernel-Rekompilierung."

o Recommended by a Danish linuxer:

http://www.sslug.dk/emailarkiv/novice/1999_04/msg00013.html To: sslug-novice@sslug.dk Subject: Newbie site! From: "Thomas Mørch" <thm@eci.dk> Date: Wed, 14 Apr 1999 10:16:45 +0200 Delivered-To: mailing list sslug-novice@sslug.dk Mailing-List: contact sslug-novice-help@sslug.dk; run by ezmlm. Newsgroups: sslug.novice. Organization: SSLUG. Reply-to: sslug-novice@sslug.dk. Hejsa Så lige denne site, der blev annonceret på njlug's liste.. Tænkte at den måske kunne gøre nytte her :) <http://sunsite.auc.dk/linux-newbie/> - Thomas.

o Recommended by Brazilian: <http://www.dicas-l.unicamp.br/dicas-l/20000809.shtml> [...]Eu já li e imprimir este manual e realmente é uma excelente fonte de consulta, inclusive já tendo sido divulgado anteriormente nesta lista. O manual é gratuito e licenciado sob a GPL, ou seja, com total liberdade de uso, modificação e distribuição. Para os interessados, o endereço é <http://sunsite.auc.dk/linux-newbie/>

o Reviewed by Italian general interest portal "100-Links":

<http://www.100links.com/archivio/990322.html>:

Una guida per chi è proprio agli inizi: la primissima parte è dedicata agli indecisi (per vedere se conviene o meno passare al Linux), dopodiché si arriva alle istruzioni per l'installazione, all'elenco delle principali risorse presenti sulla Rete e ad una pratica lista dei comandi più diffusi, che almeno all'inizio rischiano di essere difficili da ricordare. <http://sunsite.auc.dk/linux-newbie/>

o Pointed to by linux.com, the top Linux web portal: <http://www.linux.com/firststep/>

o Mentioned by Polish newssite "linuxnews.pl": <http://linuxnews.pl/news.html?id=12247>: [...] jeden z najwiêkszych darmowych podrêczników Linuksa. [Jeden] z najpe³niejszych ¼ród³ informacji o Linuksie dla pocz±tkuj±cych. Dystrybuowany na zasadach licencji GPL, dwustustronicowy podrêcznik dostêpny jest w wielu jêzykach, w tym równie¿ po polsku [...]

o Pointed to by the "Linux Documentation Project" website, the mother of all Linux documentation: <http://www.linuxdoc.org/links/links.html>

Please let us know if you write a word on our pages or make a link or whatever--we will be happy to mention you here.

Changes

Feb. , 2001. Added: How to disable ping for security. How to increase the max number of open files. How to use file redirection with the licq fifo file. How to use kernel SysRq key combinations. How can the linux sourcecode be useful for a newbie. How are directory permissions different from those for regular files. The LNAG html files are now optimized using htmlclean.

February 03, 2001 - version 0.105. This is the 105th increamental update. The zero in front means that the guide is still not finished. The Licence was changed from GPL to modified Open Contents Licence (OCL). Minor updates and cosmetics all over the guide. Change history and the description of LNAG sources were included.

February 1999 - version 0.01 released under the terms of the General Public Licence (GPL)

LNAG Sources

The Linux Newbie Administrator Guide (LNAG) is written in HTML using many editors. Before posting, the heading are inserted to the source files for consistency of the versioning and release date, and the postscript, pdf, and zip files are updated using a customized bash script with calls to the utilities: sed, htmlclean, html2ps, ps2pdf, and zip. Email us if you needed the original sources or the script.

LICENCE

The Linux Newbie Administrator Guide (LNAG) ("The Guide") is distributed under the Open Content Licence (<http://opencontent.org/openpub/>) with the following addition:

THE GUIDE IS BEING INCREMENTALLY UPDATED. THEREFORE, TO PROVIDE THE BEST VALUE TO OUR READERS, UNLESS THE GUIDE HAS BEEN SUBSTANTIALLY MODIFIED BY AUTHOR(S) OTHER THAN THE ORIGINAL GUIDE AUTHORS, ANY DISTRIBUTOR SHALL DISTRIBUTE A REASONABLY RECENT VERSION OF THE GUIDE, I.E., THE MOST RECENT VERSION OR A VERSION NOT OLDER THEN ONE YEAR ON THE DATE OF WEB SERVING, CD WRITING, OR HARD COPY PRINTING. THE MOST RECENT VERSION OF THE GUIDE IS AVAILABLE AT [HTTP://SUNSITE.DK/LINUX-NEWBIE](http://SUNSITE.DK/LINUX-NEWBIE).

THIS LICENCE MEANS THAT ANY PUBLICLY ACCESSIBLE MIRROR MUST BE UPDATED AT LEAST ONCE A YEAR, IF A NEWER VERSION OF THE GUIDE IS AVAILABLE. PLEASE DO NOT CREATE A MIRROR IF YOU DO NOT INTEND TO UPDATE IT.

THE MAINTAINER OF THE GUIDE CAN BE CONTACTED BY EMAIL:
LINUX_NAG@CANADA.COM