Linux+ Certification Workshop

Student Workbook

itcourseware
TRAINING MATERIALS FOR IT PROFESSIONALS

Linux+ Certification Workshop

Jeff Howell and Rob Roselius

Published by ITCourseware, LLC., 7245 South Havana Street, Suite 100, Centennial, CO 80112

Contributing Authors: Jef Barnhart, Brandon Caldwell, Mark Copley, Jeff Howell, and James Lopeman.

Editor: Rick Sussenbach

Assistant Editor: Jan Waleri

Special thanks to: Many instructors whose ideas and careful review have contributed to the quality of this workbook, including Larry Burley and Richard Raab, to Linus Torvalds and the many anonymous Linux developers and volunteers who created and maintain Linux and the Linux Documentation Project, and the many students who have offered comments, suggestions, criticisms, and insights.

Contents

CHAPTER 1 - COURSE INTRODUCTION

COURSE OBJECTIVES

- ❈ Manage files and directories by creating, copying, renaming, removing, editing, and performing other operations on them.
- ❈ Create and modify files using the **vi** editor.
- ❈ Use many Linux utilities for text file manipulation, file system management, and communication with other users.
- ❈ Take advantage of Linux security mechanisms to protect files and programs from unauthorized access and to configure shared access among work groups.
- ❈ Increase your productivity by generating file names using “wild card” pattern matching.
- ❈ Look up commands and other information in the on-line Linux reference manuals.
- ❈ Explain the purpose of shell programs.
- ❈ Recognize applications for shell programs.
- ❈ Design and write shell programs of moderate complexity.
- ❈ Explain the responsibilities of a Linux system administrator and perform many of the “hands-on” tasks required to manage a Linux system.
- ❈ Manage user accounts, including adding and deleting users, changing user account attributes, and controlling password requirements.
- ❈ Describe the design and operation of Linux file systems.
- ❈ Administer disks and file systems.
- ❈ Use standard Linux utilities to backup, archive, and compress files.

- ❈ Manage Linux processes, including starting and killing jobs in the background, and scheduling jobs to run once or repeatedly at selected times.
- ❈ Startup and shutdown Linux systems.
- ❈ Identify and control every process and service running at every run level.
- ❈ Give details of Linux security issues and implement techniques for secure system software and hardware.
- ❈ Monitor the performance of a multi-user Linux system and do simple performance tuning operations.
- ❈ Configure network interfaces, name resolution, routes and services, and file sharing.
- ❈ Administer the print spooling system.
- ❈ Install and manage software packages.
- ❈ Configure common internet servers.

COURSE OVERVIEW

- **Audience**: This course is designed for users who are new to Linux and seeking the CompTIA Certification in Linux+. This course will help you prepare for the Certification exam, however completion of this course alone is *not* adequate preparation to pass the exam. The actual Linux+ Certification is designed to measure the competencies of Linux professionals who have six to twelve months of practical experience with the Linux operating system. Thus, after taking this course you need more study, practice, and experience prior to taking the exam.

- **Prerequisites**: We assume that you have used a computer before and have a basic understanding of how to interact with one. Any previous UNIX or Linux experience you may have will be very helpful, whether as a user, administrator, or programmer.

- **Student Materials**:

  - Student Workbook

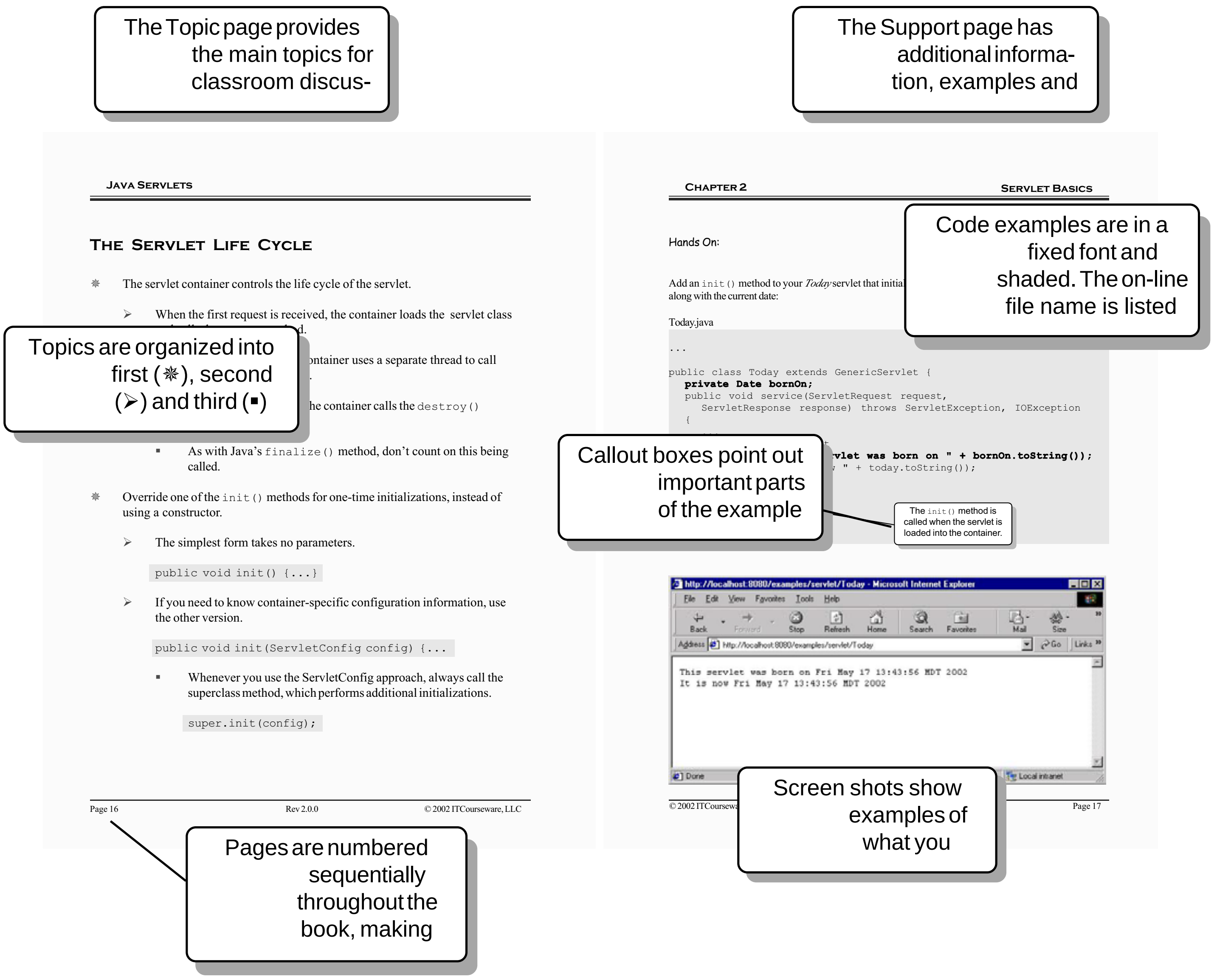
- **Classroom Environment:**

  - Linux system with one terminal per student.

Using the Workbook

This workbook design is based on a page-pair, consisting of a Topic page and a Support page. When you lay the workbook open flat, the Topic page is on the left and the Support page is on the right. The Topic page contains the points to be discussed in class. The Support page has code examples, diagrams, screen shots and additional information. **Hands On** sections provide opportunities for practical application of key concepts. **Try It** and **Investigate** sections help direct individual discovery.

In addition, there is an index for quick look-up. Printed lab solutions are in the back of the book as well as on-line if you need a little help.


The Topic page provides the main topics for classroom discus-
The Support page has additional informa-tion, examples and
Topics are organized into first (❋), second (➢) and third (▪)
Code examples are in a fixed font and shaded. The on-line file name is listed
Callout boxes point out important parts of the example
Screen shots show examples of what you
Pages are numbered sequentially throughout the book, making

JAVA SERVLETS

THE SERVLET LIFE CYCLE

❋ The servlet container controls the life cycle of the servlet.

➢ When the first request is received, the container loads the servlet class

d.

ontainer uses a separate thread to call

.

he container calls the `destroy()`

▪ As with Java's `finalize()` method, don't count on this being called.

❋ Override one of the `init()` methods for one-time initializations, instead of using a constructor.

➢ The simplest form takes no parameters.

```
public void init() {...}
```

➢ If you need to know container-specific configuration information, use the other version.

```
public void init(ServletConfig config) {...
```

▪ Whenever you use the ServletConfig approach, always call the superclass method, which performs additional initializations.

```
super.init(config);
```

Page 16 Rev 2.0.0 © 2002 ITCourseware, LLC

CHAPTER 2 SERVLET BASICS

Hands On:

Add an `init()` method to your *Today* servlet that initial along with the current date:

Today.java

```
...

public class Today extends GenericServlet {
  private Date bornOn;
  public void service(ServletRequest request,
    ServletResponse response) throws ServletException, IOException
  {
                    vlet was born on " + bornOn.toString());
                    " + today.toString());
```

The `init()` method is called when the servlet is loaded into the container.

```
This servlet was born on Fri May 17 13:43:56 MDT 2002
It is now Fri May 17 13:43:56 MDT 2002
```

© 2002 ITCoursew Page 17

Suggested References

Albitz, Paul and Cricket Liu. 2001. ***DNS and BIND, Fourth Edition***. O'Reilly & Associates, Sebastopol, CA. ISBN 0596001584.

Bailey, Edward. 1997. ***Maximum RPM***. SAMS, Indianapolis, IN. ISBN 0672311054.

Frisch, Aeleen. 2002. ***Essential System Administration, Third Edition***. O'Reilly & Associates, Sebastopol, CA. ISBN 0596003439.

Garfinkel, Simson and Gene Spafford. 2003. ***Practical Unix and Internet Security, Third Edition***. O'Reilly & Associates, Sebastopol, CA. ISBN 0563003234.

Hekman, Jessica Perry, et al. 2003. ***Linux in a Nutshell, Fourth Edition***. O'Reilly & Associates, Sebastopol, CA. ISBN 0596004826.

Hunt, Craig. 2002. ***TCP/IP Network Administration, Third Edition***. O'Reilly & Associates, Sebastopol, CA. ISBN 0596002971.

Kirch, Olaf and Terry Dawson. 2000. ***Linux Network Administrator's Guide, Second Edition***. O'Reilly & Associates, Sebastopol, CA. ISBN 1565924002.

Nemeth, Evi, Garth Snyder and Trent R. Hein. 2002. ***Linux System Administration Handbook***. Prentice Hall, Upper Saddle River, NJ. ISBN 0130084662.

Stern, Hal, Mike Eisler, and Ricardo Labiaga. 2001. ***Managing NFS and NIS, Second Edition***. O'Reilly & Associates, Sebastopol, CA. ISBN 1565925106.

Ts, Jay, Robert Eckstein, and David Collier-Brown. 2003. ***Using Samba***. O'Reilly & Associates, Sebastopol, CA. ISBN 0596002564.

Welsh, Matt, Dallheimer, Matthias and Kaufman, Lar. 2002. ***Running Linux , Fourth Edition***. O'Reilly & Associates, Sebastopol, CA. ISBN 0596002726.

The facing page lists many fine books on Linux. Of course, the best information on Linux is on-line:

man	The trusty manpages.
/usr/share/doc/	Documentation on Linux and packages.
/usr/src/	The source code!
http://www.tldp.org	Linux Documentation Project (LDP) official site.
http://www.kernel.org	Linux kernel source official site.
http://www.linux.com	Linux meta-site: news, Linux links.
http://www.linux.org	Linux meta-site: news, Linux links.
http://www.linux.org.uk/cgi-bin/portaloo	Linux news portal.
http://www.freshmeat.net	Today's new software.
news:comp.os.linux	General Linux discussion, questions, and answers.
news:comp.os.linux.admin	Linux administration.
news:comp.os.linux.advocacy	Religion.
news:comp.os.linux.development.apps	For application developers.
news:comp.os.linux.development.system	For Linux hackers.
news:comp.os.linux.help	Help for beginners.
news:comp.os.linux.misc	General Linux discussion, questions, and answers.
news:comp.os.linux.networking	Linux networking.
news:comp.os.linux.questions	General Linux questions and answers.
news:comp.os.linux.security	Linux security issues, discussions, and techniques.
news:comp.os.linux.setup	Linux installation and configuration.
news:comp.os.linux.x	X Windows setup and configuration.

Linux Journal - The Monthly Magazine of the Linux Community. SSC, Seattle, WA. Monthly.

Sys Admin - The Journal for Unix Systems Administrators. Miller Freeman, San Francisco, CA. Monthly.

CHAPTER 2 - GETTING STARTED

OBJECTIVES

- ❅ Provide a high-level overview of the background of UNIX and Linux.
- ❅ Explain the relationship between UNIX and Linux.
- ❅ Log in and out of the system.
- ❅ Run a few commands.
- ❅ Change your password.
- ❅ Look up commands in the on-line Linux reference manuals.

What is UNIX?

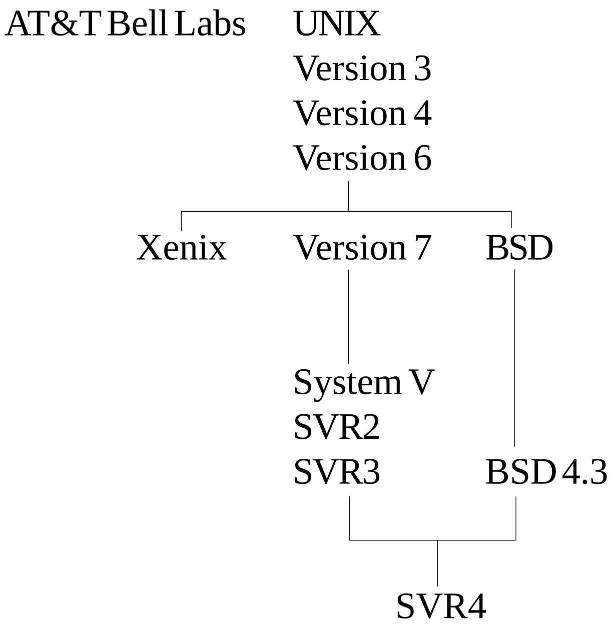
- UNIX is a general purpose, multi-tasking, multi-user, interactive, scalable, computer operating system.
- HP-UX, SunOS, Solaris, AIX, SCO UNIX, System V.4, etc., are all versions of UNIX.

Why UNIX?

- In the 1980s, UNIX became commercially popular for several reasons:
  - Customer demand for the benefits of open systems:
    - Application portability
    - Vendor independence
    - Connectivity and interoperability in multivendor environments
    - User portability
  - New workstation hardware could be brought to market more quickly with an existing operating system.
  - Major vendors (such as Sun, DEC, HP, IBM) implemented UNIX-based product lines.
  - UNIX provides excellent networking capabilities.

UNIX is an operating system (OS), not a programming language. It supports software development in almost all languages, such as C, C++, Java, COBOL, Pascal, FORTRAN, BASIC, Smalltalk, LISP, etc. It is also widely used in client/server configurations, as the OS that supports server applications.

A Brief History of UNIX



1970	UNIX invented as a pet project by Ken Thomson.
1975	Microsoft founded.
1977	UNIX, rewritten in C, ported to various hardware.
1980	Xenix introduced.
1981	MS-DOS introduced.
1982	Sun Microsystems founded.
1984	AT&T releases System V.
1980s	UNIX market growth and fragmentation.
1989	SVR4 introduced.
1991	Linux invented. World Wide Web invented.
1993	Windows NT launched.
1994	Java invented.

- ❈ UNIX was originally written at Bell Laboratories in 1970.
- ❈ In the mid-1970s, the University of California at Berkeley began making additions and enhancements to UNIX.
- ❈ In the early 1980s, AT&T began offering support for AT&T System III UNIX.
- ❈ In 1989, AT&T released System V, Release 4 (SVR4).
  - ➢ SVR4 merged features of System V, BSD, and Xenix.
  - ➢ Sun used SVR4 as the basis of Solaris.

UNIX was originally designed and written by Ken Thompson for the purpose of aiding in computer science research. AT&T provided UNIX source code at a low cost to many universities, including the University of California At Berkeley (UCB). Berkeley UNIX, built upon the Sixth Edition, added, over the years, many utilities, such as **vi** and **csh**; it was distributed under the name BSD (Berkley Software Distribution). Much research and development was done in the areas of file systems and networking. The history is well documented in several books, such as: McKusick, Marshall Kirk, et. al. 1996. ***The Design and Implementation of the 4.4BSD UNIX Operating System***. Addison-Wesley, Reading, MA. ISBN 0201549794.

AT&T, in 1982, merged several internal versions of UNIX and began licensing UNIX to vendors, such as Hewlett-Packard. In 1985, AT&T began shipping UNIX System V, and committed to support it and maintain backward compatibility in future versions of UNIX.

In 1989, AT&T's UNIX System Labs released System V, Release 4 (SVR4). SVR4 merged features of System V, BSD, and Microsoft's Xenix.

The diagram on the facing page is oversimplified with respect to the number of actual versions and variants of UNIX and its relatives, and with respect to the cross-influencing that the various versions have had on each other.

Linux

- In 1991, Linus Torvalds began writing his own UNIX-like operating system kernel, while studying an academic UNIX variant called Minix.

  - Working on a 386 PC using the GNU C compiler (which he ported to Minix), he posted his source code on the Internet.

  - Other programmers began contributing to this new, free operating system, which Linus named *Linux*.

- Linux itself is the operating system kernel, which includes:

  - Basic bootstrap code
  - Kernel routines (process scheduling, memory mgt., etc.)
  - Hardware device drivers (disk, sound, video, network, etc.)
  - File system drivers

- To make a Linux system useful, you'll install your choices from among the thousands of freely available (or commercial) programs.

  - GNU software (UNIX utilities, development tools, etc.)
  - User command-line shells (*bash*, *ksh*, etc.)
  - Network servers (Apache httpd, WU ftpd, Sendmail, etc.)
  - Graphical User Interfaces (Gnome, KDE, AfterStep, etc.)
  - Office applications (Netscape, StarOffice, etc.)

- You can download the latest Linux kernel source code from a variety of locations.

  - To install and run it, you must compile the kernel, set up a filesystem, install the kernel, build the programs you want, etc.

  - This is no simple task; it's much easier to use one of the pre-built, pre-packaged *Linux distributions*.

MINIX is a small UNIX-like system for IBM PCs and compatibles. Little used anymore, Minix is a copyrighted product that can be downloaded in source code form for educational and research uses. In April of 1991, 21-year-old student and Minix enthusiast Linus Torvalds, in order to learn about operating systems and the Intel 386 architecture, began writing a Minix-like operating system, which he named Linux (a weak pun on "Unix" and "Linus"). He developed it using a Minix 386.

By the October of 1991, he had a working system (version 0.2) that he announced on the *comp.os.minix* USENET newsgroup. He released version 0.12, the stable version that began the spread of Linux, on January 5, 1992. His main interest was sharing information with fellow hobbyists, and he didn't believe it would ever be possible to port Linux to any non-386 platform. From the beginning, he provided the Linux source code to anyone at no charge under the GNU Public License (GPL).

Between 1992 and 1998, Linus and an ever-growing community of developers around the world continued enhancing and porting Linux. Version 1.0 came out in 1994, and in 1995 they succeeded in porting Linux to the Alpha and PowerPC platforms. Enhancements included such sophisticated operating system features as IPC, threading, symmetric multi-processing, a sophisticated file system, and networking.

While Linux was, and still is, freely available, building and installing it from source code is not trivial. Various groups began in the mid-1990s to package pre-built, ready-to-boot Linux *distributions,* usually including various pre-built utilities. Early, influential distributions included Yggdrasil's easy-to-install Plug-and-Play Linux and Slackware, geared toward Linux developers.

Linux has been ported to a remarkable number of different hardware platforms, from high-end servers and parallel clusters to a Web server host the size of a matchbox. Many new porting projects are underway all the time.

In the mid- to late-1990s, system and network administrators, frustrated with trying to run network services on NT systems, but unable to afford commercial UNIX hosts, snuck Linux PCs into their shops to run Web, file, print, DNS, firewall, and other servers. Corporations, encouraged by such open-source successes as Perl and the Apache Web server, began to embrace low-cost, high-power Linux solutions. Distributors packaged high-quality, enterprise-ready Linux distributions, and Linux was launched.

The Toolkit Philosophy

- Provide the user with simple utilities (sometimes called *filters* or *tools*) that do one thing well.

- Provide the user with simple ways to combine the various tools to build more powerful applications.

- Filters - take in a text stream (**stdin**) and produce a text stream (**stdout**).

  - Error messages are reported on a third stream (**stderr**).

    ```
    $ cat phonelist | grep "(303)" | sort | more
    ```

    - *phonelist* is fed to **grep,** which finds all (303) numbers and feeds them to **sort,** which sorts them and feeds them to **more,** which pages them to screen.

- Pipes - the pipe character (|) does "plumbing" to connect the output of one command to the input of the next.

  - A series of commands connected by the pipe character is called a *pipeline*.

- Command Substitution - another powerful method of combining tools is to capture their output into a variable:

  ```
  $ textfile_count=`ls *.txt | wc -l`
  ```

  - The backticks (`) cause the output of the pipeline to be converted to a string, which is then saved in the variable *textfile_count*.

- Try to keep the toolkit philosophy in mind as you learn all the various Linux commands that follow. Think of how you could use them, not only by themselves, but also in combination with other commands.

There are many people who use UNIX or Linux who, in my humble opinion, do not understand UNIX. UNIX is not just an operating system, it is a way of doing things, and the shell plays a key role by providing the glue that makes it work. The UNIX methodology relies heavily on reuse of a set of tools rather than on building monolithic applications.

— David Korn, Author of The Korn Shell

Perhaps the most powerful method of combining our Linux tools into powerful applications is the use of shell scripts. With scripts we can automate any combination of commands we know and also take advantage of programming language constructs like loops, conditional tests, variables, and input/output operations and other shell capabilities.

Linux Distributions

- ❈ Although Linux is free, building and installing it from scratch is challenging and time consuming, even for experienced engineers.
- ❈ Linux distributions package a pre-built, tested Linux operating system along with various utilities and applications.
- ❈ Most distributions are very low cost (<$100, compared to >$1000 for comparable commercial UNIX systems).
  - ➢ In addition, most vendors allow you to download their distribution at no charge over the Internet.
- ❈ Just about any distribution will include all the essential software for running a network server or development workstation, plus much more stuff than you'll ever need.
- ❈ Linux, by its nature, evolves constantly.
  - ➢ New Linux versions incorporate fixes and new features.
  - ➢ Distibutors offer new features, software, and services with new distributions.
  - ➢ Some changes, for better or worse, may break compatibility with existing applications.
- ❈ The Linux operating system itself runs with great stability, reliability, and high performance.
  - ➢ Typical Linux hosts can run for months or years without rebooting.

Dozens of Linux distributions have sprung up, all sharing the same Linux kernel source and differing in bundled software, packaging, installation, and support.

In 1995, Red Hat Linux began marketing its commercially-oriented distribution of Linux, bundled with hundreds of free applications and its Red Hat Package Management system (RPM). They released Red Hat Linux 4.0 for the Intel, Alpha, and Sparc platforms in 1995, and Red Hat Linux 6.0 in 1999.

The Debian project began in 1993, under the auspices of the Free Software Foundation. Its intention is to provide a packaged operating system built primarily from GNU utilities and the Linux kernel, with the work done by a distributed collection of volunteer developers and maintainers. It includes a package management system, *dpkg*, and a large amount of packaged software. In 1998, they released Debian GNU/Linux 2.0 for the Intel and Motorola platforms, which included over 1500 packages maintained by over 400 developers. Debian's GNU/Linux 2.1, adding support for Alpha and Sparc, came out in 1999, as did GNU/HURD, a version of Debian featuring GNU's HURD kernel instead of the Linux kernel.

Caldera Systems, founded in 1994, markets the OpenLinux commercial distribution. It uses the Red Hat Package Manager for software package installation and management. Caldera released OpenLinux 2.2 in 1999.

SuSE, founded in Germany as *Gesellschaft für Software-und Systementwicklung mbH* in 1992, shipped its SuSE Linux 1.0 distribution in 1992 and was the leading Linux distributor in Europe as of the late 1990s. It features RPM, the StarOffice office suite, and YaST, a graphical system administration tool. SuSE Linux 6.2 for Intel was released in 1999.

Linux is a registered trademark of Linus Torvalds.

FREE SOFTWARE AND OPEN SOURCE MOVEMENTS

- Linux provided a full operating system kernel.
  - It was immediately useful, and ultimately successful, due to the vast array of utilities and libraries available from the GNU Project.
  - As it grew, these also became available from other Free and Open Source projects.
- The GNU Project (for GNU is Not Unix) and the Free Software Foundation, started by Richard Stallman (the foremost guru and advocate of free software), set out in 1984 to "develop a complete Unix-like operating system which is free software."
  - To legally define what "free software" meant, the FSF developed the GNU General Public License (GPL) and the term *copyleft* to replace the term copyright (as in "there's always a copy left!").
- When Linux was released, it filled in a missing piece for GNU, which was the yet-to-be completed kernel.
  - Since the majority of the code that makes Linux usable comes from GNU, Stallman and others hold that it should be referred as "GNU/Linux."
- Numerous other projects producing free software have emerged.
  - Although, with mild ideological splits between camps, some projects are billed as "Open Source" rather than "Free Software."
    - For the most part, the basic intention is the same: software that is freely available to use, modify, and propagate for the good of all.
- Various entities have evolved to support Open Source and Free Software development: consortiums, foundations, nonprofit organizations, university projects, and corporate initiatives.

Amazing amounts of full-featured, high-quality, reliable software have come out of the Open Source and Free Software movements. Just a few …

- X Window System - X.Org Consortium, *www.x.org*
- GNU Project - Free Software Foundation, *www.gnu.org*
- OpenOffice - OpenOffice.org Source Project, *www.openoffice.org*
- Gnome Office - Gnome Project, *www.gnome.org/gnomeoffice*
- Apache - The Apache Software Foundation, *www.apache.org*
- Mozilla - Mozilla Foundation, *www.mozilla.org*

From the FSF web page "The Free Software Definition" by Richard Stallman:

"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."

Free software is a matter of the users' freedom to run, copy, distribute, study, change and improve the software. More precisely, it refers to four kinds of freedom, for the users of the software:

- *The freedom to run the program, for any purpose (freedom 0).*
- *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.*
- *The freedom to redistribute copies so you can help your neighbor (freedom 2).*
- *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.*

From the paper "The Cathedral and The Bazaar" by Eric S. 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?

I had been preaching the Unix gospel of small tools, rapid prototyping and evolutionary programming for years. But I also believed there was a certain critical complexity above which a more centralized, a priori approach was required. I believed that the most important software (operating systems and really large tools like Emacs) needed to be built like cathedrals, carefully crafted by individual wizards or small bands of mages working in splendid isolation, with no beta to be released before its time.

Linus Torvalds'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 differing 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.

LOGGING IN

❋ To be able to use a Linux system interactively, you must first have a user account that you can log in to.

❋ Linux will prompt by displaying **login:**, and you log in by typing your user account name. Your instructor will tell you what your user name will be for this class.

```
login: tsmith
Password:    (You may not have a password, but if you do, it will not be
              echoed when you type it.)

$ ls         ( A list of file names will appear as output of the ls command.)
$
```

❋ The dollar sign is your prompt.

- ➢ The prompt you see may be different.
- ➢ Prompts are configurable by the user.

❋ **ls** is a command that you type.

- ➢ It means "LiSt the names of my files."
- ➢ The **ls** command has nothing to do with logging in; it is just a very common command that you can type *after* you log in.

Each user of a particular Linux system has a unique user account name. This name is also known as a User ID. The user names on a system may follow a convention of "First initial - last name" or "Last name - first initial," or maybe even "First name - last initial." They may be employee numbers. Unless you are the System Administrator of a machine, your account name will be assigned to you by the System Administrator.

Who do you think **tsmith** is?

It is possible for an account to have no password. Most administrators require that all users put passwords on their accounts. We will see how to create and change a password shortly.

The prompt that Linux displays is configurable by each user. In this workbook, we will always show the prompt as a dollar sign, like this: `$`

LOGGING OUT

- To log out, type **exit**.
- You can also log out by pressing <Ctrl>d.

```
$ exit

login:
```

- Now you are ready to log in again on your terminal.

The **exit** command will always log you out.

A *shell* is the program that runs on your terminal (or window) when you log in. It displays the prompt and waits for your command. Thus, a shell is sometimes called a *command interpreter*. The word "shell" comes from the idea that the program is acting as a shell around the Linux operating system, i.e., acting as the interface between you and Linux. By the way, the innermost part of the Linux operating system is called the *kernel*, so the analogy is that we have a shell around a kernel.

There are several shell programs available on Linux, and some provide additional ways to log out. The C shell (**csh**) lets you type **exit** or **logout**. All shells will let you log out with <Ctrl>d, but they may have an option set that ignores <Ctrl>d so that you don't inadvertently log out by accidentally pressing <Ctrl>d.

We will study shells in much more detail later in the course. For now, just use **exit** to log out, and be aware that there are other ways. (Of course, you might want to try logout and <Ctrl>d. Please experiment in this class!)

TRY A FEW MORE COMMANDS

- After logging in, try each of these commands, just to get some hands-on practice.

  - Remember, the dollar sign (**$**) represents the Linux shell prompt. Don't type the dollar sign!

`$ who`	Shows who is currently logged on. (May not work properly on an X-terminal.)
`$ who am i`	Shows who you are.
`$ date`	Shows the date and time.
`$ cal`	Shows calendar of the current month.
`$ cas`	A non-existent command we use to demonstrate an error message.

CHANGING YOUR PASSWORD

❋ You can create a password if you don't have one, or change an existing one, with the **passwd** command.

❋ In the following example, the user **tsmith** selects the password **fl0wer?**:

```
$ passwd
Changing password for tsmith
(current) UNIX password:                 tsmith's password
New UNIX password: fl0wer?               Not echoed
Retype new UNIX password: fl0wer?        Not echoed
```

❋ The new password must be used at the next login.

There are rules that apply to passwords. To read them, type:

```
   $ man passwd
```

This will display the manual entry for the **passwd** command.

The next page describes how to use the on-line Linux reference manuals.

If you forget your password, it must be reset by the System Administrator. Not even the SA can find out what your password is.

ONLINE DOCUMENTATION - MAN PAGES

❈ Most Linux systems keep electronic copies of the Linux reference manuals in files on disk. They are accessible with the **man** command (**man** is short for manual).

❈ To read the manual entry for a particular command, for example **passwd**, type:

```
$ man passwd
```

❈ The entry for the **passwd** command will be displayed on your terminal, pausing when a full page has been displayed.

❈ Press the <Enter> key for the next line, the <Space> bar to display the next page.

❈ To quit without displaying the rest of the manual entry, use the q (for quit) key.

❈ Sometimes you see the names of commands written in the form **command(*N*)** where ***N*** is a number. For example:

```
passwd (1)           - change user password
passwd (5)           - password file
```

- ➢ This means that there is documentation for **passwd** in both Sections 1 and 5 of the Linux reference manuals (specified by including it in the **man** command).

```
$ man 5 passwd
```

Descriptions of commands in the on-line documentation are known for their terseness. It has been said that there are three rules to be aware of when reading Linux manual entries:

Rule 1. Every word counts.
Rule 2. They only say it once.
Rule 3. You must know a fair amount of terminology before reading most entries.

We recommend that during your Linux learning curve you have a more tutorially-oriented book available, such as those listed under Suggested References and Reading in Chapter 1 of this workbook. Over time, the on-line manuals can become a more efficient way for you to look things up.

On another note, the **man** command works fine if you know the name of the command that you want to read about. But what if you want to change your password and you don't know the name of the command to use? Well, the **man** command has a useful option, **-k**, which will display the titles of all commands that contain the specified keyword. For example:

```
$ man -k password
chpasswd (8)          - update password file in batch
dpasswd (8)           - change dialup password
gpasswd (1)           - administer the /etc/group file
htpasswd (1)          - Create and update user authentication files
mkpasswd (8)          - Update passwd and group database files
passwd (1)            - change user password
passwd (5)            - password file
```

After perusing the titles of the commands, you can pick the one that you need.

The Linux manual is divided into topical sections as follows:

Section 1	Commands	Section 5	File Formats
Section 2	System Calls	Section 6	Games
Section 3	Library Functions	Section 7	Miscellany
Section 4	Special Files	Section 8	System Administration Commands

Online Documentation - info Pages

❋ A large portion of the commands used in Linux were developed or influenced by the GNU Project of the Free Software Foundation

❋ The preferred method for GNU Project documentation is the *Texinfo* file.

➢ A *Texinfo* file is structured like a book, with chapters, sections, sub-sections, (and even sub-sub-sections), each making up a node of the documentation.

➢ A single *Texinfo* file can be processed by different programs to produce its documentation in a variety of forms, including **man** pages, Info files, HTML, and print.

❋ The **info** command (with a user interface based on Emacs) provides a character-based hypertext system to browse the nodes of an Info file.

➢ **info** can be run with no arguments to access the top level node of the documentation (also called the *directory node*), or with the name of a topic or command to access specific information.

```
$ info               access the top level of the documentation
$ info passwd        get information on the passwd command
$ info regex         get information on the regular expression library
```

❋ A given node can connect to others with hyperlinks or cross-references.

➢ A cross-reference shows as a line beginning with an asterisk.

```
* Invoking GCC::       Command options supported by `gcc'
```

❋ To visit a cross-reference, merely move the cursor anywhere on that line and press <Enter>.

There are limitless ways of navigating documentation in **info**. Almost anything you know from Emacs will work. For the full story, use the **info** command on itself: **info info**. In lieu of that, here are some addition useful navigation commands:

<Tab>	move to the next cross reference
<Space>	page down
<Delete>	page up
<Ctrl>L	redraw the screen

B	Beginning of node
D	move to Directory node
E	End of node
I	search node Index for text
L	move to Last node visited (like back on a web browser)
M	Select a Menu item by name
N	Next node
P	Previous node
Q	Quit!
S	Search node for text
T	move to Top level node
U	Up from this node

In addition to the **man** pages and Info files, Linux systems will often have a wide variety of information available in the directories */usr/doc* or */usr/share/doc*. Here you can find quick reference Frequently Asked Questions files in the *FAQ* subdirectory; longer, tutorial "how to" files in the *HOWTO* subdirectory; and documentation for many of the installed software packages in the various other subdirectories.

If this documentation was omitted when Linux was installed on your machine, you can find most of the same information (and often more up-to-date) on the web. A great starting place is the Linux Documentation Project at *www.tldp.org*.

Labs

❶ Study the manual entry for the **cal** command. On what day of the week were you born?

❷ Execute the **who** command with the command-line option that puts a header on the output. Options on Linux commands are preceded by a hyphen.

```
$ ls -l
$ man -k who
```

❸ Use the **wc** command to print the number of bytes, words, and lines in the file *memo*.

❹ Read about some of the options to **wc** in its manual entry. Use **wc** to print just the number of lines in *memo* and *bigfile*.

Questions you need answers to in order to be successful on your system when you get back to work:

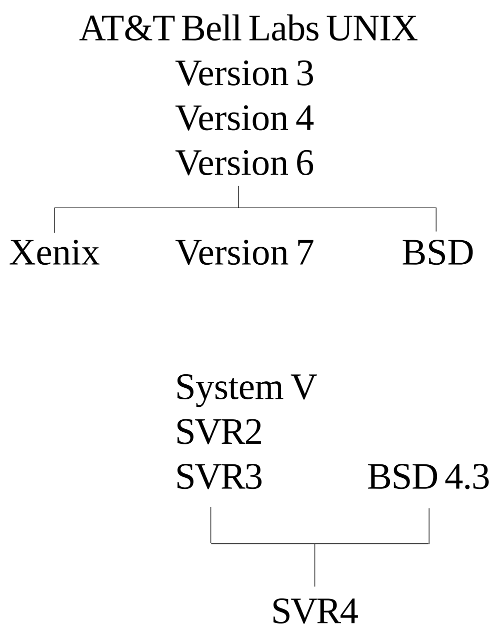
1. What is my login name?
2. What is my password?
3. Which key backspaces?
4. How do I interrupt a command? (Usually <Ctrl>c)

Chapter 14 - Overview of System Administration

Objectives

- ❋ Explain the history and current status of UNIX and Linux.
- ❋ Work with the different flavors of Linux.
- ❋ Use the Linux documentation effectively.

A BRIEF HISTORY OF UNIX



1970	UNIX invented as a pet project by Ken Thompson.
1975	Microsoft founded.
1977	UNIX, rewritten in C, ported to various hardware.
1980	Xenix introduced.
1981	MS-DOS introduced.
1982	Sun Microsystems founded.
1984	AT&T releases System V.
1980s	UNIX market growth and fragmentation.
1989	SVR4 introduced. World Wide Web invented.
1991	Linux invented.
1993	Windows NT launched.
1994	Java invented.

❈ In the early days of UNIX, the UNIX operating system source code was freely available from its inventor, Ken Thompson of Bell Labs.

- ➢ Many computer scientists and students obtained it for study.
- ➢ Many companies licensed it for commercial use.

❈ Many commercial variants of UNIX were marketed, by such vendors as Sun, IBM, Hewlett-Packard, SCO, SGI, etc.

- ➢ Although there have been many derivatives of UNIX, the most recognized versions are based on AT&T System V and BSD.

❈ Source code for a few variants (mostly BSD) is still available, but most modern UNIX products are proprietary and expensive.

The MULTICS project, a collaboration between AT&T Bell Labs, MIT, and GE, collapsed in 1969. With their free time and expertise, several of the Bell Labs project members, notably Ken Thompson and Dennis Ritchie, began development of a small, simple time-sharing operating system, which they named UNIX (a weak pun on "Multics"). It ran on a DEC PDP-7, and was multi-user and multi-tasking.

Between 1973 and 1977, Thompson and Ritchie rewrote the bulk of UNIX in C, with the goal of easy portability. In 1977, they and others succeeded in porting UNIX to several different computer architectures, and in 1979 Bell Labs released UNIX Version 7.

While AT&T used UNIX internally, and licensed it to other companies and to universities, it could not market it as a product. This was due to a 1956 consent decree prohibiting AT&T from participating in the computer market.

In 1979, Thompson taught an Operating Systems class at the University of California at Berkeley (UCB) and left behind a copy of UNIX Version 6. Bill Joy (later a co-founder of Sun Microsystems) and others at UCB began making enhancements to the operating system, and distributed it as BSD (Berkeley Software Distribution) UNIX. Many important enhancements came out of Berkeley.

When AT&T divested its telephone monopoly in 1984, it was allowed to compete in the computer market, and aggressively marketed UNIX. In 1982, it had released a version of UNIX which it called System III; in 1984, it released System V, and subsequent releases have retained this name.

Dozens of UNIX variants proliferated over the years. Xenix, Microsoft's port of UNIX to the microcomputer, was an influential product. FreeBSD, an open-source variant of Berkeley UNIX, gained a wide following among both hobbyists and professionals.

In 1989, AT&T's UNIX System Labs (USL), released System V, Release 4 (SVR4). This release merged features of AT&T, BSD, and Xenix. Sun Microsystems subsequently abandoned its BSD-based SunOS, for SVR4-based Solaris. Sun released Solaris 2.6 in 1997 and Solaris 7 in 1999.

In 1993, Novell bought USL, which had released SVR4.2 (a release geared toward desktop computers). Novell ceded the UNIX trademark to X/Open, and marketed SVR4.2 as UnixWare. In 1995, SCO took control of SVR4.2 from Novell, and in 1998 released UnixWare7.

LINUX

- ❋ In 1991, Linus Torvalds began writing his own UNIX-like operating system kernel while studying an academic UNIX variant called Minix.
  - ➢ Working on a 386 PC using the GNU C compiler (which he ported to Minix), he posted his source code on the Internet.
  - ➢ Other programmers began contributing to this new, free operating system, which Linus named *Linux*.
- ❋ Linux itself is the operating system kernel, which includes:
  - ➢ Basic bootstrap code.
  - ➢ Kernel routines (process scheduling, memory mgt., etc.).
  - ➢ Hardware device drivers (disk, sound, video, network, etc.).
  - ➢ File system drivers.
- ❋ To make a Linux system useful, you'll install your choices from among the thousands of freely available (or commercial) programs.
  - ➢ GNU software (UNIX utilities, development tools, etc).
  - ➢ User command-line shells (*bash*, *ksh*, etc.).
  - ➢ Network servers (Apache httpd, WU ftpd, Sendmail, etc.).
  - ➢ Graphical User Interfaces (Gnome, KDE, AfterStep, etc.).
  - ➢ Office applications (Netscape, StarOffice, etc.).
- ❋ You can download the latest Linux kernel source code from a variety of locations.
  - ➢ To install and run it, you must compile the kernel, set up a filesystem, install the kernel, build the programs you want, etc.
  - ➢ This is no simple task; it's much easier to use one of the pre-built, prepackaged *Linux distributions*.

MINIX is a small UNIX-like system for IBM PCs and compatibles. Little-used any more, Minix is a copyrighted product that can be downloaded in source code form for educational and research uses. In April of 1991, 21-year-old student and Minix enthusiast Linus Torvalds, in order to learn about operating systems and the Intel 386 architecture, began writing a Minix-like operating system which he named Linux (a weak pun on "Unix" and "Linus"). He developed it using a Minix 386.

By October of 1991, he had a working system (version 0.2), which he announced on the *comp.os.minix* USENET newsgroup. He released version 0.12, the stable version that began the spread of Linux, on January 5, 1992. His main interest was sharing information with fellow hobbyists, and he didn't believe it would ever be possible to port Linux to any non-386 platform. From the beginning, he provided the Linux source code to anyone at no charge under the GNU Public License (GPL).

Between 1992 and 1998, Linus and an ever-growing community of developers around the world continued enhancing and porting Linux. Version 1.0 came out in 1994, and in 1995 they succeeded in porting Linux to the Alpha and PowerPC platforms. Enhancements included such sophisticated operating system features as IPC, threading, symmetric multiprocessing, a sophisticated file system, and networking.

While Linux was, and still is, freely available, building and installing it from source code is not trivial. Various groups began in the mid-1990s to package prebuilt, ready-to-boot Linux *distributions*, usually including various prebuilt utilities. Early, influential distributions included Yggdrasil's easy-to-install Plug-and-Play Linux and Slackware, geared toward Linux developers.

Linux has been ported to a remarkable number of different hardware platforms, from high-end servers and parallel clusters to a web server host the size of a matchbox. Many new porting projects are underway all the time.

In the mid- to late-1990s, system and network administrators, frustrated with trying to run network services on NT systems but unable to afford commercial Unix hosts, snuck Linux PCs into their shops to run web, file, print, DNS, firewall, and other servers. Corporations, encouraged by such open-source successes as Perl and the Apache web server, began to embrace low-cost, high-power Linux solutions. Distributors packaged high-quality, enterprise-ready Linux distributions and Linux was launched.

The authoritative source of Linux itself, that is, the kernel, is *http://www.kernel.org*.

Linux Distributions

- Although Linux is free, building and installing it from scratch is challenging and time consuming, even for experienced engineers.

- Linux distributions package a pre-built, tested Linux operating system, along with various utilities and applications.

- Most distributions are very low-cost (<$100, compared to >$1000 for comparable commercial UNIX systems).

  - In addition, most vendors allow you to download their distribution at no charge over the Internet.

- Just about any distribution will include all of the essential software for running a network server or development workstation, plus much more stuff than you'll ever need.

- Linux, by its nature, evolves constantly.

  - New Linux versions incorporate fixes and new features.

  - Distibutors offer new features, software, and services with new distributions.

  - Some changes, for better or worse, may break compatibility with existing applications.

- The Linux operating system itself runs with great stability, reliability, and high performance.

  - Typical Linux hosts can run for months or years without rebooting.

Scores of Linux distributions have sprung up, all sharing the same Linux kernel sources and differing in bundled software, packaging, installation, and support. Among the most commercially important are:

Red Hat Linux: Red Hat began marketing its commercially-oriented distribution of Linux bundled with hundreds of free applications and its Red Hat Package Management (RPM) system. In 2002, Red Hat released their Red Hat Enterprise, with a modified Linux kernel, less frequent release cycle, and subscription-based distribution model. Red Hat stopped supporting their "standard" distribution after Red Hat Linux 9, and, renaming it **Fedora**, began sponsoring it as a community-supported project.

Debian: The Debian project began in 1993 under the auspices of the Free Software Foundation. Its intention is to provide a packaged operating system built primarily from GNU utilities and the Linux kernel, with the work done by a distributed collection of volunteer developers and maintainers. It includes a package management system, **dpkg**, and a large amount of packaged software.

SuSE: SuSE, founded in Germany as *Gesellschaft für Software-und Systementwicklung mbH*, shipped its SuSE Linux 1.0 distribution in 1992, and was the leading Linux distributor in Europe in the late 1990s. It features RPM, the StarOffice suite, and YaST, a graphical system administration tool.

Other popular or influential distributions include:

TurboLinux: The predominant commercial Linux distribution for Asia and the Pacific rim.

Slackware: Traditionally the "hacker's Linux," Slackware attempts to be the most UNIX-like distribution.

Mandrake: Following the spirit of the Red Hat Linux products, Mandrake is a popular commercial distribution built around ease of installation and use, features, and packaging.

Knoppix: A bootable CD-ROM distribution, Knoppix allows you to run Linux on your PC without installing it on your hard drive.

Gentoo: Another hacker's or power-user's Linux, with software packaging based on BSD's portage system.

Linux Online! (*http://www.linux.org/*) maintains an extensive list of known distributions. As of mid-2004, some 225 were listed. Linux has been adapted and packaged to serve widely-varying needs, including international language support, real-time (RT) applications, high-availability, embedded systems, beginning-user support, super-compact distributions, security, etc.

Linux is a registered trademark of Linus Torvalds.

ONLINE DOCUMENTATION - THE MAN PAGES

❋ Linux is extraordinarily well-documented, starting with traditional UNIX manpages for all system utilities.

- Manpages are organized into sections:

  1. User commands
  2. System calls
  3. Library functions
  4. Special files
  5. File formats
  6. Games
  7. Miscellany
  8. Administration and privileged commands

- The **man** command returns the first manpage it finds, unless you specify a section (see **man(1)**).

man passwd	# The **passwd** command, in section 1
man 1 passwd	# Same thing
man 5 passwd	# Format of the */etc/passwd* file, in section 5

❋ Documentation for packages included in your distibution should be available under */usr/share/doc/*.

❋ The Internet provides many Linux resources, especially the Linux Documentation Project (LDP).

- LDP has detailed guides, succinct how-tos, and informative FAQs.

❋ And finally, for the brave, remember that you can always read the source code!

Mirrored worldwide, the Linux Documentation Project's official Web site is:

http://www.tldp.org/

ONLINE DOCUMENTATION - THE info PAGES

- For some commands, you can find detailed documentation in the **info** pages.

```
info ls
```

- The **info** pages consist of multiple, linked nodes (pages) you can navigate with your keyboard.

n	Next node
p	Previous node
u	Go up one menu
<Space>	Scroll down
<Backspace>	Scroll up
h	Help

- Just about every Linux command has a **man** page; only certain commands have **info** pages.

CHAPTER 24 - NETWORKING FUNDAMENTALS

OBJECTIVES

- ❋ Describe the information that is needed in order to communicate on ethernet or TCP/IP.
- ❋ Explain the types of Internet name resolution.
- ❋ Use DNS tools to resolve domain names.

IP ADDRESSES AND NETMASKS

❈ The Internet Protocol (IP) distinguishes network hosts by assigning each one a unique number: its *IP address*.

❈ An IP address is a 32-bit number that we usually write as 4 smaller (8-bit, if you will) numbers separated by dots: *192.168.1.1*

❈ The *netmask* determines how much of the IP address specifies the network.

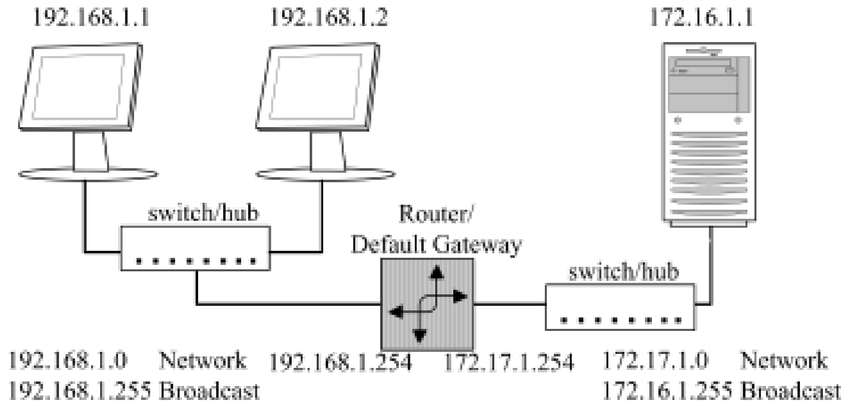
Network	Host
255.255.255	0
192.168.1	1

❈ Data is broadcast to the entire network; each host listens only for data addressed to its own IP address.

❈ All hosts also listen to the special host address called the *broadcast address* for the network: *192.168.1.255*.

❈ For two hosts to communicate, they must be on the same network (have the same network values).

- ➢ *Routers* allow hosts in different networks to communicate by forwarding data from one network to another.



Originally, the range of all possible IP addresses was organized into network classes, based on the first part of the address. CIDR is now a standard for describing IP addresses / Subnet standards.

Class A: 1.0.0.0 - 127.0.0.0
  Subnet: 255.0.0.0
  CIDR notation: 10.0.0.0/8
    Allowing 1.6 million hosts per network.
  Reserved IP address range: 10.0.0.0 - 10.255.255.255 and 127.0.0.1

Class B: 128.0.0.0 - 191.255.0.0
  Subnet: 255.255.0.0
  CIDR notation: 10.0.0.0/16
    Allowing 65,024 hosts per network.
  Reserved IP address range: 172.16.0.0 - 172.31.0.0

Class C: 192.0.0.0 - 223.255.255.0,
  Subnet: 255.255.255.0
  CIDR notation: 10.0.0.0/24
    Allowing 254 hosts per network.
  Reserved IP address range: 192.168.0.0 - 192.168.255.0

Reserved IP address ranges should never be attached directly to the internet.
Read RFC1918 for more information.

(The remaining IP addresses, 224.0.0.0 on up, were reserved for experimental or other uses.)

This class organization has been largely abandoned to make more addresses available, though you'll still hear reference to such classes.

Linux also supports a new version of the Internet Protocol: IPv6 (Internet Protocol version 6). IPv6 is designed to improve upon the traditional IP's (known as IPv4) scalability, security, ease-of-configuration, and network management. IPv6 addresses are 128 bits instead of 32 bits, and are typically written in colon-separated hexidecimal form:

FEDC:BA98:7654:3210:8:800:200C:417A

NAME RESOLUTION

- For the convenience of human users, network hosts are given names.
- When we reference a host by name, a TCP/IP program must first resolve the hostname into an IP address.
- Hostname-to-IP address resolution is handled by library functions.
  - This allows the resolver library to do the work, instead of the application.
- Two of the central functions are **gethostbyname()** and **gethostbyaddress()**.
- When these are called, the resolver code reads a local config file to determine how to resolve the name.
  - Two files are used for configuration:
    - */etc/host.conf (lib5)*
    - */etc/nsswitch.conf (glibc2/lib6)*
- The most common methods of name resolution are:
  - A local host file: */etc/hosts.*
  - A query of a remote DNS server (which usually runs the BIND implementation of DNS).

The old */etc/host.conf* file:

```
order hosts,bind
multi on
```

multi on allows for multiple IPs for single host in */etc/hosts*.

The */etc/nsswitch.conf* file:

```
#hosts:     db files nisplus nis dns
hosts:      files dns
```

THE /ETC/HOSTS FILE

- The */etc/hosts* file lists the IP addresses associated with known hostnames.

- Entries in */etc/hosts* consist of an IP address and a hostname, which can be followed by optional aliases for that host.

```
# Do not remove the following line, or various progams
# that require network functionality will fail.
127.0.0.1       localhost         localhost.localdomain
192.168.1.1     studentpc         studentpc.example.com
65.105.210.210  www.example.com   www
```

- DNS (the Domain Name System) simplifies name resolution, but each host still can have its own */etc/hosts* file.

The first one or two entries in */etc/hosts* usually refer to the local host. One special IP address on every system, 127.0.0.1, always refers to the local host.

DNS CONFIGURATION

- For the resolver to use DNS, it must be able to contact a name server.
- The configuration is held in the */etc/resolv.conf* file.
- */etc/resolv.conf* normally contains the following:
  - **nameserver** - the IP addresses for one or more name servers.
  - Either **search** (a list of domains to search, in order, for hostname lookup) or **domain** (the local domain name).

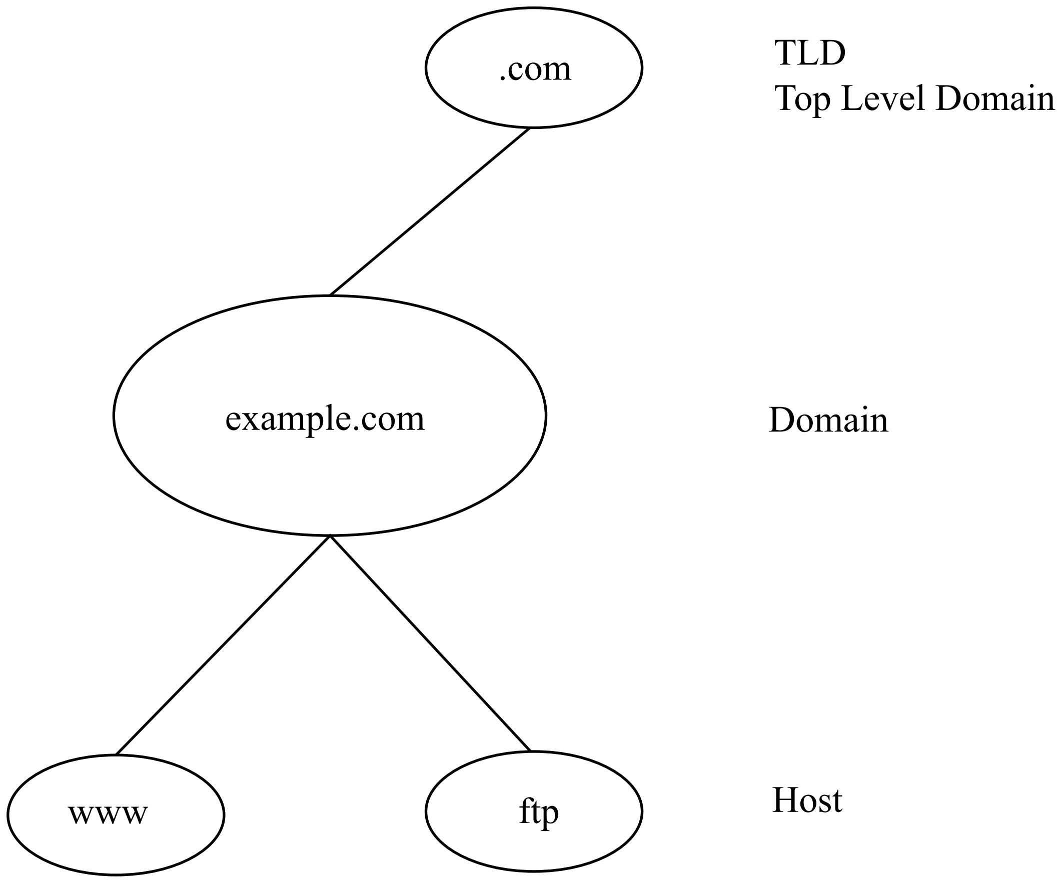
```
search example.com alt.example.com
nameserver 192.168.1.1
nameserver 65.105.210.222
```

or

```
domain example.com
nameserver 192.168.1.1
nameserver 65.105.210.222
```

DNS groups hostnames into domains, whose names and IP addresses are managed separately. In DNS, a single machine maintains a database of hostnames and IP addresses for its domain.

To get the IP address of a host in some other domain, a program contacts a local *nameserver* and gives it the hostname. The local nameserver contacts the nameserver of the other domain and passes the hostname to look up. To find the remote nameserver, the local nameserver may first have to query the root nameserver for the remote domain's top-level domain (TLD). If the hostname is in the other domain's database, the remote nameserver provides the IP address for the host. The local nameserver relays the IP address back to the client program. The local nameserver remembers (*caches*) the IP address for future reference.



DNS Tools

- You can query DNS records directly using the **nslookup**, **host**, and **dig** commands.
- These tools allow you to convert a hostname to an IP or an IP to a hostname.
    - If you do not specify a server to query, the tool will use the server or servers in the */etc/resolv.conf* file on your local host.
- **nslookup** has two modes of use, interactive and non-interactive.
    - Non-interactive is used if a hostname or IP address is given as an argument.
    - Interactive is used when no hostname is given as an argument.
        - This will give you a prompt and you can then enter a hostname or IP address.
        - Use **exit** (or <Ctrl>D) to quit.
- **host** is a simple utility to return a hostname or IP adresses.
- **dig** also has two modes of use – command-line arguments and batch mode.
    - Command line arguments is the more common use for this tool.
    - Batch mode allows you to do queries using a file.
- **whois** will query the domain name registration databases maintained by domain registrars.
    - The information that is returned contains the following:
        - the registrant
        - who it was registered through
        - administrative contact
        - technical contact
        - domain servers

```
nslookup -sil www.example.com
Server:    172.17.0.1
Address:172.17.0.1#53

Name: www.example.com
Address: 65.105.210.210""
nslookup -sil
>
```

```
host www.example.com
www.example.com has address 65.105.210.210

host 65.105.210.210
210.210.105.65.in-addr.arpa domain name pointer www.example.com.
```

```
dig www.example.com
; <<>> DiG 9.2.2-P3 <<>> www.example.com
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 31909
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 2

;; QUESTION SECTION:
;www.example.com.     IN A

;; ANSWER SECTION:
www.example.com. 3600  IN A  65.105.210.210

;; AUTHORITY SECTION:
example.com.  3600  IN NS ns2.example.com.
example.com.  3600  IN NS ns1.example.com.

;; ADDITIONAL SECTION:
ns2.example.com. 3600  IN A  65.105.210.222
ns1.example.com. 3600  IN A  65.105.210.217

;; Query time: 1 msec
;; SERVER: 172.17.0.1#53(172.17.0.1)
;; WHEN: Tue Jun  8 11:45:23 2004
;; MSG SIZE  rcvd: 122
```

Default Route

- The *Default Route* or *Default Gateway* is the destination for any traffic that is not destined for the local network or local machine.
- You can use the command **route** or **netstat -r** to view the routing table.
- When the system starts up, it reads the default route from the */etc/sysconfig/network* file (on Red Hat systems.)

```
NETWORKING=yes
HOSTNAME=classtek.batky-howell.com
GATEWAY=172.17.0.254
```

- A datagram will be matched like this:
  - Any datagrams that match the network address of 172.17.0.0 and 169.254.0.0 will be given to the ethernet adapter.
  - Any datagrams that match the network address of 127.0.0.0 will be given to the local loopback adapter.
  - Any datagrams that do not match the network address of 127.0.0.0, 172.17.0.0 or 169.254.0.0 will be routed through the ethernet and sent to the gateway address, to be sent to the next router.

```
route
169.254.0.0      *            255.255.0.0      U     0     0     0     eth0
172.17.0.0       *            255.255.0.0      U     0     0     0     eth0
127.0.0.0        *            255.0.0.0        U     0     0     0     lo
default    172.17.0.254      0.0.0.0          UG    0     0     0     eth0
```

According to the Internet's Request for Comments (RFC) 1594, a *datagram* is:

A self-contained, independent entity of data carrying sufficient information to be routed from the source to the destination computer without reliance on earlier exchanges between this source and destination computer and the transporting network.

Labs

❶ When resolving names, will your classroom host's resolver use the local hosts file first, or DNS?

❷ In order to find a remote host's IP address using DNS, what must be true?

❸ If your classroom host uses DNS, what nameserver or nameservers will it query?

❹ If DNS is available in you classroom, use the DNS tools to look up some popular Web sites. For a domain of your choice, what are its nameservers? What is the mail exchange host? Who is the domain name registered to?

❺ What default route is configured for your classroom host?