Distribuzioni Linux principali

0. Linux

0.1 Origini di Linux

- Le primissime "Distribuzioni" di Linux, intese come un kernel Linux accoppiato a strumenti GNU, risalgono a fine 1991, poco dopo il rilascio del kernel (versione 0.01) da parte di Linus Torvalds nel settembre 1991. Prima versione stabile del kernel (versione 0.02) ottobre 1991.
    - **MCC Interim Linux:** Sviluppata al Manchester Computing Centre, fu resa disponibile tra il novembre 1991 e il febbraio 1992. Era molto basilare.
    - Softlanding Linux System (**SLS**): Spesso citata come la prima vera distribuzione completa. Fu creata da Peter MacDonald nell'agosto 1992.
    - Entrambe non sono più manutenute.

0.1.1 Slackware (da **luglio 1993**, esiste ancora e funziona)

- **io l'ho usata dal 1994, installazione con 10 floppy disk su un PC con CPU i386.**
- **Init**: tradizionalmente *BSD-style / SysV-like scripts* (`/etc/rc.d/rc.S`, `/etc/rc.d/rc.<runlevel>`).
- **Bootloader**: spesso LILO in passato, oggi GRUB è comune.
- **Caratteristiche**: filosofia minimalista, poche automazioni invasive, gli script `/etc/rc.d/` eseguono passo-passo; amministrazione manuale.
- **Package manager**: installpkgc, primitivo, nessuna risoluzione automatica delle dipendenze, i pacchetti sono semplici archivi compressi che contengono i file di sistema e uno script di installazione.
-
    - Packet Radio: Bug nel kernel, si perdeva interrupt. Nel 1995 debuggato e corretto per poter usare un modulo di comunicazione a base IP su una rete radio-amatoriale. Si usava il protocollo AX.25 (Amateur X.25) è un protocollo del livello di collegamento dati (Layer 2) derivato dal protocollo X.25 e adattato per l'uso da parte dei radioamatori.con. Fornisce le funzionalità necessarie per stabilire connessioni e trasferire dati in modo affidabile attraverso il canale radioamatoriale (il "**Packet Radio**").
    - Incapsulamento IP: Per far transitare i pacchetti IP (Internet Protocol, Layer 3) sulla rete radio, si utilizza l'incapsulamento IP su AX.25. Questo permetteva al sistema di diventare parte di una rete più ampia, nota come AMPRNet o 44Net (chiamata così per l'allocazione dello spazio di indirizzamento IP classe A, 44.0.0.0/8, riservato ai radioamatori). Questo veniva configurato utilizzando strumenti specifici come kissattach per **creare un'interfaccia di rete virtuale (es. ax0)** su cui poi veniva configurato un indirizzo IP, proprio come si farebbe con una scheda Ethernet (ifconfig).
    - Il protocollo AX.25 era implementato in parte nel kernel Linux. Ma aveva problemi, si perdeva qualche interrupt. Nel 1995 ho dovuto debuggarlo per correggerlo e far funzionare le comunicazioni via rete radioamatoriale tra Cesena e Castel San Pietro Terme.
    - Nota di orgoglio: Partendo da quelle prime esperienze di radio-comunicazione, io e tre miei amici, Giovanni Pau, Pierluigi Mangani e Stefano Daddona (eravamo studenti) abbiamo progettato il sistema di ponti radio per connettere le sedi della Romagna dell'università di Bologna con la sede centrale di Bologna usando frequenze militari sui 2 GHz e sui 7 GHz e i tralicci dei ripetitori RAI di Monte Maggio (Bertinoro) e di Monte Calderaro. E' stata la mia tesi di laurea http://www.cs.unibo.it/~ghini/didattica/reti_lpr/lpr/pr_radio.pdf
    - E' entrato in funzione nel 1999-2000 ed è stato usato per 10 annetti facendo risparmiare un sacco di soldini a Unibo.

0.2 Debian e suoi derivati

- **Debian**: storicamente SysVInit, dal 2015 (Debian 8 "Jessie") ha adottato **systemd** come init predefinito (puoi comunque usare altri init).
- **Derivati di Debian** (esempi): *Raspbian (Raspberry Pi OS), Devuan (fork senza systemd), MX Linux (derivato anti-social? basato su Debian),* ***Linux Mint Debian Edition*** *(LMDE)*.
    - Nota: *Devuan* è esplicitamente creato per evitare systemd (usa sysvinit o openrc).
- **Bootloader**: GRUB2.
- **Particolarità**: Debian ha policy conservative, ampia compatibilità, **MOLTO stabile**.

0.3 Ubuntu e suoi derivati

- **Ubuntu**: ha adottato **systemd** come init di default dal 15.04 in poi.
- **Derivati**: *Kubuntu, Xubuntu, Lubuntu, Ubuntu Server, Pop!_OS (basato su Ubuntu), Linux Mint (Ubuntu-based edition)*.
- **Caratteristiche**: Ubuntu estende tool e integrazioni (cloud-init su cloud images, snapd, upstart in passato). Ubuntu Server immagini cloud sono fortemente integrate con systemd/cloud-init.
- **Il gestore di pacchetti snapd è comodo ma spreca una marea di spazio disco perché duplica le dipendenze**.

0.4 Red Hat Enterprise Linux (RHEL) e derivati

- **RHEL**: usa **systemd** da RHEL 7 (prima SysV-style).
- Originata da distribuzione Mandrake, meravigliosa.
- **Derivati**: *CentOS (storico), AlmaLinux, Rocky Linux* (questi due sono RHEL-compatible dopo la fine di CentOS Linux 8). *Fedora* è upstream di RHEL (anch'esso usa systemd).
- **Bootloader**: GRUB2.
- **Caratteristiche**: focus aziendale, SELinux, politiche di stabilità; systemd integrato con strumenti come `systemctl` e `journalctl`. **Buono ma commerciale**.

0.5 Rocky Linux

- **Rocky**: distribuzione compatibile RHEL, usa **systemd**. È nata come fork per sostituire CentOS Linux (derivata da Red Hat) quando anche questa è diventata commerciale.

0.6 Alpine Linux

- **Init**: **OpenRC** (leggero) su base BusyBox in ambienti strettamente minimali; Alpine è costruita per essere leggera e spesso usata in container.
- **Bootloader**: può usare GRUB ma in molti usi embedded/container non serve.
- **Caratteristiche**: musl libc, apk package manager, OpenRC semplice e leggero.
- **Ottima per costruire container che debbano essere semplici e leggeri, qualche difficoltà per gestire servizi complessi.**

0.7 Arch Linux

- **Arch Linux**: ha adottato **systemd** come init di default.
    - Arch Linux è costruita per essere Minimalista, pulita, configurazione manuale.
    - L'utente deve costruire il sistema da zero (scegliendo ogni componente).
    - Rivolta a utenti esperti che vogliono comprendere ogni parte del sistema,
- **Bootloader**: può usare GRUB.
- Arch è di tipo **Rolling Release (Rilascio continuo)**, con i pacchetti sempre aggiornati all'ultima versione. Non è simile a Debian/Ubuntu (Fixed Release): Queste sono "a rilascio fisso", significano che le versioni dei pacchetti sono vecchie ma stabili.
- Package manager: Pacman.

0.8 Gentoo

- **Init:** consigliato **OpenRC** ma può usare systemd.
- **Source-Based** (Basata sul Codice Sorgente): **Ricompila ogni pacchetto che installi, kernel e moduli del kernel compresi.** Per fuori di testa (io l'ho usato per anni). E' la caratteristica distintiva. La maggior parte del software viene compilata localmente sul sistema dell'utente, anziché scaricare pacchetti binari precompilati (come fanno Debian, Ubuntu, o Red Hat). L'utente può ottimizzare il software specificamente per la propria architettura CPU (es. $x86\_64$, ARM, ecc.), ottenendo **potenzialmente** migliori prestazioni.
- Rivolta a utenti esperti che vogliono ottimizzare ogni parte del sistema.
- **Bootloader**: può usare GRUB ma si può scegliere altro, LILO (Linux Loader) o systemd-boot (compatibile solo con UEFI).
- Package manager: Portage, con comando "emerge".

Come avviene il boot di un sistema Linux e il ruolo di init

1.1 Panoramica ad alto livello (fasi del boot)

1. **Firmware** (BIOS/UEFI) — inizializza hardware, trova bootloader su disco.
2. **Bootloader** (es. GRUB, syslinux) — carica kernel e initramfs (ramdisk iniziale).
3. **Kernel** — initramfs init (mount root temporaneo), rileva dispositivi, monta root definitivo.
4. **PID 1 (init system)** — il kernel avvia il processo con PID 1 (tradizionalmente `/sbin/init`); è il primo processo utente e coordina il resto del boot (lancia servizi, gestisce runlevel/targets, gestisce shutdow/reboot).
5. **Servizi e demoni** — l'init system avvia servizi (networking, filesystem, demoni).
6. **Login / sessione utente** — getty, display manager, ecc.

1.2 initramfs - Panoramica

Cos'è l'initramfs? initramfs = initial RAM filesystem

È un **piccolo filesystem compresso** (di solito in formato `cpio` + `gzip`/`xz`) che viene:

1. **caricato in RAM** direttamente dal bootloader (GRUB, syslinux, ecc.)
2. **montato da Linux all'avvio**
3. usato come filesystem "temporaneo" per avere gli strumenti per preparare l'avvio del vero sistema

In pratica è una **mini–Linux** che vive nella RAM, usata prima del vero /root.

A cosa serve l'initramfs?

Serve a permettere al kernel di **Caricare driver necessari per montare il filesystem root**

Infatti, il kernel all'inizio non ha:

- alcuni eseguibili
- driver del disco
- driver RAID
- driver LVM
- driver filesystem (es. ext4, btrfs)
- driver NVMe
- driver per chipset storage
- moduli per filesystem di rete (NFS, iSCSI)

L'initramfs contiene questi moduli, li si carica in memoria e li si usa.

Grazie a questi moduli diventa possibile montare il filesystem vero che potrebbe risiedere su

- su un disco locale
- su sistemi RAID
- su storage multipli tipo Btrfs
- su dispositivi criptati (LUKS)
- su storage di rete (iSCSI, NFS, CIFS)

Ad esempio, se il disco è criptato,
- initramfs chiede la **passphrase**
- apre il volume criptato
- monta il root

Senza initramfs, i sistemi criptati **non potrebbero avviarsi**.

Se qualcosa va storto nel boot, initramfs fornisce una **shell di emergenza** che ha un prompt così:
`(initramfs) _`
che è una busybox minimale per fare debug.

Come funziona il flusso di lavoro di initramfs?

Flow completo del boot:

```
[BIOS/UEFI]
      ↓
[Bootloader: GRUB]
      ↓ carica kernel + initramfs
[Kernel]
      ↓
Montaggio initramfs in RAM
      ↓
Esecuzione di /init (script principale)
      ↓
Caricamento moduli, attivazione LVM/RAID, decrypt, ecc.
      ↓
Montaggio /root del filesystem reale
      ↓
Avvio del vero sistema (es. /sbin/init → systemd)
```

1.3 Ruolo del processo init (PID 1)

- È il **genitore di tutti i processi** creati durante e dopo il boot.
- Recupera processi orfani (`SIGCHLD`) e previene zombie.
- È responsabile di: inizializzare i servizi, gestire runlevels/targets, eseguire lo shutdown/reboot in modo ordinato, gestire il logging di sistema (tramite journald/syslog se previsto).
- Se PID 1 muore o si blocca, il sistema è in grave difficoltà (il kernel può invocare panic).

Diverse distribuzioni usano init diversi (systemd, SysVInit, OpenRC, runit, s6). Il comportamento del boot e la gestione dei servizi possono cambiare in base all'init system.