



Linux base per HPC

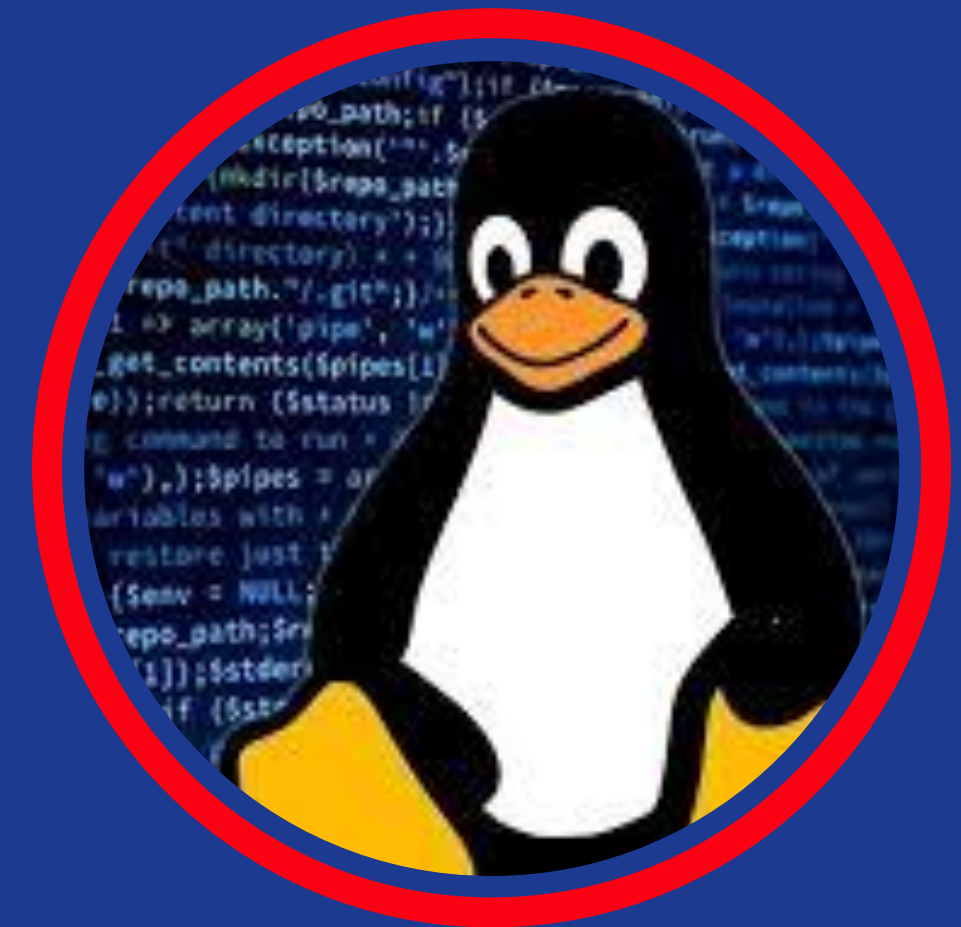
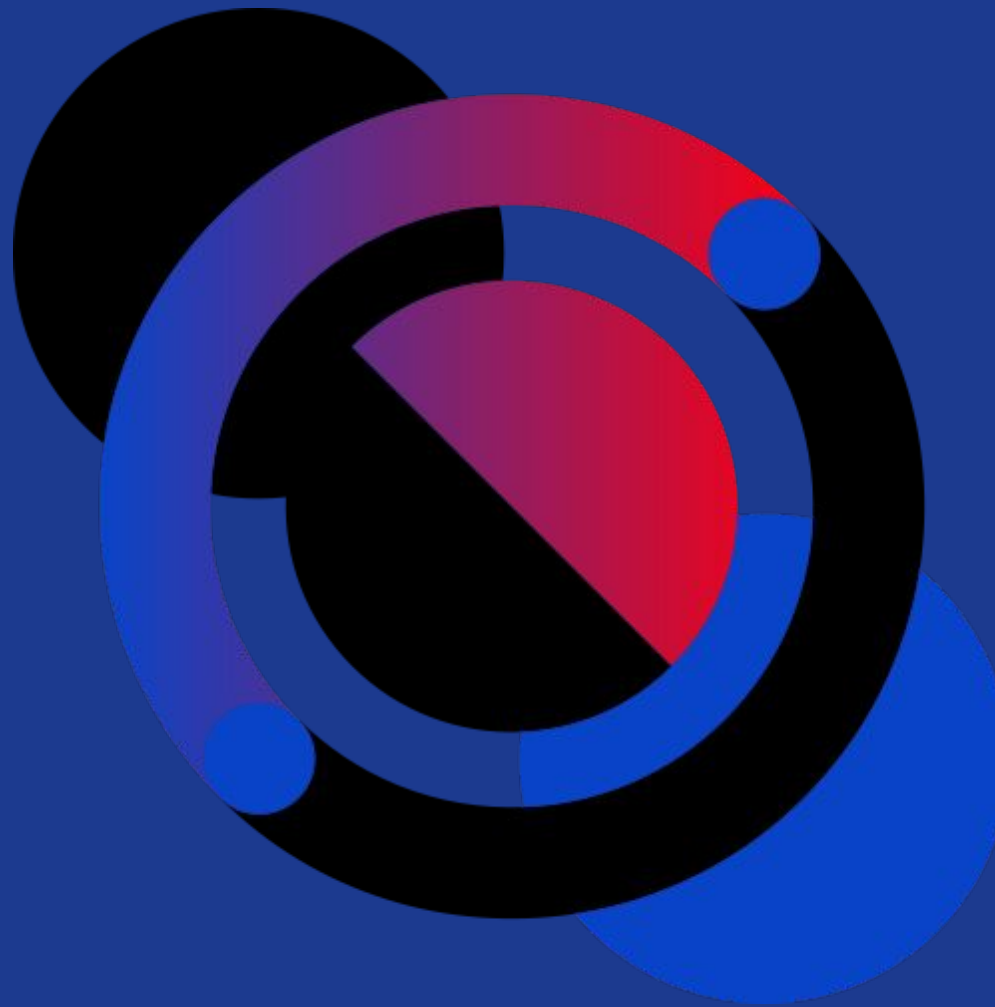


Michele Lombardi

micro-Y

COSA FACCIAMO OGGI?

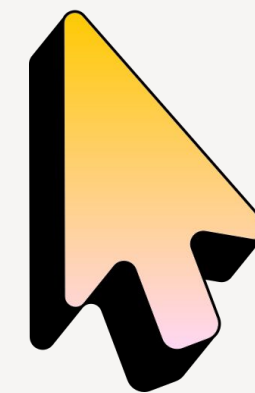
Impariamo a sopravvivere a linux



Per lavorare su di un HPC non
avrete un interfaccia come siete
abituati ma una shell.
Oggi imparerete ad usarla!

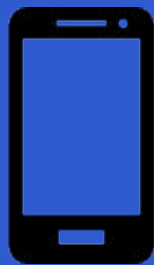
AGENDA

- 01** Storia di Linux
- 02** Come collegarsi
- 03** Kit di sopravvivenza
- 04** Esercizi e domande



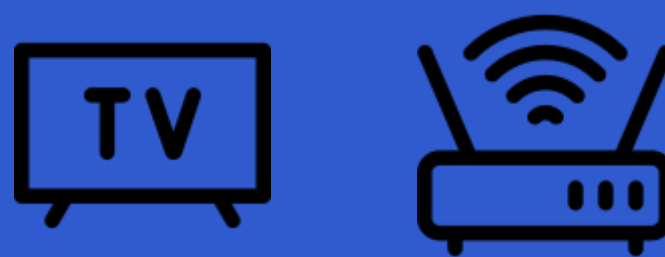
Dove si trova Linux solo su HPC?

01



Android usa il kernel linux come base del sistema aggiungendo sopra un interfaccia e app e voi non ve ne accorgete.

02



Molti dispositivi che usate tutti i giorni hanno come base linux, per esempio le smart tv, il router del wifi le console da gaming e molto altro.

03



Server di molti tipi, tra cui HPC
Linux è stabile, automatizzabile ottimo per il multi utente e il calcolo e ha moltissimi strumenti scientifici e ambienti software già standard

Mini storia linux



01

Anni '70

Nasce l'idea di un sistema multiutente dove tutto è un file

02

Anni '80

Si aggiungono librerie, comandi e compilatori

03

1991

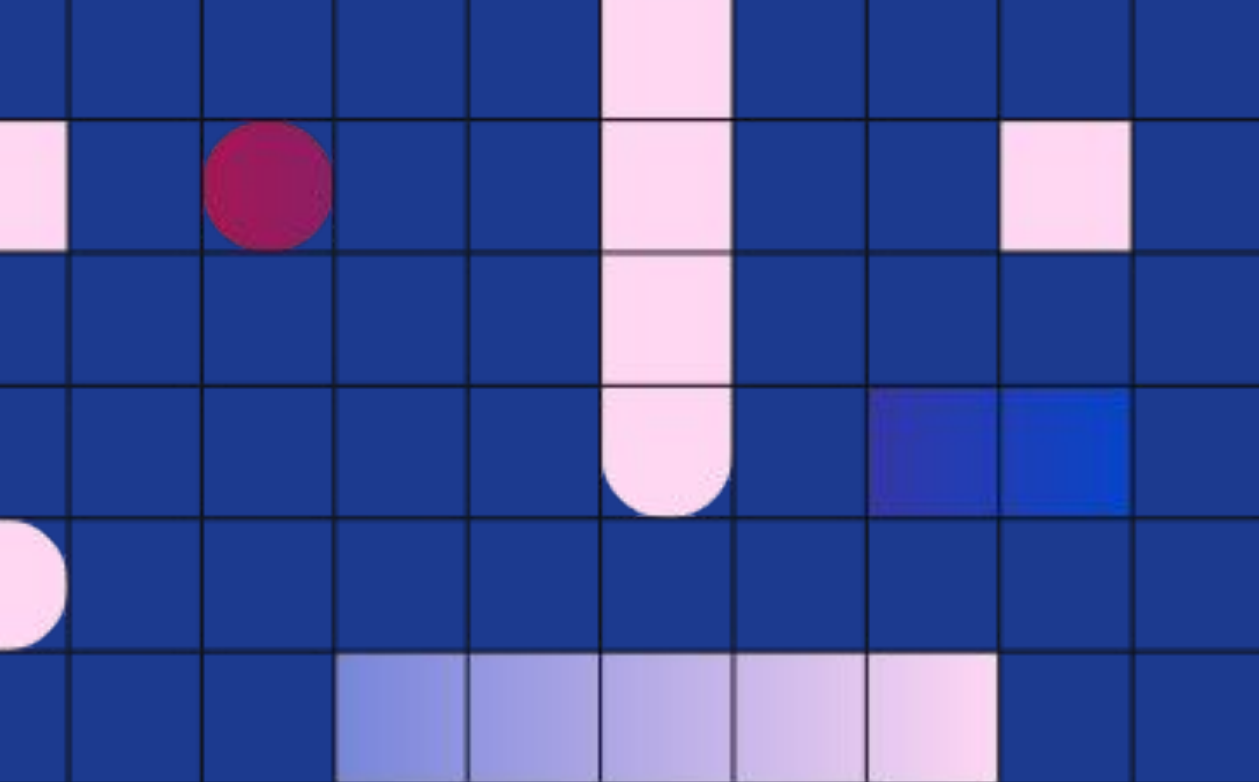
Nasce il kernel linux e si unisce agli strumenti GNU

04

Oggi

Standard per server, cloude e HPC



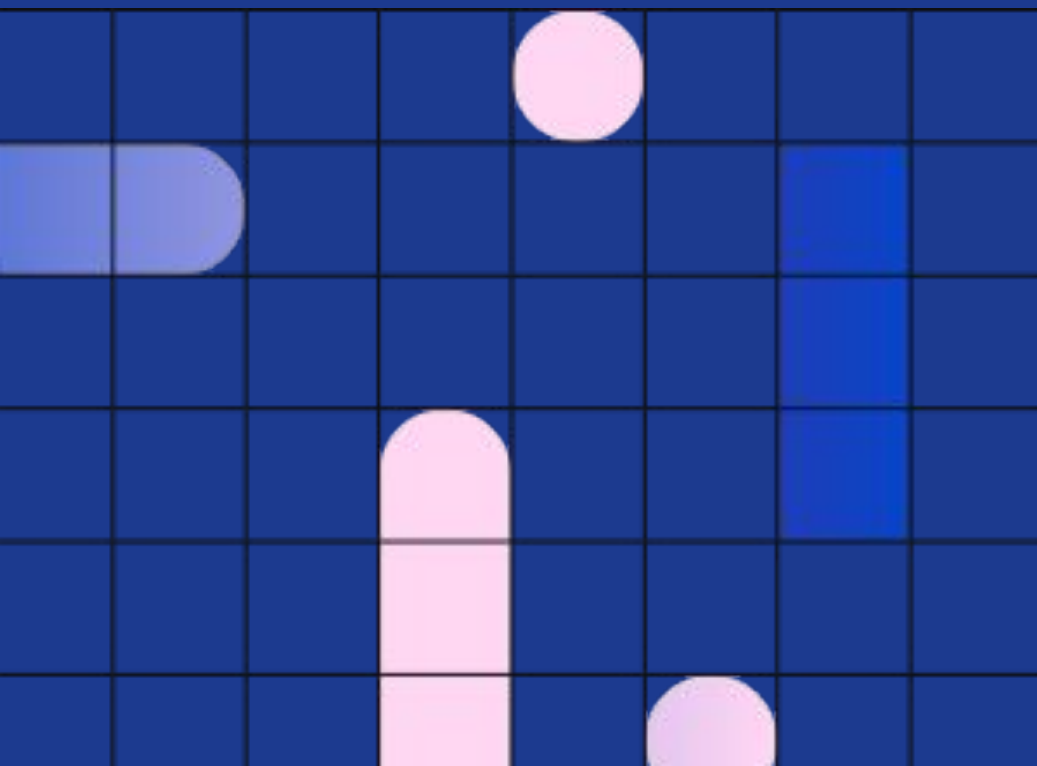


Kernel

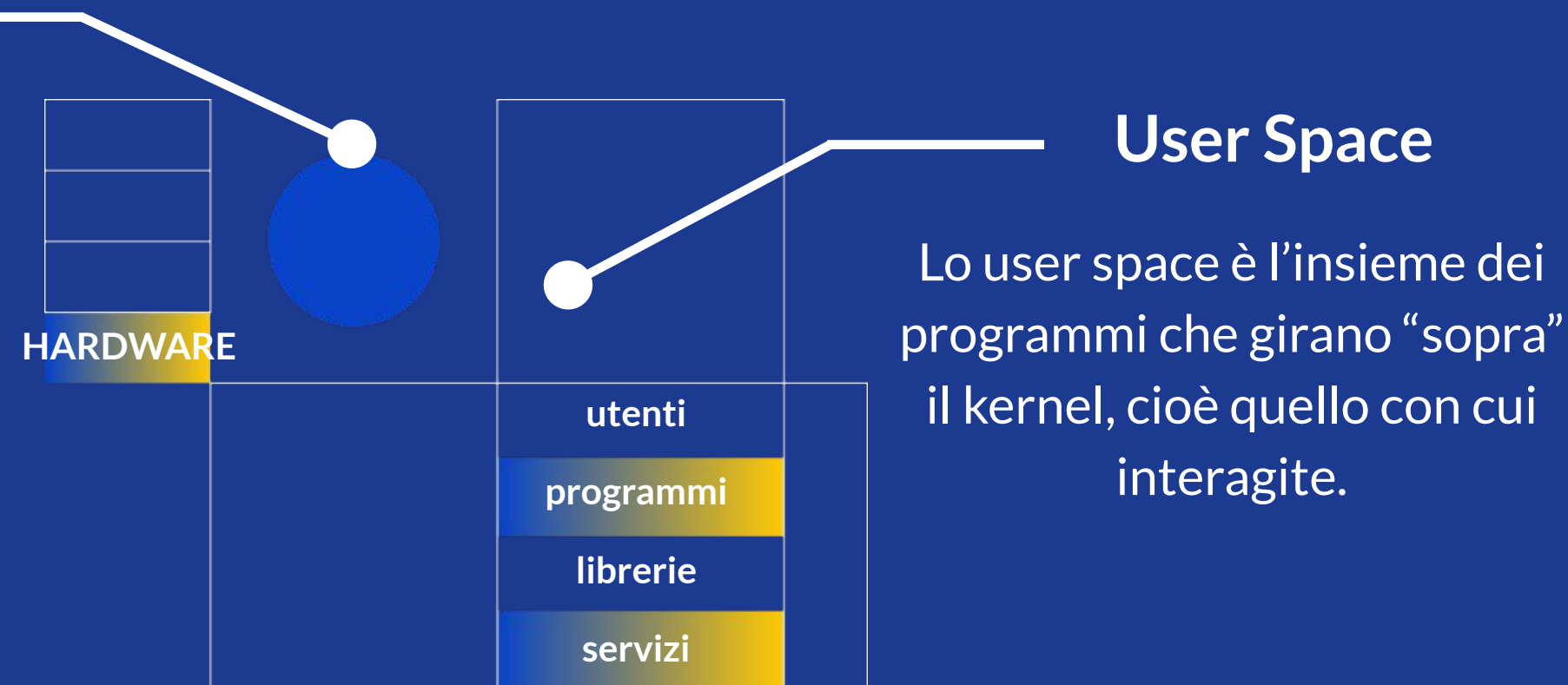
Il kernel è il software che sta più vicino all'hardware e decide come usare le risorse.

- CPU
- RAM
- DISCHI
- ECC

Ma Linux?



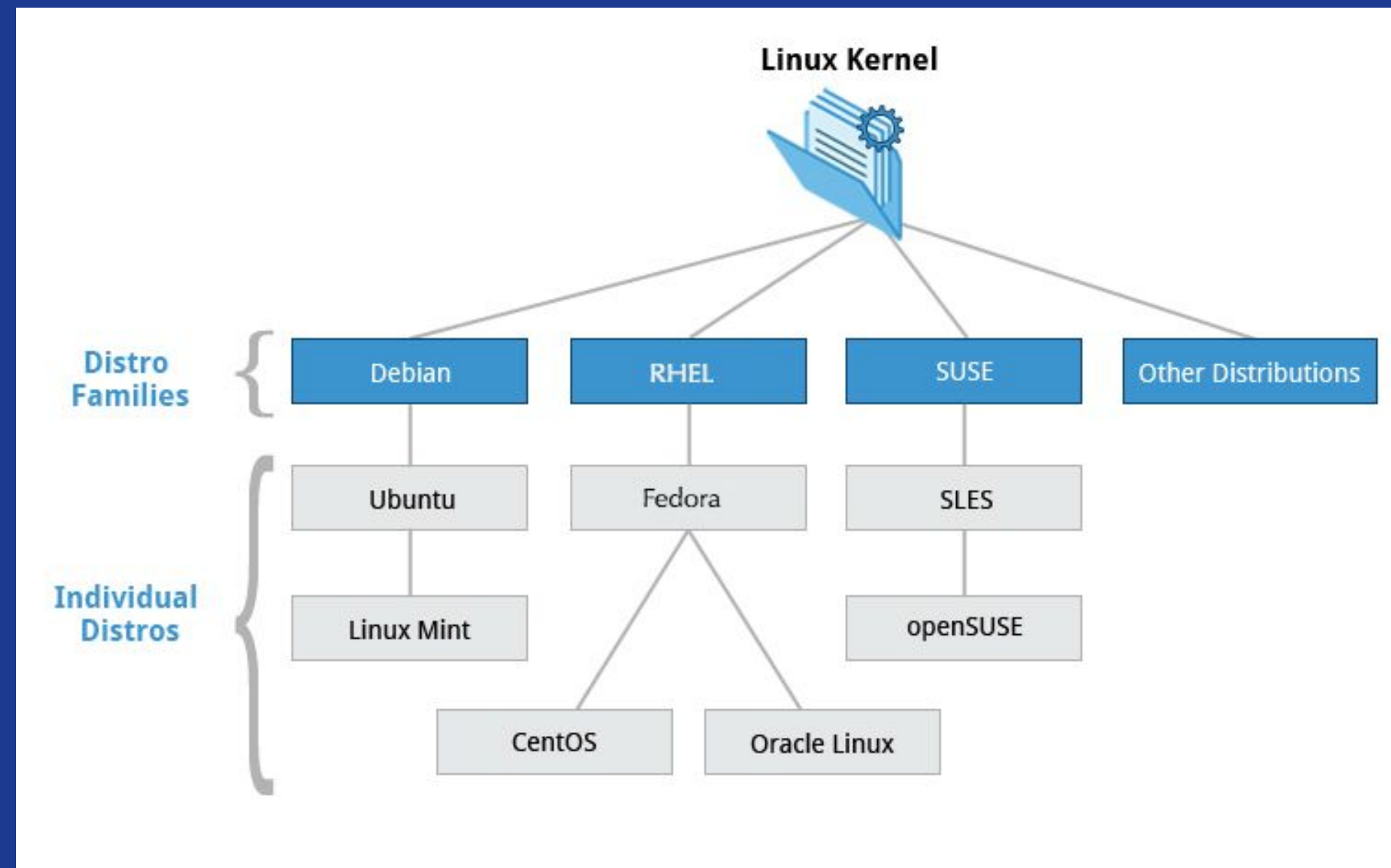
La GUI è
opzionale



Una distro è un sistema Linux "confezionato":

- Kernel
- Pacchetti (programmi, librerie, tool)
- Configurazioni (servizi, utenti, rete, ecc.)
- Package manager (apt, dnf, pacman...) per installare/aggiornare software

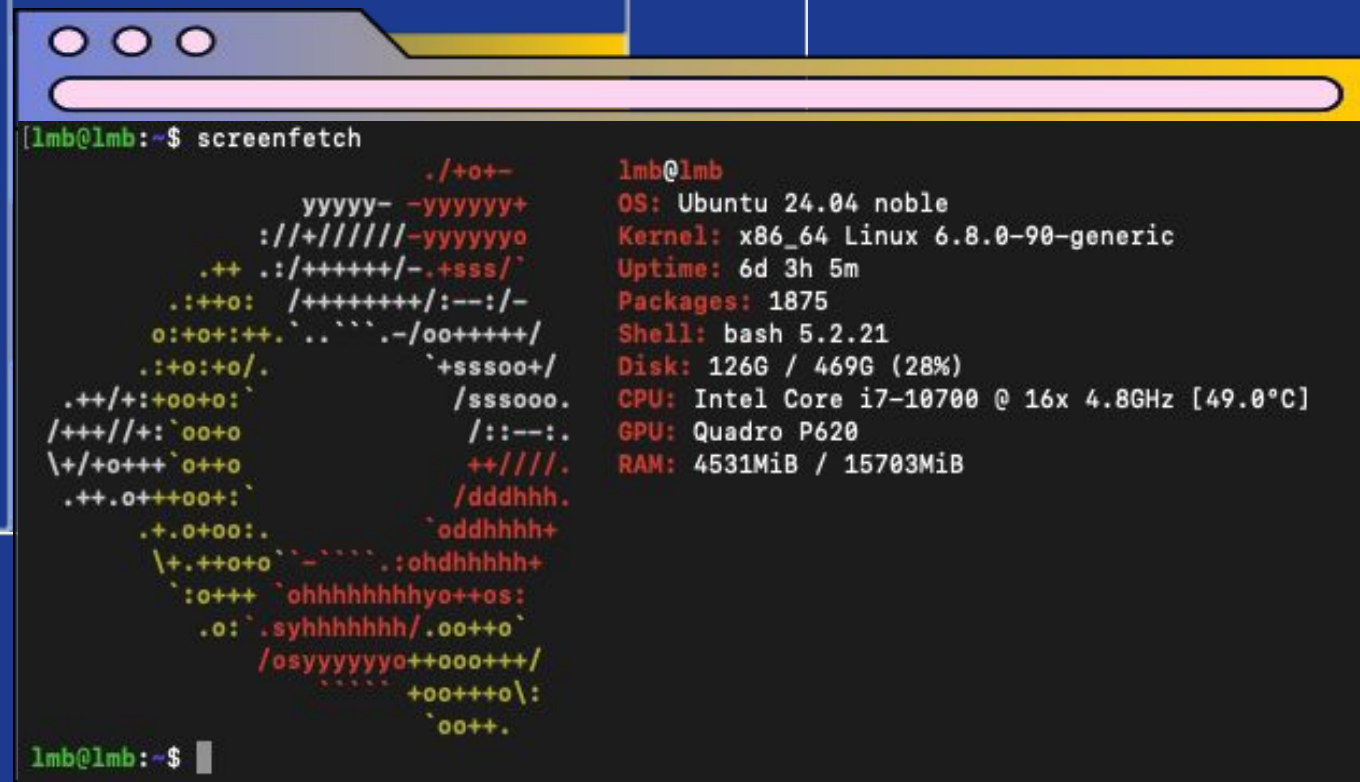
DISTRIBUZIONI



Un disastro, ma per vostra fortuna i
comandi base sono comuni

Confrontiamolo con WINDOWS

PRIMA COSA: FUNZIONA!



```
lmb@lmb:~$ screenfetch
      ./+o+-
    yyyyy- -yyyyyy+
  ://+///// -yyyyyyo
    .++ .:/+++++/-+.sss/`
  .:++o: /+++++/:--:/-
    o:+o+:+. `..``.-/oo++++/
  .:o:+o/.      +sssoo+/
.++/+:+oo+o:`   /sssoo.
/+++//+:`oo+o   /:-:--:
\+/+o+++`o++o   ++////.
.++.o+++oo+:`   /dddhhh.
+.o+oo:.      `oddhhhh+
\+.++o+o`-`-`-`.:ohdhhhh+
`o+++ `ohhhhhhhh+o++os:
.o:`.syhhhhhhh/.oo++o`
  /osyyyyyyo++ooo+++/
      +oo++o\.:
      `oo++.
```

lmb@lmb:~\$

01

Percorsi

/(non C:\) → es. /home/nomeutente/

02

Case sensitive

Dati ≠ dati

03

Estensioni

Estensione non decide tutto: conta tipo + permessi

04

Tutto é file

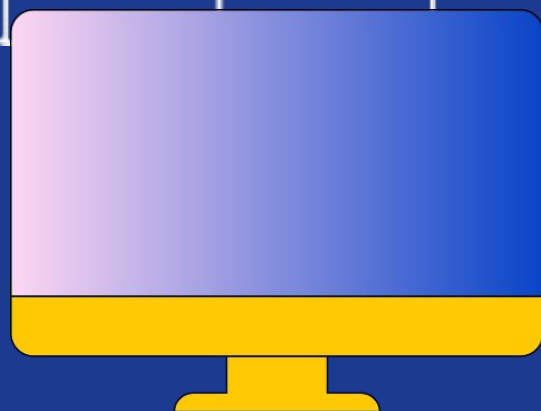
configurazioni, log, dispositivi

05

utenti

owner / group / others

2 modi per usarlo

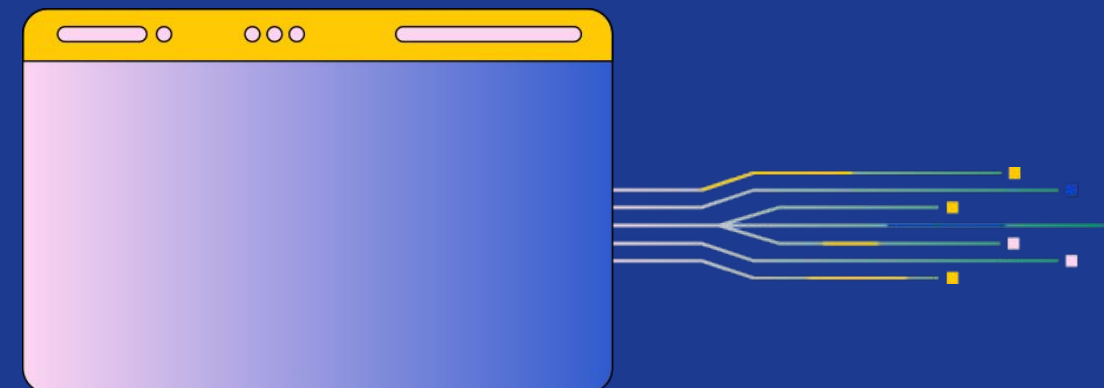


Locale

Linux installato / VM / WSL (Windows)



Utile per fare i test ed
imparare



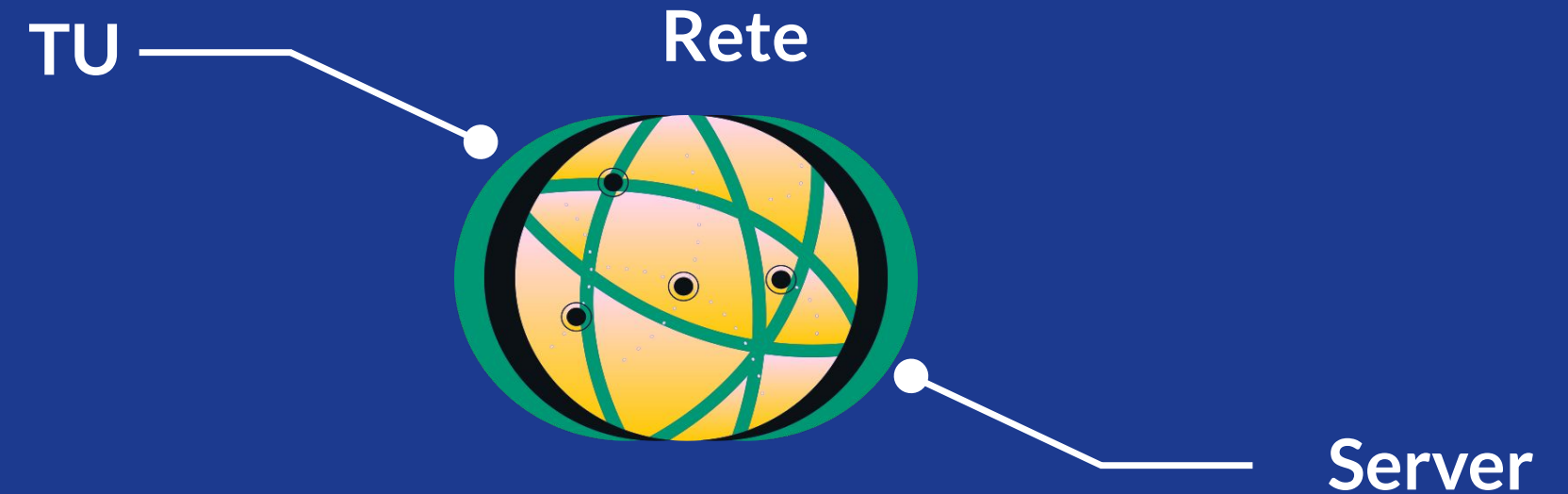
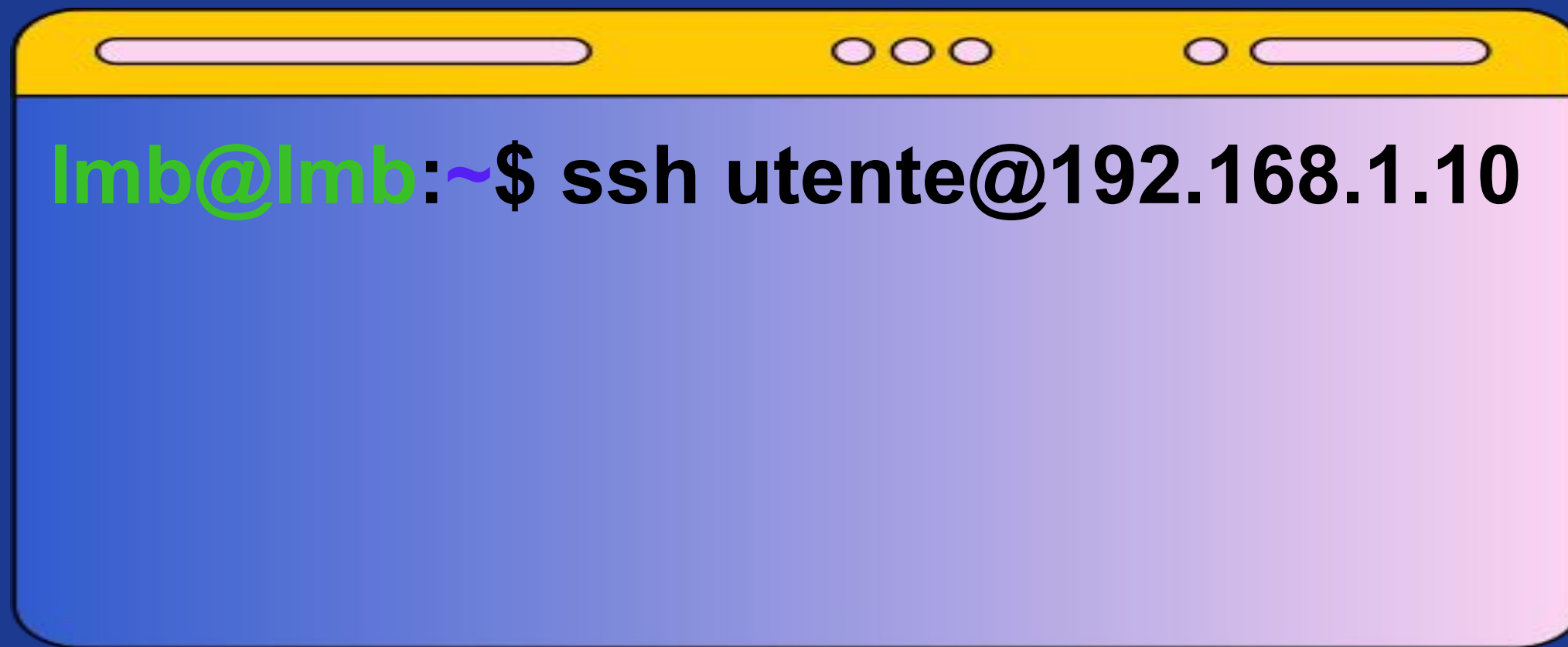
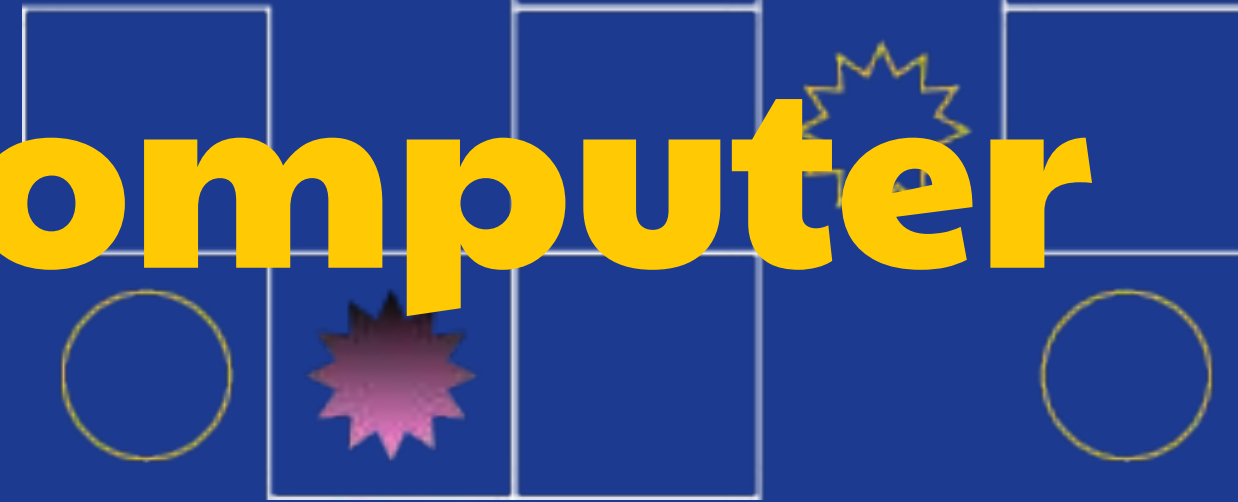
Remoto

vi collegate a una macchina del cluster via SSH



Il vostro pc diventa una
tastiera+schermo

SSH: entrare in un computer remoto



ti “ritrovi” nel terminale della macchina remota

i file e comandi che usi sono del server, non del tuo PC

tramite ssh puoi inviare file

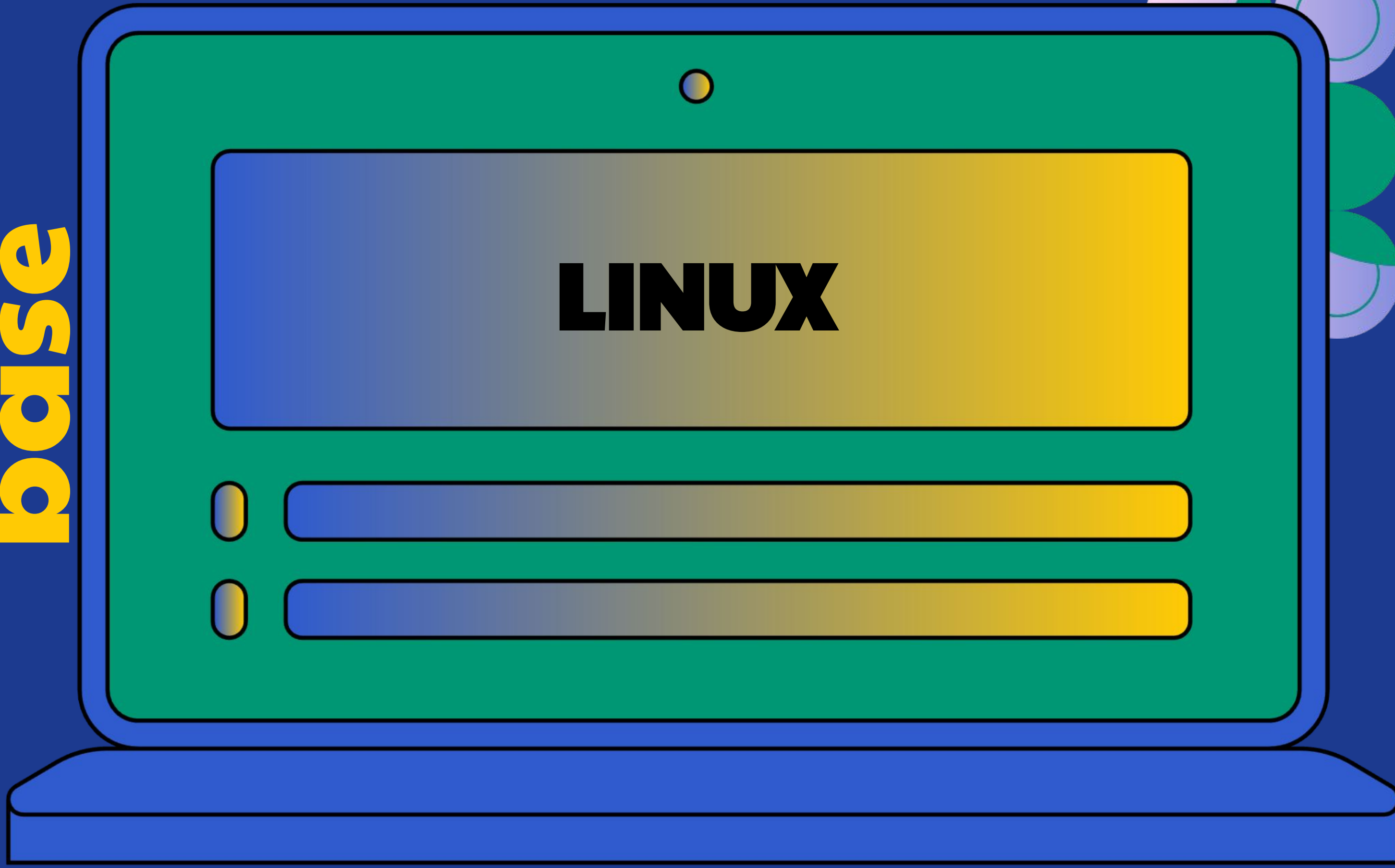


```
lmb@lmb:~$ scp ./file.txt utente@192.168.1.10:/home/lmb/  
lmb@lmb:~$ scp -r ./progetto utente@192.168.1.10:/home/lmb/
```

permette di copiare file o cartelle (con -r) dal proprio pc al server in un percorso specifico

commandi

base



Prima di tutto dove sono?

Dove mi trovo ?

`pwd`

restituisce il percorso completo dalla route /

Cosa c'è qui?

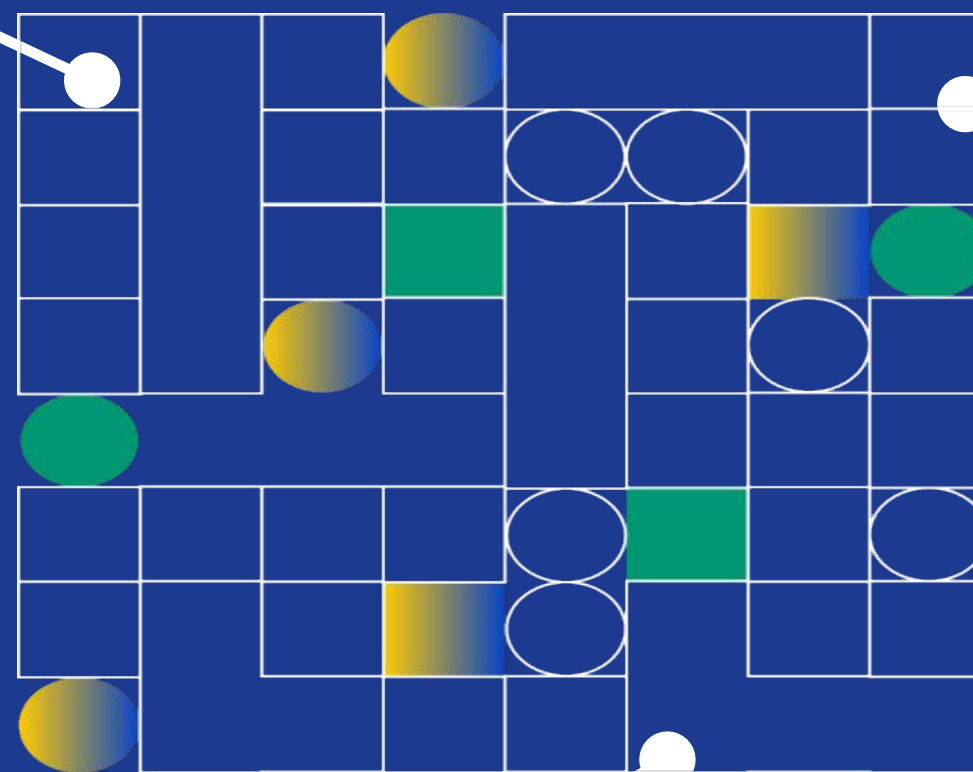
`ls`

restituisce lista file e cartelle del punto in cui ti trovi

come mi sposto ?

`cd`

ti permette di andare avanti
tornare indietro o saltare in
una alta parte



pwd



significa print working directory



Ti dice il percorso completo (assoluto) della cartella corrente

```
lmb@lmb ~/micro-y-hub pwd
/home/lmb/micro-y-hub
```

evita di lavorare nella cartella sbagliata

e ti aiuta a capire percorsi assoluti vs relativi

ls



abbreviazione di list



Serve ad elencare il contenuto della cartella corrente

```
lmb@lmb ~/micro-y-hub ls
backend      docs
bun.lockb    eslint.config.js
components.json index.html
README.md    tsconfig.app.json
docker       node_modules
scripts      tsconfig.json
```

■ ls -l → dettagli (permessi, owner, dimensione, data)

■ ls -a → include file nascosti (.config, .ssh...)

■ ls -h → dimensioni “umane” (KB, MB...)

■ ls -lah → combo tipica per lavorare bene

cd



significa change directory



Serve per navigare nel file system

```
lmb@lmb ~/micro-y-hub$ cd backend
lmb@lmb:~/micro-y-hub/backend$ ls
app db Dockerfile requirements.txt
lmb@lmb:~/micro-y-hub/backend$ cd
lmb@lmb:~$ pwd
/home/lmb
```

- cd cartella → entra in una cartella
- cd .. → sali di un livello
- cd → torna alla home
- cd - → torna alla cartella precedente (toggle)

Un bel casino le cartelle



- / radice (root) del filesystem
- /home/<utente> i vostri file (zona “sicura”)
- /tmp temporanei (possono sparire / pulizia automatica)
- /etc configurazioni di sistema
- /var/log log (diagnostica, errori)
- /bin, /usr/bin programmi/comandi

mkdir



significa **make directory**



Serve per creare una nuova cartella o piú

```
lmb@lmb ~$ ls
gestionale micro-y-hub
lmb@lmb:~$ mkdir -p
progetto/{input,output,log}
lmb@lmb:~$ cd progetto/
lmb@lmb:~/progetto$ ls
input log output
```

■ `mkdir nome_cartella` → crea una cartella

■ `mkdir -p nome_cartella/a nome_cartella/b nome_cartella/c`

→ crea anche le cartelle “intermedie” se non esistono

creare un file

touch



Serve per creare una nuova file

echo



Serve per scrivere

```
lmb@lmb:~/progetto/input$ touch readme.md
lmb@lmb:~/progetto/input$ ls
readme.md
lmb@lmb:~/progetto/input$ echo "ciao" > readme.md
lmb@lmb:~/progetto/input$ echo "a tutti" >> readme.md
```

- touch file.txt → crea un file chiamato file.txt
- echo "ciao" > file.txt → sovrascrivi ciao all'interno
- echo "ciao" >> file.txt → aggiungi ciao all'interno (in coda)

leggere un file

cat, less, head, tail



Servono per visualizzare il contenuto di un file



```
lmb@lmb:~/progetto/input$ cat readme.md
ciao
a tutti
```

■ cat file.txt → stampa tutto (ok per file piccoli)

■ less file.txt → scorri (esci con q)

✓ preferito

■ head -n 20 file.txt → prime 20 righe

■ tail -n 20 file.txt → ultime 20 righe (utile per log/output)

modificare un file

nano



Serve per aprire e modificare un file (consigliato per modifiche minime)



■ nano file.txt → apre o crea il file se non esiste

Salva: Ctrl + O (Write Out) → Invio

Esci: Ctrl + X

Cerca: Ctrl + W

copiare un file

cp



Significa copy e serve per copiare file o cartelle

```
lmb@lmb:~/progetto/input$ cp readme.md readme_old.md
lmb@lmb:~/progetto/input$ ls
readme.md  readme_old.md
lmb@lmb:~/progetto/input$ cat readme_old.md
ciao
a tutti
lmb@lmb:~/progetto/input$
```

■ cp sorgente destinazione

○ ■ cp -r cartella1 cartella2

spostare o rinominare un file

mv



Significa move e serve per spostare file o cartelle o rinominare

```
lmb@lmb:~/progetto/input$ ls
readme.md  readme_old.md
lmb@lmb:~/progetto/input$ mv readme_old.md readme_route.md
lmb@lmb:~/progetto/input$ ls
readme.md  readme_route.md
lmb@lmb:~/progetto/input$ mv readme_route.md ../
lmb@lmb:~/progetto/input$ ls
readme.md
lmb@lmb:~/progetto/input$ cd ..
lmb@lmb:~/progetto$ ls
input log output readme_route.md
```

■ mv vecchio_nome nuovo_nome

○ ■ mv file.txt cartella/

cancellare un file

rm



Significa remove e serve per eliminare file o cartelle

```
lmb@lmb:~/progetto/trash$ ls
a.txt b.txt c.txt
lmb@lmb:~/progetto/trash$ rm a.txt
lmb@lmb:~/progetto/trash$ ls
b.txt c.txt
lmb@lmb:~/progetto/trash$ cd ..
lmb@lmb:~/progetto$ rm trash/b.txt
lmb@lmb:~/progetto$ cd trash
lmb@lmb:~/progetto/trash$ ls
c.txt
lmb@lmb:~/progetto/trash$ cd ..
lmb@lmb:~/progetto$ ls
input log output readme_route.md trash
lmb@lmb:~/progetto$ rm -r trash/
lmb@lmb:~/progetto$ ls
input log output readme_route.md
```

■ rm file.txt

○ rm -r cartella/

nel terminale non
c'è un cestino

Cercare file e testo

Cerca file o cartella, restituisce
la lista di file trovati



find—

■ `find . -name "readme.txt"`

○ `find . -name "*.txt"`

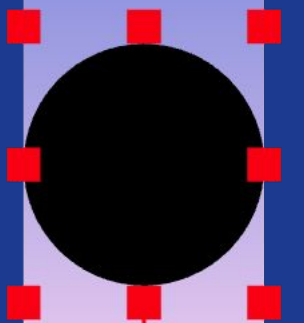
grep—



■ `grep "ciao" readme.txt`

○ `grep -R "errore" .`

Cerca testo dentro i file, restituisce la
linea del file con la parola cercata



Comandi utili

man comando

restituisce il manuale del comando inserito
dopo spesso a più pagine

history

restituisce la lista completa dei comandi
precedentemente eseguiti

clear

pulisce il terminale da tutte le cose precedenti

Ctrl + c

permette di interrompere comandi o
programmi bloccati

I permessi

versione semplice

di chi é?

ogni file ha un owner, group,
others



i permessi principali

r

leggere

w

scrivere

x

eseguire

drwxr-xr-x → cartella: owner tutto, altri possono entrare e leggere

Cambiare i permessi

versione semplice



script.sh

hai un file e vuoi eseguirlo

```
chmod +x script.sh
```

ora puoi lanciarlo

modifiche ai permessi comuni

```
chmod 644 file.txt
```

owner rw, altri r

```
chmod 755 file.txt
```

owner rwx, altri rx

Grazie per l'attenzione