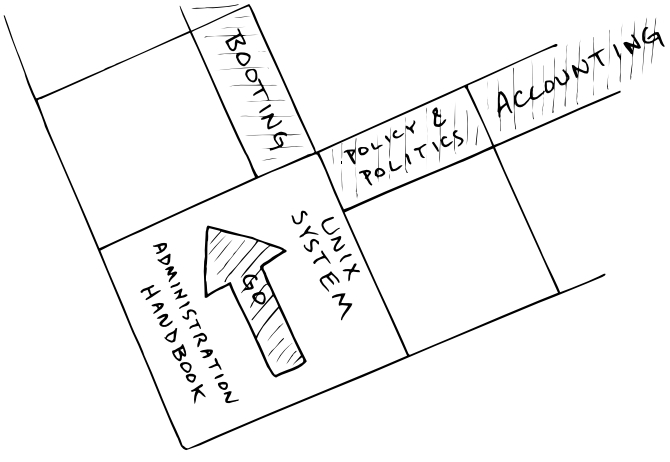
1 Where to Start



We set out to write a book that could be a system administrator's trusty companion, providing the practical advice, comfort, and basic system administration theory that you can't get from reading manual pages. As a result, this book is designed to complement—not replace—your system's documentation.

We think this book will help you in five ways:

- It will give you an overview of the major administrative systems, identifying the different pieces of each and explaining how they work together.
- It will introduce general administrative techniques that we have found, through experience, to be worthwhile.
- It will help you choose solutions that continue to work well as your site grows in size and complexity.
- It will help you sort good ideas from bad and educate you about various atrocities of taste committed by operating system developers.
- It will summarize common procedures so that you don't have to dig through the excessive detail of the manuals to accomplish simple tasks.

It's impossible to perform these functions with perfect objectivity, but we think we've made our biases fairly clear throughout the text. One of the interesting things about system administration is the fact that reasonable people can have dramatically different notions of what constitute the most appropriate policies and procedures. We

offer our subjective opinions to you as raw data. You'll have to decide for yourself how much to accept and to what extent our comments apply to your environment.

1.1 SUGGESTED BACKGROUND

We assume in this book that you have a certain amount of UNIX experience. In particular, you should have a general concept of how UNIX looks and feels from the user's perspective before jumping into administration. There are several good books that can get you up to speed; see the reading list on page 11.

You perform most administrative tasks by editing configuration files and writing scripts, so you must be familiar with a text editor. We strongly recommend that you learn **vi**. It is standard on all UNIX systems, and though it may appear a bit pale when compared with glitzier offerings such as **emacs**, it is perfectly usable. If you become addicted to another editor, you may soon tire of dragging it along with you to install on every new system. To the dismay of many, using Microsoft Word as one's only text editor is a significant impediment to effective system administration.

One of the mainstays of UNIX administration (and a theme that runs throughout this book) is the use of scripts to automate administrative tasks. To be an effective administrator, you must be able to read and modify **sh** scripts. Scripts that you write from scratch can be written in the shell or scripting language of your choice.

For new scripting projects, we recommend Perl. As a programming language, it is a little strange (OK, more than a little). However, it does include many features that are indispensable for administrators. The O'Reilly book *Programming Perl* by Larry Wall et al. is the standard text; it's also a model of good technical writing. A full citation is given on page 11.

We also recommend that you learn **expect**, which is discussed in a bit more detail starting on page 519. You will most likely pick up **expect** quite rapidly.

1.2 THE SORDID HISTORY OF UNIX

UNIX originated as a research project at AT&T Bell Labs in 1969. In 1976, it was made available at no charge to universities and thus became the basis of many operating systems classes and academic research projects.

In the late 1970s, AT&T created its UNIX Support Group (USG, later spun off as Unix System Laboratories, USL) to deploy UNIX as a commercial product. Bell Labs and USG both continued the development of UNIX, but the two groups' efforts diverged. USL's releases, System III and System V, were widely distributed and have had a proportionately greater impact on modern systems.

Berkeley UNIX began in 1977 when the Computer Systems Research Group (CSRG) at the University of California, Berkeley, licensed code from AT&T. Berkeley's releases (called BSD, for Berkeley Software Distribution) began in 1977 with 1BSD for the PDP-11 and culminated in 1993 with 4.4BSD.

Source licenses for AT&T releases were always expensive for nonacademic users. At first, the licenses were cheap or free for universities, but as UNIX gained commercial acceptance, the price rose rapidly. Eventually, Berkeley set the long-term goal of removing AT&T's code from BSD, a tedious and time-consuming process. Before the work could be completed, Berkeley lost funding for operating systems research and the CSRG was disbanded.

Before collapsing, the CSRG released its final collection of AT&T-free code, known as 4.4BSD-Lite. Most current versions of BSD UNIX (including BSD/OS, FreeBSD, NetBSD, and OpenBSD) claim the 4.4BSD-Lite package as their grandparent.

Although BSD and System V are the core systems from which most other versions of UNIX are derived, neither of these "pure" systems was ever much of a force in the commercial market. Typically, a vendor would start with a vanilla AT&T or BSD system and proceed to develop it independently. Some vendors didn't want to commit to one flavor of UNIX and supported both or developed hybrids that combined both sets of features. Not surprisingly, these UNIX variants tended to diverge from one another over time.

The most recent major development in the UNIX world has been the advent of the Linux kernel and the many UNIX systems that are now based upon it. Linux is a soup-to-nuts reimplementation of the UNIX kernel that was begun in 1991 as a personal project of Linus Torvalds, a Finnish graduate student. Over the years, the Linux project accumulated many developers, users, and enthusiasts. It has grown into a full-featured, production-quality kernel that many vendors support as their primary operating system. Many big-ticket commercial software packages (such as Oracle) have also been ported to Linux.

1.3 EXAMPLE UNIX SYSTEMS

In this book, we have chosen four popular UNIX variants as our examples: Solaris 2.7, HP-UX 11.00, Red Hat Linux 6.2, and FreeBSD 3.4. These systems are representative of the overall UNIX marketplace, and they're all so common that it's hard to find a UNIX site that doesn't use at least one of them.

Sun Microsystems' Solaris is a System V derivative with many extensions. Sun UNIX (yes, that's what it was called in the mid-80s) was originally the progeny of Berkeley UNIX, but a (now historic) corporate partnership between Sun and AT&T forced a change of platform.

HP's HP-UX is a hybrid of the System V and Berkeley UNIX trees, but with odd surprises of its own.

Several free UNIX systems are available for Intel hardware, and of these Linux is currently the most popular.[1] Linux itself is just a kernel; you must add on a full comple-

1. Linux has been ported to a variety of other hardware platforms, including the Nintendo64 video game system. Who says Nintendo doesn't make real computers?

ment of commands, utilities, and daemons to form a complete UNIX system. The various Linux "distributions" bundle the kernel with the other components needed for a full installation. Linux distributors make many choices while assembling their products, so versions of Linux can be quite different from one another. A few companies (including Red Hat, SuSE, and Corel) provide production-grade distributions with full support.

FreeBSD is a system based on Berkeley's 4.4BSD-Lite release. Like Linux, it runs on a variety of Intel platforms. A commercially supported version is available from BSDI.

1.4 NOTATION AND TYPOGRAPHICAL CONVENTIONS

In this book, filenames, commands, and literal arguments to commands are shown in boldface. Placeholders (e.g., command arguments that should not be taken literally) are in italics. For example, in the command

cp *file directory*

you're supposed to replace *file* and *directory* with the names of an actual file and an actual directory.

Excerpts from configuration files and terminal sessions are shown in a fixed-width font.[2] Sometimes, we annotate interactive sessions with italic text. For example:

```
% grep Bob /pub/phonelist          /* Look up Bob's phone # */
Bob Knowles 555-2834
Bob Smith 555-2311
```

Outside of these specific cases, we have tried to keep special fonts and formatting conventions to a minimum so long as we could do so without compromising intelligibility. For example, we often talk about entities such as the UNIX group daemon and the printer anchor-lw with no special formatting at all.

In general, we use the same conventions as the UNIX manual pages for indicating the syntax of commands:

- Anything between square brackets ("[" and "]") is optional.
- Anything followed by an ellipsis ("…") can be repeated.
- Curly braces ("{" and "}") indicate that you should select one of the items separated by vertical bars ("|").

For example, the specification

bork [**-x**] { **on** | **off** } *filename …*

would match any of the following commands:

```
bork on /etc/passwd
bork -x off /etc/passwd /etc/termcap
bork off /usr/lib/tmac
```

2. Actually, it's not really a fixed-width font, but it looks like one. We liked it better than the real fixed-width fonts that we tried. That's why the columns in some examples may not all line up perfectly.

We use shell-style globbing characters for pattern matching:

- A star (*) matches zero or more characters.
- A question mark (?) matches one character.
- A tilde or "twiddle" (~) means the home directory of the current user.
- ~*user* means the home directory of *user*.

For example, we sometimes refer to the BSD startup scripts **/etc/rc**, **/etc/rc.boot**, and **/etc/rc.local** with the shorthand pattern **/etc/rc***.

Text within quotation marks often has a precise technical meaning. In these cases, we ignore the normal rules of English and put punctuation outside the quotation marks so that there can be no confusion about what's included and what's not.

System-specific information

Information in this book generally applies to all four of our example systems unless a specific attribution is given. Details particular to one system are marked with the vendor's logo:


Solaris 2.7


HP-UX 11.00


Red Hat Linux 6.2


FreeBSD 3.4

These logos are used with the permission of their respective owners. However, the vendors have neither reviewed nor endorsed the contents of this book.

1.5 How to use your manuals

The UNIX manuals contain all the information needed to keep the system running, yet that information is sometimes hard to find and often cryptic. You *must* have access to a complete set of manuals for your version of UNIX. However, that does not necessarily mean that you need to buy printed books. Most documentation is available in electronic form, either as part of the system installation procedure or from the vendor's web site.

UNIX systems typically come with two types of documentation: "man pages" and supplemental documents. Man pages (so called because they are designed for use with the **man** command) are concise descriptions of individual commands, file formats, or library routines. They are usually kept on-line but may also be supplied in printed form.

Supplemental documents can include both individual articles and book or pamphlet-length treatments of particular topics. The supplemental materials are not limited to describing just one command, so they can adopt a tutorial or procedural approach. Many pieces of software have both a man page and an article. For example, the man page for **vi** tells you about the command-line arguments that **vi** understands, but you have to go to the supplement to learn how to actually edit a file.

Since the man pages are closely tied to the software they describe, vendors tend not to change them very much unless they modify the software itself.[3] Not so with the supplements—many vendors have entirely replaced the traditional manuals with new books and documents.

Many of the most important parts of UNIX are maintained by neutral third parties such as the Internet Software Consortium and the Apache Software Foundation. These groups typically provide adequate documentation for the packages they distribute. Vendors sometimes ship the software but skimp on the documentation, so it's often useful to check back with the original source to see if additional materials are available.

Another useful source of information about the design of many UNIX software packages is the "Request for Comments" document series, which describes the protocols and procedures used on the Internet. See page 263 for more information.

Organization of the man pages

All UNIX systems divide the man pages into sections. However, the exact definition of each section varies among systems. The basic organization of the man pages is shown in Table 1.1.

Table 1.1 Sections of the UNIX man pages

Solaris HP-UX	Linux FreeBSD	Contents
1	1	User-level commands and applications
2	2	System calls and kernel error codes
3	3	Library calls
4	5	Standard file formats
5	7	Miscellaneous files and documents
6	6	Games and demonstrations
7	4	Device drivers and network protocols
1m	8	System administration commands
9	9	Obscure kernel specs and interfaces

Many systems further subdivide the man pages in each section. For example, section 3m often contains man pages about the system's math library. There is also considerable variation in the exact distribution of pages; some systems leave section 8 empty and lump the system administration commands into section 1. A lot of systems have discontinued games and demos, leaving nothing in section 6.

Most systems allow you to create a section of the manuals called "l" for local man pages. Another common convention is section "n" for software that isn't strictly local but isn't standard, either.

3. But this is not always the case. HP has done an excellent job of editing the man pages.

troff input for man pages is traditionally kept in the directories **/usr/man/man***X*, where *X* is a digit **1** through **9**, or **l** or **n**. Formatted versions of the manuals are kept in **/usr/man/cat***X*. The **man** command will format man pages on the fly; if the **cat** directories are writable, **man** will also deposit the formatted pages as they are created, generating a cache of commonly read man pages. You can preformat all man pages at once with the **catman** command if space permits. On some systems, such as FreeBSD, the man pages have been moved to **/usr/share/man**. Many systems compress their man pages with **compress** or **gzip** to save space.

Solaris now uses SGML as the formatting language for most man pages instead of the traditional **troff**. Man pages in **troff** format are still supported, but they are kept in separate directories.

man: read manual pages

man *title* formats a specific manual page and sends it to your terminal via **more** (or whatever program is specified in your PAGER environment variable). *title* is usually a command, device, or filename. The sections of the manual are searched in roughly numeric order, although sections that describe commands (sections 1, 8, and 6) are usually searched first.

The form **man** *section title* gets you a man page from a particular section. Thus, **man tty** gets you the man page for the **tty** command and **man 4 tty** gets you the man page for the serial driver.

Under Solaris, you must preface the section number with the **-s** flag, for example, **man -s 4 tty**.

Almost all versions of **man** check to see if you have defined the MANPATH environment variable, which should contain a colon-separated list of directories if it exists. MANPATH overrides or extends the list of directories that **man** searches. For example, the command

```
setenv MANPATH /home/share/localman:/usr/share/man
```

in your **.login** file would cause **man** to search a hierarchy of local man pages before **/usr/man**. The **sh** version would be

```
MANPATH=/home/share/localman:/usr/share/man
export MANPATH
```

On some systems, MANPATH completely overrides the default search path, so you must explicitly include the default directory if you want to continue to see your vendor's man pages.

man -k *keyword* prints a list of man pages that have *keyword* in their one-line synopses. For example:

```
% man -k translate
gftype (1L)  - translate a font file for humans to read
pktype (1L)  - translate a packed font file
tr (1)       - translate characters
```

The keywords database is normally kept in a file called **whatis** in the root of the man page hierarchy (**/usr/man** or **/usr/share/man**). If you add additional man pages to your system, you may need to rebuild this file with **catman -w**.

1.6 ESSENTIAL TASKS OF THE SYSTEM ADMINISTRATOR

The sections below give an overview of some tasks that system administrators are typically expected to perform. These duties need not necessarily be performed by one person, and at many sites the work is distributed among several people. However, there does need to be at least one person who understands all of the chores and makes sure that someone is doing them.

Adding and removing users

See Chapter 6 for more information about adding new users.

The system administrator adds accounts for new users and removes the accounts of users that are no longer active. The process of adding and removing users can be automated, but certain administrative decisions (where to put the user's home directory, on which machines to create the account, etc.) must still be made before a new user can be added.

When a user should no longer have access to the system, the user's account must be disabled. All of the files owned by the account must be backed up to tape and disposed of so that the system does not accumulate unwanted baggage over time.

Adding and removing hardware

See Chapters 8, 12, and 23 for more information about these topics.

When new hardware is purchased or when hardware is moved from one machine to another, the system must be configured to recognize and use that hardware. Hardware-support chores can range from the simple task of adding a printer to the more complex job of adding a disk drive.

Performing backups

See Chapter 10 for more information about backups.

Performing backups is perhaps the most important job of the system administrator, and it is also the job that is most often ignored or sloppily done. Backups are time-consuming and boring, but they are absolutely necessary. Backups can be automated and delegated to an underling, but it is still the system administrator's job to make sure that backups are executed correctly and on schedule.

Installing new software

When new software is acquired, it must be installed and tested, often under several versions of UNIX and on several types of hardware. Once the software is working correctly, users must be informed of its availability and location. Local software should be installed in a place that makes it easy to differentiate from the system software. This organization simplifies the task of upgrading the operating system since the local software won't be overwritten by the upgrade procedure.

Monitoring the system

Large UNIX installations require vigilant supervision. Daily activities include making sure that email and web service are working correctly, watching log files for early signs of trouble, ensuring that local networks are all properly connected, and keeping an eye on the availability of system resources such as disk space.

Troubleshooting

UNIX systems and the hardware they run on occasionally break down. It is the administrator's job to play mechanic by diagnosing problems and calling in experts if needed. Finding the problem is often harder than fixing it.

Maintaining local documentation

See page 809 for suggestions regarding documentation.

As the system is changed to suit an organization's needs, it begins to differ from the plain-vanilla system described by the documentation. It is the system administrator's duty to document aspects of the system that are specific to the local environment. This chore includes documenting any software that is installed but did not come with the operating system, documenting where cables are run and how they are constructed, keeping maintenance records for all hardware, recording the status of backups, and documenting local procedures and policies.

Auditing security

See Chapter 21 for more information about security.

The system administrator must implement a security policy and periodically check to be sure that the security of the system has not been violated. On low-security systems, this chore might only involve only a few cursory checks for unauthorized access. On a high-security system, it can include an elaborate network of traps and auditing programs.

Helping users

Although helping users with their various problems is rarely included in a system administrator's job description, it claims a significant portion of most administrators' workdays. System administrators are bombarded with problems ranging from "My program worked yesterday and now it doesn't! What did you change?" to "I spilled coffee on my keyboard! Should I pour water on it to wash it out?"

1.7 How to find files on the Internet

A wealth of information about system administration is available, in many forms. A list of "starter" resources can be found in Chapter 27, *Policy and Politics*.

The Internet is by far the richest deposit of information. You can type questions about system administration topics into any of the popular search engines, such as www.yahoo.com, www.altavista.com, www.google.com, and www.webopedia.com.

Many sites cater directly to the needs of system administrators. Here are a few that we especially like:

- freshmeat.com – a huge collection of Linux software
- www.ugu.com – the Unix Guru Universe; lots of stuff for sysadmins
- www.stokely.com – a good collection of links to sysadmin resources
- www.tucows.com – Windows and Mac software, filtered for quality
- slashdot.org – "the place" for geek news
- www.cpan.org – a central source for Perl scripts and libraries
- securityfocus.com – security info; huge searchable vulnerability database

1.8 SYSTEM ADMINISTRATION UNDER DURESS

System administrators wear many hats. In the real world, they are often people with other jobs who have been asked to look after a few computers on the side. If you are in this situation, you may want to think a bit about where it might eventually lead.

The more you learn about UNIX, the more the user community will come to depend on you. Networks invariably grow, and you may be pressured to spend an increasing portion of your time on administration. You will soon find that you are the only person in your organization who knows how to perform a variety of important tasks.

Once coworkers come to think of you as the local system administrator, it is difficult to extricate yourself from this role. We know several people that have changed jobs to escape it. Since many administrative tasks are intangible, you may also find that you're expected to be both a full-time administrator and a full-time engineer, writer, or secretary.

Some unwilling administrators try to fend off requests by adopting an ornery attitude and providing poor service. We do not recommend this approach; it makes you look bad and creates additional problems.

Instead, we suggest that you document the time you spend on system administration. Your goal should be to keep the work at a manageable level and to assemble evidence that you can use when you ask to be relieved of administrative duties. In most organizations, you will need to lobby the management from six months to a year to get yourself replaced, so plan ahead.

On the other hand, you may find that you enjoy system administration and that you yearn to be a full-time administrator. You will have no problem finding a job. Unfortunately, your political problems will probably intensify. Refer to Chapter 27, *Policy and Politics*, for a preview of the horrors in store.

System Administration Personality Syndrome

One unfortunate but common clinical condition resulting from working as a system administrator is System Administration Personality Syndrome. The onset of this condition usually begins early in the third year of a system administrator's career and the

syndrome can last well into retirement. Characteristic symptoms include but are not limited to:

- Acute phantom pagerphobia: the disturbing feeling that your pager has gone off (when it really hasn't) and that your peaceful evening with your significant other is about to abruptly end, resulting in a 72-hour work marathon without food
- User voodoographia: the compulsive creation of voodoo-doll representations of the subset of your user population that doesn't seem to understand that their persistent lack of planning doesn't constitute an emergency in your world
- Idiopathic anal tapereadaplexia: the sudden, late-night urge to mount backup tapes to see if they're actually readable and labeled correctly
- Scientifica inapplicia: the strong desire to violently shake fellow system administrators who seem never to have encountered the scientific method

Many curative therapies can be used to treat this unfortunate condition. The most effective are a well-developed sense of humor and the construction of a small but well-endowed office wine cellar. You might also consider the more meditative approach of silently staring off into space and clicking your heels together whenever the words "Is the server down again?" are spoken in your vicinity. If all else fails, take a vacation.

1.9 Recommended reading

Anderson, Gail, and Paul Anderson. *The UNIX C Shell Field Guide*. Englewood Cliffs, NJ: Prentice Hall. 1986.

Hewlett-Packard Company. *The Ultimate Guide to the VI and EX Text Editors*. Redwood City, CA: Benjamin/Cummings. 1990.

Abrahams, Paul W., and Bruce A. Larson. *UNIX for the Impatient, 2nd Edition*. Reading, MA: Addison-Wesley. 1995.

Peek, Jerry, Tim O'Reilly, and Mike Loukides. *UNIX Power Tools, 2nd Edition.* Sebastopol, CA: O'Reilly & Associates. 1997.

Montgomery, John, and Woody Leonhard. *The Underground Guide to Unix: Slightly Askew Advice from a Unix Guru*. Reading, MA: Addison-Wesley. 1995.

Reichard, Kevin, and Eric Foster-Johnson. *Unix in Plain English, 3rd Edition.* Foster City, CA: IDG Books Worldwide. 1999.

Rankin, Bob. *The No BS Guide to Linux.* No Starch Press. 1997.

Wall, Larry, Tom Christiansen, and Randal L. Schwartz. *Programming Perl, 2nd Edition.* Sebastopol, CA: O'Reilly & Associates. 1997.