Introduction to using the Unix shell

Damiano Ravalico

University of Trieste

A.Y. 2022/2023

Fundamental concepts

Users and Groups

Definition

Every user of the system has a unique login name (*username*) and a corresponding numeric user ID (*UID*).

Users and Groups

For each user, these are defined by a line in the system password file, **/etc/passwd**, which includes the following additional information:

- *Group ID:* group ID of the first of the groups of which the user is a member
- *Home directory:* the initial directory into which the user is placed after logging in
- *Login shell:* the name of the program to be executed to interpret user commands

Users and Groups

Definition

A group is a set of users, divided into it for for administrative purposes (i.e. for controlling access to files and other system resources).

Users and Groups

Each group is identified by a single line in the system group file, **/etc/group**, which contains:

- *Group name:* the (unique) name of the group
- *Group ID (GID):* the numeric ID associated with this group
- *User list:* a comma-separated list of login names of users who are members of this group

Users and Groups

Definition

There is a one special user, known as superuser, that has special privileges within the system; its account has *ID 0*. The superuser bypasses all permission checks in the system.

Unix filesystem

Single Directory Hierarchy

- The kernel maintains a single hierarchical directory structure to organize all files in the system
- At the base there is the root directory, named / (slash)
- All files and directories are children or descendants of the root directory

Example

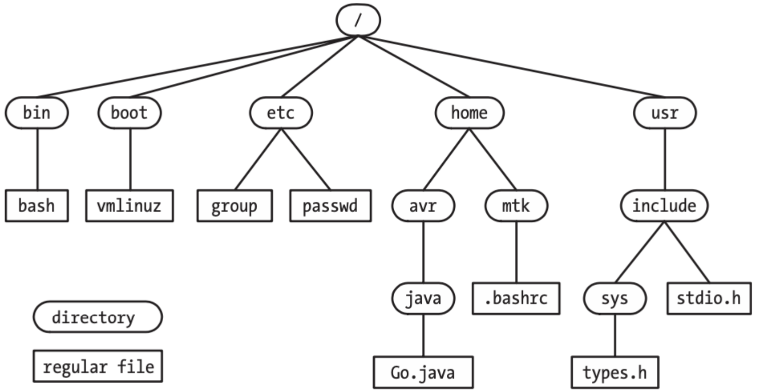


Figure: Example of unix filesystem organization

Files and types

- Within the file system, each file is marked with a type, indicating what kind of file it is (e.g. **.txt**).
- Ordinary data files are usually called regular
- Note that the term *file* is commonly used to denote a file of any type, not just a regular file

Directories and Links

Definition

A directory is a special file whose contents take the form of a table composed of couples in the form {filenames, references} to the corresponding files; this association is called link.
Filenames can be up to 255 characters long and may contain any characters except slashes (/) and null characters (\0).

Directories and Links

- Directories may contain links both to files and to other directories
- Every directory contains at least two entries: . (dot), which is a link to itself, and .. (dot-dot), which is a link to its parent directory

Pathnames

Definition

A pathname is a string consisting of an *optional* initial slash (/) followed by a series of filenames separated by slashes. All but the last of these component filenames identifies a directory: it may identify any type of file, including a directory.

Pathnames

- Absolute pathname begins with a slash (/) and specifies the location of a file with respect to the root directory
- Relative pathname specifies the location of a file relative to the current directory

Shell and Terminal Emulator

What is shell?

Definition

Shell is a special-purpose program that reads text commands entered by the user, interprets them and passes them to the underlying operating system, in particular the kernel.

Definition

Terminal emulator is a program that opens a GUI (Graphical User Interface) to allow interaction with the shell.

In most Linux distributions the default textual shell is called Bash.

In practice

A command is a sequence of characters ending with the newline symbol (*enter* key). These commands are predefined; they can be run alone, combined in a pipeline, or combined within a script.

Bash allows you to execute an pipeline of commands, where the output of the previous command is the input of the next command.

File Permission and Access Modes

File Permissions

For the purpose of accessing a file, the system divides users into three categories:

- *Owner* of the file
- *Group:* users who are members of the group matching the file's group ID
- *Others:* the rest of the world

Access Modes

Three permission bits may be set for each of these categories (a total of nine permission bits):

- *Read:* user can read the contents of the file
- *Write:* user can modify the contents of the file
- *Execute:* user can execute the file (typically a script or program)

Access Modes

If the file is a directory there are some differences:

- Read permission allows the contents of (i.e., the filenames in) the directory to be listed
- Write permission allows the contents of the directory to be changed (i.e., filenames can be added, removed, and changed)
- Execute permission allows access to files within the directory

Symbolic notation

Using the **ls -l** command in the terminal, it is possible to view the access modes of the files. This notation is called symbolic notation.

Symbolic notation

The first character indicates the file type and is not related to permissions. The remaining nine characters are in three sets, each representing a class of permissions as three characters.

- The first set represents the user category
- The second set represents the group category
- The third set represents the others category

Symbolic notation

Each of the three characters represent the permissions:

- **r** if reading is permitted, - if it is not
- **w** if writing is permitted, - if it is not
- **x** if execution is permitted, - if it is not

Symbolic notation - examples

Example

- **-rwxr-xr-x**: a regular file whose user category has full permissions and whose group and others categories have only the read and execute permissions
- **dr-x——**: a directory whose user category has read and execute permissions and whose group and others categories have no permissions

Octal notation

Binary value (**rwx**)	Octal value	Permissions
000	0	none
001	1	execution only
010	2	writing only
011	3	writing and execution
100	4	read only
101	5	reading and execution
110	6	reading and writing
111	7	read, write and execute

Table: Permission representations

Main commands

Command format

command_name [option(s)] [parameter(s)]

- **command_name:** is the name of the command to perform
- **option**: modifies a command's operation. To invoke it, use hyphens (–) or double hyphens (—)
- **parameter:** specifies any necessary information for the command

Note that:

- A command may contain an option or a parameter
- Commands are case-sensitive

List

ls: lists the contents of the current directory

Example

- **ls *.pdf** lists all files ending with the suffix .pdf located in the current directory
- **ls -a** shows hidden files in addition to the visible ones
- **ls -lh** shows the file sizes in easily readable formats

Change directory

cd: changes the current folder to the specified one

Example

- **cd** change the current directory to the home of the current user
- **cd /temp** change the current directory to **temp**
- **cd ..** moves one directory up
- **cd-** moves to your previous directory

Concatenate

cat: lists, combines, and writes file content to the standard output

Example

- **cat filename.txt** displays content
- **tac filename.txt** displays content in reverse order

Copy

cp: copies files or directories and their content

Example

- To copy one file from the current directory to another, enter **cp** followed by the file name and the destination director
- To copy files to a directory, enter the file names followed by the destination directory
- **cp -R dir_to_copy /destination/path/** copies the directory and the entire subtree connected at that point in the specified path

Create directory

mkdir: creates one or multiple directories at once

Example

mkdir test creates a directory named **test** in the current directory

Remove

rm: deletes file and directory

Example

- **rm filename1 filename2** deletes all three files
- **rm -r** delete an entire folder (even if not empty)

Create and modify files

touch: creates an empty file with the specified name
nano: allows users to edit and manage files via text editor
vi: another text editor, standard, unlike **nano** which is only present in some linux distros

Change mode

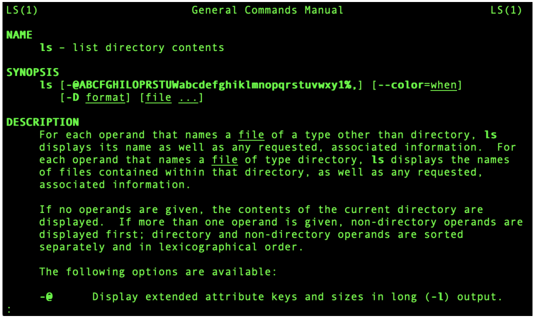
chmod: modifies a file or directory's read, write, and execute permissions

Example

chmod 777 test.md changes the file permissions to the **-rwxrwxrwx** permission type, whose numeric value is 777

User manual

man command_name provides a user manual of any commands or utilities you can run in Terminal, including the name, description, and options. To exit from it, press *q*.



```
LS(1)                        General Commands Manual                       LS(1)

NAME
     ls - list directory contents

SYNOPSIS
     ls [-@ABCFGHILOPRSTUWabcdefghiklmnopqrstuvwxy1%,] [--color=when]
        [-D format] [file ...]

DESCRIPTION
     For each operand that names a file of a type other than directory, ls
     displays its name as well as any requested, associated information.  For
     each operand that names a file of type directory, ls displays the names
     of files contained within that directory, as well as any requested,
     associated information.

     If no operands are given, the contents of the current directory are
     displayed.  If more than one operand is given, non-directory operands are
     displayed first; directory and non-directory operands are sorted
     separately and in lexicographical order.

     The following options are available:

     -@      Display extended attribute keys and sizes in long (-l) output.
:
```

Figure: First page of the **ls** command manual

Pipes

File descriptors

Definition

A file descriptor is a non-negative integer representing any type of file opened by a process and on which the process can perform input/output operations. Each process has its own set of file descriptors.

File descriptors

File descriptor	Purpose
0	standard input
1	standard output
1	standard error

Table: Standard file descriptors

Pipe

Definition

An pipe is a tool that allows processes to communicate with each other. In particular, it is a method for linking the stdout of one program to the stdin of the next.

Pipe

- A pipe is a unidirectional communication channel
- A pipe is a byte stream: there is no concept of messages or message boundaries when using a pipe
- Pipes have a limited capacity
- To concatenate commands use the |, to redirected the stdout of a command to a file use the >

Pipe - example

ls | wc -l

- In order to execute the above command, the shell creates two processes, executing **ls** and **wc**
- **ls** lists the file in the current directory
- | symbol indicates that the output of the first command is the input of the second
- **wc -l** uses that list to print the number of lines (**-l** option) i.e. the length of the list in this case