Unit 2

OS and Tools

Alberto Casagrande
Email: `acasagrande@units.it`

a.a. 2019/2020

As Programmers . . .

we are interested in:

- reading data from an input device
- implementing functions to operate on data sets
- providing results to an output device

As Programmers . . .

we are NOT interested in:

- how the I/O devices work
- where the data are physically stored in memory
- how our programs will be executed
- . . .

As Programmers . . .

we are NOT interested in:

- how the I/O devices work
- where the data are physically stored in memory
- how our programs will be executed
- . . .

We need an abtraction layer between HW and programs

As Programmers . . .

we are NOT interested in:

- how the I/O devices work
- where the data are physically stored in memory
- how our programs will be executed
- . . .

We need an abtraction layer between HW and programs

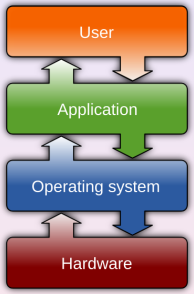
We need an Operative System

Operative Systems

Software that manage resources

- memory
- disks
- CPUs
- ...

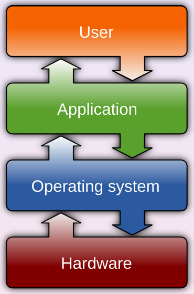
Provide interfaces to programs (API) and users



Operative Systems

Software that manage resources

- memory
- disks
- CPUs
- ...



Provide interfaces to programs (API) and users

There exist hundreds of operative systems e.g., Windows, OSX, BeOS, GNU/Linux, iOS, Android, ReactOS

POSIX standard

Is a IEEE standard about:

- Process (i.e., programs in execution) creation and control
- File and directory operations
- C library
- I/O port interface and control
- Command interpreter
- Standard utility and command
- ...

POSIX standard

Is a IEEE standard about:

- Process (i.e., programs in execution) creation and control
- File and directory operations
- C library
- I/O port interface and control
- Command interpreter
- Standard utility and command
- ...

OSX is POSIX-certified, Windows is not POSIX compliant.

GNU/Linux

Is a free (as in speach) OS.

- Linux is the core
- GNU is a project that provides tools and command, e.g., GNU C Compiler - GCC

GNU/Linux

Is a free (as in speach) OS.

- Linux is the core
- GNU is a project that provides tools and command, e.g., GNU C Compiler - GCC

There exist many different distributions. They customize:

- installer
- software manager
- additional software

GNU/Linux

Is a free (as in speach) OS.

- Linux is the core
- GNU is a project that provides tools and command, e.g., GNU C Compiler - GCC

There exist many different distributions. They customize:

- installer
- software manager
- additional software

POSIX compliant, but not certificate (freedom has a price)

GNU/Linux

You will extensively use Linux during DSSC program

GNU/Linux

You will extensively use Linux during DSSC program

Let's see how to install it and use it

We will focus on Ubuntu distribution (not the "best", but user-friendly)

How to install Ubuntu

We need:

1. a PC with a USB port
2. 16GB of free disk space
3. a USB key (at least 2GB)
4. a network connection

How to install Ubuntu

We need:

1. a PC with a USB port
2. 16GB of free disk space
3. a USB key (at least 2GB)
4. a network connection

or

1. a PC running a virtualization environment, e.g., VirtualBox
2. 16GB of free disk space
3. a network connection

How to install Ubuntu

If you opt for a “real” installation, you need to:

- download Ubuntu
- download and install Etcher
- prepare a bootable USB Live key by using Etcher
- reboot your PC and select the USB key ad boot device
- follow the instructions (pay attention and do not delete your OS!!!)

How to install Ubuntu

If you opt for a “virtual” installation, you need to:

- download Ubuntu
- download and install VirtualBox
- create your VM
- attach the Ubuntu ISO to your VM and boot from it
- follow the instructions (don’t worry about messing up with the VM’s disk)

How to install Ubuntu

Demo session

Users in Modern OS

Modern OS are multi-users, i.e., they support many users on the same system

Every user has a reserved disk space (dubbed home in POSIX) where they can store personal data and program

So, after the boot, the system asks for a username and a password

Say "Hello" to Command Line

Ubuntu has a fully functional Graphical User Interface (GUI)

We will use the command-line user interface (a.k.a. shell) because it's:

Say “Hello” to Command Line

Ubuntu has a fully functional Graphical User Interface (GUI)

We will use the command-line user interface (a.k.a. shell) because it’s:

- easier to use (for experts)
- powerful
- programmable
- cool

Say “Hello” to Command Line

Ubuntu has a fully functional Graphical User Interface (GUI)

We will use the command-line user interface (a.k.a. shell) because it’s:

- easier to use (for experts)
- powerful
- programmable
- cool

The default shell in Ubuntu is BASH.

Few info about secondary memory

Data are maintained in disks by an OS component called file system

Many kinds of FS e.g., VFAT, Ext4 (GNU/Linux “default”), APFS (OSX), NTFS (Windows)

Data are organized in a *tree* of directories (branches of the tree).

File Systems in POSIX

- the symbol / to distinguish branch levels, e.g., `/home/al`
- / is the root of the tree
- /home contains the users' homes
- ./ is the current directory
- ../ is the parent level
- ~ denotes the current user's home

File Systems in POSIX

- the symbol / to distinguish branch levels, e.g., `/home/al`
- / is the root of the tree
- /home contains the users' homes
- ./ is the current directory
- ../ is the parent level
- ~ denotes the current user's home

Directory names can be composed to specify a *path*

- absolute paths start from the root e.g., `/home/al/Desktop/` or `/user`
- relative paths start from the current active/directory, e.g., `Desktop`, `./Download/`, or `Download/../Desktop`

Demo session

Some simple BASH commands

- ls lists the content of a directory

```
al@foo:~/$ ls
Desktop             Download    Documents
examples.desktop    Music       Pictures
Templates           Public      Video
```

- pwd prints the name of the current/working directory

```
al@foo:~/$ pwd
/home/al
```

Some simple BASH commands (Cont'd)

- mkdir create new directories

```
al@foo:~/$ mkdir test
al@foo:~/$ ls
Desktop            Download   Documents
examples.desktop   Music      Pictures
Templates          test       Public
Video
```

- cd change directory. Without parameter means "go to home"

```
al@foo:~/$ cd test
al@foo:~/test$ cd ../../usr
al@foo:/usr$ cd
al@foo:~/$ cd foo
bash: cd: foo: No such file or directory
```

Some simple BASH commands (Cont'd 2)

- rm delete files/directories

```
al@foo:~/$ rm -r test
```

- chmod change access permissions
- man print command manual pages

```
al@foo:~/$ man ls
```

- apropos search words in manual pages

```
al@foo:~/$ apropos compress
```

Some simple BASH commands (Cont'd 3)

- grep print lines matching a pattern

```
al@foo:~/$ grep "\[it\]" examples.desktop
Name[it]=Esempi
Comment[it]=Contenuti di esempio per Ubuntu
```

- cat print a file on the stdout
- less print a file on terminal one screenful at a time
- head output the first part of files

```
al@foo:~/$ head -n 3 examples.desktop
```

Pipelining

We can use | (pipe) to use the output of a command as the input of another

```
al@foo:~/$ echo "Hello,␣men!"
Hello, man!
al@foo:~/$ echo "Hello,␣men!" | sed s/Hello/Hi/
Hi, man!
```

Output Redirection

```
al@foo:~/$ echo "Hello,␣men!"
Hello, man!
al@foo:~/$ echo "Hello,␣men!" > msg.txt
al@foo:~/$ cat msg.txt
Hello, man!
al@foo:~/$ echo "This␣is␣cool" > msg.txt
al@foo:~/$ cat msg.txt
This is cool
```

Output Redirection (Cont'd)

```
al@foo:~/$ echo "This␣is␣much␣better" >> msg.txt
al@foo:~/$ cat msg.txt
This is cool
This is much better
```

File Descriptors

Are positive numbers that “name” files

POSIX systems handle everything as files

stdin, stdout, and stderr have FD 0, 1, and 2.

File Descriptors (Cont'd)

We can stream data to either stdout and stderr

```
al@foo:~/$ echo "WHAT?!?!" >&1
WHAT?!?!
al@foo:~/$ cat msg.txt
This is cool
This is much better
al@foo:~/$ cat msg.txt | grep cool
This is cool
al@foo:~/$ cat msg.txt >&2 | grep cool
This is cool
This is much better
```

File Descriptors (Cont'd 2)

We can also data from either stdin, stdout and stderr

```
al@foo:~/$ cat test.txt
cat: test.txt: No such file or directory
al@foo:~/$ (echo "N" >&1; echo "Y" >&2)
N
Y
al@foo:~/$ (echo "N">&1;echo "Y">&2)1>test.txt
Y
al@foo:~/$ cat test.txt
N
al@foo:~/$ (echo "N!";echo "Y">&2)1>test.txt
Y
al@foo:~/$ cat test.txt
N!
```

(Extended) Regular Expressions

Are patterns to describe strings.

E.g., `[az]T.` describes strings beginning with either `aT` or `zT` and having 3 characters

.	any single character
-	denotes a range e.g., `a-z`
?	the prev item at most once e.g., `a?`
+	the prev item at least once
*	the prev item occurs from 0 to many times
()	bound a sub-RE
|	match both RE e.g., `(it)|(comm)`
[ ]	choose one in the set e.g., `[a-z]`

(Extended) Regular Expressions (Cont'd)

^ and $ denote begin and end of a line, respectively.

```
al@foo:~/$ grep u.*b.* msg.txt
This is much better
al@foo:~/$ grep "\(b.*\)\|\(cool\)" msg.txt
This is cool
This is much better
```

The escape character \ is needed because `grep` uses basic regular expression by default.

Programmability

Shells can be programmed to perfom complex tasks

```
al@foo:~/$ for i in $(seq 1 3); do
> echo test_${i};
> done > test.txt
al@foo:~/$ cat test.txt
test_1
test_2
test_2
```

If we have enough time, we will focus on it next week.

A user to rule them all . . .

Not all users must have the same privileges

E.g.,

- The system owner should be able to do everything
- A “host user” should not mess-up other users’ data

A user to rule them all . . .

Not all users must have the same privileges

E.g.,

- The system owner should be able to do everything
- A “host user” should not mess-up other users’ data

Modern OS provide a “superuser” to rule all the system files/programs.

root in POSIX systems, administrator in Windows.

Impersonate Superuser

Ubuntu and OSX implement a mechanism to impersonate superuser.

sudo (Super User DO) lets authorized users to impersonate superuser.

```
al@foo:~/$ mkdir /test
mkdir: cannot create directory '/test': Permission
denied
al@foo:~/$ sudo mkdir /test
[sudo] password for al:
al@foo:~/$
```

Package Managers and APT

Every GNU/Linux distribution provides a package manager.

It is useful to install, update, and remove software.

Ubuntu adopts the Advanced Packaging Tool (apt) which also implements:

- dependency handling
- download
- package security validation (digital signature)

It download and maintain from a set of sources a DB of packages and description.

Installing basic development tools

To install basic development tools, follow the next steps:

1.
   ```
   al@foo:~/$ sudo apt-get update
   ```
2. enter user's password
3.
   ```
   al@foo:~/$ sudo apt-get install build-essential
   ```
4. press "Y" and "Enter"

A Distro-Independent Package Manager

Snappy is a distribution independent package manager for applications

All the features of APT, but easier to maintain

```
al@foo:~/$ sudo snap list "visual␣studio"
Name                    Version    Publisher   Notes
Summary
code                    3db7e09f   vscode      classic   Code
code-insiders           211fa02e   vscode      classic   Code
epipolar-consistency    0.1        thahamsta   -
Consistency Conditions for any two X-ray images.
al@foo:~/$ sudo snap install code --classic
code 3db7e09f from Visual Studio Code (vscode) installed
```

Coming soon. . .

- the first C program
- types
- variables
- assignments
- numeric expressions
- output