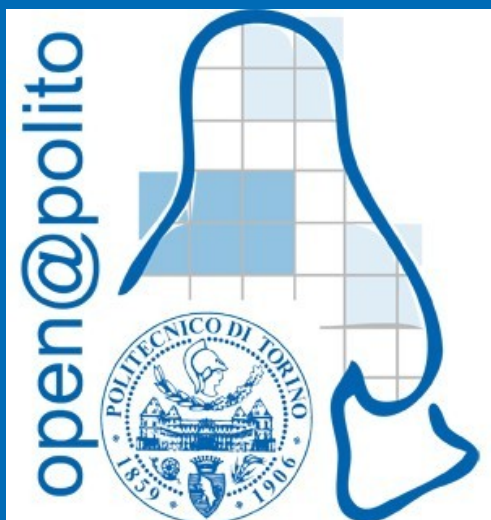




Con il supporto di:



Gestione dei processi con Systemmd

9 dicembre 2025
Andrea Gabbani

Cosa e' Systemd?

systemd
system daemon

Cosa e' Systemd?

E' un **init system** e **system manager** open-source
per sistemi GNU/Linux:

- Avvio parallelo dei servizi.
- Gestione delle dipendenze.
- Registrazione centralizzata con journald

Sostituisce i vecchi sistemi di init (come SysVinit) con un design moderno ed efficiente.

Storia

Rilasciato per la prima volta nel 2010
dallo sviluppatore tedesco Lennart Poettering



init?

Nei sistemi UNIX o UNIX-like (tra cui Linux),
L' **init** e' quel particolare programma che viene lanciato per primo
all'avvio del sistema operativo

Il suo Process ID e' e sarà sempre 1 (**PID 1**), ed e' per definizione
l'antenato, diretto o indiretto, di tutti i processi del sistema

E' quindi responsabile dell'**avvio** e della **gestione** di tutti gli altri processi

Cosa fa Systemd?

systemd Utilities

systemctl journalctl notify analyze cgl's cgtop loginctl nspawn

systemd Daemons

systemd
journald networkd
logind user session

systemd Targets

bootmode basic multi-user graphical user-session
shutdown reboot dbus telephony display service
dlog logind user-session tizen service

systemd Core

manager unit login namespace log
systemd service timer mount target multiseat inhibit
snapshot path socket swap session pam cgroup dbus

systemd Libraries

dbus-1 libpam libcap libcryptsetup tcpwrapper libaudit libnotify

Linux Kernel

cgroups autofs kdbus

Perché systemd

E' nato con l'obiettivo di **uniformare** la gestione dei sistemi Linux...
...e ci e' riuscito!

A partire dal 2010, e' progressivamente diventato l'**init system**
piu' adottato dalle distribuzioni Linux

Red Hat Enterprise Linux (RHEL) ha iniziato ad adottare
systemd dalla versione 7.0

Le controversie

Systemd va contro la **Filosofia UNIX** (Thompson e Ritchie), dove i programmi devono:

- Fare **una** cosa e farla **bene** (evitare il “*feature creep*”)
- Essere visti e progettati come componenti **modulari** fatti per **lavorare assieme**
- Lavorare con i **flussi di caratteri**, in quanto sono un'interfaccia **universale**

Secondo molti Systemd fa *troppo*, e in maniera troppo *monolitica*

Le controverse



Unit

Unit: **Systemd** organizza i **task** in **unit**. Tipi di unit comuni:

- **Service** (.service): Gestisce i servizi.
- **Timer** (.timer): Pianifica le attività.
- **Target** (.target): Raggruppa unità (es. multi-user.target per la modalità multiutente).
- **Socket** (.socket): Per l'attivazione basata su socket.

Strumenti a riga di comando:

- **systemctl**: Gestisce e interroga lo stato delle **unit**.
- **journalctl**: Accede ai **log**.

Ciclo di vita di una Unit

Una unit puo' esistere in vari stati:

- **Active:** Il servizio è in esecuzione.
- **Inactive:** Il servizio è fermo.
- **Failed:** Il servizio ha incontrato un errore.

Dipendenze delle unità: Usa **Before**, **After**, **Requires** e **Wants** per gestire le dipendenze.

Service Unit

Un file **.service** descrive un servizio gestito da systemd.

Struttura di un file di servizio:

- **[Unit]**: Metadati e dipendenze.
- **[Service]**: Comportamento del servizio (comandi, policy di riavvio).
- **[Install]**: Opzioni di abilitazione.

Facciamo pratica

```
systemctl --version
```

```
systemctl list-units
```

Facciamo pratica

Crea un file per lo script `hello.sh`

```
#!/bin/bash
```

```
echo "Hello da WEEE Open & Netstudent!"
```

Assegna i permessi di esecuzione:

```
chmod +x hello.sh
```

Facciamo pratica

Crea la service unit `hello.service`

```
[Unit]
```

```
Description=Hello World Service
```

```
[Service]
```

```
ExecStart=/home/[user]/hello.sh
```

```
Restart=no
```

```
[Install]
```

```
WantedBy=multi-user.target
```

Facciamo pratica

```
sudo cp hello.service /etc/systemd/system/  
sudo systemctl daemon-reload  
sudo systemctl start hello.service  
sudo systemctl enable hello.service
```


Comandi utili

```
systemctl status hello.service
```

```
systemctl stop hello.service
```

```
systemctl reset-failed hello.service
```

```
journalctl -u hello.service
```

Giochiamo con i timer

Un file **.timer** pianifica le attività in base al tempo o agli intervalli.

Casi d'uso comuni:

- Automazione dei backup.
- Pianificazione dell'esecuzione di script.
- Necessita' di maggiore controllo rispetto a **crond**

Giochiamo con i timer

Sezioni fondamentali per la **.timer** unit:

- **[Unit]**: Metadati.
- **[Timer]**: Dettagli di pianificazione (OnCalendar=, OnBootSec=, OnUnitActiveSec=).

Giochiamo con i timer

Crea la timer unit `hello.timer`:

```
[Unit]
```

```
Description=Esegui Hello Service ogni minuto
```

```
[Timer]
```

```
OnBootSec=10s
```

```
OnUnitActiveSec=1min
```

```
[Install]
```

```
WantedBy=timers.target
```

Giochiamo con i timer

```
sudo cp hello.timer /etc/systemd/system/
```

```
sudo systemctl start hello.timer
```

```
sudo systemctl enable hello.timer
```

Comandi utili

systemd-analyze

systemd-analyze blame

Opzioni avanzate

[Timer]

OnCalendar=daily

OnUnitActiveSec=5m

OnStartupSec=60m

RandomizedDelaySec=60m

Persistent=true

OnCalendar=* - * - * 06..18:00/30