Lecture Notes on GNU/Linux

Joseph Hesse

December 2009

Contents

Preface

0.1 About the Book

I wrote my book, "Lecture Notes on GNU/Linux (LNGL)", for use in the Linux courses I teach at Saint Paul College, formerly Saint Paul Technical College. The intended audience consists of working computer professionals who have a serious desire to learn Linux, either for personal or professional reasons.

The original title of the book was "Linux For Smart People." It seemed too flippant so I changed the title to LNGL. Also, I didn't want to get sued by the "Dummies" people.

0.2 How the Book is Organized

The book is organized into 6 major sections. They are:

1. Introduction to Linux and Open Source
2. Installing Linux On Your Computer
3. Using Your Linux Computer
4. The Linux Graphical Interface
5. Configuring Your Linux Computer
6. Networking with Other Computers

The names should explain themselves.

0.3 Goals

The book has enough detailed technical material for a two semester course in Linux, the first semester being a thorough introduction to Linux and the second semester covering Linux system administration. A student finishing the two courses should be able to set up and administer a small office network of Linux computers and Windows computers. He/she should be able to choose an appropriate Linux distribution, create user accounts, help users and learn the skills to fill gaps in his own knowledge.

The course assumes you are familiar with general computer usage and know how to use a graphical user interface (GUI). You should have experience using a text editor, a word processor, an email client, a file manager and a web browser. Familiarity with Microsoft Windows office applications should be sufficient. Some TCP/IP networking knowledge and the ability to read C programs is also helpful.

The reader should have access to one or more computers that he/she can dedicate to doing the exercises and use to experiment with Linux. Needless to say, these computers should not be production systems since the exercises and experiments may crash the system.

Besides covering material that is Linux specific, another goal is to teach more than perfunctory material on Open Source. The reason for this is that Open Source is new, evolving, and revolutionary. A person who professes to be knowledgeable about Linux should be able to intelligently discuss Open Source. The main ideas about Open Source that I cover are:

- Linux distributions, as we know them, would not exist without Richard Stallman's work on the GNU utilities and GPL license.
- The Open Source Definition (OSD) states what an Open Source License should consist of and gets rid of Stallman's ambiguous usage of the word free.
- Open Source has given rise to an astoundingly different method of software development. This is described in Eric Raymond's book "The Cathedral and the Bazaar."
- Microsoft has reacted to and is dealing with Open Source.

A book with a title like "Distro X Linux" is intended for people who want to learn "Distro X Linux." Such a book doesn't have to deal with differences, sometimes major, between Linux distributions. A well written "Distro X Linux" book can get the independent reader well on his way to having a working Linux system. The down side of such a book is that the reader gets the erroneous impression that every Linux distribution works the same way. For these reasons, LNGL uses and discusses two major Linux distributions, Fedora 10 and Ubuntu 8.10 and concentrates on the system aspects of these distributions rather than specific applications like OpenOffice.org. For those wanting a distribution specific book, there is nothing wrong with owning such a book and my LNGL.

0.4 How I Use the Book

My students at Saint Paul College get a printed copy of the book. A typical four hour weekly class meeting starts with my lecture on the current topic. The students see a projected copy of the pages I am discussing. I use the projected pages as memory jogs and don't read the printed copy. I try to keep my lecture to under an hour. The rest of the class is devoted to hands on class exercises which are done on the student's own computer. I usually float around the classroom and help the students.

The book is also intended for short one or two week courses at companies. It can be custom printed to add or omit topics, as requested by the company. A book written by the course instructor and available for examination prior to a possible course adds credibility both to the instructor and to the quality of the future course.

0.5 About the Author

I have an M.S. in Physics and a Ph.D in Mathematics. I have extensive college and university teaching experience and have been teaching computer subjects at corporations and schools since 1990.

I have no training in educational pedagogy.

I first became interested in Linux as a C++ programmer using the GNU C/C++ compiler as an alternative to the Microsoft one. Also, when I was a software developer using Unix on corporate computers, the Unix operating system cost over $100,000 (pre-1990) and I never had root access to the system. With Linux I have a Unix equivalent, at almost $0 cost, and have complete access to the system.

My web site and email address are `http://www.actcx.com` and `joe_hesse@actcx.com`.

0.6 Notes to a Prospective Publisher

The large amount of time I spent writing LNGL was to satisfy the author's teaching goals. Nevertheless, it might be useful to make it available to more people than the author's immediate students.

Tentatively, suppose the book is published and it appears, along with other general Linux books, in the Linux section of a major book store. How would Sam, a prospective Linux book buyer, choose the right book. When I say "right book" I mean the "right book", one that he will use to some degree, benefit from, and not ignore or return. Here is how I see it.

1. The books competing with LNGL are the non-distribution ones such as O'Reilly's "Running Linux". For the distribution specific titles, there are usually two or more titles for each of the major Linux distributions Debian, Fedora, SUSE, and Ubuntu. How is Sam going to make a choice?

2. If Sam has it in his mind that he wants a book on Ubuntu, there is not much choice about the books he will look at. Sam might choose Ubuntu because he is a beginner without anyone to help him or he already has Ubuntu running and wants to learn more about what is specifically available on a Ubuntu system. He looks at the available books on Ubuntu Linux, peruses them and chooses the one that he feels is best suited to what he want to do with Ubuntu.

3. Suppose Sam already knows something about Linux. He might have used Linux in a programming course at college, he might be an active member of one of the many Linux User Groups around the country or, he might be a teacher choosing a Linux book for his students. Sam then might look at the non-distribution specific books and come across LNGL. He flips through the pages of LNGL and sees that:

    - It covers two major distributions. The installation, partitioning and post installation of these distributions are discussed. "I didn't know that the partitioning part of the Fedora installation program doesn't allow me to create extended partitions."

- It is system oriented and does not spend much time on user applications. Sam already knows that there are other books that concentrate on Linux applications. He knows that GNOME, KDE, OpenOffice.org, Firefox, Ethereal, LaTeX, etc. all have readable, easily available documentation, both from within the application and from their respective web sites. LNGL collects this documentation and makes it available on the CD for the book.
- The history of Open Source is discussed in detail. "Now I understand why Torvald's kernel would have gone nowhere if the GNU Utilities and the GPL weren't already in place."
- There are exercises giving Sam something to do to test his learning.
- "It covers LVM, wow!"
- Both KDE and GNOME are discussed. "Now I know the political reasons why there are two great desktops available instead of one. I can install and learn to use both of them."
- "Unbelievable, there is clear documentation on the GNU info help system." No other Linux book has anything like this.
- There are some simple C programs to help understand things like signals. LNGL has a program that responds to Ctrl-C with "Don't bother me, I am busy." I can read the source code and see how it works.
- "Another C program helps me really understand the difference between stdin, stdout, stderr and how to capture their output." Again, I can read the source code and see how it works.
- Standards, wow! I though Linux was a "free for all." "Now I can learn about POSIX, LSB, and the File Hierarchy Standard."
- Extended regular expressions are better than regular expressions. "I should always use egrep rather than grep."
- There is a simple shell script written by the author that lets me try out different regular expressions (regex's).
- Sed and awk, "I didn't realize there were such things."
- "I just checked the web and found that Fedora 10 and Ubuntu 8.10 are the latest available versions and, guess what, LNGL covers these versions, not older ones."
- "I never really understood networks, now I can configure one." Network addresses, computer addresses, DHCP, DNS, routers are all clearly explained.
- "What happened to /etc/inittab?" It appears in recent releases of Fedora but not in Ubuntu. What is this "upstart" thing?
- LNGL shows how to set up a web server and other servers. "I want to try that."

4. Sam sees much more. He decides there is a lot of useful and readable information in LNGL that is not available in any other single book on Linux. He buys LNGL.

Part I

Introduction to Linux and Open Source

Chapter 1

Linux

1.1 About Linux

1.1.1 What is Linux?

A computer operating system is the software that allows programs to run on a computer. It provides the environment for running both user and system programs. Operating system software is responsible for scheduling and managing processes, managing memory, and accessing hardware such as disk drives, video, keyboard, mouse, and networking interfaces.

The software for an operating system is very complex. Modern operating systems are written by more than one person and represent many person years of effort. They also evolve over time.

Linux is a computer operating system like Windows, the Macintosh or UNIX. You can run applications on Linux the same as you can with other operating systems. Linux closely resembles the UNIX operating system.

Most of the time you see Linux running on a PC. Linux has nothing to do with Microsoft Windows even though they can be installed as a dual boot option on the same computer and share files.

Linux is a multiuser operating system. More than one user can be logged on to a Linux system at the same time. This is usually done by having multiple logins from hosts on the network. Of course you can always log in from the console.

Linux is named after Linus Torvalds who wrote the first version of the Linux kernel in 1991 when he was 22 years old. Even though a Linux distribution is made up of many more components than the kernel, most people refer to Linux distributions as Linux. This is probably unfair since the people and organizations who contributed to the other components are not getting the recognition they deserve.

The official mascot of Linux is Tux the penguin. Tux was chosen by Linus Torvalds. See Figure 1.1 on page 4. The figure is reproduced with the permission of Larry Ewing. More Tux images can be seen on the web site `http://www.isc.tamu.edu/~lewing/linux/`.

Linux, as we know it, would not exist if it weren't for the GNU Project. The official GNU mascot is the GNU. See Figure 1.2 on page 4. It was downloaded from the GNU web site, `http://www.gnu.org`. The GNU Project will be discussed later in these notes.

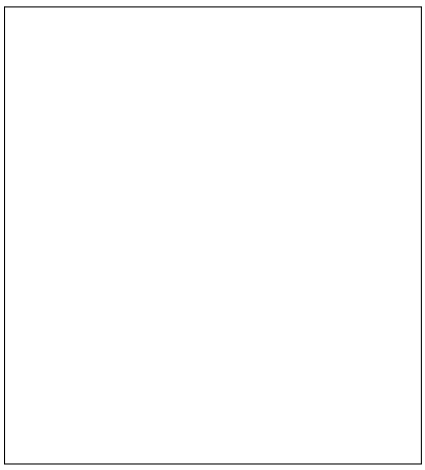

Figure 1.1: Tux the Penguin

Figure 1.2: A GNU

1.1.2 Components of a Linux Distribution

There are 6 components that make up a Linux distribution. They are:

Linux Kernel The kernel of the operating system is responsible for managing processes, managing memory, interfacing with the hardware, and creating an environment for programmers to write applications. Some distributions modify the current kernel to create custom kernels specific to that distribution.

System Programs Command shells, text editors and compilers are examples of these programs. Most of them come from the Free Software Foundation's GNU Project. Included here are the processes that the user doesn't directly interact with such as the `crond` daemon. These processes are started when Linux boots or are started on demand.

Install and Update Most Linux distributions have installation programs and programs to keep the system up to date.

System Administration Linux distributions come with tools to do system administration and configuration.

X Windows The X Window system provides the graphical interface. It comes from the X.org Foundation. Even though the URL of the Foundation is `http://www.x.org`, the company name is also the X.org Foundation.

Applications Strictly speaking, applications are not part of the operating system. However, without them, there is no point in having an operating system. The choice of individual applications depends on the Linux distribution. More can be added by the user.

1.1.3 Linux Applications

Linux distributions include hundreds of applications, probably more than you will ever want or use. Included among them are email clients, web browsers, graphics programs, and the image manipulation program GIMP which is like Photoshop. There are also games and system tools. Most applications have documentation included in the distribution or the documentation is readily available on the Internet. If a particular Linux distribution doesn't have the application you are looking for, there is a good chance that it is available somewhere as a free download.

One of the most popular set of Linux applications is OpenOffice.org. Even though the name looks like a web site address, it is the name of a product. As you might guess, the URL of the company is `http://www.openoffice.org`. OpenOffice.org includes a word-processor, a spreadsheet, a presentation program and a drawing program. OpenOffice.org can read documents created with Microsoft Office. OpenOffice.org can also be installed under Microsoft Windows as a replacement for Microsoft Office.

These notes were prepared with LaTeX. LaTeX is a document preparation system included in most Linux distributions. The TeX in LaTeX refers to the TeX typesetting system created by the distinguished computer scientist Donald Knuth. LaTeX is a macro package on top of TeX. It enables authors to typeset and print documents at the highest level of typographic quality. LaTeX is pronounced "lay-tech" or "lah-tech." LaTeX is a markup language like HTML. It is different from WYSIWYG word processors like Microsoft Word or OpenOffice.org Writer. Beautifully typeset complex mathematical equations can be created in LaTeX e.g.

$$I(z) = \sin(\frac{\pi}{2}z^2)\sum_{n=0}^{\infty}\frac{(-1)^n\pi^{2n}}{1\cdot 3\cdots(4n+1)}z^{4n+1} - \cos(\frac{\pi}{2}z^2)\sum_{n=0}^{\infty}\frac{(-1)^n\pi^{2n+1}}{1\cdot 3\cdots(4n+3)}z^{4n+3}$$

For this reason current mathematics research papers and Ph.D theses are usually written in LaTeX.

1.1.4 Where Do I Get Linux?

If you want to install Linux on your computer you have to obtain a Linux distribution. There are over 100 to choose from. Most of them are available as a free download. The

process consists of downloading one or more ISO images and then burning a CD or DVD from each ISO image. The details of this process are discussed later in these notes.

If you don't have a high speed Internet connection you can purchase Linux distributions on a CD or DVD for very low cost. Some of the companies providing Linux distributions have paid support.

1.2 Popular Linux Distributions

The six components of any Linux distribution were mentioned earlier, see page 4. There are over 100 Linux distributions, see `http://distrowatch.com`, a unique web site with a wealth of information about the major Linux distributions.

Some of the organizations that distribute Linux are multi-million dollar businesses, e.g., Red Hat and SUSE. Others appear to be labors of love and solicit contributions to defray their operating costs.

The purpose of these notes is to teach Linux, not to promote any particular distribution. Since teaching Linux requires working with Linux, the author has chosen the two popular Linux distributions Fedora 10 and Ubuntu 8.10. In addition, we will discuss Debian, SUSE and Knoppix. They can be downloaded at no cost from their respective web sites. All of them are likely to be around in a few years. These notes are not a collection of detailed information about any particular brand of Linux. There are books that cover individual Linux distributions in great detail.

It is easy to install several Linux distributions on a single PC. A boot manager is required to make this happen. Windows can be included as one of the boot options. Files on one distribution can be read with the other distributions.

A brief history of the previously mentioned Linux distributions follows. They appear in alphabetical order.

1.2.1 Debian

The Debian Project was founded by Ian Murdock on August 16th, 1993. At that time, the concept of a Linux distribution was new. Ian intended Debian to be a distribution which would be made openly, in the spirit of Linux and the GNU project (discussed later in these notes). The creation of Debian was sponsored by the Free Software's GNU project from November 1994 to November 1995. The official pronunciation of Debian is 'deb ee n'. The name comes from the names of the creator of Debian, Ian Murdock, and his wife, Debra.

Ian Murdock led Debian until March 1996. Afterwards, Bruce Perens of Open Source Definition fame, led Debian from April 1996 to December 1997. More information on Debian history is in the document `http://www.us.debian.org/doc/manuals/project-history/ch-leaders.en.html`.

Debian always has three releases available. They are: stable, testing and unstable. The stable distribution is the latest official release of Debian. The testing distribution contains packages that have not made their way into a stable release. The unstable distribution is the one where current development takes place. Frequent changes take place in the unstable distribution.

The latest stable version, Debian 4.0r5, codenamed "etch", was released on October 23, 2008. The current testing version is codenamed "lenny." If you want to experiment with Debian Linux and are not doing mission critical computing, I suggest you download and install "lenny", the current testing version. You should do a network install where you download a minimal system and then complete the install over the Internet.

More information on Debian Linux can be obtained by going to `http://www.debian.org`. Also, see Krafft [7] and Hill et al. [6].

1.2.2 Fedora

Red Hat Linux was founded by Marc Ewing in 1993 and was bought in 1995 by Bob Young who took on the role of CEO. Bob Young was succeeded by Matthew Szulik in 1999. The company went public in August of 1999.

The Fedora Project began in July of 2003 as a descendant of Red Hat Linux. There is now Red Hat Enterprise Linux and the Fedora Project. Red Hat Enterprise Linux costs money and paid support is available. The Fedora Project is sponsored by Red Hat and is guided by the Fedora Project Board. There is no charge to download Fedora Linux. Red Hat does not provide support, paid or otherwise, for Fedora Linux distributions. See `http://www.redhat.com` for information on Red Hat Enterprise Linux and `http://fedoraproject.org` for information on Fedora.

The Fedora web site states:

> What is Fedora? The Fedora Project is a collection of projects sponsored by Red Hat and developed as a partnership between the open source community and Red Hat engineers. The goal of Fedora is the rapid progress of free and open source software and content. Public forums. Open processes. Rapid innovation. Meritocracy and transparency. All in pursuit of the best operating system and platform that free software `http://www.fsf.org/licensing/essays/free-sw.html` can provide.

Releases up to and including version 6 were named "Fedora Core *x*." Current releases omit the name "Core."

Fedora 10, codename Cambridge, is the current version and was released on November 25, 2008. Fedora 9, codename Sulphur was released on May 13, 2008. The previous release dates of Fedora were FC1 on 11/5/03, FC2 on 5/18/04, FC3 on 11/8/04, FC4 on 6/13/05, FC5 on 3/20/06, FC6 on 10/24/06, F7 on 5/31/07 and F8 on 11/8/07.

1.2.3 Knoppix

Knoppix is a Linux distribution that boots from a CD or DVD. It runs from the CD or DVD, nothing is written to the hard drive. It uses the KDE desktop. Knoppix is based on the Debian Linux distribution and was created by Klaus Knopper.

Some of the uses of Knoppix are:

1. You can demonstrate Linux to your friends without writing anything to the hard drive.

2. You can test if the hardware on a PC or laptop is Linux compatible. If Knoppix doesn't boot correctly then you might have a hardware compatibility problem. This is especially true for new motherboards or new graphics adapters.

3. You can use Knoppix to fix a problem with your current Linux distribution. Booting from Knoppix gives you complete access to the partitions on your hard drive. Also, you can use Knoppix to do the initial partitioning on your hard drive, before installing your favorite Linux distribution.

Go to `http://www.knoppix.net` for information in English and a download link. The latest release of Knoppix is 5.3.1, dated 3/28/08.

1.2.4 SUSE

SuSE Linux was originally based on Slackware Linux. The name S.u.S.E., later shortened to just SuSE, was originally an acronym for the German phrase "Software und System Entwicklung" ("Software and System Development").

The current correct spelling is "SUSE", other spellings are used for historical accuracy.

S.u.S.E was founded in late 1992 as a UNIX consulting group. The first important release was S.u.S.E Linux 4.2 in 1996. SUSE is popular in Germany.

On November 4, 2003, Novell announced it would acquire SUSE Linux. The acquisition was finalized in January 2004. Novell's corporate technology stated that Novell would not "in the medium term" alter the way in which SuSE continues to be developed. At Novell's annual BrainShare gathering in 2004, all computers ran SUSE Linux for the first time. At this gathering it was also announced that the proprietary SUSE administration program YaST2 would be released into the public under the GPL license.

On August 4th, 2005, Novell announced that the development of the SUSE Professional series will become more open and, under the community name openSUSE, will try to reach a wider audience of users and developers. The result of this is that now there is SUSE Linux Enterprise and openSUSE Linux. Novell states on their web site, `http://www.novell.com/products/opensuse/comparative.html`:

> Novell provides Linux operating systems such as SUSE Linux Enterprise Server and SUSE Linux Enterprise Desktop for businesses that need a hardened, supported operating system for running IT infrastructure and corporate applications. Novell also provides openSUSE for technical individuals who want to keep up to speed with the latest and greatest open source code. The following table outlines the differences between Novell enterprise products and its openSUSE product.
>
> Although not intended for business deployments, openSUSE 10.3 provides corporate evaluators with an excellent first look at technologies like Novell AppArmor (security), Xen (virtualization), OpenOffice.org (productivity suite), Xgl (3-D desktop acceleration), Beagle (desktop search) and Mono (open source development environment). These technologies, when refined and fully supported, may appear in future Novell enterprise Linux offerings.

Go to `http://en.opensuse.org` to freely download openSUSE 11.0 and documentation.

You can freely download SUSE Linux Enterprise 10 from `http://www.novell.com/linux`. This free download does not include support, software patches and updates. If you want support you have to pay for it.

It appears that Red Hat and Novell are operating in similar ways since they both have supported and unsupported versions of their Linux distributions.

See McCallister [9] for a Linux book devoted entirely to SUSE Linux 10.

1.2.5 Ubuntu

Ubuntu Linux is based on Debian's unstable branch. The first release of Ubuntu Linux was on October 20, 2004. The word Ubuntu comes from the Zulu and Xhosa South African languages and represents an ideology. Roughly translated, it means "humanity towards others." There is no charge to download Ubuntu.

Ubuntu is funded by Canonical Ltd., a South African company founded by Mark Shuttleworth. Canonical Ltd. announced the creation of the Ubuntu Foundation whose purpose is to provide support for future versions of Ubuntu. The Ubuntu foundation was given an initial funding of $10 million. The foundation is described by Shuttleworth as an emergency fund.

The latest supported version is Ubuntu 8.10, codenamed "Intrepid." It was released on October 30, 2008. Ubuntu plans to release new versions every six months.

Ubuntu uses the GNOME desktop. There is also Kubuntu which uses the KDE desktop, Xubuntu which is a lightweight version using the Xfce desktop and Edubuntu, a version for schools. Each of these have their own websites. They are `http://www.ubuntu.com`, `http://www.kubuntu.org`, `http://www.xubuntu.org` and `http://www.edubuntu.org`. It is not a misprint that the first URL has the high level domain name `com` and the other three are `org`.

Ubuntu can be run from the CD or DVD, much like Knoppix, or it can be installed on the computer's hard drive.

Ubuntu installs with no root or administrator password. This means you can't log in as root unless you deliberately set a root password. This is done for the sake of security. If there is no root password you can't log in as root. The Linux command `sudo` is used to do administrative tasks.

Ubuntu uses the Debian package format. Here is what Ubuntu has to say about their software packages. The quote was taken from `http://www.ubuntu.com/community/ubuntustory/components`.

> Components
>
> The Ubuntu software repository is divided into four components, main, restricted, universe and multiverse on the basis of our ability to support that software, and whether or not it meets the goals laid out in our Free Software Philosophy.
>
> The standard Ubuntu installation is a subset of software available from the main and restricted components.

. . .

"main" component

The main distribution component contains applications that are free software, can freely be redistributed and are fully supported by the Ubuntu team. This includes the most popular and most reliable open source applications available, much of which is installed by default when you install Ubuntu.

Software in main includes a hand-selected list of applications that the Ubuntu developers, community, and users feel are important and that the Ubuntu security and distribution team are willing to support. When you install software from the main component you are assured that the software will come with security updates and technical support.

We believe that the software in main includes everything most people will need for a fully functional desktop or internet server running only open source software.

. . .

"restricted" component

The restricted component is reserved for software that is very commonly used, and which is supported by the Ubuntu team even though it is not available under a completely free licence. Please note that it may not be possible to provide complete support for this software since we are unable to fix the software ourselves, but can only forward problem reports to the actual authors.

Some software from restricted will be installed on Ubuntu CDs but is clearly separated to ensure that it is easy to remove. We include this software because it is essential in order for Ubuntu to run on certain machines - typical examples are the binary drivers that some video card vendors publish, which are the only way for Ubuntu to run on those machines. By default, we will only use open source software unless there is simply no other way to install Ubuntu. The Ubuntu team works with such vendors to accelerate the open-sourcing of their software to ensure that as much software as possible is available under a Free licence.

"universe" component

The universe component is a snapshot of the free, open source, and Linux world. In universe you can find almost every piece of open source software, and software available under a variety of less open licences, all built automatically from a variety of public sources. All of this software is compiled against the libraries and using the tools that form part of main, so it should install and work well with the software in main, but it comes with no guarantee of security fixes and support. The universe component includes thousands of pieces of software. Through universe, users are able to have the diversity and flexibility offered by the vast open source world on top of a stable Ubuntu core.

. . .

Popular or well supported pieces of software will move from universe into main if they are backed by maintainers willing to meet the standards set for main by the Ubuntu team.

"multiverse" component

The "multiverse" component contains software that is "not free", which means the licensing requirements of this software do not meet the Ubuntu "main" Component Licence Policy.

The onus is on you to verify your rights to use this software and comply with the licensing terms of the copyright holder.

...

Exercise 1.2.1: *Browse the web sites for each of the above distributions.*

1.3 Standards

There are standards for Unix and Linux. The purpose of this section is to state the names of the standards and briefly describe them. It is outside the scope of these notes to go into any detail.

1.3.1 POSIX

Applications written for Unix and Linux involve system calls made in the source code, e.g., the call to open a file or the call to write to a file. Collectively, these system calls are called the Application Programming Interface (API).

POSIX specifies the user and software interfaces to the OS in some 15 different documents. The standard user command line and scripting interface was based on the Korn shell. Other user-level programs, services and utilities include awk, echo, ed, and hundreds of others. Required program-level services include basic I/O (file, terminal, and network) services. POSIX also defines a standard threading library API which is supported by most modern operating systems.

Currently POSIX documentation is divided in three parts:

1. POSIX Kernel APIs (which include extensions for POSIX.1, Real-time Services, Threads Interface, Real-time Extensions, Security Interface, Network File Access and Network Process-to-Process Communications)

2. POSIX Commands and Utilities (with User Portability Extensions, Corrections and Extensions, Protection and Control Utilities and Batch System Utilities)

3. POSIX Conformance Testing. A test suite for POSIX accompanies the standard. It is called PCTS or the POSIX Conformance Test Suite.

Since the IEEE charges very high rates for POSIX documentation and does not allow on-line publication of the standards, there has been a tendency toward the "Single UNIX

Specification" standard, which is open, accepts input from anyone, and is freely available on the Internet. Beginning in 1998 a joint working group, the Austin Group, began to develop a combined standard that would be known as the Single UNIX Specification Version 3.

The name POSIX was suggested by Richard Stallman and is an acronym for Portable Operating System. Linux is not officially certified as POSIX compatible but it mostly conforms to it.

Most of the above information on POSIX was quoted from the Wikipedia web site, `http://en.wikipedia.org/wiki/POSIX`.

1.3.2 The Linux Foundation

Here is some information about the Linux Foundation taken from their web site `http://www.linux-foundation.org`.

> The Linux Foundation is a nonprofit consortium dedicated to fostering the growth of Linux. Founded in 2007 by the merger of the Open Source Development Labs and the Free Standards Group, it sponsors the work of Linux creator Linus Torvalds and is supported by leading Linux and open source companies and developers from around the world. The Linux Foundation promotes, protects and standardizes Linux by providing unified resources and services needed for open source to successfully compete with closed platforms.
>
> Since OSDL and the FSG were each formed more than six years ago, Linux has grown significantly in server, desktop, and embedded usage around the world. Moreover, the open source model has transformed development by providing faster demand-side learning, higher quality, better security, shorter development cycles, and lower prices than closed platform development models. OSDL and the FSG were important forces behind open source adoption and played key roles in preventing fragmentation of the Linux market.
>
> For Linux to remain open and attain the greatest ubiquity possible, important services must be provided, including legal protection, standardization, promotion and collaboration. Successful proprietary software companies, for instance, do several important things well: backwards compatibility, promotion, interoperability, developer support, and more. In the voluntary and distributed world of Linux development, the industry continues to successfully use the consortia model to rapidly improve these value attributes for Linux. The Linux Foundation has been founded to help close the gap between open source and proprietary platforms, while sustaining the openness, freedom of choice and technical superiority inherent in open source software. The Linux Foundations Activities
>
> The Linux Foundation does not build Linux, nor does it compete with existing Linux companies. Rather it fosters the growth of Linux by focusing on the following areas: Protecting Linux by sponsoring key Linux developers and providing legal services
>
> Its vitally important that Linux creator Linus Torvalds and other key kernel developers remain independent. The Linux Foundation sponsors them so they

> can work full time on improving Linux. The Linux Foundation also manages the Linux trademark and offers developers legal intellectual property protection through such initiatives as the Open Source as Prior Art project, the Patent Commons Project, and sponsorship of the Linux Legal Defense Fund. Standardizing Linux and improving it as a platform for software development
>
> A platform is only as strong as the applications that support it. The Linux Foundation offers application developers standardization services and support that make Linux an attractive target for their development efforts. These include the Linux Standard Base (LSB) and the Linux Developer Network. All major Linux distributions comply with the LSB. Providing a neutral forum for Collaboration and Promotion
>
> The Linux Foundation serves as a neutral spokesperson to advance the interests of Linux and respond with authority to competitors attacks. It also fosters innovation by hosting collaboration events among the Linux technical community, application developers, industry and end users to solve pressing issues facing the Linux ecosystem in such areas as desktop interfaces, accessibility, printing, application packaging, and many others.

The Linux Foundation supports the Linux Standard Base. Basically the LSB standard allows a developer to build an application so the binary executable will run on any LSB conforming Linux implementation. The following quote is taken from `http://www.linux-foundation.org/en/LSB`.

> The Linux Standard Base delivers interoperability between applications and the Linux operating system. Currently all major distributions comply with the LSB and many major application vendors, like MySQL, RealNetworks and SAP, are certifying. The LSB offers a cost-effective way for application vendors to target multiple Linux distributions while building only one software package. For end-users, the LSB and its mark of interoperability preserves choice by allowing them to select the applications and distributions they want while avoiding vendor lock-in. LSB certification of distributions results in more applications being ported to Linux and ensures that distribution vendors are compatible with those applications. In short, the LSB ensures Linux does not fragment.
>
> ...

1.3.3 Filesystem Hierarchy Standard

Unix and Linux have a hierarchical file structure. The name and location of directories and their contents in this hierarchical file structure have been specified in the Filesystem Hierarchy Standard (FHS). The FHS is based on the historical placement of these directories and files. Implementations conforming to the FHS should have system files and directories in the same place.

The Filesystem Hierarchy Standard (FHS) is part of the Linux Standard Base (LSB) specifications, and does a pretty good job of laying out where software should be installed,

depending on whether the files are coming from the distribution vendor, ISV, or theyre installed by the local admin.

For instance, distributions should install software under /usr. If you're administering a Linux system and want to install packages from source, they should be installed under /usr/local, and ISVs should install software under /opt.

Here are the directories required by the FHS. They are in section 3.2 and 3.3 of the FHS document.

```
3.2 Requirements

The following directories, or symbolic links to directories, are
required in /.

Directory                   Description
bin       Essential command binaries
boot      Static files of the boot loader
dev       Device files
etc       Host-specific system configuration
lib       Essential shared libraries and kernel modules
media     Mount point for removeable media
mnt       Mount point for mounting a filesystem temporarily
opt       Add-on application software packages
sbin      Essential system binaries
srv       Data for services provided by this system
tmp       Temporary files
usr       Secondary hierarchy
var       Variable data

3.3 Specific Options

The following directories, or symbolic links to directories, must be
in /, if the corresponding subsystem is installed:

Directory                    Description
home      User home directories (optional)
lib<qual> Alternate format essential shared libraries (optional)
root      Home directory for the root user (optional)
```


The latest version of this document is version 2.3, January 29, 2004. The document is easy to read. It can be obtained at `http://www.pathname.com/fhs/`.

Chapter 2

Open Source

2.1 Why Study Linux History?

Many technical subjects are taught with little or no history of the technology. Calculus books, for example, mention that Newton and Leibniz discovered calculus but little else is mentioned about how the historical times influenced these men. This approach is OK for a calculus course since there is nothing controversial or changing about calculus.

Linux, on the other hand, is brand new. It was created in 1991 and is still under active development. The Open Source way that Linux and other applications are distributed is still new and not completely understood by many computer professionals. If you get involved in Linux and Open Source you can reasonably expect to be asked by your friends and colleagues to explain, and possibly defend, some of these concepts. For this reason, we include more than perfunctory material on the history of Linux and Open Source.

2.2 Linux is Free

Linux is distributed as free software. The word free in free software does not mean that the acquisition cost is zero. It means, as will be explained below, a set of freedoms or rights.

Even though free software does not mean zero cost, it turns out that most Linux distributions cost nothing to download. If it weren't for this fact, Linux would not have achieved the popularity it has today. There are, however, Linux distributions costing thousands of dollars, e.g., at the time of this writing, Red Hat Enterprise Linux Advanced Platform costs $2499 for a one year Premium Subscription.

The term free software was created by Richard M. Stallman. Stallman created the Free Software Foundation (FSF) in October 1985. If it weren't for Richard Stallman and his GNU project, Linux would probably not exist (more about this later). Stallman defines free software (1985) as:

1. The freedom to run the program, for any purpose (freedom 0).
2. The freedom to study how the program works, and adapt it to your needs (freedom 1). Access to the source code is a precondition for this.

3. The freedom to redistribute copies so you can help your neighbor (freedom 2).
4. The freedom to improve the program, and release your improvements to the public, so that the whole community benefits (freedom 3). Access to the source code is a precondition for this.

Stallman states in his often quoted comment:

> "Free software" is a matter of liberty, not price. To understand the concept, you should think of "free" as in "free speech", not as in "free beer."

Part of Stallman's Free Software concept is that free software may be distributed for no charge or for a fee, regardless of how you obtained the original copy.

2.3 What is Open Source?

In English, the word free has a dual meaning. It can mean both freedom and gratis. The word free in free software has the freedom meaning. Because of the dual meaning of free in English, a new term evolved that both expresses and elaborates on the same idea. It is called Open Source. The term was coined in June 1997 by Bruce Perens and Eric Raymond. Bruce Perens took the concept and produced the Open Source Definition consisting of 10 major points. Here is a summary of what Open Source is about.

2.3.1 The Definition

Open Source is an adjective applied to a class of software license agreements. An Open Source software license agreement must meet certain conditions. These conditions are:

- Include the source code for the software.
- Not restrict any party from selling or giving away the software.
- Allow modifications and derived works. Allow them to be distributed under the same terms as the original software.
- The rights attached to the software applies to all to whom the program is distributed.
- + more (see Open Source Definition from OSI)

Perens points out:

> Note that the Open Source Definition is not itself a software license. It is a specification of what is permissible in a software license for that software to be referred to as Open Source. ...

The first and most famous Open Source license was written by Richard Stallman. It is the GNU General Public License. Version 2 (GPLv2), was released in June 1991 by the Free Software Foundation. Even though the term Open Source was not invented in 1991, GPLv2 is a bonafide Open Source license. It should be mentioned that Stallman does not approve of the term Open Source, see `http://www.gnu.org/philosophy/free-software-for-freedom.html`. GPLv3 was released 16 years later in June 2007.

2.3.2 What Open Source is NOT

Open Source is NOT:

- Freeware - Free but copyrighted.
- Shareware - Try before you buy.
- Public Domain Software - The software is free and can be used without any restrictions.

2.3.3 FAQs about Open Source

Some important facts about Open Source software are:

- There are over 55 Open Source licenses.
- Open Source does not mean $0 even though much Open Source software costs $0. The $0 cost is primarily responsible for the popularity of Open Source software.
- While most Open Source software is of high quality, the term Open Source does not necessarily imply high quality.
- There are no warranties with Open Source software. As an example, from the file `/etc/motd` on the Ubuntu GNU/Linux Distribution, version 7.10:

```
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
```

  Every time you log in to an Ubuntu system, you see the above quote.

2.3.4 How Linux is Distributed

At this point in time, most people say that Linux is distributed under an Open Source License or that Linux is Open Source. More information on the Open Source Definition will follow.

Besides Linux, there are hundreds of software products that are currently being distributed under Open Source Licenses. Among them are the Mozilla and Firefox web browsers and the OpenOffice.org suite of office software.

2.3.5 Why is Open Source Important?

- Because of the $0 cost, we are seeing Open Source software replace Microsoft software in schools and workplaces.
- Entire countries are adopting Open Source software.
- Software companies are integrating Open Source software into their products.

2.3.6 Alternative Meaning for Open Source

Even though Open Source refers to a licensing agreement, the term is used by many people to refer to the collection of software licensed under Open Source rather than the license itself. To them, Open Source software means Linux, Apache, Firefox, OpenOffice.org, etc.

2.3.7 What Is Your Company's Open Source Strategy?

If you are asked "What is your company's Open Source strategy?", the question usually means one of two things:

- If you are not a software company the question means; how are you going to save money by having your staff use Linux and OpenOffice.org? How can you use Open Source products to optimize your company's data center? For data centers the biggest concerns are flexibility, cost and mission-criticality.
- If you are a company that develops and sells software the question means; how are you going to integrate Open Source software into your company's products? The biggest concerns here are about choosing appropriate software components and Open Source licensing models.

2.3.8 Where Did Open Source Come From?

The ideas behind Open Source did not take place overnight. They evolved over a time period of approximately 25 years and had their roots in the early history of the Unix Operating System. Unix was created in 1969 by Dennis Ritchie at AT&T. Much of the early Unix development took place at AT&T and the University of California at Berkeley. During this time, AT&T, because of the 1956 Consent Decree, made the source code for Unix available at a nominal cost to universities and research institutions. After the AT&T divestiture in 1984, they ceased distributing Unix at a nominal cost. Berkeley always made the source code of their latest versions available at a nominal charge. See Salus [11] for a very readable history of Unix.

2.4 Richard Stallman

2.4.1 About Richard Stallman

If anyone can be called the father of the Open Source movement it is Richard M. Stallman. He is an extremely talented and principled individual. He has received many awards, including the MacArthur Genius Grant for $240,000, the ACM's Grace Murray Hopper Award, and several honorary doctorates. He created the GNU project and is the author of the GNU Emacs text editor, the GNU C compiler, and the GNU Debugger.

In 1984 Stallman resigned his position at MIT because he felt that the freedoms, as in his definition of Free Software, were not available to him at MIT. This resignation was a matter of conscience.

2.4.2 The GNU Manifesto

Stallman's GNU project was created in September 1983. His Free Software Foundation (FSF) was created in October 1985 to raise funds for GNU. It is a tax-exempt charity and the principal organizational sponsor of the GNU project. The GNU Manifesto (September 1985) was written by Richard Stallman at the beginning of the GNU Project.

> GNU, which stands for Gnu's Not Unix, is the name for the complete Unix-compatible software system which I am writing so I can give it away free, to everyone who can use it. ...
>
> ...
>
> I consider that the golden rule requires that if I like a program I must share it with other people who like it. Software sellers want to divide the users and conquer them, making each user agree not to share with others. I refuse to break solidarity with other users in this way. I cannot in good conscience sign a nondisclosure agreement or a software license agreement. For years I worked within the Artificial Intelligence Lab to resist such tendencies and other inhospitalities, but eventually they had gone too far: I could not remain in an institution where such things are done for me against my will.
>
> So that I can continue to use computers without dishonor, I have decided to put together a sufficient body of free software so that I will be able to get along without any software that is not free. I have resigned from the AI labs to deny MIT any legal excuse to prevent me from giving GNU away. ...

In 1993 Stallman added a footnote to the GNU Manifesto to clarify his use of the word free.

> (1) The wording here was careless. The intention was that nobody would have to pay for *permission* to use the GNU system. But the words don't make this clear, and people often interpret them as saying that copies of GNU should always be distributed at little or no charge. That was never the intent; later on, the manifesto mentions the possibility of companies providing the service of distribution for a profit. Subsequently, I have learned to distinguish carefully between "free" in the sense of freedom and "free" in the sense of price. ...

Exercise 2.4.1: *The complete text of the GNU Manifesto is on the CD for this book. Take a look at it.*

The kernel of the GNU operating system is called the Hurd. On `http://www.gnu.org/software/hurd/hurd.html` we find:

> The GNU Hurd is the GNU project's replacement for the Unix kernel. The Hurd is a collection of servers that run on the Mach microkernel to implement file systems, network protocols, file access control, and other features that are implemented by the Unix kernel or similar kernels (such as Linux). ...

Development on the Hurd started in 1990. Even though there is a working version of the Hurd, it does not have the popularity of the Linux kernel. Work on the Hurd appears to be at a standstill.

The GNU logo (picture of a GNU) is shown in Figure 1.2 on page 4.

2.4.3 The GNU General Public License

2.4.3.1 What is the GPL

The Free Software Foundation created a software license called "The GNU General Public License (GPL)." It is a licensing agreement that reflects the ideas of free software. If you write software you can distribute it under the GPL or one of its variants. There are currently two versions of the GPL, version 2(GPLv2), completed in June 1991 and the most recent, version 3 (GPLv3), completed 16 years later in June 2007.

The GPL consists of three parts, a Preamble which give the rationale for the license, paragraphs of terms and conditions, and finally, an appendix telling you how to apply the terms to your new programs.

The GPLv2 is still in active use. The Linux Kernel, Sun's Java, and MySQL are all licensed under GPLv2.

2.4.3.2 The Lesser GPL

There is a weaker form of the GPL called the "Lesser GPL." The Lesser GPL permits the use of free software libraries in proprietary programs. For example, you could write a proprietary software product that is compiled using the GNU C++ compiler and is linked with both the GNU C++ library and your own proprietary library functions.

Here is what Stallman has to say about the Lesser GPL.

> Why you shouldn't use the Lesser GPL for your next library
>
> The GNU Project has two principal licenses to use for libraries. One is the GNU Lesser GPL; the other is the ordinary GNU GPL. The choice of license makes a big difference: using the Lesser GPL permits use of the library in proprietary programs; using the ordinary GPL for a library makes it available only for free programs.
>
> Which license is best for a given library is a matter of strategy, and it depends on the details of the situation. At present, most GNU libraries are covered by the Lesser GPL, and that means we are using only one of these two strategies, neglecting the other. So we are now seeking more libraries to release under the ordinary GPL.
>
> Proprietary software developers have the advantage of money; free software developers need to make advantages for each other. Using the ordinary GPL for a library gives free software developers an advantage over proprietary developers: a library that they can use, while proprietary developers cannot use it.
>
> Using the ordinary GPL is not advantageous for every library. There are reasons that can make it better to use the Lesser GPL in certain cases. The most

common case is when a free library's features are readily available for proprietary software through other alternative libraries. In that case, the library cannot give free software any particular advantage, so it is better to use the Lesser GPL for that library.

This is why we used the Lesser GPL for the GNU C library. After all, there are plenty of other C libraries; using the GPL for ours would have driven proprietary software developers to use another-no problem for them, only for us.

...

2.4.3.3 The GPL Version 3

Stallman states that you should upgrade to the GPLv3. Here are some of his comments taken from `http://www.gnu.org/licenses/rms-why-gplv3.html`.

...

Keeping a program under GPLv2 won't create problems. The reason to migrate is because of the existing problems which GPLv3 will address.

One major danger that GPLv3 will block is tivoization. Tivoization means computers (called "appliances") contain GPL-covered software that you can't change, because the appliance shuts down if it detects modified software. The usual motive for tivoization is that the software has features the manufacturer thinks lots of people won't like. The manufacturers of these computers take advantage of the freedom that free software provides, but they don't let you do likewise.

...

In the crucial area of Digital Restrictions Management-nasty features designed to restrict your use of the data in your computer-competition is no help, because relevant competition is forbidden. Under the Digital Millennium Copyright Act and similar laws, it is illegal, in the US and many other countries, to distribute DVD players unless they restrict the user according to the official rules of the DVD conspiracy (its web site is http://www.dvdcca.org/, but the rules do not seem to be published there). The public can't reject DRM by buying non-DRM players, because none are available. No matter how many products you can choose from, they all have equivalent digital handcuffs.

GPLv3 ensures you are free to remove the handcuffs. It doesn't forbid DRM, or any kind of feature. It places no limits on the substantive functionality you can add to a program, or remove from it. Rather, it makes sure that you are just as free to remove nasty features as the distributor of your copy was to add them. Tivoization is the way they deny you that freedom; to protect your freedom, GPLv3 forbids tivoization.

...

> Another threat that GPLv3 resists is that of patent deals like the Novell-Microsoft deal. Microsoft wants to use its thousands of patents to make GNU-Linux users pay Microsoft for the privilege, and made this deal to try to get that. The deal offers Novell's customers rather limited protection from Microsoft patents.
>
> Microsoft made a few mistakes in the Novell-Microsoft deal, and GPLv3 is designed to turn them against Microsoft, extending that limited patent protection to the whole community. In order to take advantage of this, programs need to use GPLv3.
>
> Microsoft's lawyers are not stupid, and next time they may manage to avoid those mistakes. GPLv3 therefore says they don't get a "next time". Releasing a program under GPL version 3 protects it from Microsoft's future attempts to make redistributors collect Microsoft royalties from the program's users.
>
> GPLv3 also provides for explicit patent protection of the users from the program's contributors and redistributors. With GPLv2, users rely on an implicit patent license to make sure that the company which provided them a copy won't sue them, or the people they redistribute copies to, for patent infringement.
>
> The explicit patent license in GPLv3 does not go as far as we might have liked. Ideally, we would make everyone who redistributes GPL-covered code surrender all software patents, along with everyone who does not redistribute GPL-covered code. Software patents are a vicious and absurd system that puts all software developers in danger of being sued by companies they have never heard of, as well as by all the megacorporations in the field. Large programs typically combine thousands of ideas, so it is no surprise if they implement ideas covered by hundreds of patents. Megacorporations collect thousands of patents, and use those patents to bully smaller developers. Patents already obstruct free software development.
>
> ...
>
> Copyright © 2007 Richard Stallman Verbatim copying and distribution of this entire article are permitted worldwide without royalty in any medium provided this notice is preserved.

Exercise 2.4.2: *The GPL is worth reading. Read the GPL, versions 2 and 3, included on the book's CD or go to* `http://www.gnu.org/licenses/gpl.html`.

2.4.3.4 Copyleft

Stallman uses the term copyleft to refer to the software freedoms mentioned above. He states, `http://www.gnu.org/copyleft`:

> *Copyleft* is a general method for making a program free software and requiring all modified and extended versions of the program to be free software as well.
>
> ...

> To copyleft a program, we first state that it is copyrighted; then we add distribution terms, which are a legal instrument that gives everyone the rights to use, modify, and redistribute the program's code or *any program derived from it* but only if the distribution terms are unchanged. Thus, the code and the freedoms become legally inseparable.
>
> ...
>
> Copyleft is a general concept; there are many ways to fill in the details. In the GNU Project, the specific distribution terms that we use are contained in the GNU General Public License ...
>
> ...

2.5 Linus Torvalds

2.5.1 The Birth of Linux

The Linux operating system was created in 1991, as a hobby, by Linus Torvalds at the University of Helsinki. He used Unix at the university and wanted a Unix system to experiment with on his personal PC. The commercial versions of Unix were too expensive and ran on, at that time, expensive hardware.

An inexpensive option called Minix was available. Minix was a Unix clone created by Andrew Tannenbaum for teaching purposes. The full source code for Minix was available at no cost. Torvalds decided to develop an operating system that was better than Minix. He succeeded and posted, in August 1991, the result on `comp.os.minix`. Here is the full text of that now famous post.

```
From: torvalds@klaava.Helsinki.FI (Linus Benedict Torvalds)
Newsgroups: comp.os.minix
Subject: What would you like to see most in minix?
Summary: small poll for my new operating system
Message-ID: <1991Aug25.205708.9541@klaava.Helsinki.FI>
Date: 25 Aug 91 20:57:08 GMT
Organization: University of Helsinki

Hello everybody out there using minix -

I'm doing a (free) operating system (just a hobby, won't be big and
professional like gnu) for 386(486) AT clones. This has been brewing
since april, and is starting to get ready. I'd like any feedback on
things people like/dislike in minix, as my OS resembles it somewhat
(same physical layout of the file-system (due to practical reasons)
among other things).

I've currently ported bash(1.08) and gcc(1.40), and things seem to work.
This implies that I'll get something practical within a few months, and
I'd like to know what features most people would want. Any suggestions
are welcome, but I won't promise I'll implement them :-)

Linus (torvalds@kruuna.helsinki.fi)
```

```
PS. Yes - it's free of any minix code, and it has a multi-threaded fs.
It is NOT portable (uses 386 task switching etc), and it probably never
will support anything other than AT-harddisks, as that's all I have :-(.
```

Torvalds released kernel versions prior to 0.12 at no charge and required that there be no charge for redistributing it. He changed that with version 0.12. He now required that subsequent versions of the Linux kernel be released under the GPL. On January 5, 1992 he wrote:

```
                RELEASE NOTES FOR LINUX v0.12


This file mostly contains info on changed features of Linux, and using
old versions as a help-reference might be a good idea.


                COPYRIGHT

The Linux copyright will change: I've had a couple of requests to make
it compatible with the GNU copyleft, removing the "you may not
distribute it for money" condition.  I agree.  I propose that the
copyright be changed so that it confirms to GNU - pending approval of
the persons who have helped write code.  I assume this is going to be no
problem for anybody: If you have grievances ("I wrote that code assuming
the copyright would stay the same") mail me.  Otherwise The GNU copyleft
takes effect as of the first of February.  If you do not know the gist
of the GNU copyright - read it.

...
```

The kernel numbering system is `major.minor.patch`. An even `minor` number indicates a stable version of the kernel, an odd `minor` number indicates a test release for developers only. The current version is `2.6.xxx`.

Torvalds is still the key figure in Linux kernel development. After he put the Linux kernel under the GPL, thousands of developers contributed to its development. The kernel is still under active development.

2.5.2 GNU's Invaluable Contribution to Linux

Two very important things were in place at the time Torvalds wrote the Linux kernel. Without them, Linux would not, as we know it, exist!

1. The GNU utilities. This provided, among other things, a command shell, a compiler, and a text editor. Without these, Torvald's kernel could not be used to create programs and allow user interaction with the computer.

2. The GPL. This allowed further Kernel development to take place and "protect" the rights of the community of hackers who contributed to the kernel.

Because of this, the title of these notes refers to GNU/Linux.

Torvalds has the following opinions on the GPLv3. He states on the Linux Kernel Developers mailing list:

- I was impressed in the sense that it was a hell of a lot better than the disaster that were the earlier drafts.
- I still think GPLv2 is simply the better license.
- I consider dual-licensing unlikely (and technically quite hard), but at least possible in theory. I have yet to see any actual *reasons* for licensing under the GPLv3, though. All I've heard are shrill voices about "tivoization" (which I expressly think is ok) and panicked worries about Novell-MS (which seems way overblown, and quite frankly, the argument seems to not so much be about the Novell deal, as about an excuse to push the GPLv3).

2.6 Open Source and Free Software

2.6.1 The Open Source Definition

Bruce Perens wrote the first draft of "The Open Source Definition" (OSD) in June 1997. The OSD began as a policy document for the Debian Linux distribution (Debian Social Contract).

As mentioned before, the OSD is not a license but a set of 10 specifications that must be present in a license claiming to be Open Source. There are over 50 licenses that conform to the OSD, see `http://www.opensource.org/licenses/`. The OSD is a descendant of Richard Stallman's GPL. Perens calls the OSD a bill of rights for the computer user. The current version of the OSD is version 1.9.

It is the OSD that makes products like Linux possible. Bruce Perens states in his OSD document:

> The volunteers who made products like Linux possible are only there, and the companies are only able to cooperate, because of the rights that come with Open Source. The average computer programmer would feel stupid if he put lots of work into a program, only to have the owner of the program sell his improvement without giving anything back. ...

Exercise 2.6.3: *The OSD is too lengthy to reproduce here. It is important that you look at it. Read the annotated OSD included on the book's CD or go to* ***`http://www.opensource.org/docs/definition.php`****.*

A recent interesting book on producing open source software is Fogel [3]. A legal copy is on the CD for these notes.

2.6.2 The Debian Free Software Guidelines

The following lengthy quote is taken from the Debian web site, `http://www.debian.org/intro/free`. It is put here in these notes so you can read a slightly different viewpoint (different from RMS) on the previous ideas.

...

Many people new to free software find themselves confused because the word "free" in the term "free software" is not used the way they expect. To them free means "at no cost". An English dictionary lists almost twenty different meanings for "free". Only one of them is "at no cost". The rest refer to liberty and lack of constraint. When we speak of Free Software, we mean freedom, not price.

Software that is free only in the sense that you don't need to pay to use it is hardly free at all. You may be forbidden to pass it on, and you are almost certainly prevented from improving it. Software licensed at no cost is usually a weapon in a marketing campaign to promote a related product or to drive a smaller competitor out of business. There is no guarantee that it will stay free.

Truly free software is always free. Software that is placed in the public domain can be snapped up and put into non-free programs. Any improvements then made are lost to society. To stay free, software must be copyrighted and licensed.

To the uninitiated, either a piece of software is free or it isn't. Real life is much more complicated than that. To understand what kinds of things people are implying when they call software free we must take a little detour into the world of software licenses.

Copyrights are a method of protecting the rights of the creator of certain types of works. In most countries, software you write is automatically copyrighted. A license is the authors way of allowing use of their creation (software in this case), by others, in ways that are acceptable to them. It is up to the author to include a license which declares in what ways the software may be used. For a proper discussion of copyright see `http://www.copyright.gov/`.

Of course, different circumstances call for different licenses. Software companies are looking to protect their assets so they only release compiled code (which isn't human readable) and put many restrictions on the use of the software. Authors of free software on the other hand are generally looking for some combination of the following:

- Not allowing use of their code in proprietary software. Since they are releasing their code for all to use, they don't want to see others steal it. In this case, use of the code is seen as a trust: you may use it, as long as you play by the same rules.
- Protecting identity of authorship of the code. People take great pride in their work and do not want someone else to come along and remove their name from it or claim that they wrote it.
- Distribution of source code. One of the problems with most commercial code is that you can't fix bugs or customize it since the source code is not available. Also, the company may decide to stop supporting the hardware

you use. Many free licenses force the distribution of the source code. This protects the user by allowing them to customize the software for their needs. This also has other ramifications which will be discussed later.

- Forcing any work that includes part of their work (such works are called derived works in copyright discussions) to use the same license.

Many people write their own license. This is frowned upon as writing a license that does what you want involves subtle issues. Too often the wording used is either ambiguous or people create conditions that conflict with each other. Writing a license that would hold up in court is even harder. Luckily, there are a number of licenses already written that probably do what you want.

Three of the most widely found licenses are:

- The GNU General Public License (GPL). Some good background information on software licenses and a copy of the license can be found at the GNU web site. This is the most common free license in use in the world.
- Artistic License.
- BSD style license.

Some of the features these licenses have in common.

- You can install the software on as many machines as you want.
- Any number of people may use the software at one time.
- You can make as many copies of the software as you want and give them to whomever you want (free or open redistribution).
- There are no restrictions on modifying the software (except for keeping certain notices intact).
- There is no restriction on distributing, or even selling, the software.

This last point, which allows the software to be sold for money seems to go against the whole idea of free software. It is actually one of its strengths. Since the license allows free redistribution, once one person gets a copy they can distribute it themselves. They can even try to sell it. In practice, it costs essentially no money to make electronic copies of software. Supply and demand will keep the cost down. If it is convenient for a large piece of software or an aggregate of software to be distributed by some media, such as CD, the vendor is free to charge what they like. If the profit margin is too high, however, new vendors will enter the market and competition will drive the price down. As a result, you can buy a Debian release on several CDs for just a few USD.

While free software is not totally free of constraints (only putting something in the public domain does that) it gives the user the flexibility to do what they need in order to get work done. At the same time, it protects the rights of the author. Now that's freedom.

The Debian project is a strong supporter of free software. Since many different licenses are used on software, a set of guidelines, the Debian Free Software Guidelines (DFSG) were developed to come up with a reasonable definition of what constitutes free software. Only software that complies with the DFSG is allowed in the main distribution of Debian.

2.6.3 Stallman's View of Open Source Software

The following is quoted from `http://www.gnu.org/philosophy/open-source-misses-the-point.html`.

> Why "Open Source" misses the point of Free Software
>
> ...
>
> When we call software "free," we mean that it respects the users' essential freedoms: the freedom to run it, to study and change it, and to redistribute copies with or without changes. This is a matter of freedom, not price, so think of "free speech," not "free beer."
>
> These freedoms are vitally important. They are essential, not just for the individual users' sake, but because they promote social solidarity, that is, sharing and cooperation. They become even more important as more and more of our culture and life activities are digitized. In a world of digital sounds, images and words, free software comes increasingly to equate with freedom in general.
>
> ...
>
> However, not all of the users and developers of free software agreed with the goals of the free software movement. In 1998, a part of the free software community splintered off and began campaigning in the name of "open source." The term was originally proposed to avoid a possible misunderstanding of the term "free software," but it soon became associated with philosophical views quite different from those of the free software movement.
>
> Some of the proponents of "open source" considered it a "marketing campaign for free software," which would appeal to business executives by citing practical benefits, while avoiding ideas of right and wrong that they might not like to hear. Other proponents flatly rejected the free software movement's ethical and social values. Whichever their views, when campaigning for "open source" they did not cite or advocate those values. The term "open source" quickly became associated with the practice of citing only practical values, such as making powerful, reliable software. Most of the supporters of "open source" have come to it since then, and that practice is what they take it to mean.
>
> Nearly all open source software is free software; the two terms describe almost the same category of software. But they stand for views based on fundamentally different values. Open source is a development methodology; free software is a social movement. For the free software movement, free software is an ethical imperative, because only free software respects the users' freedom. By contrast,

> the philosophy of open source considers issues in terms of how to make software "better" in a practical sense only. It says that non-free software is a suboptimal solution. For the free software movement, however, non-free software is a social problem, and moving to free software is the solution.
>
> Free software. Open source. If it's the same software, does it matter which name you use? Yes, because different words convey different ideas. While a free program by any other name would give you the same freedom today, establishing freedom in a lasting way depends above all on teaching people to value freedom. If you want to help do this, it is essential to speak about "free software."
>
> We in the free software movement don't think of the open source camp as an enemy; the enemy is proprietary (non-free) software. But we want people to know we stand for freedom, so we do not accept being misidentified as open source supporters.
>
> ...
>
> Conclusion
>
> As the advocates of open source draw new users into our community, we free software activists have to work even more to bring the issue of freedom to those new users' attention. We have to say, "It's free software and it gives you freedom!" more and louder than ever. Every time you say "free software" rather than "open source," you help our campaign.

2.6.4 Open Source Licenses

There are more than 50 different open source licenses. It is beyond the scope of these notes and the author's knowledge to discuss them. See `http://www.opensource.org/licenses/` for information about individual open source licenses.

2.7 The Cathedral and the Bazaar

The Cathedral and the Bazaar by Eric S. Raymond, Raymond [10], is considered an important book in the Open Source movement. Its title comes from an essay that Raymond delivered at the May 1997 Linux Kongress. See `http://www.linux-kongress.org/1997/raymond.html`.

Raymond uses cathedrals and bazaars as contrasting symbols of organizational structure. A cathedral is built according to a master plan and under the control of a central authority. A bazaar is non-hierarchical and is a collection of different approaches and agendas.

Raymond's thesis is that, amazingly, the bazaar approach has produced Linux and other high quality software. According to Raymond:

> Linux is subversive. Who would have thought even five years ago that a world-class operating system could coalesce as if by magic out of part-time hacking by several thousand developers scattered all over the planet, connected only by the tenuous strands of the Internet?
>
> ...
>
> Linus Torvald's style of development - release early and often, delegate everything you can, be open to the point of promiscuity - came as a surprise. No quiet, reverent cathedral-building here - rather, the Linux community seemed to resemble a great babbling bazaar of different agendas and approaches (aptly symbolized by the Linux archive sites, who'd take submissions from anyone) out of which a coherent and stable system could seemingly emerge only by a succession of miracles.

Raymond's book is a chronicle of his work to deliberately develop a piece of software using the bazaar approach. He was successful beyond all expectation. The resulting software is fetchmail, a mail-retrieval and forwarding utility.

Exercise 2.7.4: *A legal copy of The Cathedral and the Bazaar and other documents by Eric Raymond are included on the book's CD. Take a look at them.*

2.8 The SCO Case

In February 1985, IBM signed a license agreement with AT&T to use their Unix code. IBM used the code to produce AIX, their version of Unix. In 1995 the Santa Cruz Organization (SCO) purchased, from AT&T, all rights to Unix.

Subsequently, IBM has been giving away its AIX code to the Linux community. SCO claims that IBM doesn't own the code so they have no right to give it away. In 2003, SCO filed a $1 billion dollar law suit against IBM.

This caused a reaction at Sun Microsystems since their Solaris operating system is based on Unix. Sun immediately assured their customers that the Solaris licenses they purchased were valid.

In July 2005, Novell filed in the U.S. District court of Utah that it, and not SCO, owned Unix's copyrights. On August 13, 2007, a U.S. district judge ruled that SCO does not own the Unix operating system, thus dealing SCO a significant legal blow.

This case is important because organizations will be reluctant to use Linux if they are going to be threatened with a lawsuit.

2.9 Open Source and Microsoft

Here are some items that give a flavor of how Microsoft feels about Linux and Open Source. These should be viewed as informative, not a section on Microsoft vs. Open Source.

2.9.1 The Halloween Documents

In October of 1998, Eric Raymond received a confidential Microsoft document concerning their strategy against Open Source. A brief quote from Raymond follows. See `http://opensource.feratech.com/halloween/`.

> In the last week of October 1998, a confidential Microsoft memorandum on Redmond's strategy against Linux and Open Source software was leaked to me by a source who shall remain nameless. I annotated this memorandum with explanation and commentary over Halloween Weekend and released it to the national press. Microsoft was forced to acknowledge its authenticity. ...

Here are some Eric Raymond annotated quotes from the Halloween documents on Open Source Software (OSS).

> OSS poses a direct, short-term revenue and platform threat to Microsoft, particularly in server space. Additionally, the intrinsic parallelism and free idea exchange in OSS has benefits that are not replicable with our current licensing model and therefore present a long term developer mindshare threat.
>
> Recent case studies (the Internet) provide very dramatic evidence ... that commercial quality can be achieved / exceeded by OSS projects.
>
> ...to understand how to compete against OSS, we must target a process rather than a company.
>
> OSS is long-term credible ... FUD tactics can not be used to combat it.
>
> Linux and other OSS advocates are making a progressively more credible arguments that OSS software is at least as robust – if not more – than commercial alternatives. The Internet provides an ideal, high-visibility showcase for the OSS world.
>
> Linux has been deployed in mission critical, commercial environments with an excellent pool of public testimonials. ... Linux outperforms many other UNIXes ... Linux is on track to eventually own the x86 UNIX market ...
>
> Linux can win as long as services / protocols are commodities.
>
> OSS projects have been able to gain a foothold in many server applications because of the wide utility of highly commoditized, simple protocols. By extending these protocols and developing new protocols, we can deny OSS projects entry into the market.
>
> The ability of the OSS process to collect and harness the collective IQ of thousands of individuals across the Internet is simply amazing. More importantly, OSS evangelization scales with the size of the Internet much faster than our own evangelization efforts appear to scale.

2.9.2 Ballmer: "Linux is a cancer"

In a June 1, 2001 Chicago Sun-Times interview, Microsoft CEO Steve Ballmer, in response to the question "Do you view Linux and the open-source movement as a threat to Microsoft", said:

> Yeah. It's good competition. It will force us to be innovative. It will force us to justify the prices and value that we deliver. And that's only healthy. The only thing we have a problem with is when the government funds open-source work. Government funding should be for work that is available to everybody. Open source is not available to commercial companies. The way the license is written, if you use any open-source software, you have to make the rest of your software open-source. If the government wants to put something in the public domain, it should. Linux is not in the public domain. Linux is a cancer that attaches itself in an intellectual property sense to everything it touches. That's the way the license works.

The use of the word "cancer" is not meant in the fatal sense but more in the "spreading" sense. Any software that Open Source components "touch" acquire, because of how Open Source licenses are written, an Open Source license. If a company develops and sells software, it has to be careful about which Open Source components are used in the final product.

2.9.3 Get The Facts

2.9.3.1 Part I

Microsoft has been promoting, since January 2004, a "Get The Facts" campaign comparing Windows and Linux. See `http://www.microsoft.com/windowsserver/facts/default.mspx`. They provide 10+ cases in which the Total Cost of Ownership (TCO) is lower with Microsoft products compared with Open Source products. Here is a quote from one of their case studies. The title of the case study is "Windows Server Delivers Lower TCO than Linux in China."

> Many white papers have shown that Windows has lower total cost of ownership (TCO) than Linux in countries where labor rates are high. However, little work has been done in low labor rate markets. Chinese Computer World (CCW) has done ground-breaking research in this area by studying the TCO of five Windows and Linux workloads in China.
>
> ...
>
> Information technology labor is a major cost driver for chief information officers (CIOs) in all markets. In the United States, 60 percent of server TCO is IT staffing. In China, where labor rates are 90 percent less, IT staffing accounts for 27 percent of TCO. IT labor remains a major cost driver in China since organizations use less automation and more people to manage the same servers when compared to more expensive labor markets.

In China, Windows has lower TCO than Linux in 4 out of 5 workloads:

Windows application/database servers have 41 percent lower TCO than Linux.

Windows file/print servers have 12 percent lower TCO than Linux.

Windows web servers have 8 percent higher TCO than Linux.

Windows networking servers have 1 percent lower TCO than Linux.

Windows mail servers have 12 percent lower TCO than Linux.

2.9.3.2 Part II

Here is a quote, August 2007, from the IDG News Service.

> San Francisco (IDGNS) - Microsoft has replaced its controversial anti-Linux "Get the Facts" Web site with a kinder, gentler site explaining how its Windows Server operating system compares to open-source Linux as well as other competitive OSes.
>
> The new WindowsServer/Compare Web site provides information about how Windows Server stacks up in total cost of ownership, reliability, security, manageability and interoperability with Linux, Unix, and IBM's mainframe architecture.
>
> Microsoft has posted customer information, feedback from industry experts, white papers and resources about the capabilities of Windows Server on the site. It also offers information for developers building applications on Windows Server.
>
> Microsoft said the new site is an evolution of its Get the Facts campaign, launched in mid-2003 and seen by many as a direct slam against Linux and open source.
>
> The campaign compared Windows Server favorably against Linux and other technologies in terms of some of the same factors handled on the Compare site. Get the Facts was panned by Linux proponents. Their ire in part may have been due to the outspoken swagger of then Microsoft rising star Martin Taylor who led the campaign. After 13 years at Microsoft, Taylor abruptly left the company in June 2006 and no explanation was given for his departure.
>
> ...

The new WindowsServer/Compare Web site is located at `http://www.microsoft.com/windowsserver/compare`.

2.9.4 Microsoft and Intellectual Property Protection

Microsoft claims that open source software violates 235 Microsoft patents. They have not made public which of their patents have been violated. They are attempting to sign agreements with various Linux vendors to save them from being sued by Microsoft for patent infringement.

2.9.4.1 Microsoft and Novell

On November 2, 2006, Microsoft and Novell (SUSE Linux) announced they would collaborate on Windows and Linux interoperability and support. Here is a quote from the announcement.

> Microsoft Corp. and Novell Inc. today announced a set of broad business and technical collaboration agreements to build, market and support a series of new solutions to make Novell and Microsoft products work better together. The two companies also announced an agreement to provide each other customers with patent coverage for their respective products. These agreements will be in place until at least 2012. Under this new model, customers will realize unprecedented choice and flexibility through improved interoperability and manageability between Windows and Linux.
>
> "They said it couldn't be done. This is a new model and a true evolution of our relationship that we think customers will immediately find compelling because it delivers practical value by bringing two of their most important platform investments closer together," said Steve Ballmer, CEO of Microsoft. "We're excited to work with Novell, whose strengths include its heritage as a mixed-source company. Resolving our patent issues enables a combined focus on virtualization and Web services management to create new opportunities for our companies and our customers."
>
> ...

2.9.4.2 Microsoft and Linspire

On June 14, 2007, the CEO of Linspire, a popular Linux distribution announced:

> Microsoft Will Help Deliver a "Better" Linux
>
> ...
>
> Today, Linspire announced our latest partnership, one with Microsoft, to bring even more choices to desktop Linux users, and together, offer a "better" Linux experience. Just as Steve Jobs announced in 1997 that "the era of setting this up as a competition between Apple and Microsoft is over," I too believe it's time for Linux to do the same. Rather than isolating Linux, I believe we need to understand, as Apple did in 1997, that Linux exists in an ecosystem and must work with and interoperate within that ecosystem. As unpopular as it may appear to some, Linspire is willing to take a lead in this effort. Some people booed Steve Jobs back in 1997, but if you trace the history of his announcement, I think it was an incredibly smart move for both Microsoft and Apple, issuing in a new era for both.
>
> ...

2.9.4.3 Microsoft and Red Hat

From an eWEEK article dated July 3, 2007, `http://www.eweek.com/article2/0,1895,2154521,00.asp`.

> Even though patent talks between Microsoft and Red Hat broke down last year before Microsoft went on to sign a technical collaboration and patent indemnity deal with Novell, Red Hat is still willing to work with the Redmond software maker on the interoperability front.
>
> But the Linux vendor wants to limit those talks to pure interoperability between Windows and Red Hat Linux, with the goal of solving real customer problems, Paul Cormier, Red Hat's executive vice president of engineering, told eWEEK.
>
> "I want to talk to the folks at Microsoft about our two operating systems and how we can work together to solve real customer problems without attaching any unrelated strings, such as intellectual property," he said.
>
> While Cormier declined to comment on why its earlier talks with Microsoft fell through, he ruled out any possibility of Red Hat doing a deal with Microsoft like the controversial patent agreement and covenant not to sue that Redmond penned with Novell last year, especially after viewing the limited information that is publicly available on that deal.
>
> But Microsoft officials said their position is that the issues of interoperability and intellectual property are not completely separate, and have to be considered together, meaning there is a de facto standoff between it and Red Hat on this issue.

2.9.5 Expressing Your Opinion

You probably have heard people express their opinion on the relative merits of Windows and Linux. Sometimes the rhetoric gets heated; four letter words are used and corporate business practices and ethics are thrown in the mix. Many people who hear these words are not computer professionals. They don't understand the technical issues and these opinions are likely to turn them off to Linux. If you give a public opinion on these issues, be polite, discuss the technical and legal issues and don't confuse corporate ethics with the quality of a corporation's products.

In the author's opinion, one of the best things about Linux is that it offers competition to Windows. Competition is always a good thing for the consumer.

Exercise 2.9.5: *What do you think would happen if Microsoft "Open Sourced" the Windows operating system?*

2.10 Open Source Timeline

The events in this chapter were not presented in strict chronological order. This was done for the sake of readability. Table 2.1 on page 36 lists the events in this chapter in chronological order.

Date	Event
1969	Unix, Dennis Ritchie, AT&T
1983	GNU Project
1984	Stallman resigns from MIT
1985	Free Software Definition, Stallman
1985	The GNU Manifesto
1985	Free Software Foundation created
1985	IBM gets Unix license to create AIX
1990	Hurd kernel development begins (GNU)
1991	Last version of GPLv2
1991	First Linux kernel from Torvalds
1993	Stallman clarifies use of word "free"
1995	SCO purchases Unix from AT&T
1997	Open Source Definition, First draft, Bruce Perens
1997	Cathedral and the Bazaar, Eric Raymond
1998	Halloween Documents, leaked to Eric Raymond
2001	Ballmer, "Linux is a cancer"
2003	SCO sues IBM over licensing
2004	Microsft starts "Get the Facts" campaign
2005	Novell sues SCO over licensing
2006	Microsoft and Novell collaborate
2007	Microsoft and Linspire collaborate
2007	Microsoft and Redhat do not collaborate
2007	GPLv3
2007	Novell wins against SCO
2007	Microsoft revises "Get the Facts" web site
2007	Latest version of Joe Hesse's Linux notes

Table 2.1: Timeline for Open Source

2.11 Current Open Source News

The author has found the web site `http://news.yahoo.com/i/1817` to be a good source of current Open Source News.

Exercise 2.11.6: *Look at the `http://news.yahoo.com/i/1817` web site. Do a google search on Open Source and see what comes up.*

Part II

Installing Linux On Your Computer