Chapter 1

Introduction to Linux

Linux is quite possibly the most important achievement of free software since the original *Space War*, or, more recently, Emacs. This book will be your guide to its shifting and many-faceted world. Linux has developed into the operating system for businesses, education, and personal productivity. It is no longer just for UNIX wizards who sit for hours in front of the glowing console (although we assure you that quite a number of users still fall into this category). This book will help you get the most out of it.

Linux (pronounced with a short *i*, as in *LIH-nucks*) is a clone of the UNIX operating system that runs on Intel 80386 and 80486 computers. It supports a wide range of software, from TEX to X Windows to the GNU C/C++ compiler to TCP/IP. It's a versatile, bona fide implementation of UNIX, freely distributed by the terms of the GNU General Public License (see Appendix E).

Linux can turn any 386 or 486 PC into a workstation. It will give you the full power of UNIX at your fingertips. Businesses are installing Linux on entire networks of machines, using the operating system to manage financial and hospital records, a distributed user computing environment, telecommunications, and more. Universities worldwide are using Linux for teaching courses on operating systems programming and design. And, of course, computing enthusiasts everywhere are using Linux at home, for programming, productivity, and all-around hacking.

What makes Linux so different is that it is a *free* implementation of UNIX. It was and still is developed by a group of volunteers, primarily on the Internet, exchanging code, reporting bugs, and fixing problems in an open-ended environment. Anyone is welcome to join in the Linux development effort: all it takes is interest in hacking a free UNIX clone and some kind of programming know-how. The book that you hold in your hands is your tour guide.

1.1 About This Book

This book is an installation and entry-level guide to the Linux system. The purpose is to get new users up and running with the system by consolodating as much important material as possible into one book. Instead of covering many of the volatile technical details, those things which tend to change with rapid development, we give you enough background to find out more on your own.

Linux is not difficult to install and use. However, as with any implementation of UNIX, there is often some black magic involved to get everything working correctly. We hope that this book will get you on the Linux tourbus and show you how groovy this operating system can be.

In this book, we cover the following topics.

- What is Linux? The design and philosophy of this unique operating system, and what it can do for you.
- All of the details of what is needed to run Linux, including suggestions on what kind of hardware configuration is recommended for a complete system.
- How to obtain and install Linux. There are many distributions of the Linux software. We present a general discussion of Linux software distributions, how to obtain them, and generic instructions for installing the software (which should be applicable to any distribution).
- A brief introductory UNIX tutorial, for those users who have never had experience with UNIX before. This tutorial should, hopefully, provide enough material for complete novices to have enough basic know-how to find their way around the system.
- An introduction to systems administration with Linux. This covers the most important tasks that new Linux administrators will need to be familiar with, such as creating users, managing filesystems, and so forth.
- A cursory overview of advanced features of Linux, such as the X Window System, networking with TCP/IP and SLIP, and the setup of electronic mail and news systems.

This book is for the personal computer user wishing to get started with Linux. We don't assume previous UNIX experience, but do expect novices to refer to other materials along the way. For those unfamiliar with UNIX, a list of useful sources of information is given in Appendix A. In general, this book is meant to be read along with another book on basic UNIX concepts.

1.2 A Brief History of Linux

UNIX is one of the most popular operating systems worldwide because of its large support base and distribution. It was originally developed as a multitasking system for minicomputers and mainframes in the mid-1970's, but has since grown to become one of the most widely used operating systems anywhere, despite its sometimes confusing interface and lack of central standardization.

The real reason for UNIX's popularity? Many hackers feel that UNIX is the Right Thing—the One True Operating System. Hence, the development of Linux by an expanding group of UNIX hackers who want to get their hands dirty with their own system.

Versions of UNIX exist for many systems—ranging from personal computers to supercomputers such as the Cray Y-MP. Most versions of UNIX for personal computers are quite expensive and cumbersome. At the time of this writing, a one-machine version of AT&T's System V for the 386 runs at about US$1500.

Linux is a freely distributable version of UNIX developed primarily by Linus Torvalds [1] at the University of Helsinki in Finland. Linux was developed with the help of many UNIX programmers and wizards across the Internet, allowing anyone with enough know-how and gumption the ability to develop and change the system. The Linux kernel uses no code from AT&T or any other proprietary source, and much of the software available for Linux is developed by the GNU project at the Free Software Foundation in Cambridge, Massachusetts. However, programmers all over the world have contributed to the growing pool of Linux software.

Linux was originally developed as a hobby project by Linus Torvalds. It was inspired by Minix, a small UNIX system developed by Andy Tanenbaum, and the first discussions about Linux were on the USENET newsgroup `comp.os.minix`. These discussions were concerned mostly with the development of a small, academic UNIX system for Minix users who wanted more.

The very early development of Linux was mostly dealing with the task-switching features of the 80386 protected-mode interface, all written in assembly code. Linus writes,

> "After that it was plain sailing: hairy coding still, but I had some devices, and debugging was easier. I started using C at this stage, and it certainly speeds up developement. This is also when I start to get serious about my megalomaniac ideas to make 'a better Minix than Minix'. I was hoping I'd be able to recompile `gcc` under Linux some day...
>
> "Two months for basic setup, but then only slightly longer until I had a disk-driver (seriously buggy, but it happened to work on my machine) and a small filesystem. That was about when I made 0.01 available [around late August of 1991]: it wasn't pretty, it had no floppy driver, and it couldn't do much anything. I don't think anybody ever compiled that version. But by then I was hooked, and didn't want to stop until I could chuck out Minix."

No announcement was ever made for Linux version 0.01. The 0.01 sources weren't even executable: they contained only the bare rudiments of the kernel source, and assumed that you had access to a Minix machine to compile and play with them.

On 5 October 1991, Linus announced the first "official" version of Linux, version 0.02. At this point, Linus was able to run `bash` (the GNU Bourne Again Shell) and `gcc` (the GNU C compiler), but not very much else was working. Again, this was intended as a hacker's system. The primary focus was kernel development—none of the issues of user support, documentation, distribution, and so on had even been addressed. Today, the Linux community still seems to treat these ergonomic issues as secondary to the "real programming"—kernel development.

Linus wrote in `comp.os.minix`,

> "Do you pine for the nice days of Minix-1.1, when men were men and wrote their own device drivers? Are you without a nice project and just dying to cut your teeth on a OS you can try to modify for your needs? Are you finding it frustrating when everything

[1] `torvalds@kruuna.helsinki.fi`.

> works on Minix? No more all-nighters to get a nifty program working? Then this post might be just for you.
>
> "As I mentioned a month ago, I'm working on a free version of a Minix-lookalike for AT-386 computers. It has finally reached the stage where it's even usable (though may not be depending on what you want), and I am willing to put out the sources for wider distribution. It is just version 0.02... but I've successfully run `bash`, `gcc`, `gnu-make`, `gnu-sed`, `compress`, etc. under it."

After version 0.03, Linus bumped the version number up to 0.10, as more people started to work on the system. After several further revisions, Linus increased the version number to 0.95, to reflect his expectation that the system was ready for an "official" release very soon. (Generally, software is not assigned the version number 1.0 until it is theoretically complete or bug-free.) This was in March of 1992. Almost a year and a half later, in late December of 1993, the Linux kernel was still at version 0.99.pl14—asymptotically approaching 1.0. Some felt that version 1.0 would never make it.

Today, Linux is a complete UNIX, capable of running X Windows, TCP/IP, Emacs, UUCP, mail and news software, you name it. Almost all of the major free software packages have been ported to Linux, and commercial software is becoming available. Much more hardware is supported than in original versions of the kernel. Many people have executed benchmarks on 80486 Linux systems and found them comparable with mid-range workstations from Sun Microsystems and Digital Equipment Corporation. Who would have ever guessed that this "little" UNIX clone would have grown up to take on the entire world of personal computing?

1.3 System Features

Linux supports most of the features found in other implementations of UNIX, plus quite a few that aren't found elsewhere. This section is a nickel tour of the Linux kernel features.

Linux is a complete multitasking, multiuser operating system (just like all other versions of UNIX). This means that many users can be logged into the same machine at once, running multiple programs simultaneously.

The Linux system is mostly compatible with a number of UNIX standards (inasmuch as UNIX has standards) on the source level, including IEEE POSIX.1, System V, and BSD features. It was developed with source portability in mind: therefore, you are most likely to find commonly-used features in the Linux system which are shared across multiple implementations. A great deal of free UNIX software available on the Internet and elsewhere compiles on Linux out of the box. In addition, all source code for the Linux system, including the kernel, device drivers, libraries, user programs, and development tools, is freely distributable.

Other specific internal features of Linux include POSIX job control (used by shells such as `csh` and `bash`), pseudoterminals (`pty` devices), and support for national or customized keyboards using dynamically-loadable keyboard drivers. Linux also supports **virtual consoles**, which allow you to switch between multiple login sessions from the system console in text mode. Users of the "`screen`" program will find the Linux virtual console implementation familiar.

The kernel is able to emulate 387-FPU instructions itself, so that systems without a math co-processor can run programs that require floating-point math instructions.

Linux supports various filesystem types for storing data. Various filesystems, such as the *ext2fs* filesystem, have been developed specifically for Linux. Other filesystem types, such as the Minix-1 and Xenix filesystems, are also supported. The MS-DOS filesystem has been implemented as well, allowing you to access MS-DOS files directly by mounting an MS-DOS partition or floppy. The ISO 9660 CD-ROM filesystem type, which reads all standard formats of CD-ROMs, is also supported. We'll talk more about filesystems in Chapters 2 and 4.

Linux provides a complete implementation of TCP/IP networking. This includes device drivers for many popular Ethernet cards, SLIP (Serial Line Internet Protocol, allowing you to access a TCP/IP network via a serial connection), PLIP (Parallel Line Internet Protocol), PPP (Point-to-Point Protocol), NFS, and so on. The complete range of TCP/IP clients and services is supported, such as FTP, `telnet`, NNTP, and SMTP. We'll talk more about networking in Chapter 5.

The Linux kernel is developed to use the special protected-mode features of the Intel 80386 and 80486 processors. In particular, Linux makes use of the protected-mode descriptor-based memory management paradigm and many of the other advanced features of these processors. Anyone familiar with 80386 protected-mode programming knows that this chip was designed for a multitasking system such as UNIX (or, actually, Multics). Linux exploits this functionality.

The Linux kernel supports demand-paged loaded executables. That is, only those segments of a program which are actually used are read into memory from disk. Also, copy-on-write pages are shared among executables, meaning that if several instances of a program are running at once, they will share pages in physical memory, reducing overall memory usage.

In order to increase the amount of available memory, Linux also implements disk paging: that is, up to 256 megabytes of "swap space"[2] can be allocated on disk. When the system requires more physical memory, it will swap out inactive pages to disk, thus allowing you to run larger applications and support more users at once. However, swap is no substitute for physical RAM—it is much slower due to drive access latency times.

The kernel also implements a unified memory pool for user programs and disk cache. In this way, all free memory is used for caching, and the cache is reduced when running large programs.

Executables use dynamically linked shared libraries, meaning that executables share common library code in a single library file found on disk, not unlike the SunOS shared library mechanism. This allows executable files to occupy much less space on disk, especially those that use many library functions. There are also statically-linked libraries for those who wish to use object debugging or maintain "complete" executables without the need for shared libraries to be in place. Linux shared libraries are dynamically linked at run-time, allowing the programmer to replace modules of the libraries with their own routines.

To facilitate debugging, the Linux kernel does core dumps for post-mortem analysis. Using a core dump and an executable linked with debugging support, it is possible to determine what caused a program to crash.

[2]Swap space is inappropriately named: entire processes are not swapped, but rather individual pages. Of course, in many cases entire processes will be swapped out, but this is not neccessarily always the case.

1.4 Software Features

In this section, we'll introduce you to many of the software applications available for Linux, and talk about a number of common computing tasks. After all, the most important part of the system is the wide range of software available for it. The fact that most of this software is freely distributable is even more impressive.

1.4.1 Basic commands and utilities

Virtually every utility that you would expect to find on standard imnplementations of UNIX has been ported to Linux. This includes basic commands such as `ls`, `awk`, `tr`, `sed`, `bc`, `more`, and so on. You name it, Linux has it. Therefore, you can expect your familiar working environment on other UNIX systems to be duplicated on Linux. All of the standard commands and utilities are there. (Novice Linux users should see Chapter 3 for an introduction to these basic UNIX commands.)

Many text editors are available, including `vi`, `ex`, `pico`, `jove`, as well as GNU Emacs and variants such as Lucid Emacs (which incorporates extensions for use under X Windows) and `joe`. Whatever text editor you're accustomed to using has more than likely been ported to Linux.

The choice of a text editor is an interesting one. Many UNIX users still use "simple" editors such as `vi` (in fact, the author wrote this book using `vi` under Linux). However, `vi` has many limitations, due to its age, and more modern (and complex) editors such as Emacs are gaining popularity. Emacs supports a complete LISP-based macro language and interpreter, a powerful command syntax, and other fun-filled extensions. Emacs macro packages exist to allow you to read electronic mail and news, edit the contents of directories, and even engage in an artificially intelligent psychotherapy session (indispensible for stressed-out Linux hackers).

One interesting note is that most of the basic Linux utilities are GNU software. These GNU utilities support advanced features not found in the standard versions from BSD or AT&T. For example, GNU's version of the `vi` editor, `elvis`, includes a structured macro language which differs from the original AT&T implementation. However, the GNU utilities strive to remain compatible with their BSD and System V counterparts. Many people consider the GNU versions of these programs superior to the originals.

The most important utility to many users is the **shell**. The shell is a program which reads and executes commands from the user. In addition, many shells provide features such as **job control** (allowing the user to manage several running processes at once—not as Orwellian as it sounds), input and output redirection, and a command language for writing **shell scripts**. A shell script is a file containing a program in the shell command language, analogous to a "batch file" under MS-DOS.

There are many types of shells available for Linux. The most important difference between shells is the command language. For example, the **C Shell** (`csh`) uses a command language somewhat like the C programming language. The classic **Bourne Shell** uses a different command language. One's choice of a shell is often based on the command language that it provides. The shell that you use defines, to some extent, your working environment under Linux.

No matter what shell you're accustomed to, some version of it has probably been ported to Linux. The most popular shell is the GNU Bourne Again Shell (`bash`), a Bourne shell variant which includes many advanced features, such as job control, command history, command and filename completion, an Emacs-like interface for editing the command line, and powerful extensions to the standard Bourne shell language. Another popular shell is `tcsh`, a version of the C Shell with advanced functionality similar to that found in `bash`. Other shells include `zsh`, a small Bourne-like shell; the Korn shell (`ksh`); BSD's `ash`; and `rc`, the Plan 9 shell.

What's so important about these basic utilities? Linux gives you the unique opportunity to tailor a custom system to your needs. For example, if you're the only person who uses your system, and you prefer to exclusively use the `vi` editor, and `bash` as your shell, there's no reason to install other editors or shells. The "do it yourself" attitude is prevalent among Linux hackers and users.

1.4.2 Text processing and word processing

Almost every computer user has a need for some kind of document preparation system. (How many computer enthusiasts do you know who still use pen and paper? Not many, we'll wager.) In the PC world, *word processing* is the norm: it involves editing and manipulating text (often in a "What-You-See-Is-What-You-Get" environment) and producing printed copies of the text, complete with figures, tables, and other garnishes.

In the UNIX world, *text processing* is much more common, which is quite different than the classical concept of word processing. With a text processing system, text is entered by the author using a "typesetting language", which describes how the text should be formatted. Instead of entering the text within a special word processing environment, the source may be modified with any text editor such as `vi` or Emacs. Once the source text (in the typesetting language) is complete, the user formats the text with a separate program, which converts the source to a format suitable for printing. This is somewhat analogous to programming in a language such as C, and "compiling" the document into a printable form.

There are many text processing systems available for Linux. One is `groff`, the GNU version of the classic `nroff` text formatter originally developed by Bell Labs and still used on many UNIX systems worldwide. Another modern text processing system is TeX, developed by Donald Knuth of computer science fame. Dialects of TeX, such as LaTeX, are also available.

Text processors such as TeX and `groff` differ mostly in the syntax of their formatting languages. The choice of one formatting system over another is also based upon what utilities are available to satisfy your needs, as well as personal taste.

For example, some people consider the `groff` formatting language to be a bit obscure, so they use TeX, which is more readable by humans. However, `groff` is capable of producing plain ASCII output, viewable on a terminal, while TeX is intended primarily for output to a printing device. However, various programs exist to produce plain ASCII from TeX-formatted documents, or to convert TeX to `groff`, for example.

Another text processing system is `texinfo`, an extension to TeX used for software documentation by the Free Software Foundation. `texinfo` is capable of producing a printed document, or an onlinebrowsable hypertext "Info" document from a single source file. Info files are the main format of documentation used by GNU software such as Emacs.

Text processors are used widely in the computing community for producing papers, theses, magazine articles, and books (in fact, this book was produced using LaTeX). The ability to process the source language as a plain text file opens the door to many extensions to the text processor itself. Because source documents are not stored in an obscure format, readable only by a particular word processor, programmers are able to write parsers and translators for the formatting language, extending the system.

What does such a formatting language look like? In general, the formatting language source consists mostly of the text itself, along with "control codes" to produce a particular effect, such as changing fonts, setting margins, creating lists, and so on.

As an example, take the following text:

> Mr. Torvalds:
>
> We are very upset with your current plans to implement *post-hypnotic suggestion* in the **Linux** terminal driver code. We feel this way for three reasons:
>
> 1. Planting subliminal messages in the terminal driver is not only immoral, it is a waste of time;
> 2. It has been proven that "post-hypnotic suggestions" are ineffective when used upon unsuspecting UNIX hackers;
> 3. We have already implemented high-voltage electric shocks, as a security measure, in the code for `login`.
>
> We hope you will reconsider.

This text would appear in the LaTeX formatting language as the following:

```
\begin{quote}
Mr. Torvalds:

We are very upset with your current plans to implement {\em post-hypnotic
suggestion\/} in the {\bf Linux} terminal driver code. We feel this
way for three reasons:
\begin{enumerate}
\item Planting subliminal messages in the kernel driver is not only
      immoral, it is a waste of time;
\item It has been proven that ``post-hypnotic suggestions'' are ineffective
      when used upon unsuspecting UNIX hackers;
\item We have already implemented high-voltage electric shocks, as a
      security measure, in the code for {\tt login}.
\end{enumerate}
We hope you will reconsider.
\end{quote}
```

The author enters the above "source" text using any text editor, and generates the formatted output by processing the source with LaTeX. At first glance, the typesetting language may appear to be obscure, but it's actually quite easy to learn. Using a text processing system enforces typographical standards when writing. For example, all enumerated lists within a document will look the same, unless the author modifies the definition of the enumerated list "environment". The primary goal is to allow the author to concentrate on writing the actual text, instead of worrying about typesetting conventions.

WYSIWYG word processors are attractive for many reasons; they provide a powerful (and sometimes complex) visual interface for editing the document. However, this interface is inherently limited to those aspects of text layout which are accessible to the user. For example, many word processors provide a special "format language" for producing complicated expressions such as mathematical formulae. This is identical text processing, albeit on a much smaller scale.

The subtle benefit of text processing is that the system allows you to specify exactly what you mean. Also, text processing systems allow you to edit the source text with any text editor, and the source is easily converted to other formats. The tradeoff for this flexibility and power is the lack of a WYSIWYG interface.

Many users of word processors are used to seeing the formatted text as they edit it. On the other hand, when writing with a text processor, one generally does not worry about how the text will appear when formatted. The writer learns to expect how the text should look from the formatting commands used in the source.

There are programs which allow you to view the formatted document on a graphics display before printing. For example, the `xdvi` program displays a "device independent" file generated by the TeX system under the X Windows environment. Other software applications, such as `xfig`, provide a WYSIWYG graphics interface for drawing figures and diagrams, which are subsequently converted to the text processing language for inclusion in your document.

Admittedly, text processors such as `nroff` were around long before word processing was available. However, many people still prefer to use text processing, because it is more versatile and independent of a graphics environment. In either case, the `idoc` word processor is also available for Linux, and before long we expect to see commercial word processors becoming available as well. If you absolutely don't want to give up word processing for text processing, you can always run MS-DOS, or some other operating system, in addition to Linux.

There are many other text-processing-related utilities available. The powerful METAFONT system, used for designing fonts for TeX, is included with the Linux port of TeX. Other programs include `ispell`, an interactive spell checker and corrector; `makeindex`, used for generating indicies in LaTeX documents; as well as many `groff` and TeX-based macro packages for formatting many types of documents and mathematical texts. Conversion programs to translate between TeX or `groff` source to a myriad of other formats are available.

1.4.3 Programming languages and utilities

Linux provides a complete UNIX programming environment, including all of the standard libraries, programming tools, compilers, and debuggers that you would expect to find on other UNIX systems. Within the UNIX software development world, applications and systems programming is usually done in C or C++. The standard C and C++ compiler for Linux is GNU's `gcc`, which is an advanced, modern compiler supporting many options. It also capable of compiling C++ (including AT&T 3.0 features) as well as Objective-C, another object-oriented dialect of C.

Besides C and C++, many other compiled and interpreted programming languages have been ported to Linux, such as Smalltalk, FORTRAN, Pascal, LISP, Scheme, and Ada (if you're masochistic enough to program in Ada—we're not going to stop you). In addition, various assemblers for writing protected-mode 80386 code are available, as are UNIX hacking favorites such as Perl (the script language to end all script languages) and Tcl/Tk (a shell-like command processing system including support for developing simple X Windows applications).

The advanced `gdb` debugger has been ported, which allows you to step through a program to find bugs, or examine the cause for a crash using a core dump. `gprof`, a profiling utility, will give you performance statistics for your program, letting you know where your program is spending most of its time executing. The Emacs text editor provides an interactive editing and compilation environment for various programming languages. Other tools include GNU `make` and `imake`, used to manage compilation of large applications; and RCS, a system for source locking and revision control.

Linux implements dynamically-linked shared libraries, which allow binaries to be much smaller as the subroutine code is linked at run-time. These DLL libraries also allow the applications programmer to override function definitions with their own code. For example, if a programmer wished to write her own version of the `malloc()` library routine, the linker would use the programmer's new routine instead of the one found in the libraries.

Linux is ideal for developing UNIX applications. It provides a modern programming environment with all of the bells and whistles. Various standards such as POSIX.1 are supported, allowing software written for Linux to be easily ported to other systems. Professional UNIX programmers and system administrators can use Linux to develop software at home, and then transfer the software to UNIX systems at work. This not only can save a great deal of time and money, but will also let you work in the comfort of your own home, on a single-user system.[3] Computer Science students can use Linux to learn UNIX programming and to explore other aspects of the system, such as kernel architechture.

With Linux, not only do you have access to the complete set of libraries and programming utilities, but you also have the complete kernel and library source code at your fingertips. (Just imagine the endless mischief arising there!)

[3] The author uses his Linux system to develop and test X Windows applications at home, which can be directly compiled on workstations elsewhere.

1.4.4 The X Window System

The X Window System is the standard graphics interface for UNIX machines. It is a powerful environment supporting many applications. Using X Windows, the user can have multiple terminal windows on the screen at once, each one containing a different login session. A pointing device such as a mouse is often used with the X interface, although it isn't required.

Many X-specific applications have been written, such as games, graphics utilities, programming and documentation tools, and so on. With Linux and X, your system is a bona fide workstation. Coupled with TCP/IP networking, you can even display X applications running on other machines on your Linux display, as is possible with other systems running X.

The X Window System was originally developed at MIT, and is freely distributable. However, may commercial vendors have distributed proprietary enhancements to the original X Windows software. The version of X Windows available for Linux is known as XFree86, a port of X11R5 made freely distributable for 80386-based UNIX systems such as Linux. XFree86 supports a wide range of video hardware, including VGA, Super VGA, and a number of accelerated video adaptors. This is a complete distribution of the X Windows software, containing the X server itself, many applications and utilities, programming libraries, and documentation.

Standard X applications include `xterm` (a terminal emulator used for most text-based applications within an X window); `xdm` (the X Session Manager, which handles logins); `xclock` (a simple clock display); `xman` (an X-based man page reader), and more. The many X applications available for Linux are too numerous to mention here, but the base XFree86 distribution includes the "standard" applications found in the original MIT release. Many others are available separately, and theoretically any application written for X Windows should compile cleanly under Linux.

The look and feel of the X Windows interface is controlled to a large extent by the **window manager**. This friendly program is in charge of the placement of windows, the user interface for resizing, iconifying, and moving windows, the appearance of window frames, and so on. The standard XFree86 distribution includes `twm`, the classic MIT window manager, although more advanced window managers such as the Open Look Virtual Window Manager (`olvwm`) are available as well. One window manager that is popular among Linux users is `fvwm`. This is a small window manager, requiring less than half of the memory used by `twm`. It provides a 3-D appearance for windows, as well a virtual desktop—if the user moves the mouse to the edge of the screen, the entire desktop is shifted as if the display were much larger than it actually is. `fvwm` is greatly customizable, and allows all functions to be accessed from the keyboard as well as the mouse. Many Linux distributions use `fvwm` as the standard window manager.

The XFree86 distribution contains programming libraries and include files for those wily programmers who wish to develop X applications. Various widget sets, such as Athena, Open Look, and Xaw3D are supported. All of the standard fonts, bitmaps, man pages, and documentation are included. PEX (a programming interface for 3-D graphics) is also supported.

Many X applications programmers use the proprietary Motif widget set for development. Several vendors sell single and multiple-user licenses for a binary version of Motif for Linux. Because Motif itself is relatively expensive, not many Linux users own it. However, binaries statically linked with Motif routines may be freely distributed. Therefore, if you write a program using Motif and wish to distribute it freely, you may provide a binary so that users without Motif can use the program.

The only major caveats with X Windows are the hardware and memory requirements. A 386 with 4 megabytes of RAM is capable of running X, but 8 megabytes or more of physical RAM are needed to use it comfortably. A faster processor is nice to have as well, but having enough physical RAM is much more important. In addition, to achieve really slick video performance, an accelerated video card (such as a local bus S3-chipset card) is strongly recommended. Performance ratings in excess of 140,000 xstones have been acheived with Linux and XFree86. With sufficient hardware, you'll find that running X and Linux is as fast, or faster, than running X on other UNIX workstations.

In Chapter 5 we'll discuss how to install and use X on your system.

1.4.5 Networking

Interested in communicating with the world? Yes? No? Maybe? Linux supports the two primary networking protocols for UNIX systems: **TCP/IP** and **UUCP**. TCP/IP (Transmission Control Protocol/Internet Protocol, for acronym aficionados) is the set of networking paradigms that allow systems all over the world to communicate on a single network known as the Internet. With Linux, TCP/IP, and a connection to the network, you can communicate with users and machines across the Internet via electronic mail, USENET news, file transfers with FTP, and more. There are many Linux systems currently on the Internet.

Most TCP/IP networks use Ethernet as the physical network transport. Linux supports many popular Ethernet cards and interfaces for personal computers, including the D-Link pocket Ethernet adaptor for laptops.

However, because not everyone has an Ethernet drop at home, Linux also supports **SLIP** (Serial Line Internet Protocol), which allows you to connect to the Internet via modem. In order to use SLIP, you'll need to have access to a SLIP server, a machine connected to the network which allows dial-in access. Many businesses and universities provide such SLIP servers. In fact, if your Linux system has an Ethernet connection as well as a modem, you can configure it as a SLIP server for other hosts.

NFS (Network File System) allows your system to seamlessly share files with other machines on the network. FTP (File Transfer Protocol) allows you to transfer files between other machines. Other applications include `sendmail`, a system for sending and receiving electronic mail using the SMTP protocol; NNTP-based electronic news systems such as C-News and INN; `telnet`, `rlogin`, and `rsh`, which allow you to login and execute commands on other machines on the network; and `finger`, which allows you to get information on other Internet users. There are literally tons of TCP/IP-based applications and protocols out there.

The full range of mail and news readers are available for Linux, such as `elm`, `pine`, `rn`, `nn`, and `tin`. Whatever your preference, you can configure your Linux system to send and receive electronic mail and news from all over the world.

If you have experience with TCP/IP applications on other UNIX systems, Linux will be very familiar to you. The system provides a standard socket programming interface, so virtually any program which uses TCP/IP can be ported to Linux. The Linux X server also supports TCP/IP, allowing you to display applications running on other systems on your Linux display.

In Chapter 5 we'll discuss configuration and setup of TCP/IP, including SLIP, for Linux.

UUCP (UNIX-to-UNIX Copy) is an older mechanism used to transfer files, electronic mail, and electronic news between UNIX machines. Classically, UUCP machines connected to each other over the phone lines via modem, but UUCP is able to transport over a TCP/IP network as well. If you do not have access to a TCP/IP network or a SLIP server, you can configure your system to send and receive files and electronic mail using UUCP. See Chapter 5 for more information.

1.4.6 Telecommunications and BBS software

If you have a modem, you will be able to communicate with other machines using one of the telecommunications packages available for Linux. Many people use telecommunications software to access bulletin board systems (BBSs), as well as commercial online services such as Prodigy, CompuServe, and America On-Line. Other people use their modems to connect to a UNIX system at work or school. You can even use your modem and Linux system to send and receive facsimiles. Telecommunications software under Linux is very similar to that found under MS-DOS or other operating systems. Anyone who has ever used a telecommunications package will find the Linux equivalent familiar.

One of the most popular communications packages for Linux is Seyon, an X application providing a customizable, ergonomic interface, with built-in support for various file transfer protocols such as Kermit, ZModem, and so on. Other telecommunications programs include C-Kermit, `pcomm`, and `minicom`. These are similar to communications programs found on other operating systems, and are quite easy to use.

If you do not have access to a SLIP server (see the previous section), you can use `term` to multiplex your serial line. `term` will allow you to open multiple login sessions over the modem connection to a remote machine. `term` will also allow you to redirect X client connections to your local X server, through the serial line, allowing you to display remote X applications on your Linux system. Another software package, KA9Q, implements a similar SLIP-like interface.

Running a bulletin board system (BBS) is a favorite hobby (and means of income) for many people. Linux supports a wide range of BBS software, most of which is more powerful than what is available for other operating systems. With a phone line, a modem, and Linux, you can turn your system into a BBS, providing dial-in access to your system to users worldwide. BBS software for Linux includes XBBS and the UniBoard BBS packages.

Most BBS software locks the user into a menu-based system where only certain functions and applications are available. An alternative to BBS access is full UNIX access, which would allow users to dial into your system and login as a regular user. While this would require a fair amount of maintenance on the part of the system administrator, it can be done, and providing public UNIX access from your Linux system is not difficult to do. Along with a TCP/IP network, you can provide electronic mail and news access to users on your system.

If you do not have access to a TCP/IP network or UUCP feed, Linux will also allow you to communicate with a number of BBS networks, such as FidoNet, with which you can exchange electronic news and mail via the phone line. More information on telecommunications and BBS software under Linux can be found in Chapter 5.

1.4.7 Interfacing with MS-DOS

Various utilities exist to interface with the (somewhat perverse, we admit) world of MS-DOS. The most well-known application is the Linux MS-DOS Emulator, which allows you to run many MS-DOS applications directly from Linux. Although Linux and MS-DOS are completely different operating systems, the 80386 protected-mode environment allows certain tasks to behave as if they were running in 8086-emulation mode, as MS-DOS applications do.

The MS-DOS emulator is still under development, yet many popular applications run under it. Understandably, however, MS-DOS applications which use bizarre or esoteric features of the system may never be supported, because it is only an emulator. For example, you wouldn't expect to be able to run any programs which use 80386 protected-mode features, such as Microsoft Windows (in 386 enhanced mode, that is).

Applications which run successfully under the Linux MS-DOS Emulator include 4DOS (a command interpreter), Foxpro 2.0, Harvard Graphics, MathCad, Stacker 3.1, Turbo Assembler, Turbo C/C++, Turbo Pascal, Microsoft Windows 3.0 (in *real* mode), and WordPerfect 5.1. Standard MS-DOS commands and utilities (such as `PKZIP`, and so on) work with the emulator as well.

The MS-DOS Emulator is meant mostly as an ad hoc solution for those people who need MS-DOS only for a few applications, but use Linux for everything else. It's not meant to be a complete implementation of MS-DOS. Of course, if the Emulator doesn't satisfy your needs, you can always run MS-DOS as well as Linux on the same system. Using the LILO boot loader, you can specify at boot time which operating system to start. Linux can coexist with other operating systems, such as OS/2, as well.

Linux provides a seamless interface for transferring files between Linux and MS-DOS. You can mount an MS-DOS partition or floppy under Linux, and directly access MS-DOS files as you would any other.

Currently under development is a project known as **WINE**—a Microsoft Windows emulator for the X Window System under Linux. Once WINE is complete, users will be able to run MS-Windows applications directly from Linux. This is similar to the proprietary WABI Windows emulator from Sun Microsystems. At the time of this writing, WINE is still in the early stages of development, but the outlook is good.

In Chapter 5 we'll talk about the MS-DOS tools available for Linux.

1.4.8 Other applications

A host of miscellany is available for Linux, as one would expect from such a hodgepodge operating system. Linux's primary focus is currently for personal UNIX computing, but this is rapidly changing. Business and scientific software is expanding, and commercial software vendors are beginning to contribute to the growing pool of applications.

Several relational databases are available for Linux, including Postgres, Ingres, and Mbase. These are full-featured, professional client/server database applications similar to those found on other UNIX platforms. `/rdb`, a commercial database system, is available as well.

Scientific computing applications include FELT (a finite element analysis tool); `gnuplot` (a plotting and data analysis application); Octave (a symbolic mathematics package, similar to MATLAB); `xspread` (a spreadsheet calculator); `xfractint`, an X-based port of the popular Fractint fractal generator; `xlispstat` (a statistics package), and more. Other applications include Spice (a circuit design and analysis tool) and Khoros (an image/digital signal processing and visualization system).

Of course, there are many more such applications which have been, and can be, ported to run on Linux. Whatever your field, porting UNIX-based applications to Linux should be quite straightforward. Linux provides a complete UNIX programming interface, sufficient to serve as the base for any scientific application.

As with any operating system, Linux has its share of games. These include classic text-based dungeon games such as Nethack and Moria; MUDs (multi-user dungeons, which allow many users to interact in a text-based adventure) such as DikuMUD and TinyMUD; as well as a slew of X games such as `xtetris`, `netrek`, and `gnuchess`. True, UNIX never has lent itself to the modern trend of dazzling shoot-em-up arcade games, but you know what they say: "All work and no play makes Linux a dull toy."

For audiophiles, Linux has support for various sound cards and related software, such as CDplayer (a program which can control a CD-ROM drive as a conventional CD player, surprisingly enough), MIDI sequencers and editors (allowing you to compose music for playback through a synthesizer or other MIDI-controlled instrument), and sound editors for digitized sounds.

Can't find the application you're looking for? The Linux Software Map, described in Appendix A, contains a list of many software packages which have been written and ported to Linux. While this list is far from complete, it contains a great deal of software. Another way to find Linux applications is to look at the `INDEX` files found on Linux FTP sites, if you have Internet access. Just by poking around you'll find a great deal of software just waiting to be played with.

If you absolutely can't find what you need, you can always attempt to port the application from another platform to Linux. Most freely distributable UNIX-based software will compile on Linux with few problems. Or, if all else fails, you can write the application yourself. If it's a commercial application you're looking for, there may be a free "clone" available. Or, you can encourage the software company to consider releasing a Linux binary version. Several individuals have contacted software companies, asking them to port their applications to Linux, and have met with various degrees of success.

1.5 About Linux's Copyright

Linux is covered by what is known as the GNU *General Public License*, or *GPL*. The GPL was developed for the GNU project by the Free Software Foundataion. It makes a number of provisions for the distribution and modification of "free software". "Free" in this sense refers to freedom, not just cost. The GPL has always been subject to misinterpretation, and we hope that this summary will help you to understand the extent and goals of the GPL and its effect on Linux. A complete copy of the GPL is included in Appendix E.

Originally, Linus Torvalds released Linux under a license more restrictive than the GPL, which allowed the software to be freely distributed and modified, but prevented any money changing hands for its distribution and use. On the other hand, the GPL allows people to sell and make profit from free software, but does not allow them to restrict the right for others to distribute the software in any way.

First, it should be explained that "free software" covered by the GPL is *not* in the public domain. Public domain software is software which is not copyrighted, and is literally owned by the public. Software covered by the GPL, on the other hand, is copyrighted to the author or authors. This means that the software is protected by standard international copyright laws, and that the author of the software is legally defined. Just because the software may be freely distributed does not mean that it is in the public domain.

GPL-licensed software is also not "shareware". Generally, "shareware" software is owned and copyrighted by the author, but the author requires users to send in money for its use after distribution. On the other hand, software covered by the GPL may be distributed and used free of charge.

The GPL also allows people to take and modify free software, and distribute their own versions of the software. However, any derived works from GPL software must also be covered by the GPL. In other words, a company could not take Linux, modify it, and sell it under a restrictive license. If any software is derived from Linux, that software must be covered by the GPL as well.

The GPL allows free software to be distributed and used free of charge. However, it also allows a person or organization to distribute GPL software for a fee, and even to make a profit from its sale and distribution. However, in selling GPL software, the distributor cannot take those rights away from the purchaser; that is, if you purchase GPL software from some source, you may distribute the software for free, or sell it yourself as well.

This might sound like a contradiction at first. Why sell software for profit when the GPL allows anyone to obtain it for free? As an example, let's say that some company decided to bundle a large amount of free software on a CD-ROM and distribute it. That company would need to charge for the overhead of producing and distributing the CD-ROM, and the company may even decide to make profit from the sales of software. This is allowed by the GPL.

Organizations which sell free software must follow certain restrictions set forth in the GPL. First, they cannot restrict the rights of users who purchase the software. This means that if you buy a CD-ROM of GPL software, you can copy and distribute that CD-ROM free of charge, or resell it yourself. Secondly, distributors must make it obvious to users that the software is indeed covered by the GPL. Thirdly, distributors must provide, free of charge, the complete source code for the software being distributed. This will allow anyone who purchases GPL software to make modifications of that software.

Allowing a company to distribute and sell free software is a very good thing. Not everyone has access to the Internet to download software, such as Linux, for free. The GPL allows companies to sell and distribute software to those people who do not have free (cost-wise) access to the software. For example, many organizations sell Linux on diskette, tape, or CD-ROM via mail order, and make profit from these sales. The developers of Linux may never see any of this profit; that is the understanding that is reached between the developer and the distributor when software is licensed by the GPL. In other words, Linus knew that companies may wish to sell Linux, and that he may not see a penny of the profits from those sales.

In the free software world, the important issue is not money. The goal of free software is always to develop and distribute fantastic software and to allow anyone to obtain and use it. In the next section, we'll discuss how this applies to the development of Linux.

1.6 The Design and Philosophy of Linux

When new users encounter Linux, they often have a few misconceptions and false expectations of the system. Linux is a unique operating system, and it is important to understand its philosophy and design in order to use it effectively. Time enough for a soapbox. Even if you are an aged UNIX guru, what follows is probably of interest to you.

In commercial UNIX development houses, the entire system is developed with a rigorous policy of quality assurance, source and revision control systems, documentation, and bug reporting and resolution. Developers are not allowed to add features or to change key sections of code on a whim: they must validate the change as a response to a bug report and consequently "check in" all changes to the source control system, so that the changes can be backed out if necessary. Each developer is assigned one or more parts of the system code, and only that developer may alter those sections of the code while it is "checked out".

Internally, the quality assurance department runs rigorous regression test suites on each new pass of the operating system, and reports any bugs. It is the responsibility of the developers to fix these bugs as reported. A complicated system of statistical analysis is employed to ensure that a certain percentage of bugs are fixed before the next release, and that the operating system as a whole passes certain release criteria.

In all, the process used by commercial UNIX developers to maintain and support their code is very complicated, and quite reasonably so. The company must have quantitative proof that the next revision of the operating system is ready to be shipped; hence, the gathering and analysis of statistics about the operating system's performance. It is a big job to develop a commercial UNIX system, often large enough to employ hundreds (if not thousands) of programmers, testers, documentors, and administrative personel. Of course, no two commercial UNIX vendors are alike, but you get the general picture.

With Linux, you can throw out the entire concept of organized development, source control systems, structured bug reporting, or statistical analysis. Linux is, and more than likely always will be, a hacker's operating system.[4]

[4] What I mean by "hacker" is a feverishly dedicated programmer, a person who enjoys exploiting computers and generally doing interesting things with them. This is in contrast to the common denotation of "hacker" as a computer

Linux is primarily developed as a group effort by volunteers on the Internet from all over the world. Across the Internet and beyond, anyone with enough know-how has the opportunity to aid in developing and debugging the kernel, porting new software, writing documentation, or helping new users. There is no single organization responsible for developing the system. For the most part, the Linux community communicates via various mailing lists and USENET newsgroups. A number of conventions have sprung up around the development effort: for example, anyone wishing to have their code included in the "official" kernel should mail it to Linus Torvalds, which he will test and include in the kernel (as long as it doesn't break things or go against the overall design of the system, he will more than likely include it).

The system itself is designed with a very open-ended, feature-minded approach. While recently the number of new features and critical changes to the system have diminished, the general rule is that a new version of the kernel will be released about every few months (sometimes even more frequently than this). Of course, this is a very rough figure: it depends on a several factors including the number of bugs to be fixed, the amount of feedback from users testing pre-release versions of the code, and the amount of sleep that Linus has had this week.

Let it suffice to say that not every single bug has been fixed, and not every problem ironed out between releases. As long as the system appears to be free of critical or oft-manifesting bugs, it is considered "stable" and new revisions will be released. The thrust behind Linux development is not an effort to release perfect, bug-free code: it is to develop a free implementation of UNIX. Linux is *for* the developers, more than anyone else.

Anyone who has a new feature or software application to add to the system generally makes it available in an "alpha" stage—that is, a stage for testing by those brave or unwary users who want to bash out problems with the initial code. Because the Linux community is largely based on the Internet, alpha software is usually uploaded to one or more of the various Linux FTP sites (see Appendix C) and a message posted to one of the Linux USENET newsgroups about how to get and test the code. Users who download and test alpha software can then mail results, bug fixes, or questions to the author.

After the initial problems in the alpha code have been fixed, the code enters a "beta" stage, in which it is usually considered stable but not complete (that is, it works, but not all of the features may be present). Otherwise, it may go directly to a "final" stage in which the software is considered complete and usable. For kernel code, once it is complete the developer may ask Linus to include it in the standard kernel, or as an optional add-on feature to the kernel.

Keep in mind that these are only conventions—not rules. Some people feel so confident with their software that they don't need to release an alpha or test version. It is always up to the developer to make these decisions.

You might be amazed that such a nonstructured system of volunteers, programming and debugging a complete UNIX system, could get anything done at all. As it turns out, it is one of the most efficient and motivated development efforts ever employed. The entire Linux kernel was written *from scratch*, without employing any code from proprietary sources. A great deal of work was put forth by volunteers to port all of the free software under the sun to the Linux system. Libraries were

wrongdoer or outlaw.

written and ported, filesystems developed, and hardware drivers written for many popular devices.

The Linux software is generally released as a *distribution*, which is a set of pre-packaged software making up an entire system. It would be quite difficult for most users to build a complete system from the ground up, starting with the kernel, adding utilities, and installing all of the necessary software by hand. Instead, there are a number of software distributions including everything that you need to install and run a complete system. Again, there is no standard distribution—there are many, each with their own advantages and disadvantages. We'll talk more about the various available Linux distributions in Section 2.1.

Despite the completeness of the Linux software, you will still need a bit of UNIX know-how to install and run a complete system. No distribution of Linux is completely bug-free, so you may be required to fix small problems by hand after installation. Running a UNIX system is not an easy task, not even for commercial versions of UNIX. If you're serious about Linux, bear in mind that it will take a considerable amount of effort and attention on your part to keep the system running and take care of things: this is true of *any* UNIX system, and Linux is no exception. Because of the diversity of the Linux community and the many needs which the software is attempting to meet, not eveything can be taken care of for you all of the time.

1.6.1 Hints for UNIX novices

One of the greatest mistakes often made by new UNIX users is attempting to install and run a complete Linux system without much background UNIX knowledge. To put it simply, without previous UNIX experience, installing and running Linux is going to prove difficult at best. This is the case with all implementations of UNIX. Nobody can expect to go from being a UNIX novice to a UNIX system administrator overnight. Too much of the time, MS-DOS users are tempted to dive into the Linux world, expecting that it will be easy to pick up "as you go along" without any external help. As the expression goes, "This isn't Kansas anymore." UNIX is not like MS-DOS or Microsoft Windows. No implementation of UNIX is expected to run trouble- and maintenance-free. Every UNIX system needs a system administrator, and if you are the system administrator for your soon-to-be-Linux system, you need to learn quite a few things about using and running UNIX before you begin.

This may sound like a tall order, but it really isn't so bad. There are many good introductory UNIX books on the market: See Appendix A for a list. Many new UNIX users expect to install Linux in order to *learn* UNIX, but we're afraid that it's supposed to work the other way around. Installing and setting up a UNIX system of your own is a challenging task even for very experienced system administrators. This is not to say that it is difficult, but only that you will benefit greatly from UNIX experience and an understanding of the process. Should you have trouble installing the system, you will know how to investigate and fix the problem yourself without having to find outside help. You must be aptly prepared for the journey which lies ahead. Otherwise, if you're new to UNIX, you may very well become overly frustrated with the system. It is very important to do some reading and experimentation with UNIX systems before attempting to run your own Linux system.

1.6.2 Hints for UNIX gurus

Even those people with years of UNIX programming and systems administration experience may need assistance before they are able to pick up and install Linux. There are still aspects of the system that UNIX wizards will need to be familiar with before diving in. For one thing, Linux is not a commercial UNIX system. It does not attempt to uphold the same standards as other UNIX systems you have may have come across. To be more specific, while stability is an important factor in the development of Linux, it is not the *only* factor.

More important, perhaps, is functionality. In many cases, new code will make it into the standard kernel even though it is still buggy and not functionally complete. The assumption is that it is more important to release code which users can test and use than delay a release until it is "complete". As an example, WINE (the Microsoft Windows Emulator for Linux) had an "official" alpha release before it was completely tested. In this way, the Linux community at large had a chance to work with the code, test it, and help develop it, while those who found the alpha code "good enough" for their needs could use it. Commercial UNIX vendors rarely, if ever, release software in this manner.

If you have been a UNIX systems administrator for more than a decade, and have used every commercial UNIX system under the Sun (no pun intended), Linux may take some getting used to. The system is very modern and dynamic. A new kernel release is made approximately every few months. New software is constantly being released. One day your system may be completely up-to-date with the current trend, and the next day the same system is considered to be in the Stone Age.

With all of this dynamic activity, how can you be expected to keep up with the ever-changing Linux world? For the most part, it is best to upgrade incrementally; that is, upgrade only those parts of the system that *need* upgrading, and then only when you think an upgrade is necessary. For example, if you never use Emacs, there is little reason to continuously install every new release of Emacs on your system. Furthermore, even if you are an avid Emacs user, there is usually no reason to upgrade it unless you find that some feature is missing that is in the next release. There is little or no reason to always be on top of the newest version of software.

We hope that Linux will meet or exceed your expectations of a homebrew UNIX system. At the very core of Linux is the spirit of free software, of constant development and growth. The Linux community favors expansion over stability, and that is a difficult concept to swallow for many people, especially those so steeped in the world of commercial UNIX. You cannot expect Linux to be perfect; nothing ever is in the free software world. However, we believe that Linux really is as complete and useful as any other implementation of UNIX.

1.7 Differences Between Linux and Other Operating Systems

It is important to understand the differences between Linux and other operating systems, such as MS-DOS, OS/2, and other implementations of UNIX for the personal computer. First of all, it should be made clear that Linux will coexist happily with other operating systems on the same machine: that is, you can run MS-DOS and OS/2 along with Linux on the same system without problems. There are even ways to interact between the various operating systems, as we'll see.

1.7.1 Why use Linux?

Why use Linux instead of a well-known, well-tested, and well-documented commercial operating system? We could give you a thousand reasons. One of the most important, however, is that Linux is an excellent choice for personal UNIX computing. If you're a UNIX software developer, why use MS-DOS at home? Linux will allow you to develop and test UNIX software on your PC, including database and X Windows applications. If you're a student, chances are that your university computing systems run UNIX. With Linux, you can run your own UNIX system and tailor it to your own needs. Installing and running Linux is also an excellent way to learn UNIX if you don't have access to other UNIX machines.

But let's not lose sight. Linux isn't just for personal UNIX users. It is robust and complete enough to handle large tasks, as well as distributed computing needs. Many businesses—especially small ones—are moving to Linux in lieu of other UNIX-based workstation environments. Universities are finding Linux to be perfect for teaching courses in operating systems design. Larger commercial software vendors are starting to realize the opportunities that a free operating system can provide.

The following sections should point out the most important differences between Linux and other operating systems. We hope that you'll find that Linux can meet your computing needs, or (at least) enhance your current computing environment. Keep in mind that they best way to get a taste for Linux is just to try it out—you needn't even install a complete system to get a feel for it. In Chapter 2, we'll show you how.

1.7.2 Linux vs. MS-DOS

It's not uncommon to run both Linux and MS-DOS on the same system. Many Linux users rely on MS-DOS for applications such as word processing. While Linux provides its own analogues for these applications (for example, TEX), there are various reasons why a particular user would want to run MS-DOS as well as Linux. If your entire dissertation is written using WordPerfect for MS-DOS, you may not be able to easily convert it to TEX or some other format. There are many commercial applications for MS-DOS which aren't available for Linux, and there's no reason why you can't use both.

As you might know, MS-DOS does not fully utilize the functionality of the 80386 and 80486 processors. On the other hand, Linux runs completely in the processor's protected mode, and exploits all of the features of the processor. You can directly access all of your available memory (and beyond, using virtual RAM). Linux provides a complete UNIX interface not available under MS-DOS—developing and porting UNIX applications under Linux is easily done, while under MS-DOS you are limited to a small subset of the UNIX programming functionality. Because Linux is a true UNIX system, you do not have these limitations.

We could debate the pros and cons of MS-DOS and Linux for pages on end. However, let it suffice to say that Linux and MS-DOS are completely different entities. MS-DOS is inexpensive (compared to other commercial operating systems), and has a strong foothold in the PC computing world. No other operating system for the PC has reached the level of popularity of MS-DOS—largely because the cost of these other operating systems is unapproachable to most personal computer users. Very few PC users can imagine spending $1000 or more on the operating system alone. Linux, however, is free, and you finally have the chance to decide.

We will allow you to make your own judgments of Linux and MS-DOS based on your expectations and needs. Linux is not for everybody. If you have always wanted to run a complete UNIX system at home, without the high cost of other UNIX implementations for the PC, Linux may be what you're looking for.

There are tools available to allow you to interact between Linux and MS-DOS. For example, it is easy to access MS-DOS files from Linux. There is also an MS-DOS emulator available, which allows you to run many popular MS-DOS applications. A Microsoft Windows emulator is currently under development.

1.7.3 Linux vs. The Other Guys

A number of other advanced operating systems are on the rise in the PC world. Specifically, IBM's OS/2 and Microsoft's Windows NT are becoming very popular as more users move away from MS-DOS.

Both OS/2 and Windows NT are full multitasking operating systems, much like Linux. Technically, OS/2, Windows NT, and Linux are quite similar: they support roughly the same features in terms of user interface, networking, security, and so forth. However, the real difference between Linux and The Other Guys is the fact that Linux is a version of UNIX, and hence benefits from the contributions of the UNIX community at large.

What makes UNIX so important? Not only is it the most popular operating system for multiuser machines, it is also the foundation for the majority of the free software world. If you have access to the Internet, nearly all of the free software available there is written specifically for UNIX systems. (The Internet itself is largely UNIX-based.)

There are many implementations of UNIX, from many vendors, and no single organization is responsible for distribution. There is a large push in the UNIX community for standardization in the form of open systems, but no single corporation controls this design. Hence, any vendor (or, as it turns out, any hacker) may implement these standards in an implementation of UNIX.

OS/2 and Windows NT, on the other hand, are proprietary systems. The interface and design are controlled by a single corporation, and only that corporation may implement that design. (Don't expect to see a free version of OS/2 anytime in the near future.) In one sense, this kind of organization is beneficial: it sets a strict standard for the programming and user interface unlike that found even in the open systems community. OS/2 is OS/2 wherever you go—the same holds for Windows NT.

However, the UNIX interface is constantly developing and changing. Several organizations are attempting to standardize the programming model, but the task is very difficult. Linux, in particular, is mostly compliant with the POSIX.1 standard for the UNIX programming interface. As time goes on, it is expected that the system will adhere to other such standards, but standardization is not the primary issue in the Linux development community.

1.7.4 Other implementations of UNIX

There are several other implementations of UNIX for the 80386 and 80486. The 80386 architecture lends itself to the UNIX design, and a number of vendors have taken advantage of this.

Feature-wise, other implementations of UNIX for the PC are quite similar to Linux. You will see that almost all commercial versions of UNIX support roughly the same software, programming environment, and networking features. However, there are some strong differences between Linux and commercial versions of UNIX.

First of all, Linux supports a different range of hardware from commercial implementations. In general, Linux supports the most well-known hardware devices, but support is still limited to that hardware which developers actually have access to. However, commercial UNIX vendors generally have a wider support base, and tend to support more hardware, although Linux is not far behind. We'll cover the hardware requirements for Linux in Section 1.8.

Secondly, commercial implementations of UNIX usually come bundled with a complete set of documentation as well as user support from the vendor. In contrast, most of the documentation for Linux is limited to documents available on the Internet—and books such as this one. In Section 1.9 we'll list sources of Linux documentation and other information.

As far as stability and robustness are concerned, many users have reported that Linux is at least as stable as commercial UNIX systems. Linux is still under development, and certain features (such TCP/IP networking) are less stable but improve as time goes by.

The most important factor to consider for many users is price. The Linux software is free, if you have access to the Internet (or another computer network) and can download it. If you do not have access to such a network, you may need to purchase it via mail order on diskette, tape, or CD-ROM (see Appendix B). Of course, you may copy Linux from a friend who may already have the software, or share the cost of purchasing it with someone else. If you are planning to install Linux on a large number of machines, you need only purchase a single copy of the software—Linux is not distributed on a "single machine" license.

The value of commercial UNIX implementations should not be demeaned: along with the price of the software itself, one usually pays for documentation, support, and assurance of quality. These are very important factors for large institutions, but personal computer users may not require these benefits. In any case, many businesses and universities are finding that running Linux on a lab of inexpensive personal computers is preferrable to running a commercial version of UNIX in a lab of workstations. Linux can provide the functionality of a workstation on PC hardware at a fraction of the cost.

As a "real-world" example of Linux's use within the computing community, Linux systems have travelled the high seas of the North Pacific, managing telecommunications and data analysis for an oceanographic research vessel. Linux systems are being used at research stations in Antarctica. As a more mundane example, perhaps, several hospitals are using Linux to maintain patient records. It is proving to be as reliable and useful as other implementations of UNIX.

There are other free or inexpensive implementations of UNIX for the 386 and 486. One of the most well-known is 386BSD, an implementation and port of BSD UNIX for the 386. 386BSD is comparable to Linux in many ways, but which one is "better" depends on your own personal needs and expectations. The only strong distinction that we can make is that Linux is developed openly (where any volunteer can aid in the development process), while 386BSD is developed within a closed team of programmers who maintain the system. Because of this, serious philosophical and design differences exist between the two projects. The goals of the two projects are entirely different: the goal of Linux is to develop a complete UNIX system from scratch (and have a lot of fun in the process), and the goal of 386BSD is in part to modify the existing BSD code for use on the 386.

NetBSD is another port of the BSD NET/2 distribution to a number of machines, including the 386. NetBSD has a slightly more open development structure, and is comparable to 386BSD in many respects.

Another project of note is HURD, an effort by the Free Software Foundation to develop and distribute a free version of UNIX for many platforms. Contact the Free Software Foundation (the address is given in Appendix E) for more information about this project. At the time of this writing, HURD is still in early stages of development.

Other inexpensive versions of UNIX exist as well, such as Coherent (available for about $99) and Minix (an academic but useful UNIX clone upon which early development of Linux was based). Some of these implementations are of mostly academic interest, while others are full-fledged systems for real productivity. Needless to say, however, many personal UNIX users are moving to Linux.

1.8 Hardware Requirements

Now you must be convinced of how wonderful Linux is, and all of the great things that it can do for you. However, before you rush out and install the software, you need to be aware of the hardware requirements and limitations that Linux has.

Keep in mind that Linux was developed by its users. This means, for the most part, that the hardware which is supported by Linux is only the hardware which the users and developers actually have access to. As it turns out, most of the popular hardware and peripherals for 80386/80486 systems are supported (in fact, Linux supports more hardware than some commercial implementations of UNIX). However, some of the more obscure and esoteric devices aren't supported yet. As time goes on, a wider range of hardware is supported, so if your favorite devices aren't listed here, chances are that support for them is forthcoming.

Another drawback for hardware support under Linux is that many companies have decided to keep the hardware interface proprietary. The upshot of this is that volunteer Linux developers simply can't write drivers for those devices (if they could, those drivers would be owned by the company that owned the interface, which would violate the GPL). The companies that maintain proprietary interfaces write their own drivers for operating systems such as MS-DOS and Microsoft Windows; the end user (that's you) never needs to know about the interface. Unfortunately, this does not allow Linux developers to write drivers for those devices.

There is very little that can be done about the situation. In some cases, programmers have attempted to write hackish drivers based on assumptions about the interface. In other cases, developers will work with the company in question and attempt to obtain information about the device interface, with varying degrees of success.

In the following sections, we'll attempt to summarize the hardware requirements for Linux. The Linux Hardware Compatibility List (see Section 1.9) contains a more complete listing of hardware supported by Linux.

Disclaimer: a good deal of hardware support for Linux is currently in the development stage. Some distributions may or may not support these experimental features. This section primarily lists hardware which has been supported for some time and is known to be stable. When in doubt, consult the documentation for the distribution of Linux you are using (see Section 2.1 for more information on Linux distributions).

1.8.1 Motherboard and CPU requirements

Linux currently supports systems with an Intel 80386 or 80486 CPU. This includes all variations on this CPU type, such as the 386SX, 486SX, 486DX, and 486DX2. Linux should also work on Intel's Pentium processor with minor changes. Non-Intel "clones", such as AMD and Cyrix processors, work with Linux as well.

If you have a 80386 or 80486SX, you may also wish to use a math coprocessor, although one isn't required (the Linux kernel can do FPU emulation if you do not have a math coprocessor). All standard FPU couplings are supported, such as IIT, Cyrix FasMath, and Intel coprocessors.

The system motherboard must use ISA or EISA bus architecture. There terms define how the system interfaces with peripherals and other components on the main bus. Most systems sold today are either ISA or EISA bus. IBM's MicroChannel (MCA) bus, found on machines such as the IBM PS/2, is not currently supported.

Systems which use a local bus architecture (for faster video and disk access) are supported as well. It is suggested that you have a standard local bus architecture such as the VESA Local Bus ("VLB").

1.8.2 Memory requirements

Linux requires very little memory to run compared to other advanced operating systems. You should have at the very least 2 megabytes of RAM; however, it is strongly suggested that you have 4 megabytes. The more memory you have, the faster the system will run.

Linux can support the full 32-bit address range of the 386/486; in other words, it will utilize all of your RAM automatically.

Linux will run happily with only 4 megabytes of RAM, including all of the bells and whistles such as X Windows, Emacs, and so on. However, having more memory is almost as important as having a faster processor. 8 megabytes is more than enough for personal use; 16 megabytes or more may be needed if you are expecting a heavy user load on the system.

Most Linux users allocate a portion of their hard drive as swap space, which is used as virtual RAM. Even if you have a great deal of physical RAM in your machine, you may wish to use swap space. While swap space is no replacement for actual physical RAM, it can allow your system to run larger applications by swapping out inactive portions of code to disk. The amount of swap space that you should allocate depends on several factors; we'll come back to this question in Section 2.2.3.

1.8.3 Hard drive controller requirements

You do not need to have a hard drive to run Linux; you can run a minimal system completely from floppy. However, this is slow and very limited, and many users have access to hard drive storage anyway. You must have an AT-standard (16-bit) controller. There is support in the kernel for XT-standard (8 bit) controllers; however, most controllers used today are AT-standard. Linux should support all MFM, RLL, ESDI, and IDE controllers.

The general rule for non-SCSI hard drive and floppy controllers is that if you can access the drive from MS-DOS or another operating system, you should be able to access it from Linux.

Linux also supports a number of popular SCSI drive controllers, although support for SCSI is more limited because of the wide range of controller interface standards. Supported SCSI controllers include the Adaptec AHA1542B, AHA1742A (BIOS version 1.34), AHA1522, AHA1740, AHA1740 (SCSI-2 controller, BIOS 1.34 in Enhanced mode); Future Domain 1680, TMC-850, TMC-950; Seagate ST-02; UltraStor SCSI; Western Digital WD7000FASST. Clones which are based on these cards should work as well.

1.8.4 Hard drive space requirements

Of course, to install Linux, you'll need to have some amount of free space on your hard drive. Linux will support multiple hard drives in the same machine; you can allocate space for Linux across multiple drives if necessary.

The *amount* of hard drive space that you will require depends greatly on your needs and the amount of software that you're installing. Linux is relatively small as UNIX implementations go; you could run a complete system in 10 to 20 megabytes of space on your drive. However, if you want to have room for expansion, and for larger packages such as X Windows, you will need more space. If you plan to allow multiple users to use the machine, you will need to allocate storage for their files.

Also, unless you have a large amount of physical RAM (16 megabytes or more), you will more than likely want to allocate swap space, to be used as virtual RAM. We will discuss all of the details of installing and using swap space in Section 2.2.3.

Each distribution of Linux usually comes with some literature that should help you to gauge the precise amount of required storage depending on the amount of software you plan to install. You can run a minimal system with less than 20 megabytes; a complete system with all of the bells and whistles in 80 megabytes or less; and a very large system with room for many users and space for future expansion in the range of 100-150 megabytes. Again, these figures are meant only as a ballpark approximation; you will have to look at your own needs and goals in order to determine your specific storage requirements.

1.8.5 Monitor and video adapator requirements

Linux supports all standard Hercules, CGA, EGA, VGA, and Super VGA video cards and monitors for the default text-based interface. In general, if the video card and monitor coupling works under another operating system such as MS-DOS, it should work fine with Linux.

Graphical environments such as the X Window System have video hardware requirements of their own. Instead of listing these requirements here, we relegate the discussion to Section 5.1.1.

1.8.6 Miscellaneous hardware

The above sections described the hardware which is required to run a Linux system. However, most users have a number of "optional" devices such as tape and CD-ROM storage, sound boards, and so on, and are interested in whether or not this hardware is supported by Linux. Read on.

1.8.6.1 Mice and other pointing devices

For the most part, you will only be using a mouse under a graphical environment such as the X Window System. However, several Linux applications not associated with a graphics environment do make use of the mouse.

Linux supports all standard serial mice, including Logitech, MM series, Mouseman, Microsoft (2-button) and Mouse Systems (3-button). Linux also supports Microsoft, Logitech, and ATIXL busmice. The PS/2 mouse interface is supported as well.

All other pointing devices, such as trackballs, which emulate the above mice, should work as well.

1.8.6.2 CD-ROM storage

Almost all CD-ROM drives use the SCSI interface. As long as you have a SCSI adaptor supported by Linux, then your CD-ROM drive should work. A number of CD-ROM drives have been verified to work under Linux, including the NEC CDR-74, Sony CDU-541, and Texel DM-3024. The Sony internal CDU-31a and the Mistsumi CD-ROM drives are supported by Linux as well.

Linux supports the standard ISO-9660 filesystem for CD-ROMs.

1.8.6.3 Tape drives

There are several types of tape drives available on the market. Most of them use the SCSI interface, all of which should be supported by Linux. Among the verified SCSI tape drives are the Sankyo

CP150SE; Tandberg 3600; Wangtek 5525ES, 5150ES, and 5099EN with the PC36 adaptor. Other QIC-02 drives should be supported as well.

Drivers are currently under development for various other tape devices, such as Colorado drives which hang off of the floppy controller.

1.8.6.4 Printers

Linux supports the complete range of parallel printers. If you are able to access your printer via the parallel port from MS-DOS or another operating system, you should be able to access it from Linux as well. The Linux printing software consists of the UNIX standard `lp` and `lpr` software. This software also allows you to print remotely via the network, if you have one available.

1.8.6.5 Modems

As with printer support, Linux supports the full range of serial modems, both internal and external. There is a great deal of telecommunications software available for Linux, including Kermit, `pcomm`, `minicom`, and Seyon. If your modem is accessible from another operating system on the same machine, you should be able to access it from Linux with no difficulty.

1.8.7 Ethernet cards

Many popular Ethernet cards and LAN adaptors are supported by Linux. These include:

- 3com 3c503, 3c503/16
- Novell NE1000, NE2000
- Western Digital WD8003, WD8013
- Hewlett Packard HP27245, HP27247, HP27250

The following clones are reported to work:

- LANNET LEC-45
- Alta Combo
- Artisoft LANtastic AE-2
- Asante Etherpak 2001/2003,
- D-Link Ethernet II
- LTC E-NET/16 P/N 8300-200-002
- Network Solutions HE-203,
- SVEC 4 Dimension Ethernet

- 4-Dimension FD0490 EtherBoard 16
- D-Link DE-600

Clones which are compatible with any of the above cards should work as well.

1.9 Sources of Linux Information

As you have probably guessed, there are many sources of information about Linux available apart from this book. In particular, there are a number of books, not specific to Linux but rather about UNIX in general, that will be of importance, especially to those readers without previous UNIX experience. If you are new to the UNIX world, we seriously suggest that you take the time to peruse one of these books before you attempt to brave the jungles of Linux. Specifically, the book *Learning the UNIX Operating System*, by Grace Todino and John Strang, is a good place to start.

Many of the following sources of information are available online in some electronic form. That is, you must have access to an online network, such as the Internet, USENET, or Fidonet, in order to access the information contained therein. If you do not have online access to any of this material, you might be able to find someone kind enough to give you hardcopies of the documents in question. Read on.

1.9.1 Online documents

If you have access to the Internet, there are many Linux documents available via anonymous FTP from archive sites all over the world. If you do not have direct Internet access, these documents may still be available to you: they are distributed on many other networks as well, such as Fidonet and CompuServe. If you are able to send mail to Internet sites, you may be able to retrieve these files using one of the `ftpmail` servers which will electronically mail you the documents or files from FTP archive sites. See Appendix C for more information on using `ftpmail`.

There is a great number of FTP archive sites which carry Linux software and related documents. A list of well-known Linux archive sites is given in Appendix C. In order to reduce network traffic, you should always use the FTP site which is geographically (network-wise) closest to you.

Appendix A contains a listing of some of the Linux documents which are available via anonymous FTP. The filenames will differ depending on the archive site in question; most sites keep Linux-related documents in the `docs` subdirectory of their Linux archive space. For example, on the FTP site `sunsite.unc.edu`, Linux files are stored in the directory `/pub/Linux`, with Linux-related documentation being found in `/pub/Linux/docs`.

Examples of available online documents are the *Linux FAQ*, a collection of freqeuently asked questions about Linux; the Linux *HOWTO* documents, each describing a specific aspect of the system—including the *Installation HOWTO*, the *Printing HOWTO*, and the *Ethernet HOWTO*; and, the Linux META-FAQ, a list of other sources of Linux information on the Internet.

Most of these documents are also regularly posted to one or more Linux-related USENET newsgroups; see Section 1.9.3 below.

1.9.2 Books and other published works

At this time, there are few published works specifically about Linux. Most noteworthy are the books from the Linux Documentation Project, a project carried out over the Internet to write and distribute a bona fide set of "manuals" for Linux. These manuals are analogues to the documentation sets available with commercial versions of UNIX: they cover everything from installing Linux, to using and running the system, programming, networking, kernel development, and more.

The Linux Documentation Project manuals are available via anonymous FTP from the Internet, as well as via mail order from several sources. Appendix A lists the manuals which are available and covers means of obtaining them in detail.

There are not many books specifically about Linux currently available. Several books have been published in Germany and Japan, including a translation of the Linux Documentation Project book *Linux Installation and Getting Started*. However, there are a large number of books about UNIX in general which are certainly applicable to Linux—as far as using and programming the system is concerned, Linux does not differ greatly from other implementations of UNIX. In short, almost everything you want to know about using and programming Linux can be found in sources meant for a general UNIX audience. In fact, this book is meant to be complemented by the large library of UNIX books currently available; here, we present the most important Linux-specific details and hope that you will look to other sources for more in-depth information.

Armed with a number of good books about using UNIX, as well as the book you hold in your hands, you should be able to tackle just about anything. Appendix A includes a list of highly-recommended UNIX books, both for UNIX newcomers and UNIX wizards alike.

There is also a monthly magazine about Linux, called the *Linux Journal*. It is distributed worldwide, and is an excellent way to keep in touch with the many goings-on in the Linux community—especially if you do not have access to USENET news (see below). See Appendix B for information on subscribing to the *Linux Journal*.

1.9.3 USENET newsgroups

USENET is a worldwide electronic news and discussion forum with a heavy contingent of so-called "newsgroups"—discussion areas devoted to a particular topic. Much of the development of Linux has been done over the waves of the Internet and USENET, and not suprisingly there are a number of USENET newsgroups available for discussions about Linux.

The original Linux newsgroup was `alt.os.linux`, and was created to move some of the discussions about Linux out of `comp.os.minix` and the various mailing lists. Soon, the traffic on `alt.os.linux` grew to be large enough that a newsgroup in the `comp` hierarchy was warranted; a vote was taken in February of 1992, and `comp.os.linux` was created.

`comp.os.linux` quickly became one of the most popular (and loudest) USENET groups; more popular than any other `comp.os` group. In December of 1992, a vote was taken to split the newsgroup in order to reduce traffic; only `comp.os.linux.announce` passed this vote. In July of 1993, the group was finally split into the new hierarchy. Almost 2000 people voted in the `comp.os.linux` reorganization, making it one of the largest USENET CFV's ever.

If you do not have direct USENET access, but are able to send and receive electronic mail from the Internet, there are mail-to-news gateways available for each of the newsgroups below.

`comp.os.linux.announce`
: `comp.os.linux.announce` is a moderated newsgroup for announcements and important postings about the Linux system (such as bug reports, important patches to software, and so on). If you read any Linux newsgroups at all, read this one. Often, the important postings in this group are not crossposted to other groups. This group also contains many periodic postings about Linux, including many of the online documents described in the last section and listed in Appendix A.

  Postings to this newsgroup must be approved by the moderators, Matt Welsh and Lars Wirzenius. If you wish to submit and article to this group, in most cases you can simply post the article as you normally would (using `Pnews` or whatever posting software that you have available); the news software will automatically forward the article to the moderators for approval. However, if your news system is not set up correctly, you may need to mail the article directly; the submission address is `linux-announce@tc.cornell.edu`.

  The rest of the Linux newsgroups listed below are unmoderated.

`comp.os.linux.help`
: This is the most popular Linux newsgroup. It is for questions and answers about using, setting up, or otherwise running a Linux system. If you are having problems with Linux, you may post to this newsgroup, and hopefully receive a reply from someone who might be able to help. However, it is strongly suggested that you read all of the available Linux documentation before posting questions to this newsgroup.

`comp.os.linux.admin`
: This newsgroup is for questions and discussion about running a Linux system, most commonly in an active, multi-user environment. Any discussion about administrative issues of Linux (such as packaging software, making backups, handling users, and so on) is welcome here.

`comp.os.linux.development`
: This is a newsgroup for discussions about development of the Linux system. All issues related to kernel and system software development should be discussed here. For example, if you are writing a kernel driver and need help with certain aspects of the programming, this would be the place to ask. This newsgroup is also for discussions about the direction and goals behind the Linux development effort, as described (somewhat) in Section 1.6.

  It should be noted that this newsgroup is not (technically) for discussions about development of software *for* Linux, but rather for discussions of development *of* Linux. That is, issues dealing with applications programming under Linux should be discussed in another Linux newsgroup; `comp.os.linux.development` is about developing the Linux system itself, including the kernel, system libraries, and so on.

`comp.os.linux.misc`
: This newsgroup is for all discussion which doesn't quite fit into the other available Linux groups. In particular, advocacy wars (the incessant "Linux versus Windows NT" thread, for example), should be waged here, as opposed to in the technical Linux groups. Any nontechnical or metadiscourse about the Linux system should remain in `comp.os.linux.misc`.

It should be noted that the newsgroup `comp.os.linux`, which was originally the only Linux group, has been superseded by the new hierarchy of groups. If you have access to `comp.os.linux`, but not to the newer Linux groups listed above, encourage your news administrator to create the new groups on your system.

1.9.4 Internet mailing lists

If you have access to Internet electronic mail, you can participate in a number of mailing lists even if you do not have USENET access. Note that if you are not directly on the Internet, you can join one of these mailing lists as long as you are able to exchange electronic mail with the Internet (for example, UUCP, FidoNET, CompuServe, and other networks all have access to Internet mail).

The "Linux Activists" mailing list is primarily for Linux developers and people interested in aiding the development process. This is a "multi-channel" mailing list, in which you join one or more "channels" based on your particular interests. Some of the available channels include: `NORMAL`, for general Linux-related issues; `KERNEL`, for kernel development; `GCC`, for discussions relating to the `gcc` compiler and library development; `NET`, for discussions about the TCP/IP networking code; `DOC`, for issues relating to writing and distributing Linux documentation; and more.

For more information about the Linux Activists mailing list, send mail to

```
linux-activists@niksula.hut.fi
```

You will receive a list of currently available channels, including information on how to subscribe and unsubscribe to particular channels on the list.

Quite a few special-purpose mailing lists about and for Linux exist as well. The best way to find out about these is to watch the Linux USENET newsgroups for announcements, as well as to read the list of publicly-available mailing lists, periodically posted to the USENET group `news.answers`.

1.10 Getting Help

You will undoubtedly require some degree of assistance during your adventures in the Linux world. Even the most wizardly of UNIX wizards occasionally is stumped by some quirk or feature of Linux, and it's important to know how and where to find help when you need it.

The primary means of getting help in the Linux world are via Internet mailing lists and USENET newsgroups, as discussed in Section 1.9. If you don't have online access to these sources, you might be able to find comparable Linux discussion forums on other online services, such as on local BBS's, CompuServe, and so on.

A number of businesses are providing commercial support for Linux. This will allow you to pay a "subscription fee" which will allow you to call the consutants for help with your Linux problems. Appendix B contains a list of commercial sources for Linux support. However, if you have access to USENET and Internet mail, you may find the free support found there to be just as useful.

Keeping the following suggestions in mind will greatly improve your experiences with Linux and will guarantee you more success in finding help to your problems.

Consult all available documentation... first! The first thing you should do when encountering a problem is consult the various sources of information listed in Section 1.9 and Appendix A. These documents were laboriously written for people like you—people who need help with the Linux system. Even books written for UNIX in general are applicable to Linux, and you should take advantage of them. More than likely, you will find the answer to your problems somewhere in this documentation, as impossible as it may seem.

If you have access to USENET news or any of the Linux-related mailing lists, be sure to actually *read* the information there before posting for help with your problem. Many times, solutions to common problems are not easy to find in documentation, and instead are well-covered in the newsgroups and mailing lists devoted to Linux. If you only post to these groups, and don't actually read them, you are asking for trouble.

Learn to appreciate self-maintenance. In most cases, it is preferable to do as much independent research and investigation into the problem as possible before seeking outside help. After all, you asked for it, by running Linux in the first place! Remember that Linux is all about hacking and fixing problems yourself. It is not a commercial operating system, nor does it try to look like one. Hacking won't kill you. In fact, it will teach you a great deal about the system to investigate and solve problems yourself—maybe even enough to one day call yourself a Linux guru. Learn to appreciate the value of hacking the system, and how to fix problems yourself. You can't expect to run a complete, homebrew Linux system without some degree of handiwork.

Remain calm. It is vital to refrain from getting frustrated with the system, at all costs. Nothing is earned by taking an axe—or worse, a powerful electromagnet—to your Linux system in a fit of anger. The authors have found that a large punching bag or similar inanimate object is a wonderful way to relieve the occasional stress attack. As Linux matures and distributions become more reliable, we hope that this problem will go away. However, even commercial UNIX implementations can be tricky at times. When all else fails, sit back, take a few deep breaths, and go after the problem again when you feel relaxed. Your mind and conscience will be clearer.

Refrain from posting spuriously. Many people make the mistake of posting or mailing messages pleading for help prematurely. When encountering a problem, do not—we repeat, do *not*—rush immediately to your nearest terminal and post a message to one of the Linux USENET newsgroups. Often, you will catch your own mistake five minutes later and find yourself in the curious situation of defending your own sanity in a public forum. Before posting anything any of the Linux mailing lists or newsgroups, first attempt to resolve the problem yourself and be absolutely certain what the problem is. Does your system not respond when switched on? Perhaps the machine is unplugged.

If you do post for help, make it worthwhile. If all else fails, you may wish to post a message for help in any of the number of electronic forums dedicated to Linux, such as USENET newsgroups and mailing lists. When posting, remember that the people reading your post are not there to help you. The network is not your personal consulting service. Therefore, it is important to remain as polite, terse, and informative as possible.

How can one accomplish this? First, you should include as much (relevant) information about your system and your problem as possible. Posting the simple request, "I cannot seem to get e-mail to work" will probably get you nowhere unless you include information on your system, what software you are using, what you have attempted to do so far and what the results were. When including technical information, it is usually a good idea to include general information on the version(s) of your software (Linux kernel version, for example), as well as a brief summary of your hardware configuration. However, don't overdo it—including information on the brand and type of monitor that you have probably is irrelevant if you're trying to configure networking software.

Secondly, remember that you need to make some attempt—however feeble—at solving your problem before you go to the Net. If you have never attempted to set up electronic mail, for instance, and first decide to ask folks on the Net how to go about doing it, you are making a big mistake. There are a number of documents available (see the Section 1.9) on how to get started with many common tasks under Linux. The idea is to get as far along as possible on your own and *then* ask for help if and when you get stuck.

Also remember that the people reading your message, however helpful, may occasionally get frustrated by seeing the same problem over and over again. Be sure to actually read the Linux newsgroups and mailing lists before posting your problems. Many times, the solution to your problem has been discussed repeatedly, and all that's required to find it is to browse the current messages.

Lastly, when posting to electronic newsgroups and mailing lists, try to be as polite as possible. It is much more effective and worthwhile to be polite, direct, and informative—more people will be willing to help you if you master a humble tone. To be sure, the flame war is an art form across many forms of electronic communication, but don't allow that to preoccupy your and other people's time. Save the network undue wear and tear by keeping bandwidth as low as possible, and by paying as much attention to other sources of information which are available to you. The network is an excellent way to get help with your Linux problems—but it is important to know how to use the network *effectively*.