Overview of Linux : Kernel

- The base of Linux is the kernel. You could replace each and every library, but as long as the Linux kernel remained, it would still be Linux.
- The kernel contains device drivers, memory management, process management and communication management.
- The kernel hacker gurus follow POSIX guidelines which sometimes makes programming easier and sometimes harder.

Overview of Linux : Kernel

- A UNIX operating system consists of a kernel and some system programs. There are also some application programs for doing work.
- The kernel is the heart of the operating system. It keeps track of files on the disk, starts programs and runs them concurrently, assigns memory and other resources to various processes, receives packets from and sends packets to the network, and so on.
- The kernel does very little by itself, but it provides tools with which all services can be built.

Overview of Linux : Kernel

- It also prevents anyone from accessing the hardware directly, forcing everyone to use the tools it provides. This way the kernel provides some protection for users from each other.
- The system programs use the tools provided by the kernel to implement the various services required from an operating system.

Overview of Linux : Kernel

- System programs, and all other programs, run `on top of the kernel', in what is called the user mode.
- The difference between system and application programs is one of intent: applications are intended for getting useful things done (or for playing, if it happens to be a game), whereas system programs are needed to get the system working.
- A word processor is an application; telnet is a system program. The difference is often somewhat blurry, however, and is important only to compulsive categorizers.

Overview of Linux : Kernel

- An operating system can also contain compilers and their corresponding libraries (GCC and the C library in particular under Linux), although not all programming languages need be part of the operating system.
- Documentation, and sometimes even games, can also be part of it.
- Traditionally, the operating system has been defined by the contents of the installation tape or disks; with Linux it is not as clear since it is spread all over the FTP sites of the world.

Overview of Linux : init

- The single most important service in a UNIX system is provided by init .
- init is started as the first process of every UNIX system, as the last thing the kernel does when it boots.
- When init starts, it continues the boot process by doing various startup chores (checking and mounting filesystems, starting daemons, etc).
- The exact list of things that init does depends on which flavor it is; there are several to choose from.

Overview of Linux : init

- init usually provides the concept of single user mode, in which no one can log in and root uses a shell at the console; the usual mode is called multiuser mode.
- Some flavors generalize this as run levels; single and multiuser modes are considered to be two run levels, and there can be additional ones as well, for example, to run X on the console.

Overview of Linux : init

- In normal operation, init makes sure getty's are working (to allow users to log in), and to adopt orphan processes (processes whose parent has died; in UNIX all processes must be in a single tree, so orphans must be adopted).
- When the system is shut down, it is init that is in charge of killing all other processes, unmounting all filesystems and stopping the processor, along with anything else it has been configured to do.

Overview of Linux : login

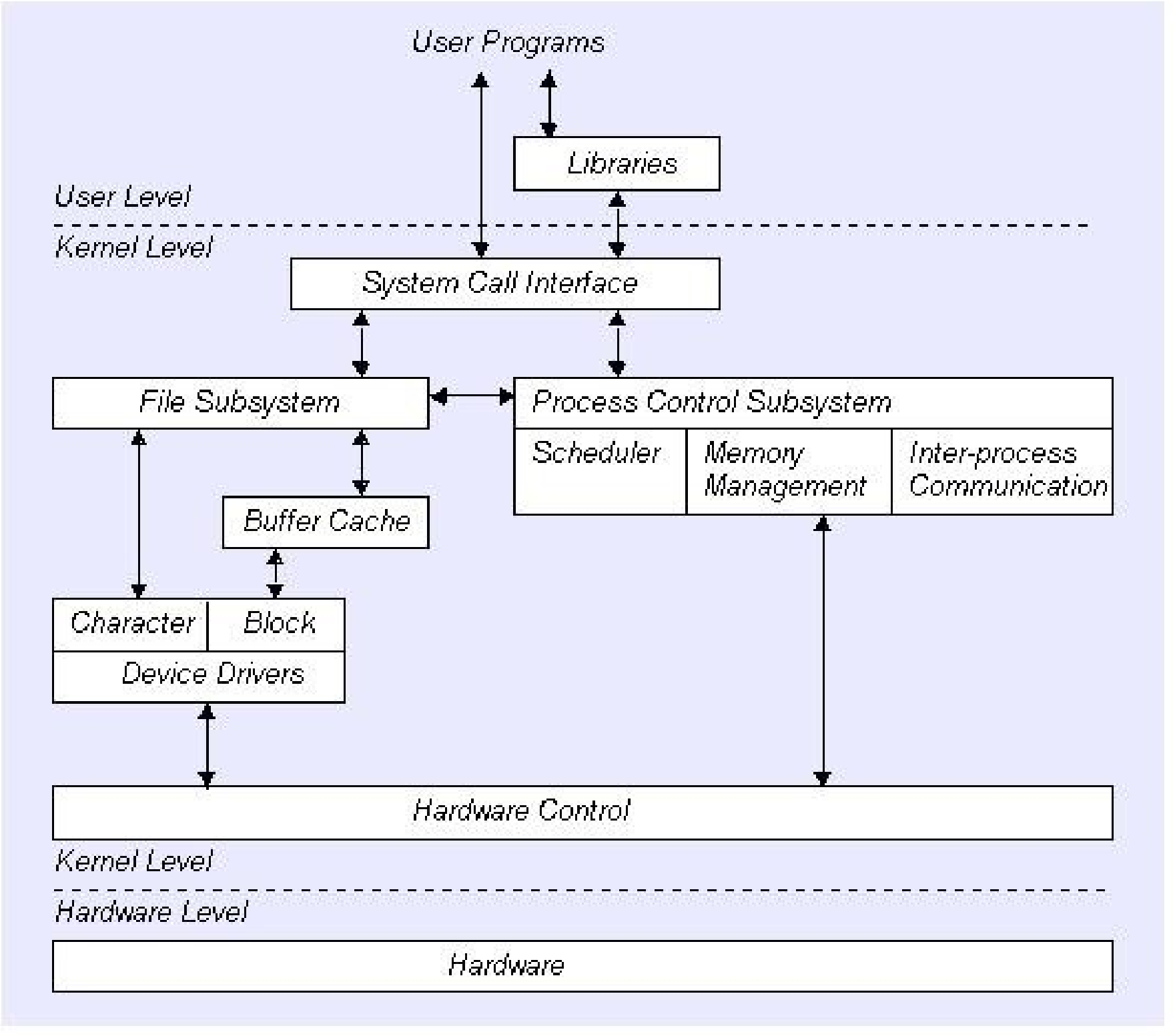
- Logins from terminals (via serial lines) and the console (when not running X) are provided by the getty program.
- init starts a separate instance of getty for each terminal for which logins are to be allowed.
- getty reads the username and runs the login program, which reads the password. If the username and password are correct, login runs the shell.
- When the shell terminates, i.e., the user logs out, or when login terminated because the username and password didn't match, init notices this and starts a new instance of getty .
- The kernel has no notion of logins, this is all handled by the system programs.

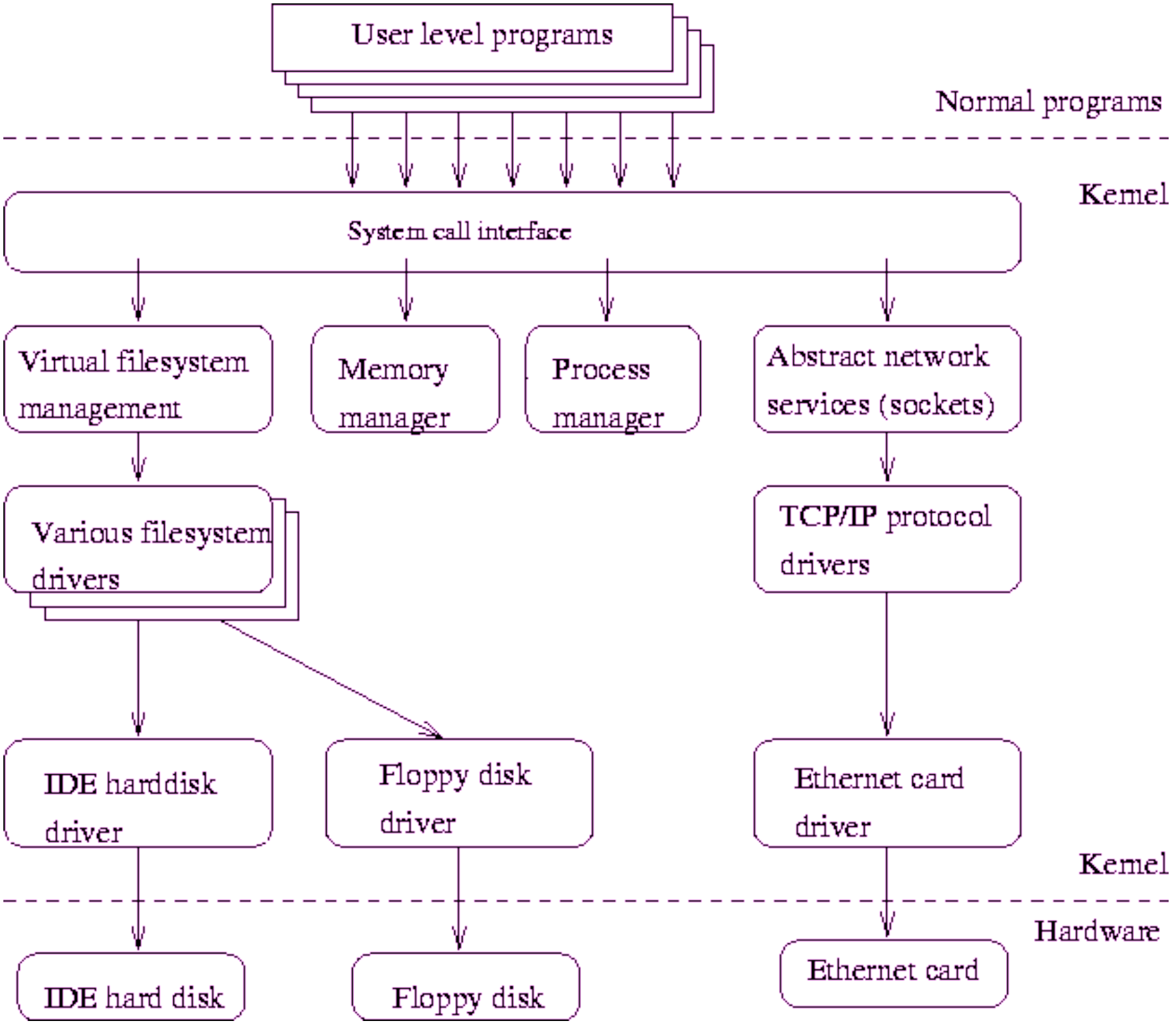
Overview of Linux : syslog

- The kernel and many system programs produce error, warning, and other messages.
- It is often important that these messages can be viewed later, even much later, so they should be written to a file. The program doing this is syslog .
- It can be configured to sort the messages to different files according to writer or degree of importance.
- For example, kernel messages are often directed to a separate file from the others, since kernel messages are often more important and need to be read regularly to spot problems.

Overview of Linux : kernel

- The Linux kernel consists of several important parts: process management, memory management, hardware device drivers, filesystem drivers, network management, and various other bits and pieces. Next figure shows some of them.


User Programs
Libraries
User Level
Kernel Level
System Call Interface
File Subsystem
Process Control Subsystem
Scheduler
Memory Management
Inter-process Communication
Buffer Cache
Character
Block
Device Drivers
Hardware Control
Kernel Level
Hardware Level
Hardware


User level programs
Normal programs
Kernel
System call interface
Virtual filesystem management
Memory manager
Process manager
Abstract network services (sockets)
Various filesystem drivers
TCP/IP protocol drivers
IDE harddisk driver
Floppy disk driver
Ethernet card driver
Kernel
Hardware
IDE hard disk
Floppy disk
Ethernet card

Overview of Linux : kernel

- Probably the most important parts of the kernel (nothing else works without them) are memory management and process management.
- Memory management takes care of assigning memory areas and swap space areas to processes, parts of the kernel, and for the buffer cache.
- Process management creates processes, and implements multitasking by switching the active process on the processor.

Overview of Linux : kernel

- At the lowest level, the kernel contains a hardware device driver for each kind of hardware it supports.
- Since the world is full of different kinds of hardware, the number of hardware device drivers is large.
- There are often many otherwise similar pieces of hardware that differ in how they are controlled by software.
- The similarities make it possible to have general classes of drivers that support similar operations.

Periodic execution : cron and at

- Both users and system administrators often need to run commands periodically. For example, the system administrator might want to run a command to clean the directories with temporary files (/tmp and /var/tmp) .
- The cron service is set up to do this. Each user has a crontab , where he lists the commands he wants to execute and the times they should be executed. The cron daemon takes care of starting the commands when specified.
- The at service is similar to cron , but it is once only: the command is executed at the given time, but it is not repeated.

Graphical User Interface

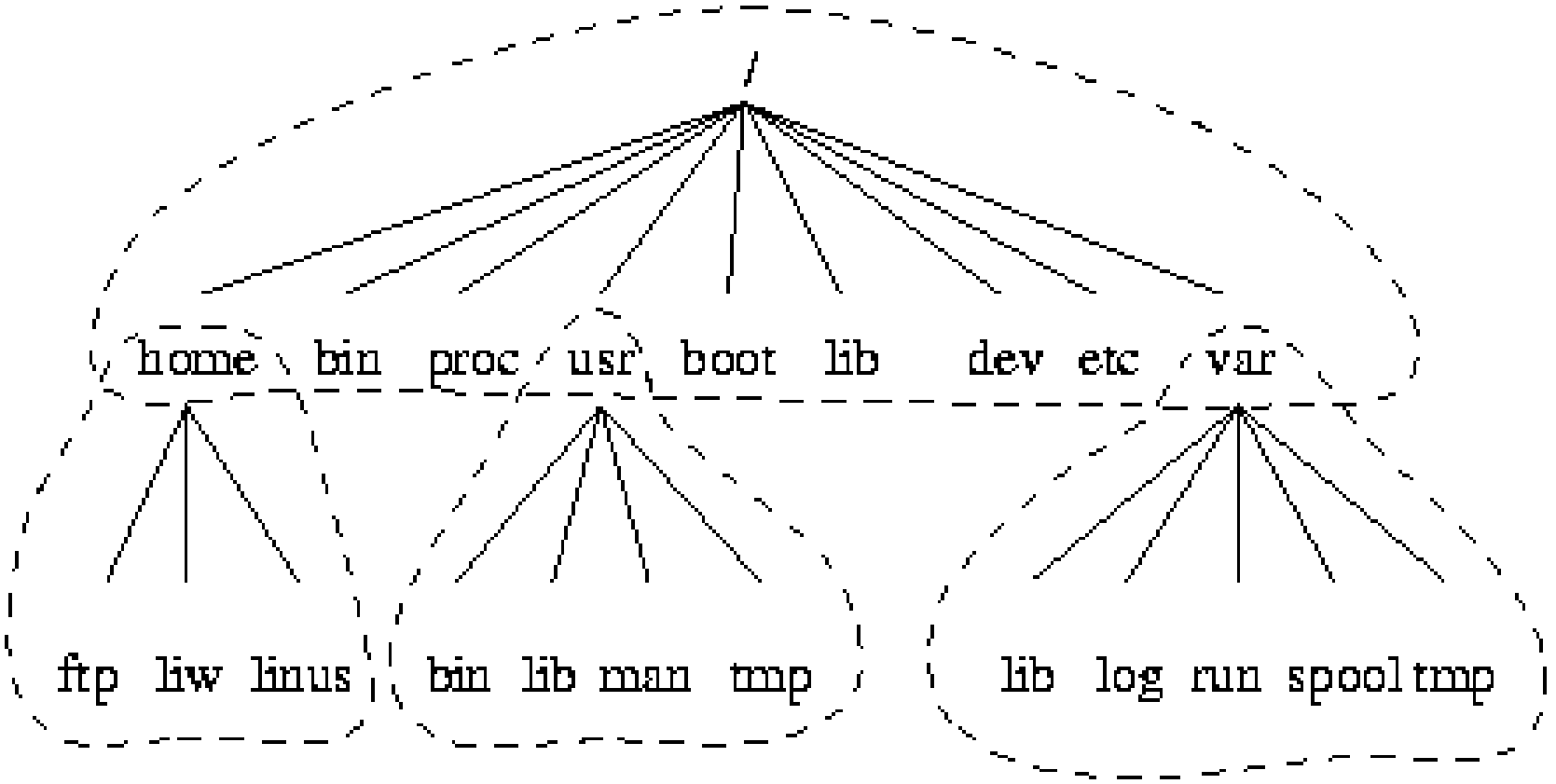
- UNIX and Linux don't incorporate the user interface into the kernel; instead, they let it be implemented by user level programs. This applies for both text mode and graphical environments.
- This arrangement makes the system more flexible, but makes the system harder to learn.
- The graphical environment primarily used with Linux is called the X Window System (X for short). X also does not implement a user interface; it only implements a window system. The three most popular user interface styles implemented over X are Athena, Motif, and Open Look.

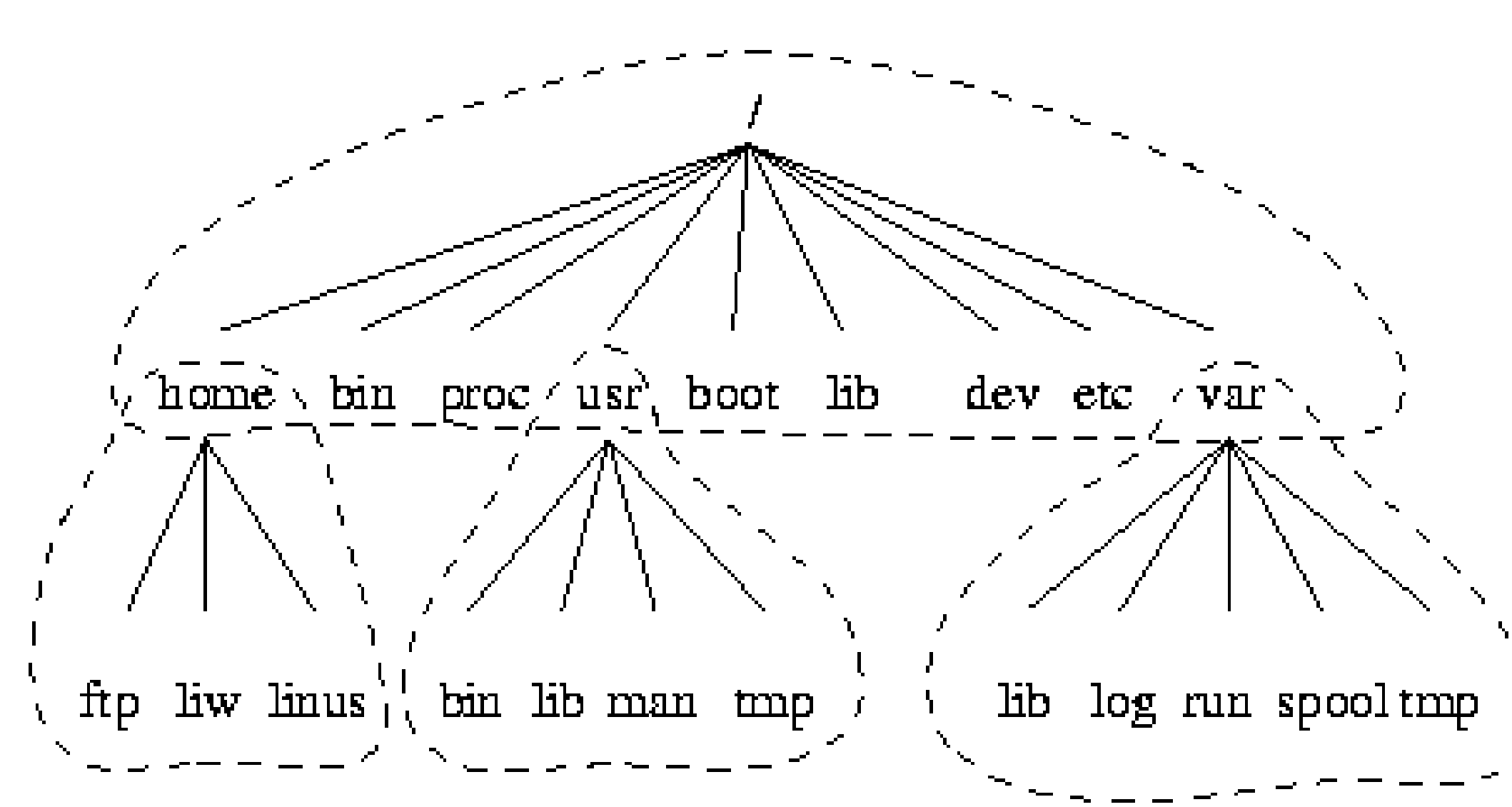
Electronic mail

- The mail system consists of many programs. The delivery of mail to local or remote mailboxes is done by one program (the mail transfer agent or MTA, e.g., sendmail or smail), while the programs users use are many and varied (mail user agent or MUA, e.g., pine or elm). The mailboxes are usually stored in /var/spool/mail .

Directory Tree Structure

- Here we describe the important parts of a standard Linux directory tree, based on the FSSTND filesystem standard. It outlines the normal way of breaking the directory tree into separate filesystems with different purposes and gives the motivation behind this particular split.




home bin proc usr boot lib dev etc var
ftp liw linus
bin lib man tmp
lib log run spool tmp

/ and /usr

- The root filesystem is specific for each machine (it is generally stored on a local disk, although it could be a ramdisk or network drive as well) and contains the files that are necessary for booting the system up, and to bring it up to such a state that the other filesystems may be mounted.
- The contents of the root filesystem will therefore be sufficient for the single user state.
- It will also contain tools for fixing a broken system, and for recovering lost files from backups.

/ and /usr

- The /usr filesystem contains all commands, libraries, manual pages, and other unchanging files needed during normal operation.
- No files in /usr should be specific for any given machine, nor should they be modified during normal use.
- This allows the files to be shared over the network, which can be cost-effective since it saves disk space (there can easily be hundreds of megabytes in /usr), and can make administration easier (only the master /usr needs to be changed when updating an application, not each machine separately).
- Even if the filesystem is on a local disk, it could be mounted read-only, to lessen the chance of filesystem corruption during a crash.

/var and /home

- The /var filesystem contains files that change, such as spool directories (for mail, news, printers, etc), log files, formatted manual pages, and temporary files.
- Traditionally everything in /var has been somewhere below /usr , but that made it impossible to mount /usr read-only.
- The /home filesystem contains the users' home directories, i.e., all the real data on the system.
- Separating home directories to their own directory tree or filesystem makes backups easier; the other parts often do not have to be backed up, or at least not as often (they seldom change).
- A big /home might have to be broken on several filesystems, which requires adding an extra naming level below /home , e.g., /home/students and /home/staff .

Directory tree

- Although the different parts have been called filesystems above, there is no requirement that they actually be on separate filesystems.
- They could easily be kept in a single one if the system is a small single-user system and the user wants to keep things simple.
- The directory tree might also be divided into filesystems differently, depending on how large the disks are, and how space is allocated for various purposes.
- The important part, though, is that all the standard names work; even if, say, /var and /usr are actually on the same partition, the names /usr/lib/libc.a and /var/adm/messages must work, for example by moving files below /var into /usr/var , and making /var a symlink to /usr/var .

Root file system

- The root filesystem should generally be small, since it contains very critical files and a small, infrequently modified filesystem has a better chance of not getting corrupted. A corrupted root filesystem will generally mean that the system becomes unbootable except with special measures (e.g., from a floppy), so you don't want to risk it.
- The root directory generally doesn't contain any files, except perhaps the standard boot image for the system, usually called /vmlinuz . All other files are in subdirectories in the root filesystems:

/bin

- Commands needed during bootup that might be used by normal users (probably after bootup).

/sbin

- Like /bin , but the commands are not intended for normal users, although they may use them if necessary and allowed.

Root file system

- /etc : Configuration files specific to the machine.
- /root : The home directory for user root.
- /lib : Shared lib needed by the programs on the root filesystem.
- /lib/modules : Loadable kernel modules, especially those that are needed to boot the system when recovering from disasters (e.g., network and filesystem drivers).
- /dev : Device files.
- /tmp : Temporary files.
- /boot : Files used by the bootstrap loader, e.g., LILO. Kernel images are often kept here instead of in the root directory.
- /mnt : Mount point for temporary mounts by the system administrator.
- /proc , /usr , /var , /home
- Mount points for the other filesystems.

/usr file system

- The /usr filesystem is often large, since all programs are installed there. All files in /usr usually come from a Linux distribution; locally installed programs and other stuff goes below /usr/local .
- /usr/X11R6 : The X Window System, all files. To simplify the development and installation of X, the X files have not been integrated into the rest of the system. There is a directory tree below /usr/X11R6 similar to that below /usr itself.
- /usr/bin : Almost all user commands. Some commands are in /bin or in /usr/local/bin .
- /usr/sbin : System administration commands that are not needed on the root filesystem, e.g., most server programs.
- /usr/man , /usr/info , /usr/doc : Manual pages, GNU Info documents, and miscellaneous other documentation files, respectively.
- /usr/include : Header files for the C programming language.

/usr file system

- /usr/lib : Unchanging data files for programs and subsystems, including some site-wide configuration files. The name lib comes from library; originally libraries of programming subroutines were stored in /usr/lib .
- /usr/local : The place for locally installed software and other files.

Special configuration in /etc/inittab

- The /etc/inittab has some special features that allow init to react to special circumstances. These special features are marked by special keywords in the third field. Some examples:
- powerwait : Allows init to shut the system down, when the power fails. This assumes the use of a UPS, and software that watches the UPS and informs init that the power is off.
- Ctrlaltdel : Allows init to reboot the system, when the user presses control-alt-del on the console keyboard. Note that the system administrator can configure the reaction to C-A-D to be something else instead, e.g., to be ignored, if the system is in a public location.
- Sysinit : Command to be run when the system is booted. This command usually cleans up /tmp , for example.

Documentation

- /usr/doc/FAQ - Directory of FAQ (Frequently Asked Questions!)
- /usr/doc/FAQ/html
- /usr/doc/FAQ/ps
- /usr/doc/FAQ/txt
- /usr/doc/FAQ/txt/FAQ

- /usr/doc/LDP - Directory of Linux Documentation Project
- /usr/doc/LDP/lpg - Linux Programmer's Guide
- /usr/doc/LDP/nag - Network Administrator's Guide
- /usr/doc/LDP/sag - System Administrator's Guide

(RedHat 7.x has put these in /usr/share/doc)

Documentation

- /usr/doc/HOWTO - Directory of HOWTO's
- /usr/doc/HOWTO/mini - Directory of mini HOWTO's
- /usr/doc/HOWTO/unmaintained - Directory of unmaintained HOWTO's

- Man Pages : Manual Pages

=> man man

- Info : Information System

=> info

- Help : Help on built in commands

=> help

- KDE Documentation : KDE Help Center

click on icon