3 Operating Systems

Content

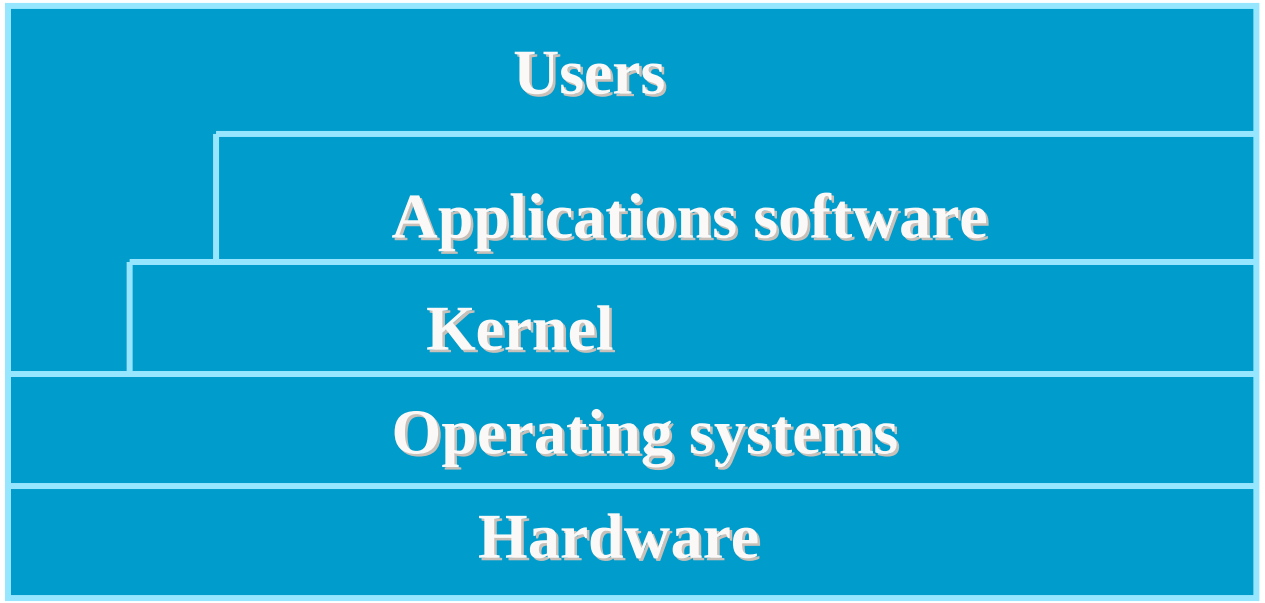
- The concept of an operating system.
- The internal architecture of an operating system.
- The architecture of the Linux operating system in more detail.
- How to log into (and out of) UNIX and change your password.
- The general format of UNIX commands.

What is an Operating System?

- An elephant is a mouse with an operating system.

What is an Operating System?

- A piece of software that provides a convenient, efficient environment for the execution of user programs.



Resource Abstraction and Sharing

- Hides the details of how the hardware operates
- Provides an abstract model of the operation of hardware components
  - generalize hardware behavior
  - limit the flexibility in which hardware can be manipulated
- Computation model: processes, threads
- Resources: memory, disk, files, cpu, etc.

Why do we need an operating system?

- **User viewpoint** – provide user interface, command interpreter, directory structure, utility programs (compilers, editors, filters)
- **Program environment viewpoint** – enhance the bare machine higher-level I/O, structure files, notion of independent processes, improved store (size, protection)
- **Efficiency viewpoint** – replace a human operator scheduling jobs, storing I/O (files), invoking necessary programs such as compiler
- **Economic viewpoint** – allow concurrent uses and good scheduling of resources

So, the goals are to make the system convenient to use (via system calls) and to manage resources efficiently.

UNIX

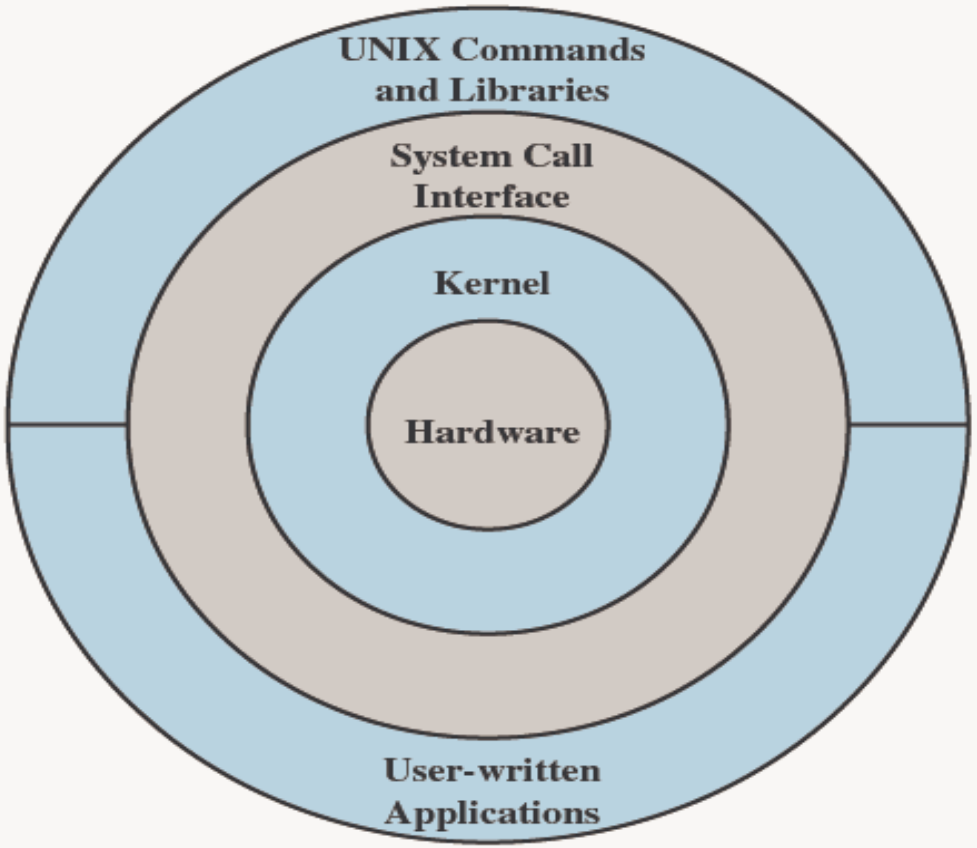


Figure 2.14 General UNIX Architecture

UNIX

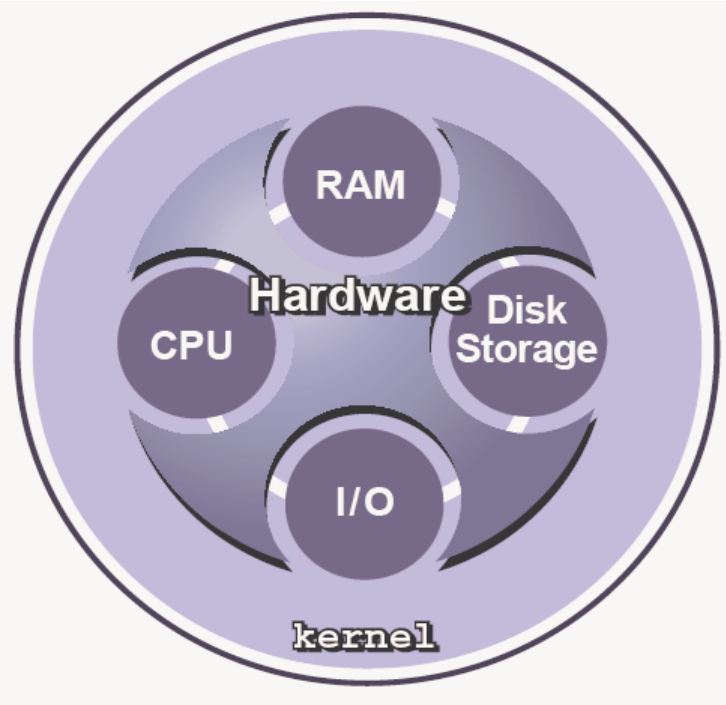
- The kernel provides low-level device, memory and processor management functions
- Basic hardware-independent kernel services are exposed to higher-level programs through a library of system calls
- Application programs (e.g. word processors, spreadsheets) and system utility programs (simple but useful application programs that come with the operating system, e.g. programs which find text inside a group of files) make use of system calls. Applications and system utilities are launched using a shell (a textual command line interface) or a graphical user interface that provides direct user interaction.

Modern UNIX Systems

- System V Release 4 (SVR4)
- Solaris 9
- 4.4BSD
- Linux

Kernel

- The kernel is the core of the OS
- Manages all the physical resources of the computer
- After the shell passes the commands, the kernel executes the commands.



Shell

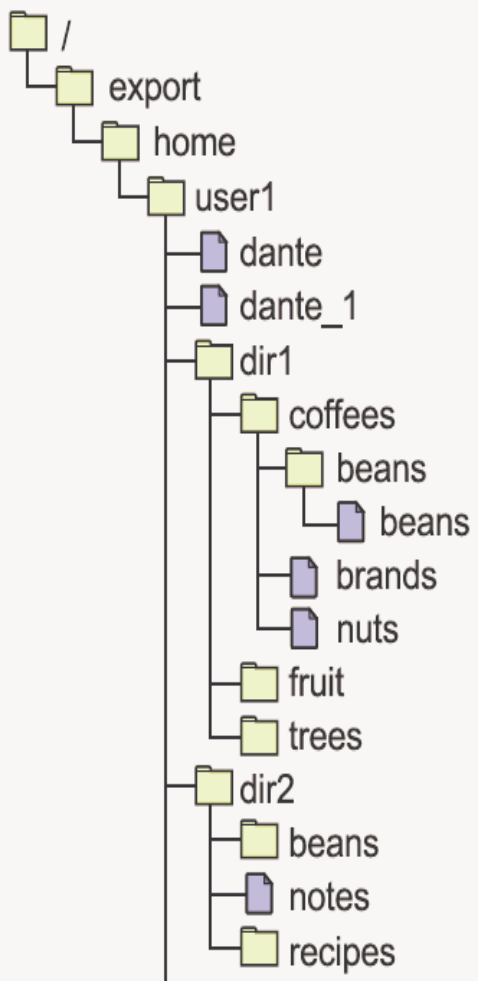
The shell performs the following:

- Acts as an interface between the user and the kernel
- Acts as a command-line interpreter
- Takes the commands that a user enters
- Processes the command line entered by a user
- Passes the interpreted command line to the kernel

Shell

- Bourne Shell — ***sh***
- Korn Shell — ***ksh***
- C Shell — ***csh***
- TC Shell — ***tcsh***
- B(ourne)A(gain)sh Shell — ***bash***

Directory Hierarchy



- The directory hierarchy contains an organized group of directories and files.

Logging In to the System

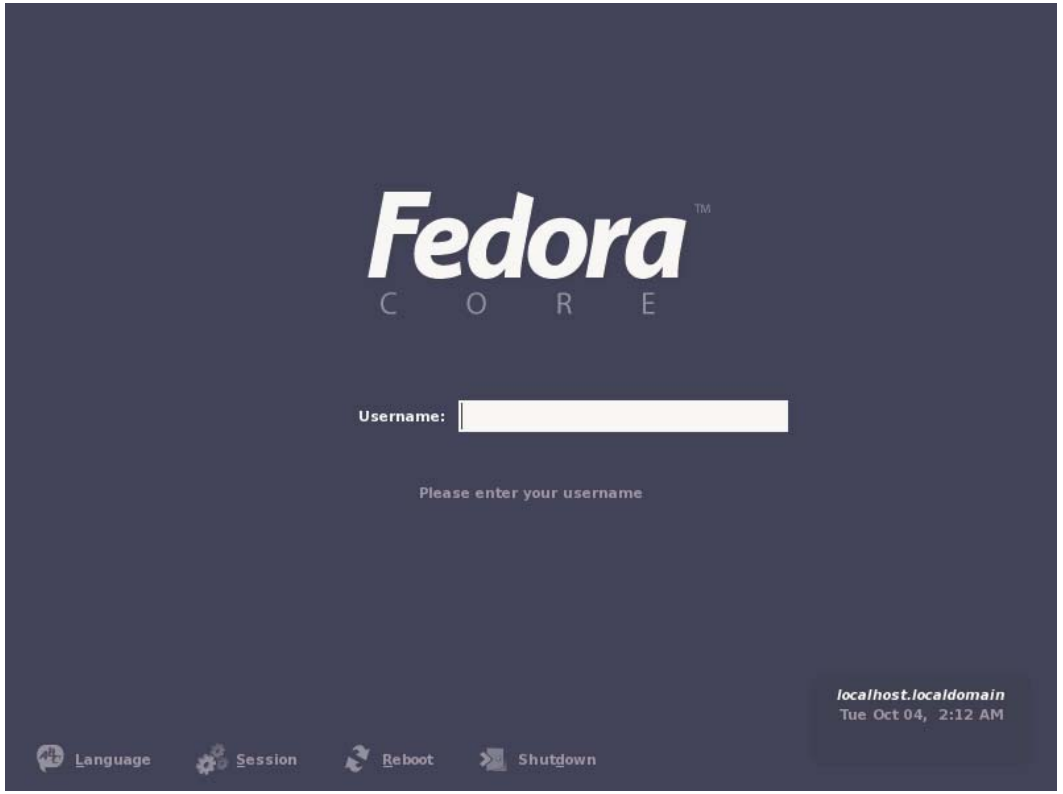
- All users must follow a login process so that the system can recognize and authenticate the user.
- The desktop Login screen, which appears on your monitor, enables you to log into the system and use the desktop.
  - Logging in using the command line
  - Changing your password

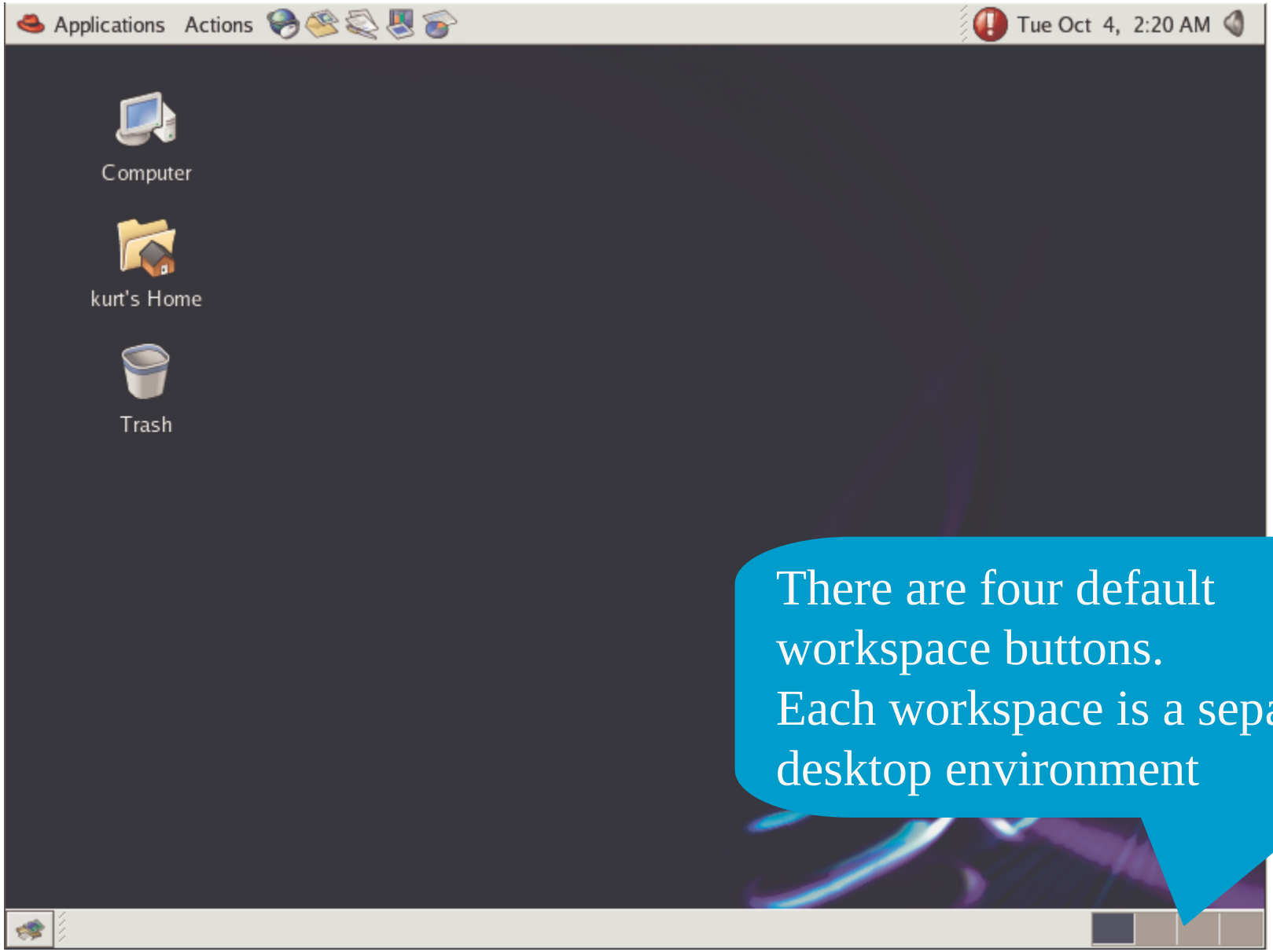
Logging In to the System

- Turn your PC on and switch to Unix OS (e.g. Fedora)

- You will have several messages from the OS.
- You use them to see whether everything is going well

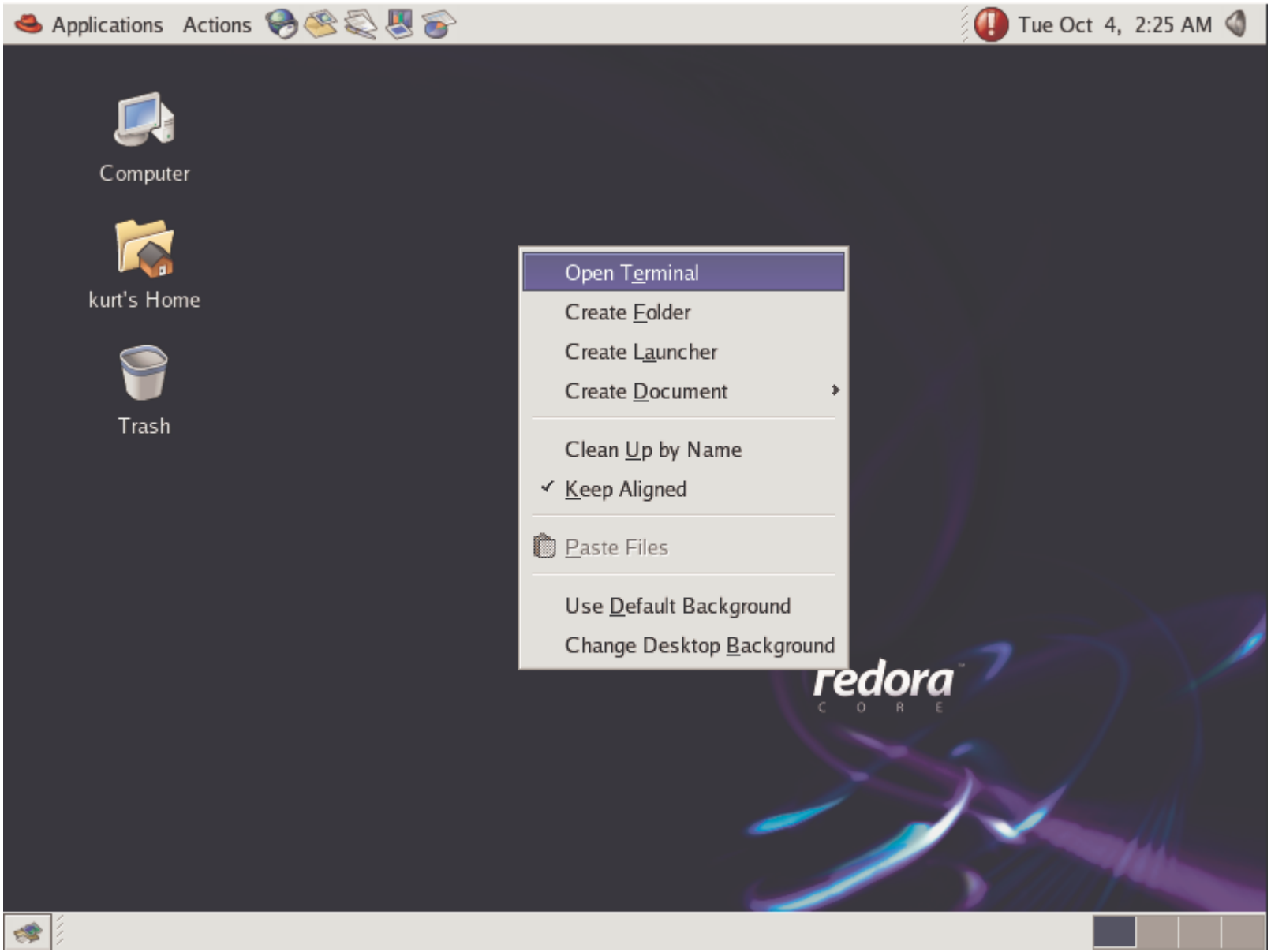
- To log in to a desktop session from the desktop environment Login screen, complete the following steps:
  1. Type your user name in the text field. Press Return or click OK.
  2. Type your password in the password text field. Press Return or click OK.
- If the login attempt fails, try again.

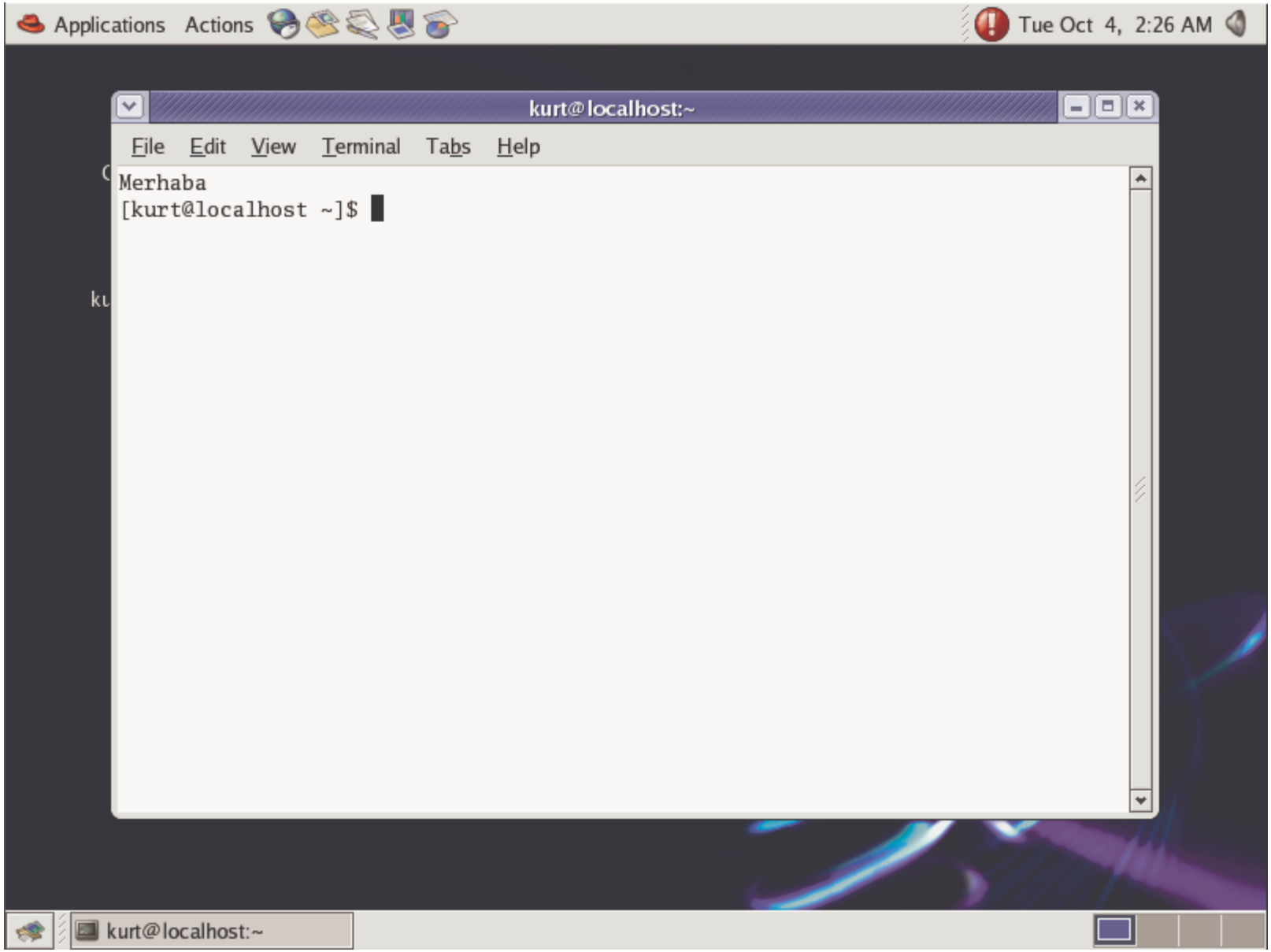



Applications
Actions
Tue Oct 4, 2:20 AM
Computer
kurt's Home
Trash
There are four default workspace buttons.
Each workspace is a separate desktop environment

Managing Files Using the Desktop Environment

- To move a file from one directory to another:
  1. Position the mouse pointer over the file icon.
  2. Hold down the left mouse button, and drag the icon to the appropriate directory icon.
  3. When the file icon is positioned over the directory icon, release the left mouse button
- The file moves to that directory.


Applications
Actions
Tue Oct 4, 2:25 AM
Computer
kurt's Home
Trash
Open Terminal
Create Folder
Create Launcher
Create Document
Clean Up by Name
Keep Aligned
Paste Files
Use Default Background
Change Desktop Background
redora
CORE


Applications
Actions
Tue Oct 4, 2:26 AM
kurt@localhost:~
File Edit View Terminal Tabs Help
Merhaba
[kurt@localhost ~]$
kurt@localhost:~

Introducing Command-Line Syntax

- You can change the behavior of command functions by using options and arguments, as shown in the following table:

Item	Description
command	Specifies what the system does (an executable).
option	Specifies how the command runs (a modifier). Options start with a dash (-) character.
argument	Specifies what is affected (a file, a directory, or text).

Entering Multiple Commands on a Single Command Line

- You can enter multiple commands on a single command line by using a semicolon (;) to separate each command.
- The shell recognizes the semicolon as a command separator.

Control Characters

Control Characters	Purpose
Control-C	Terminates the command currently running
Control-D	Indicates end-of-file or exit
Control-U	Erases all characters on the current command line
Control-W	Erases the last word on the command line
Control-S	Stops output to the screen
Control-Q	Restarts output to the screen after you have pressed Control-S
Control-Z	Stops the job in progress

Displaying the Online Manual Pages

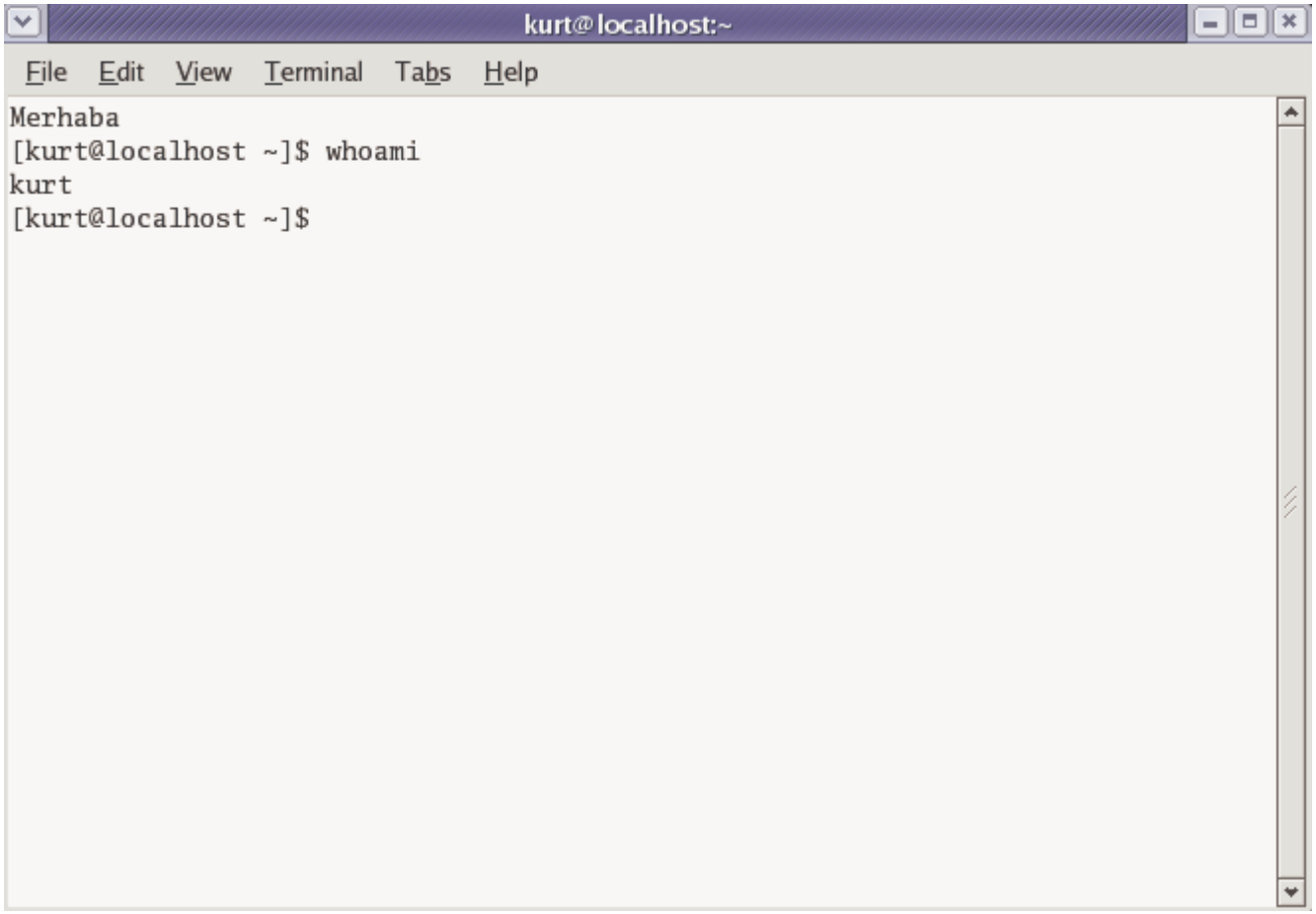
- The online UNIX Reference Manual (man) pages provide detailed descriptions of UNIX commands and how to use them.
- Use the `man` command to display the man page entry that explains a given command.

Scrolling in Man Pages

- This table shows the keys on the keyboard that you use to control the scrolling capabilities when you are in the man pages.

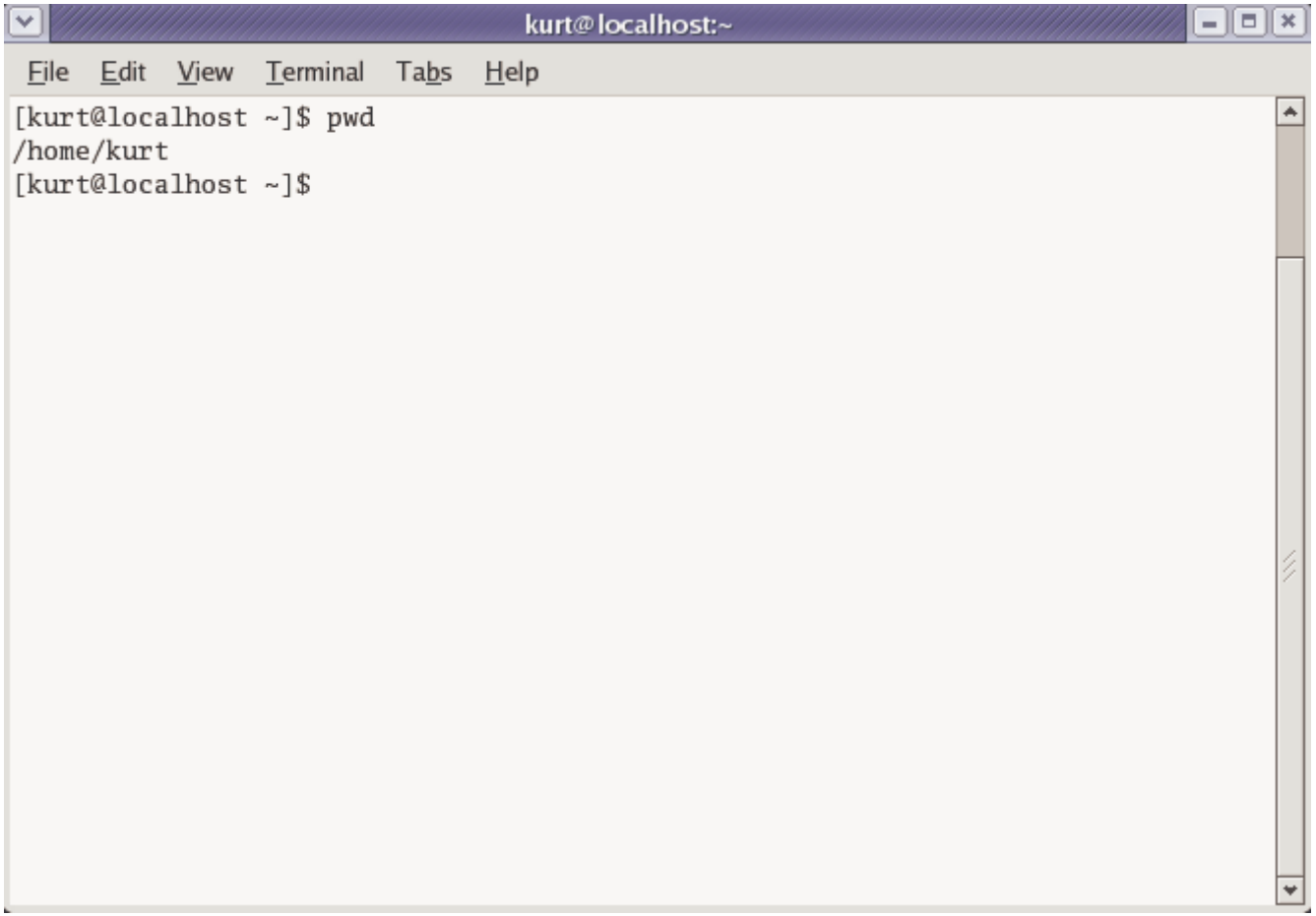
Scrolling Keys	Action
Spacebar	Displays the next screen of a man page
Return	Displays the next line of a man page
b	Moves back one full screen
/pattern	Searches forward for a pattern
n	Finds the next occurrence of a pattern after you have used */pattern*
h	Provides a description of all scrolling capabilities
q	Quits the man command

Who am i?


kurt@localhost:~
File Edit View Terminal Tabs Help
Merhaba
[kurt@localhost ~]$ whoami
kurt
[kurt@localhost ~]$

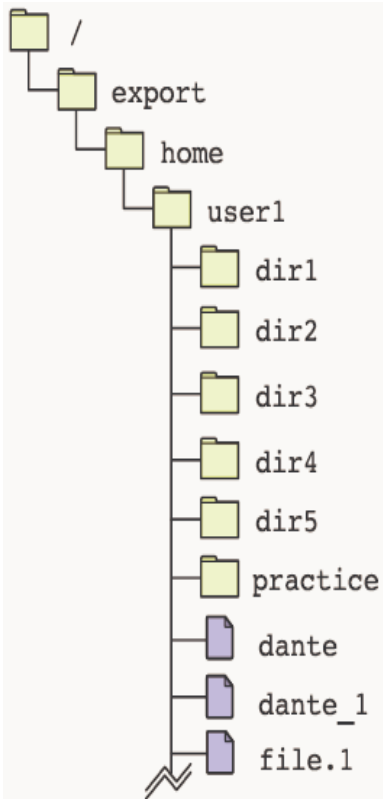
Determining the Current Directory

- The `pwd` command identifies the directory that you are currently accessing.



Displaying the Directory Contents

- You can view a hierarchy of the files and directories in the /export/home/user1 directory.



Displaying the Directory Contents

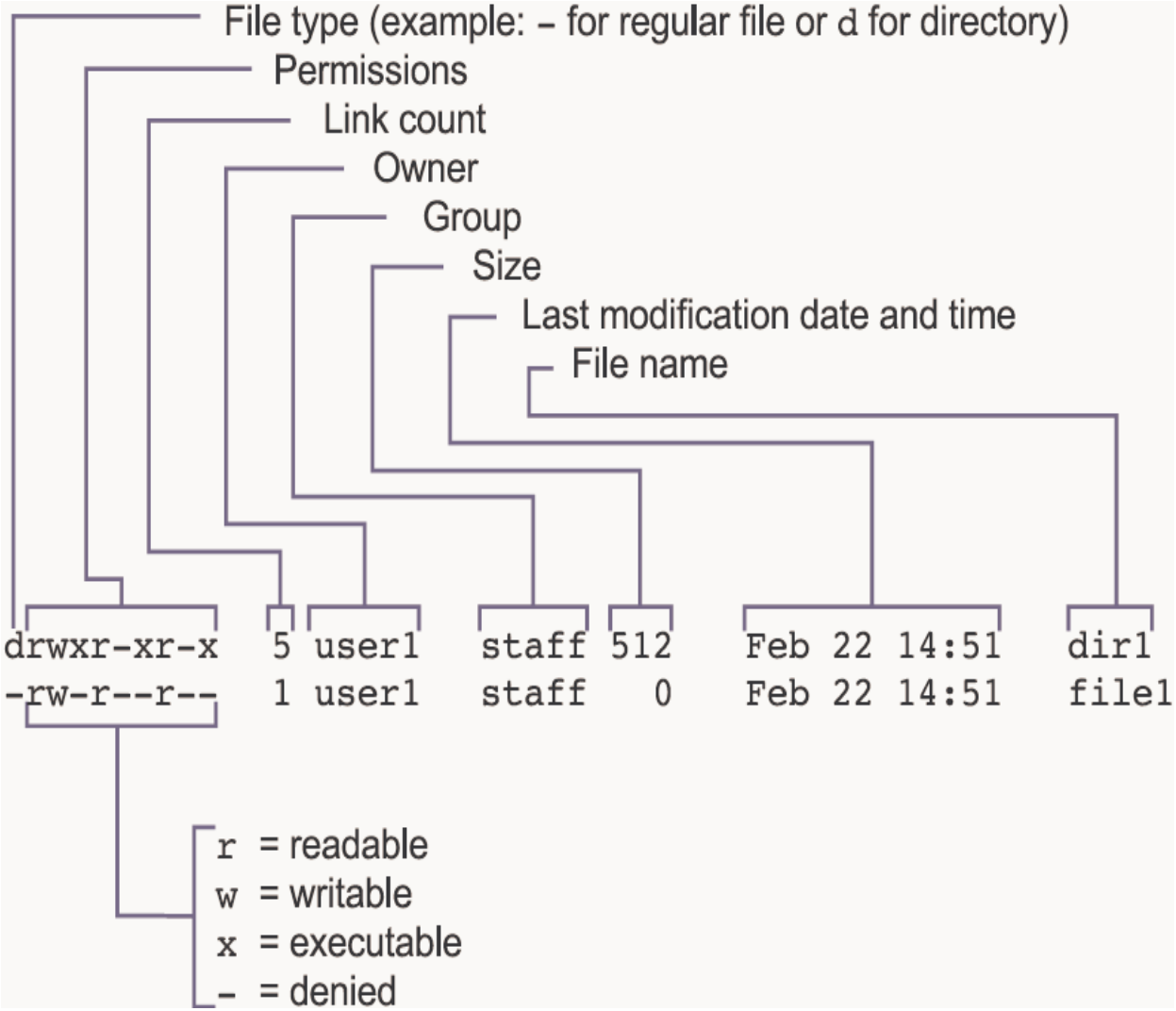
- Displaying hidden files
- Displaying a long list

  You can use the `ls -l` command to view detailed information about the contents of a directory. The output of the `ls -l` command shows a long listing of file information.

What is a hidden file?

- `touch apple`
- `ls` (what do you see? do you see apple?)
- `touch .pear`
- `ls` (do you see .pear)
- `ls -a` (what about now?)

Displaying the Directory Contents



Displaying the Directory Contents

- To view detailed information on the contents of the dir1 directory, perform the `ls -l dir1` command from the user1 directory.

```
$ ls -l dir1
total 6
drwxr-xr-x   2 user1    staff          512 Feb 22 14:51 coffees
drwxr-xr-x   2 user1    staff          512 Feb 22 14:51 fruit
drwxr-xr-x   2 user1    staff          512 Feb 22 14:51 trees
```

Displaying the Directory Contents

- Displaying individual directories
  - `ls -l /etc`
- Displaying a recursive list
  - `ls -R /etc`
  - `ls -R /`

Displaying File Types

- You can use either the `ls -F` command or the `file` command to display file types.
- This table shows the symbols used with the `ls -F` command output:

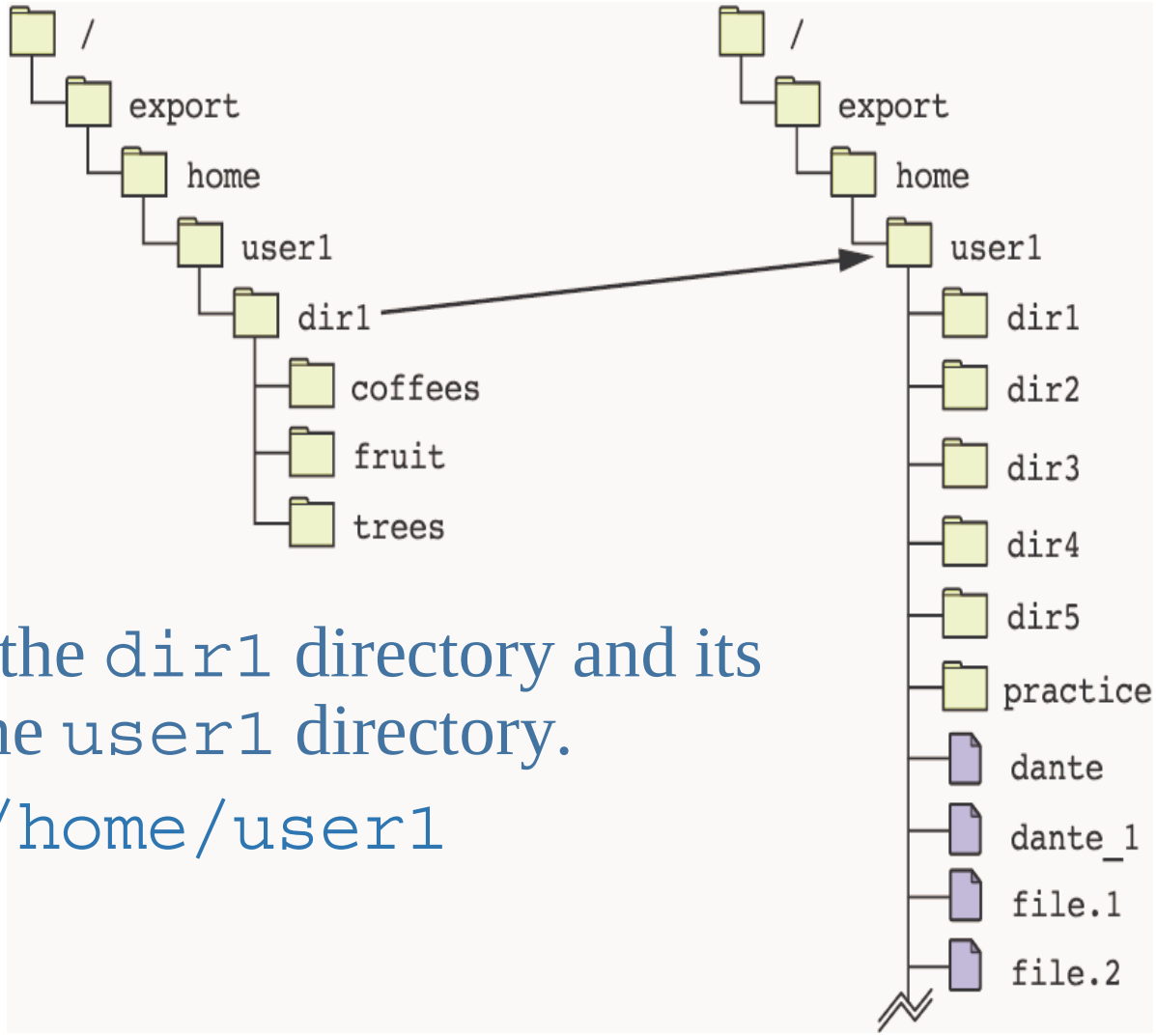
Symbol	File Type
/	Directory
*	Executable
(none)	Plain text file or ASCII
@	Symbolic link

Changing Directories

- Using Path Name Abbreviations
  - You can use path name abbreviations to easily navigate or refer to directories on the command line.

Symbol	Path Name
.	Current or working directory
. .	Parent directory, the directory directly above the current working directory

Changing Directories



- This figure shows the `dir1` directory and its parent directory, the `user1` directory.
  - `cd /export/home/user1`
- or
  - `cd ..`

Returning to your home directory

```
$ cd
$ cd ~user1/
$ cd ~/
$ cd $HOME
```

Working With Files

- There are different commands available that enable you to view file content in a read-only format or to display information about a file.
- These commands include:
  - The `cat` command
  - The `more` command
  - The `tail` command
  - The `head` command
  - The `wc` command

Viewing Files Using the `more` Command

- When the `--More--(n%)` prompt appears, you can use the keys described in the following table to scroll through the file.

Scrolling Keys	Action
Spacebar	Moves forward one screen
Return	Scrolls one line at a time
`b`	Moves back one screen
`h`	Displays a help menu of features
/string	Searches forward for *pattern*
`n`	Finds the next occurrence of *pattern*
`q`	Quits and returns to the shell prompt

Displaying Line, Word, and Character Counts

- This table shows the options that you can use with the `wc` command.

Option	Description
`-l`	Line count
`-w`	Word count
`-c`	Byte count
`-m`	Character count

Working With Files

```
$ car /etc/passwd
$ tail -2 /etc/passwd
$ head -2 /etc/passwd
$ ls -R / > dat
$ cat ./dat
$ more ./dat
$ cat ./dat | more
$ wc /etc/passwd
$ wc -l /etc/passwd
```

Copying Files

- The syntax for the `cp` command when copying files is:

```
cp -option sources target
```

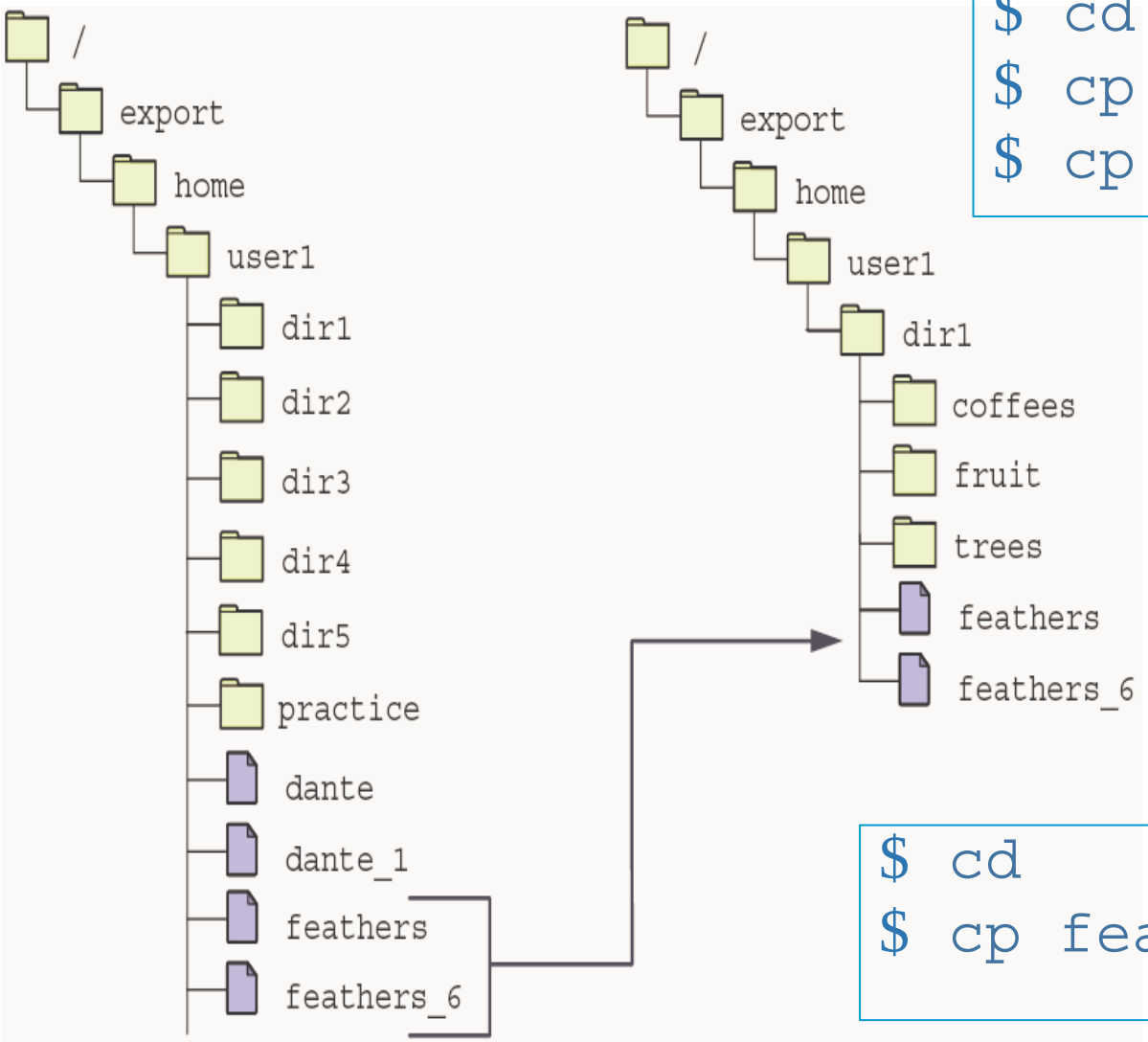
- The source option is a file.
- The target option can be a file or a directory.
- This table describes some options you can use with the `cp` command when you are copying files and directories.

Option	Description
`-i`	Prevents you from accidentally overwriting existing files or directories
`-r`	Includes the contents of a directory, including the contents of all subdirectories, when you copy a directory

Copying Files

```
$ touch apple
$ ls
$ cp apple pear
$ ls
```

Copying Multiple Files




```
$ cd                                    ①
$ cp feathers dir1/
$ cp feathers_6 dir1/
```

```
$ cd                                    ②
$ cp feathers* dir1/
```

Moving and Renaming Files and Directories

- You can use the `mv` command to:
  - Move and rename a file
  - Move a file to another directory
  - Move a directory and its contents
  - Rename a directory

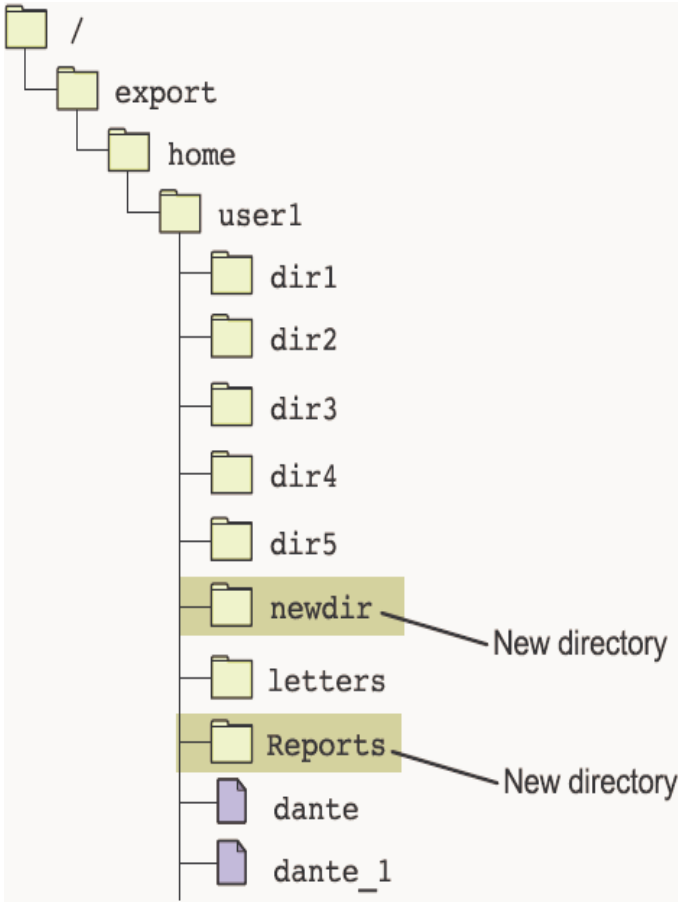
Moving and Renaming Files and Directories


```
$ cd
$ touch banana
$ ls
$ mv banana apricot
$ ls
$ mkdir dir3
$ ls
$ mv apricot dir3/
$ ls
$ ls dir3/
```

Creating Files and Directories

- You can use the touch and `mkdir` commands to:
  - Create empty files
  - Create directories

Creating Files and Directories




```
$ cd
$ mkdir newdir
$ mkdir Reports
```

Creating Files and Directories

```
$ cd
$ mkdir DIR1
$ ls
$ mkdir DIR/DIR2
$ ls
$ cd DIR1
$ ls
$ pwd
```

Removing Files and Directories

- From the directory hierarchy you can permanently:
  - Remove files
  - Remove directories (including removing a directory with content)
- This table describes the options that you can use with the `rm` command when you are removing directories.

Option	Description
`-r`	Includes the contents of a directory and the contents of all subdirectories when you remove a directory
`-i`	Prevents the accidental removal of existing files or directories

Removing Files and Directories


```
$ cd
$ mkdir fruit
$ touch
  fruit/banana
$ touch
  fruit/apple
$ cd fruit
$ touch pear
$ touch pricot
$ ls
$ rm apple
$ ls
```

```
$ rm pear
$ ls
$ cd ..
$ rm -r fruit/
$ ls
```

Using Symbolic Links

- Files (and directories) might be located on several different file systems.
- You can use symbolic links to link files that are in different file systems.
  - Introducing symbolic links
  - Creating symbolic links
  - Removing symbolic links

Using the `grep` Command

- The options that you use with the `grep` command can modify your search.
- Each option, except for the –w option, can be used with the `egrep` and `fgrep` commands.
- This table shows the options for the `grep` command.

Option	Definition
`-i`	Searches for both uppercase and lowercase characters
`-l`	Lists the names of files with matching lines
`-n`	Precedes each line with the relative line number in the file
`-v`	Inverts the search to display lines that do not match the *`pattern`*
`-c`	Counts the lines that contain the *`pattern`*
`-w`	Searches for the expression as a complete word, ignoring those matches that are substrings of larger words

Using the grep Command

```
$ cat /etc/passwd | grep root
$ grep root /etc/passwd
$ xclock &
$ ps -fe | grep xclock
```

Using the `find` Command

- The `pathname`, `expression`, and `action` arguments for the `find` command are shown in the table.

Argument	Definition
pathname	The absolute or relative path where the search originates.
expression	The search criteria specified by one or more options. Specifying multiple options causes the `find` command to use the boolean operator "and," so all listed expressions must be verified as true.
action	The action required after the files have been located. The default action is to print all path names matching the criteria to the screen.

Using the `find` Command

- `Expressions` that you can use with the find command.

Expression	Definition
`-name` *`filename`*	Finds files matching the specified *`filename`*. Metacharacters are acceptable if placed inside " ".
`-size [+|-]`*`n`*	Finds files that are larger than +*n*, smaller than –*n*, or exactly *n*. The *n* represents 512-byte blocks.
`-atime [+|-]`*`n`*	Finds files that have been accessed more than +*n* days, less than -*n* days, or exactly *n* days.
`-mtime [+|-]`*`n`*	Finds files that have been modified more than +*n* days ago, less than -*n* days ago, or exactly *n* days ago.
`-user` *`loginID`*	Finds all files that are owned by the *`loginID`* name.
`-type`	Finds a file type, for example, `f` (file) or `d` (directory).
`-perm`	Finds files that have certain access permission bits.

Using the `find` Command

- `Action` arguments for the find command.

Action	Definition
`-exec command {} \;`	Runs the specified *command* on each file located. A set of braces, {}, delimits where the file name is passed to the command from the preceding expressions. A space, backslash, and semicolon (\;) delimits the end of the command. There must be a space before the backslash (\).
`-ok command {} \;`	Requires confirmation before the `find` command applies the *command* to each file located. This is the interactive form of the -exec command.

Using the `find` Command

- Additional action arguments for the `find` command:

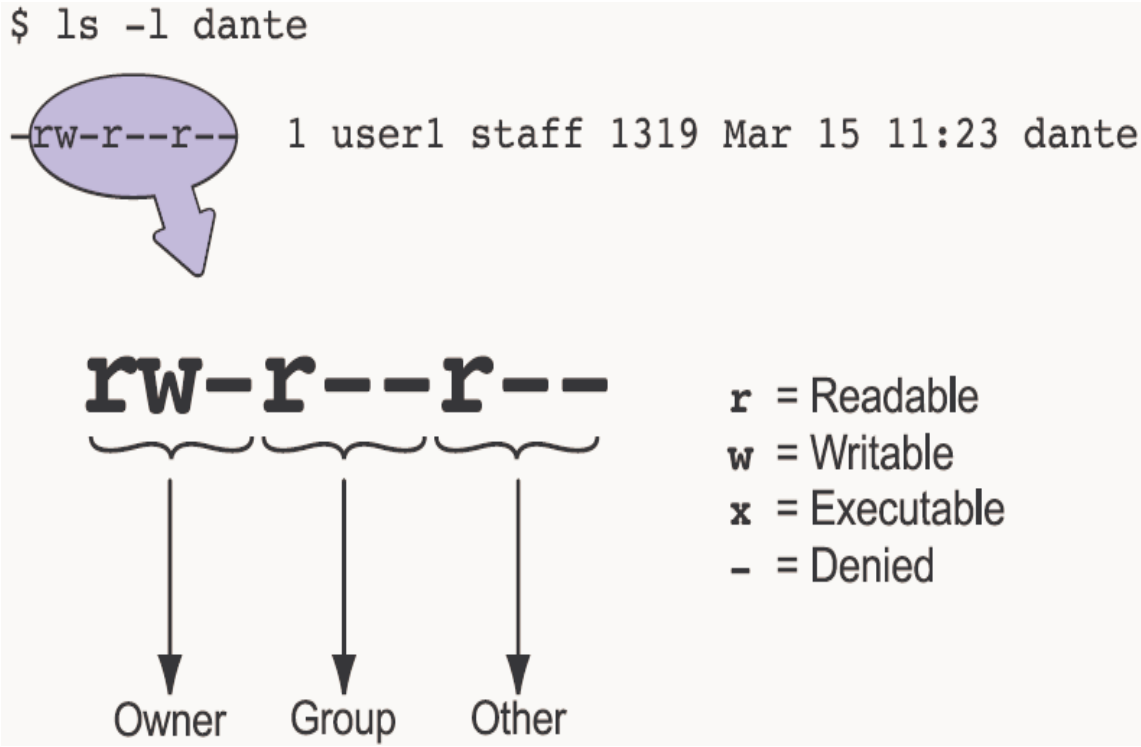
Action	Definition
`-print`	Instructs the `find` command to print the current path name to the terminal screen. This is the default.
`-ls`	Displays the current path name and associated statistics, such as: the inode number, the size in Kilobytes, protection mode, the number of hard links, and the user.

Using the `find` Command

```
$ find /etc -name “pas*” -print
$ find / -user kurt -mtime -7 -size -100 -print
$ find / -name core -exec rm {} \;
```

Viewing Permission Categories

- To view the permissions for files and directories, perform the `ls -l` command.
- This figure shows the information displayed for the dante file.



Viewing Permission Categories

- The first field of information displayed by the `ls -l` command is the file type.
- The file type typically specifies whether it is a file or a directory.
- A file is represented by a hyphen (-).
- A directory is represented by the letter d.

Viewing Permission Categories

- The remaining fields represent three types of users: owner, group, and other.
- This table describes each type of user with a brief description of each.

Field	Description
Owner	Permissions used by the assigned owner of the file or directory.
Group	Permissions used by members of the group that owns the file or directory.
Other	Permissions used by all users other than the file owner, and members of the group that owns the file or the directory.

Permission Characters

Permission	Character	Access for a File	Access for a Directory
Read	`r`	You can display file contents and copy the file.	You can list the directory contents with the `ls` command.
Write	`w`	You can modify the file contents.	If you also have execute access, you can modify the contents of the directory.
Execute	`x`	You can execute the file if it is an executable. You can execute a shell script if you also have read and execute permissions.	You can use the `cd` command to access the directory. If you also have read access, you can run the `ls -l` command on the directory to list contents.

Permission Sets

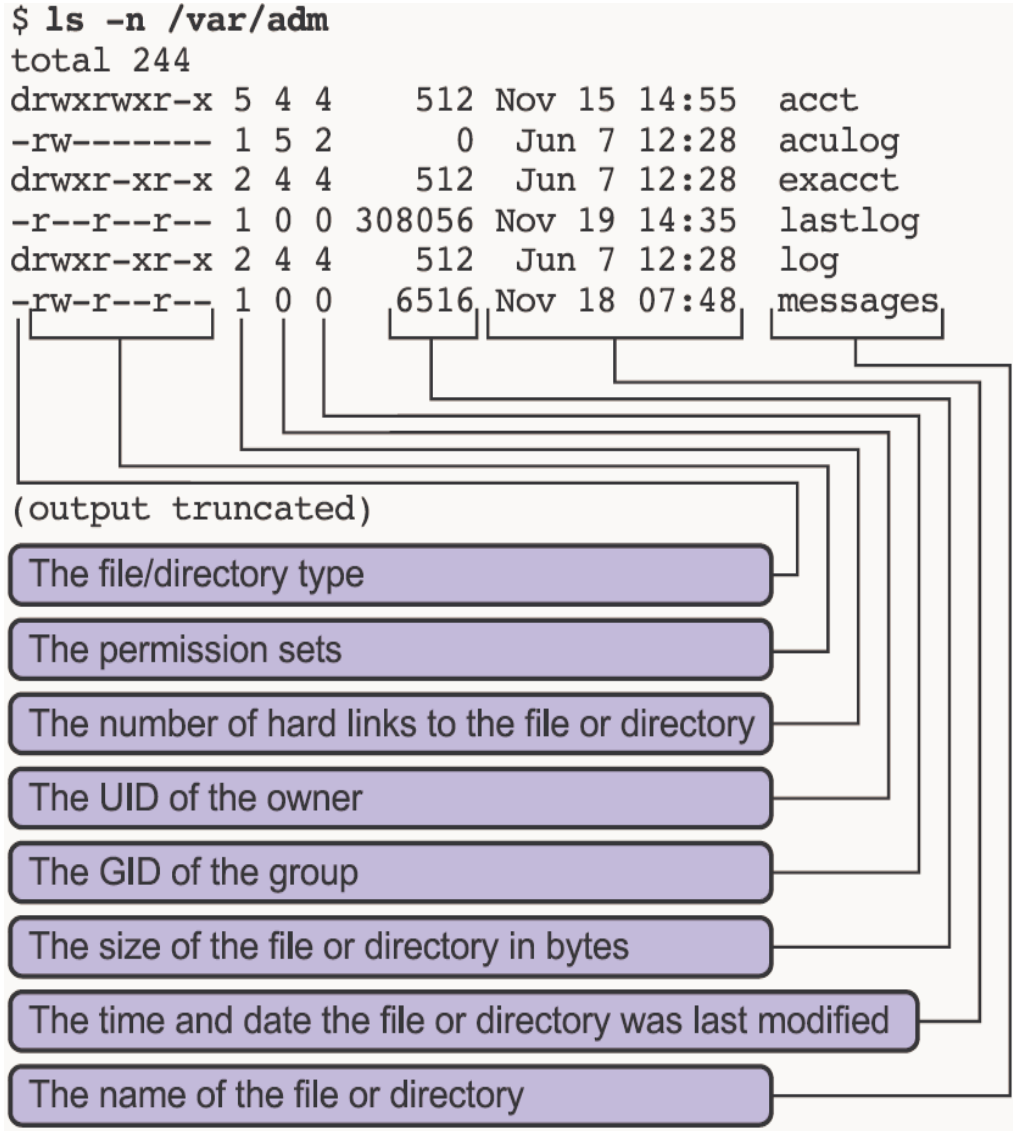
Permissions	Description
`-rwx------`	This file has read, write, and execute permissions set for the file owner only. Permissions for group and other are denied.
`dr-xr-x---`	This directory has read and execute permissions set for the directory owner and the group only.
`-rwxr-xr-x`	This file has read, write, and execute permissions set for the file owner. Read and execute permissions are set for the group and other.

Using the `ls -n` Command

- To view the user identifiers (UIDs) and group identifiers (GIDs), perform the ls -n command on the /var/adm directory.

```
$ ls -n /var/adm
total 244
drwxrwxr-x   5 4          4              512 Nov 15 14:55 acct
-rw-------   1 5          2                0 Jun  7 12:28 aculog
drwxr-xr-x   2 4          4              512 Jun  7 12:28 exacct
-r--r--r--   1 0          0           308056 Nov 19 14:35 lastlog
drwxr-xr-x   2 4          4              512 Jun  7 12:28 log
-rw-r--r--   1 0          0             6516 Nov 18 07:48 messages
(output truncated)
```

Description of the Output of the `ls -n` Command



Assigned Octal Values for Permissions

- This table shows the octal numbers for each individual permission.

Octal Value	Permission
4	Read
2	Write
1	Execute

Octal Digits for Permission Sets

- This table shows the octal numbers that represent a combined set of permissions.

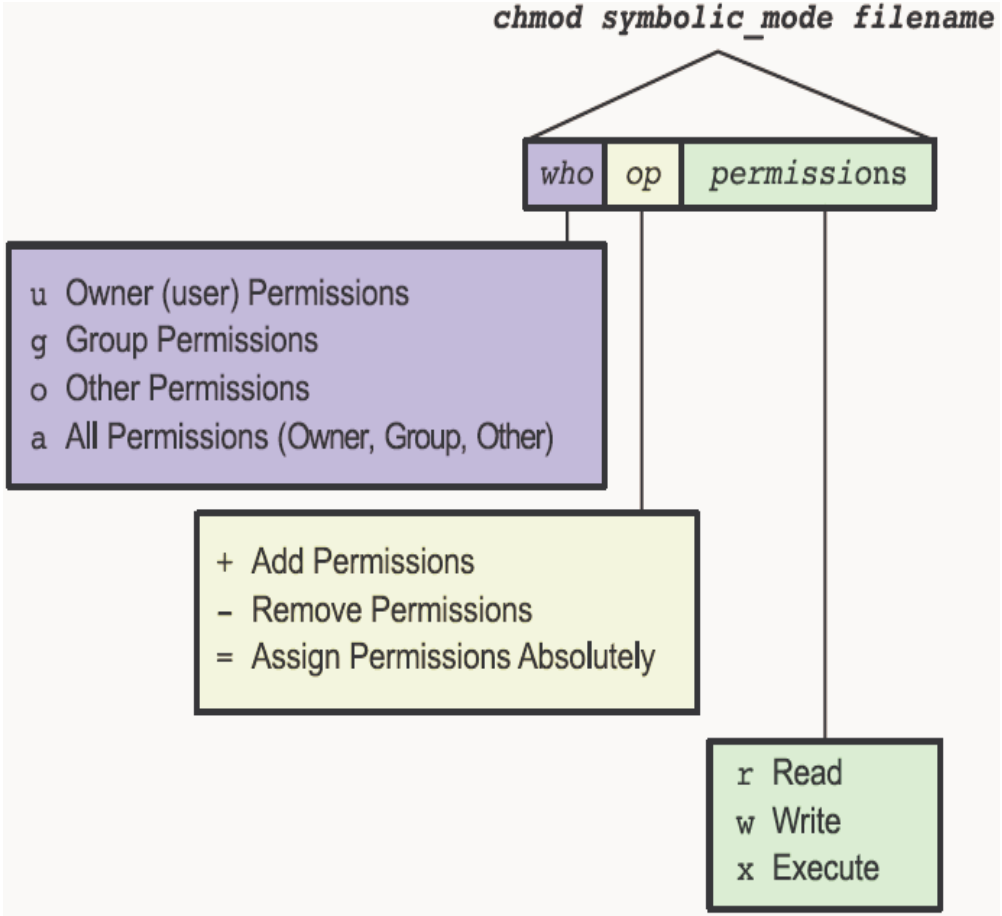
Octal Value	Permission Sets	
7	r w x	421
6	rw-	420
5	r-x	4-1
4	r--	4--
3	-wx	021
2	-w-	020
1	--x	001
0	---	000

Combined Values and Permissions

- This table shows the permission sets for the three-digit octal numbers:

Octal Mode	Permissions
644	rw-r--r--
751	rwxr-x--x
775	rwxrwxr-x
777	rwxrwxrwx

Changing permissions in symbolic mode



Changing permissions in symbolic mode

- `touch apple`
- `ls -n apple`
- `chmod g-rwx apple`
- `ls -n apple`
- `chmod o-wx apple`
- `chmod o+r apple`
- `ls -n apple`

Introducing the `umask` Utility

- This table shows the file and directory permissions that are created for each of the `umask` octal values.
- Also use these values to determine the `umask` value you want to set.

`umask` Octal Value	File Permissions	Directory Permissions
0	`rw-`	`rwx`
1	`rw-`	`rw-`
2	`r--`	`r-x`
3	`r--`	`r--`
4	`-w-`	`-wx`
5	`-w-`	`-w-`
6	`---`	`--x`
7	`---`	`---` (none)