MINI-GUIDA PER LINUX

Introduzione

Linux è un sistema operativo open-source nato negli anni ’90 e oggi utilizzato ovunque: dai supercomputer agli smartphone, dai server Internet ai dispositivi embedded. Raspberry Pi è un piccolo computer progettato per l’educazione e la prototipazione che utilizza Linux come sistema operativo principale.

Questa guida ha lo scopo di accompagnarti passo dopo passo alla comprensione di Linux su Raspberry Pi, partendo dai concetti di base fino ad arrivare all’uso pratico quotidiano. L’obiettivo non è solo “sapere cosa scrivere”, ma capire *perché* le cose funzionano in un certo modo.

Cos’è Linux e come funziona

Linux è un sistema operativo multitasking e multi-utente. Questo significa che può:

- **eseguire più programmi contemporaneamente**;
- **gestire più utenti con permessi diversi**;
- **controllare in modo efficiente le risorse del computer** (CPU, memoria, dischi, rete).

A differenza di un microcontrollore come Arduino, che esegue un solo programma in loop senza sistema operativo, Linux introduce un livello di astrazione che permette grande flessibilità, ma anche maggiore complessità.

Raspberry Pi

In questo contesto, Raspberry Pi è a tutti gli effetti un computer, contiene:

- CPU e RAM
- memoria di massa (microSD)
- porte USB, HDMI, Ethernet, Wi-Fi
- pin GPIO per interagire con l’hardware

Quando accendi un Raspberry Pi:

1. viene caricato il bootloader;
2. viene avviato il kernel Linux;
3. il sistema operativo prende il controllo dell’hardware;
4. vengono avviati i servizi e l’ambiente grafico o testuale.

Il terminale

Il terminale è l'interfaccia testuale che permette di comunicare direttamente con il sistema operativo. Anche se esiste un'interfaccia grafica, il terminale rimane lo strumento più potente e preciso. Attraverso il terminale è possibile esplorare il file system, avviare programmi, configurare il sistema e diagnosticare problemi in modo diretto.

Ogni comando segue una struttura semplice:

```
comando opzioni argomenti
```

Dove:

- **comando** indica il programma da eseguire;
- **opzioni** modificano il comportamento del comando;
- **argomenti** specificano su quali file o risorse operare.

Esempio:

```
ls -l /home
```

Comandi di base più utilizzati

Navigazione e file system

- **`pwd`** → (print working directory) mostra la cartella corrente
- **`ls`** → (list) elenca file e cartelle
- **`cd nome_cartella`** → (change directory) entra in una cartella
- **`cd ..`** → torna alla cartella superiore

Gestione file

- **`touch file.txt`** → crea un file vuoto
- **`nano`** → crea un file e consente di modificarlo
- **`cp sorgente destinazione`** → (copy) copia file o cartelle
- **`mv sorgente destinazione`** → (move) sposta o rinomina
- **`rm file.txt`** → (remove) elimina un file
- **`rm -r cartella`** → elimina una cartella e il suo contenuto, si può usare anche **`rmdir`**

Visualizzazione contenuti

- **`cat file.txt`** → mostra il contenuto di un file
- **`less file.txt`** → visualizza un file pagina per pagina

Sistema e processi

- **`top`** → monitor in tempo reale dei processi (se installato si può usare **`htop`**)
- **`ps aux`** → elenco dei processi attivi
- **`kill PID`** → termina un processo
- **`lscpu`** → mostra informazioni dettagliate sulla CPU (architettura, core, thread)

Dischi e dispositivi di memorizzazione

- **`lsblk`** → elenca dischi, partizioni e dispositivi collegati
- **`df -h`** → spazio disco utilizzato e disponibile

Rete

- **`ip a`** → mostra le interfacce di rete e gli IP
- **`ping indirizzo`** → verifica la connessione
- **`ssh utente@ip`** → connessione remota via terminale

Rete e Controllo Remoto

Uno dei punti di forza di Linux e Raspberry Pi è la gestione della rete. Il sistema può: collegarsi a Internet, comunicare con altri dispositivi ed essere controllato da remoto tramite SSH. SSH permette di usare il Raspberry Pi anche senza schermo o tastiera, trasformandolo in un vero computer remoto.

Come funziona SSH (Secure Shell)

SSH è un protocollo che permette di **aprire un terminale remoto cifrato** su un altro computer. Dal punto di vista pratico, è come se stessi usando il terminale del Raspberry Pi, ma digitando i comandi dal tuo PC.

Il modello è **client–server**:

- il **server SSH** gira sul Raspberry Pi;
- il **client SSH** è il computer da cui ti colleghi.

Tutto il traffico è cifrato: password e comandi NON viaggiano in chiaro sulla rete.

Verificare che SSH sia attivo sul Raspberry Pi

Sul Raspberry Pi:

```
sudo systemctl status ssh
```

Se non è attivo:

```
sudo systemctl enable ssh
sudo systemctl start ssh
```

Collegarsi via SSH

Dal computer client:

```
ssh utente@indirizzo_ip
```

Esempio:

```
ssh test1@192.168.1.103
```

Alla prima connessione verrà chiesto di confermare l'identità del dispositivo e poi la password dell'utente.

Specificare una porta diversa

Se il server usa una porta non standard:

```
ssh -p 2222 test1@192.168.1.103
```

Copiare file con SSH (scp)

SSH permette anche di copiare file in modo sicuro.

Copiare un file **dal Raspberry al PC**:

```
scp pi@192.168.1.42:/home/pi/file.txt .
```

Copiare un file **dal PC al Raspberry**:

```
scp file.txt pi@192.168.1.42:/home/pi/
```

Uscire da una sessione SSH

Per chiudere la connessione:

```
exit
```

SSH è uno strumento fondamentale: una volta imparato, rende il Raspberry Pi utilizzabile come server, computer remoto o nodo di rete senza essere fisicamente collegati ad esso con un monitor e tastiera.

Installare programmi

l gestore di pacchetti (apt)

In Linux i programmi non si scaricano normalmente da siti web come su Windows. Il sistema utilizza un **gestore di pacchetti**, che si occupa di:

- scaricare i programmi **da repository ufficiali**;
- installarli nel punto corretto del file system;
- gestire automaticamente le dipendenze;
- mantenere il software aggiornato.

Su Raspberry Pi OS (basato su Debian) il gestore di pacchetti principale è **apt**. È importante perché rende Linux affidabile e stabile: il software proviene da fonti controllate e il sistema evita conflitti tra librerie. È uno dei motivi per cui Linux è molto usato in ambito server e embedded.

Aggiornare la lista dei programmi

Prima di installare qualsiasi cosa è buona pratica aggiornare l'elenco dei pacchetti disponibili (questo comando **non installa nulla**, ma sincronizza il sistema con i repository):

```
sudo apt update
```

Aggiornare i programmi installati

Per aggiornare tutti i programmi già presenti:

```
sudo apt upgrade
```

Installare un programma

Per installare un software:

```
sudo apt install nome_programma
```

Esempi comuni:

```
sudo apt install python3
sudo apt install git
sudo apt install nano
```

Rimuovere un programma

sudo apt remove nome_programma

Per rimuovere anche i file di configurazione:

sudo apt purge nome_programma

Cercare un programma

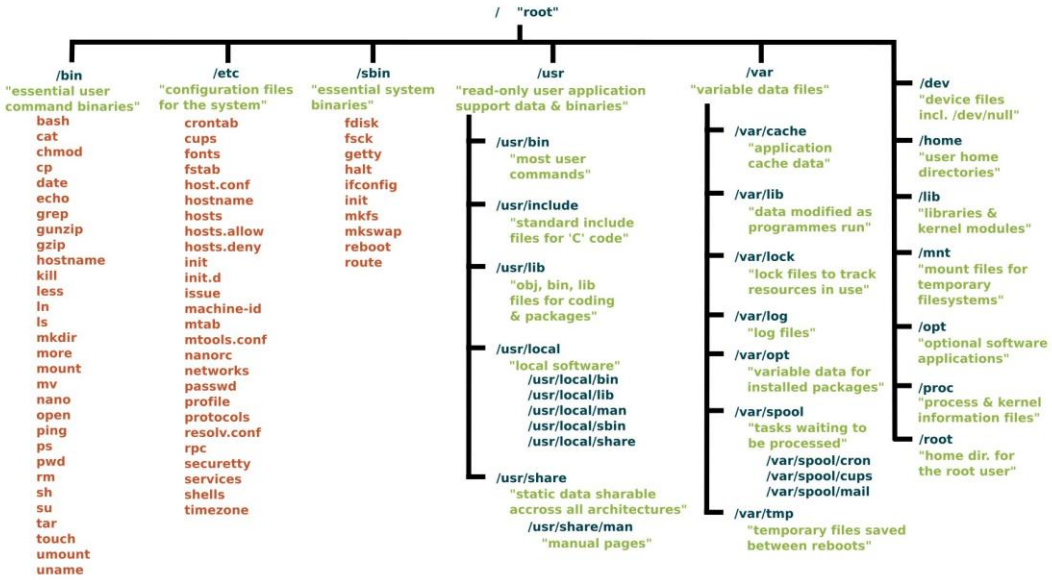
apt search parola_chiave

File system: come Linux organizza i dati

In Linux (generalmente) tutto è organizzato in una struttura ad albero che parte dalla radice `/`. Potete trovare ulteriori informazioni cercando "Filesystem Hierarchy Standard" Cartelle importanti:

- **`/home`** → dati degli utenti
- **`/etc`** → configurazioni

- **`/bin`** e **`/usr/bin`** → programmi
- **`/var`** → file variabili (log, cache)



Permessi e utenti

Ogni file in Linux ha associati dei permessi che stabiliscono chi può:

- leggere
- scrivere
- eseguire

Questi permessi sono fondamentali per la sicurezza del sistema. Il comando sudo permette a un utente normale di eseguire temporaneamente comandi come amministratore.

Il comando **`chmod`** permette di modificare i permessi di un file o di una cartella. Esempio tipico:

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

Rende il file eseguibile.

Permessi numerici (forma compatta):

- **4** → lettura (r)
- **2** → scrittura (w)
- **1** → esecuzione (x)

Esempio:

```
chmod 755 script.sh
```

Significa:

- proprietario: lettura, scrittura, esecuzione (4+2+1)
- gruppo: lettura, esecuzione (4+1)

- altri: lettura, esecuzione (4+1)

Programmi e processi

Un programma è un file eseguibile. Quando viene avviato, diventa un processo. Linux gestisce centinaia di processi contemporaneamente. È possibile:

- visualizzarli
- monitorarne l'uso di risorse
- terminarli se necessario

Rete e controllo remoto

Uno dei punti di forza di Linux e Raspberry Pi è la gestione della rete. Il sistema può:

- collegarsi a Internet
- comunicare con altri dispositivi
- essere controllato da remoto tramite SSH

SSH permette di usare il Raspberry Pi anche senza schermo o tastiera, trasformandolo in un vero computer / server remoto.

Raspberry Pi e GPIO

A differenza di un normale PC, Raspberry Pi offre pin GPIO per interagire con sensori e attuatori. Tuttavia, essendo Linux un sistema non realtime, **il controllo dei pin non è deterministico** come su Arduino.

Per questo motivo:

- **Raspberry Pi è ideale per logica, rete, interfacce**
- **Arduino è ideale per controllo temporale preciso**

Spesso i due vengono usati insieme.

Script e automazione

Uno script è un file che contiene una sequenza di comandi. In Linux gli script sono fondamentali per automatizzare operazioni ripetitive. Tipologie comuni:

- script **Bash** (.sh)
- script **Python** (.py)

Rendere uno script eseguibile significa trattarlo come un vero programma.

sudo

`sudo` (superuser do) è un comando che permette a un utente normale di eseguire **temporaneamente** un comando con i privilegi di amministratore (detto *root*).

In Linux, per motivi di sicurezza, un utente normale **non può**:

- installare o rimuovere programmi di sistema;
- modificare file critici (come quelli in /etc);
- cambiare impostazioni globali del sistema.

Quando scrivi:

```
sudo comando
```

stai dicendo al sistema: "Esegui *questo* comando come amministratore, assumendomene la responsabilità".

Perché sudo è importante

- evita di lavorare sempre come amministratore;
- riduce i danni in caso di errore;
- tiene traccia di chi ha eseguito cosa.

Esempio tipico:

```
sudo apt install git
```

Serve sudo perché stai modificando il sistema, non solo i tuoi file. **Attenzione**: usare sudo a caso può danneggiare il sistema. Usalo solo quando richiesto e solo se capisci cosa stai facendo.