

I Comandi Fondamentali di Linux

Una guida pratica basata sull'approccio 80/20

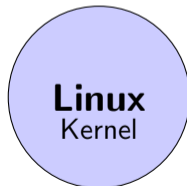
Basato su "The Linux Commands Handbook" di Flavio Copes

IIS Fermi Sacconi Ceci

24 dicembre 2025

Cos'è Linux?

- **Sistema Operativo** open source e libero
- Nato nel 1991 in Finlandia da Linus Torvalds
- Kernel del sistema GNU/Linux
- Alimenta la maggior parte dei server di Internet
- Base di Android
- Disponibile in diverse **distribuzioni**: Debian, Ubuntu, Red Hat, Fedora, ecc.



Libertà

- Libertà di fare qualsiasi cosa con il tuo computer
- Nessuna azienda può dettare cosa puoi o non puoi fare
- Codice sorgente completamente accessibile

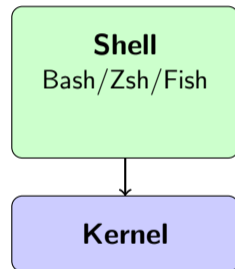
Regola 80/20

Questo corso segue la regola 80/20:

- Impara il 20% dei comandi
- Che userai l'80% del tempo
- Focus sui comandi essenziali

Cos'è una Shell?

- **Interprete di comandi** testuale
- Interfaccia tra utente e sistema operativo
- Permette di eseguire operazioni tramite comandi
- Consente di creare **script** automatizzati
- Più potente e veloce di una GUI



Principali Shell UNIX

- **sh** (Bourne Shell) - La shell originale di Steve Bourne
- **Bash** (Bourne Again Shell) - Shell predefinita su molti sistemi Linux
- **Zsh** (Z Shell) - Shell predefinita su macOS (da Catalina)
- **Fish** (Friendly Interactive Shell) - Shell moderna e user-friendly
- **Csh/Tcsh** - C Shell e TC Shell

```
$ echo $SHELL
```

Il Comando man

Manual Pages

Il primo comando da conoscere per imparare tutti gli altri!

```
$ man <comando>
```

Esempio:

```
$ man ls
```

Alternativa Moderna: tldr

tldr (Too Long; Didn't Read) fornisce esempi pratici e concisi:

```
$ tldr ls
```

Comando ls - Listare File

Lista i file in una directory

```
$ ls                # lista file directory corrente
$ ls /bin           # lista file in /bin
$ ls -l            # formato lungo (dettagliato)
$ ls -a            # mostra anche file nascosti
$ ls -al           # combinazione opzioni
$ ls -lh           # dimensioni human-readable
```

Output ls -l

```
-rw-r--r-- 1 user group 1234 Dec 7 10:30 file.txt
```

- Permessi, link, proprietario, gruppo, dimensione, data, nome

Comando cd - Cambiare Directory

Change Directory - Naviga tra le cartelle

```
$ cd cartella          # entra nella cartella
$ cd /etc              # percorso assoluto
$ cd ..               # torna alla directory padre
$ cd ~                # vai alla home directory
$ cd -                # torna alla directory precedente
```

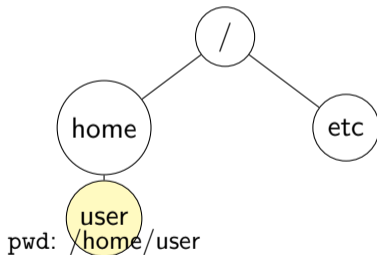
Percorsi Speciali

- . = directory corrente
- .. = directory padre
- ~ = home directory dell'utente
- / = directory radice (root)

Comando pwd - Posizione Corrente

Print Working Directory - Dove mi trovo?

```
$ pwd  
/home/username/documents
```



Comando mkdir - Creare Directory

Make Directory - Crea nuove cartelle

```
$ mkdir cartella          # crea una cartella
$ mkdir doc img src       # crea pi  cartelle
$ mkdir -p path/to/dir    # crea percorso completo
```

Esempio con -p

```
$ mkdir -p progetti/web/frontend
# Crea tutte le directory intermedie se non esistono
```

Comando rmdir - Eliminare Directory

Remove Directory - Elimina cartelle vuote

```
$ rmdir cartella          # elimina cartella vuota  
$ rmdir dir1 dir2         # elimina pi cartelle
```

Attenzione!

rmdir funziona solo con directory vuote. Per eliminare directory con contenuto:

```
$ rm -rf cartella        # -r ricorsivo, -f forza
```

Usare con cautela! Nessuna conferma richiesta.

Comando cp - Copiare File

Copy - Copia file e directory

```
$ cp file1 file2           # copia file1 in file2
$ cp file /path/to/dest    # copia in altra directory
$ cp -r dir1 dir2          # copia directory ricorsivamente
$ cp -i file dest          # chiede conferma prima di sovrascrivere
$ cp -v file dest          # verbose, mostra cosa fa
```

Opzioni Comuni

- -r o -R: ricorsivo (per directory)
- -i: interattivo (chiede conferma)
- -v: verbose (mostra dettagli)
- -p: preserva attributi (permessi, timestamp)

Comando mv - Spostare/Rinominare

Move - Sposta o rinomina file e directory

```
$ mv file1 file2           # rinomina file1 in file2
$ mv file /path/to/dest    # sposta file in altra directory
$ mv -i file dest          # chiede conferma
$ mv dir1 dir2             # rinomina/sposta directory
```

Esempi

```
$ mv documento.txt backup/documento.txt
$ mv vecchionome.txt nuovonome.txt
$ mv *.txt documenti/      # sposta tutti i file .txt
```

Touch - Crea file vuoti o aggiorna timestamp

```
$ touch file.txt           # crea file vuoto
$ touch file1 file2 file3  # crea pi file
$ touch existing.txt       # aggiorna data modifica
```

Usi Comuni

- Creare file vuoti rapidamente
- Aggiornare timestamp di file esistenti
- Creare file placeholder per test

Comando find - Cercare File

Find - Potente strumento di ricerca

```
$ find . -name "*.txt"          # cerca file .txt
$ find /home -name documento    # cerca nella directory
$ find . -type f                # solo file
$ find . -type d                # solo directory
$ find . -size +1M              # file > 1 MB
$ find . -mtime -7              # modificati ultimi 7 giorni
```

Esempio Complesso

```
$ find . -name "*.log" -type f -mtime +30 -delete
# Trova ed elimina file .log pi vecchi di 30 giorni
```

Link - Crea collegamenti tra file

```
$ ln file link # hard link
$ ln -s file link # symbolic link (soft link)
$ ln -s /path/to/file link # link simbolico
```

Hard Link

- Stesso inode del file
- Indistinguibile dall'originale
- Rimane anche se l'originale è eliminato

Symbolic Link

- Puntatore al file
- Come un "collegamento"
- Si rompe se l'originale è eliminato

Comando open - Aprire File

Open - Apre file con applicazione predefinita (macOS)

```
$ open file.txt           # apre con editor predefinito
$ open image.jpg          # apre con visualizzatore immagini
$ open -a "TextEdit" file # apre con app specifica
$ open .                  # apre directory corrente nel Finder
```

Linux

Su Linux, usare:

```
$ xdg-open file.txt
```

Comando gzip - Compressione

Gzip - Comprime file

```
$ gzip file.txt           # crea file.txt.gz
$ gzip -k file.txt        # mantiene file originale
$ gzip -v file.txt        # verbose
$ gzip -9 file.txt        # massima compressione
```

Decompressione:

```
$ gunzip file.txt.gz      # decomprime
$ gzip -d file.txt.gz     # equivalente
```

Comando tar - Archiviazione

Tape Archive - Crea e gestisce archivi

```
$ tar -cvf archive.tar dir/      # crea archivio
$ tar -xvf archive.tar           # estrae archivio
$ tar -czvf archive.tar.gz dir/  # comprimi con gzip
$ tar -xzvf archive.tar.gz       # estrai .tar.gz
$ tar -tf archive.tar            # lista contenuto
```

Opzioni Principali

- c: create (crea)
- x: extract (estrai)
- v: verbose (dettagliato)
- f: file (specifica nome file)
- z: gzip (compressione)

Alias - Crea abbreviazioni per comandi

```
$ alias ll='ls -alh'           # crea alias
$ alias gs='git status'
$ alias ..='cd ..'
$ alias                       # mostra tutti gli alias
$ unalias ll                  # rimuove alias
```

Alias Permanenti

Aggiungi nel file ~/.bashrc o ~/.zshrc:

```
alias update='sudo apt update && sudo apt upgrade'
alias cls='clear'
```

Comando cat - Visualizzare File

Concatenate - Visualizza e concatena file

```
$ cat file.txt           # visualizza contenuto
$ cat file1 file2        # concatena file
$ cat file1 file2 > nuovo # concatena in nuovo file
$ cat -n file.txt         # con numeri di riga
```

Creare File al Volo

```
$ cat > newfile.txt
Digita il testo...
(premi Ctrl+D per terminare)
```

Comando less - Visualizzatore Paginato

Less - Visualizza file grandi con navigazione

```
$ less file.txt           # apre visualizzatore
$ less -N file.txt        # con numeri di riga
```

Navigazione in less

- Spazio: pagina successiva
- b: pagina precedente
- /parola: cerca parola
- n: prossima occorrenza
- q: esci
- G: vai alla fine
- g: vai all'inizio

Comando tail - Fine File

Tail - Mostra le ultime righe di un file

```
$ tail file.txt           # ultime 10 righe
$ tail -n 20 file.txt     # ultime 20 righe
$ tail -f logfile.log     # segue il file in tempo reale
$ tail -f -n 50 log.txt   # segue con 50 righe iniziali
```

Monitoraggio Log

tail -f è essenziale per monitorare file di log in tempo reale:

```
$ tail -f /var/log/syslog
```

Comando head - Inizio File

Head - Mostra le prime righe di un file

```
$ head file.txt           # prime 10 righe
$ head -n 20 file.txt     # prime 20 righe
$ head -n 5 *.txt         # prime 5 righe di pi file
```

Combinazione head e tail

```
$ head -n 100 file.txt | tail -n 10
# Mostra le righe dalla 91 alla 100
```

Comando wc - Contare

Word Count - Conta righe, parole e caratteri

```
$ wc file.txt           # righe parole caratteri
$ wc -l file.txt        # solo righe
$ wc -w file.txt        # solo parole
$ wc -c file.txt        # solo byte
$ wc -m file.txt        # solo caratteri
```

Output

```
$ wc document.txt
  142   1024   7891  document.txt
   |       |       |
righe parole byte
```

Comando grep - Cercare Testo

Global Regular Expression Print - Cerca pattern

```
$ grep "parola" file.txt          # cerca parola
$ grep -i "parola" file.txt       # case insensitive
$ grep -r "pattern" dir/         # ricorsivo
$ grep -n "text" file.txt        # con numeri riga
$ grep -v "pattern" file.txt     # righe NON corrispondenti
$ grep -c "word" file.txt        # conta occorrenze
```

Con Pipe

```
$ ps aux | grep firefox
$ ls -l | grep "\.txt$"
```

Comando sort - Ordinare

Sort - Ordina righe di testo

```
$ sort file.txt           # ordina alfabeticamente
$ sort -r file.txt        # ordine inverso
$ sort -n file.txt        # ordine numerico
$ sort -u file.txt        # rimuove duplicati
$ sort -k 2 file.txt      # ordina per colonna 2
```

Esempio Pratico

```
$ du -h * | sort -h       # ordina per dimensione
$ ls -l | sort -k 5 -n     # ordina per dimensione file
```

Comando uniq - Righe Uniche

Unique - Rimuove o conta righe duplicate

```
$ uniq file.txt           # rimuove duplicati adiacenti
$ uniq -c file.txt        # conta occorrenze
$ uniq -d file.txt        # mostra solo duplicati
$ uniq -u file.txt        # mostra solo righe uniche
```

Nota Importante

uniq funziona solo su righe adiacenti! Usare con sort:

```
$ sort file.txt | uniq
$ sort file.txt | uniq -c | sort -nr # pi frequenti
```

Comando diff - Differenze

Diff - Confronta file

```
$ diff file1.txt file2.txt      # mostra differenze
$ diff -u file1 file2          # formato unified (Git)
$ diff -y file1 file2          # side by side
$ diff -r dir1/ dir2/          # ricorsivo per directory
$ diff -q dir1/ dir2/          # solo nomi file diversi
```

Simboli Output

- <: riga presente solo nel primo file
- >: riga presente solo nel secondo file
- a: aggiunta
- c: cambiamento
- d: cancellazione

Comando echo - Stampare Testo

Echo - Stampa argomenti sull'output

```
$ echo "Hello World"           # stampa testo
$ echo "text" > file.txt       # scrive in file (sovrascrive)
$ echo "text" >> file.txt      # aggiunge a file
$ echo $HOME                   # stampa variabile ambiente
$ echo $(pwd)                  # esegue comando
$ echo {1..10}                 # espansione range
```

Utilizzi Creativi

```
$ echo *.txt                   # espansione wildcard
$ echo "PATH is $PATH"         # interpolazione variabili
```

Comando chown - Cambiare Proprietario

Change Owner - Cambia proprietario di file/directory

```
$ chown user file.txt          # cambia proprietario
$ chown user:group file.txt    # cambia owner e group
$ chown -R user dir/          # ricorsivo
$ sudo chown root file.txt     # richiede permessi
```

Uso Comune

Spesso necessario dopo aver copiato file come root o da altri utenti.

```
$ sudo chown $USER:$USER file.txt
```

Comando chmod - Permessi File (1/2)

Change Mode - Modifica permessi di accesso

Permessi Unix

Tre tipi di permessi per tre categorie:

- **r** (read): lettura (4)
- **w** (write): scrittura (2)
- **x** (execute): esecuzione (1)

Tre categorie:

- **u** (user): proprietario
- **g** (group): gruppo
- **o** (others): altri
- **a** (all): tutti

Comando chmod - Permessi File (2/2)

Modalità Simbolica:

```
$ chmod u+x script.sh      # aggiungi esecuzione owner
$ chmod g-w file.txt       # rimuovi scrittura group
$ chmod a+r file.txt       # aggiungi lettura tutti
$ chmod o-rwx file.txt     # rimuovi tutti per others
```

Modalità Numerica:

```
$ chmod 755 script.sh      # rwxr-xr-x
$ chmod 644 file.txt       # rw-r--r--
$ chmod 777 file.txt       # rwxrwxrwx (sconsigliato!)
```

$$755 = 7 (rwx) + 5 (r-x) + 5 (r-x)$$

Comando umask - Permessi Predefiniti

Umask - Imposta permessi predefiniti

```
$ umask                # mostra umask corrente
0022
$ umask -S              # formato simbolico
u=rwx,g=rx,o=rx
$ umask 002             # imposta nuova umask
```

Valori Umask Comuni

- 0022: file creati con rw-r-r- (644)
- 0002: file creati con rw-rw-r- (664)
- 0077: file creati con rw----- (600)

Disk Usage - Calcola spazio occupato

```
$ du                                # spazio directory corrente
$ du -h                            # human readable (KB, MB, GB)
$ du -sh *                         # sommario di ogni elemento
$ du -sh dir/                      # spazio totale directory
$ du -ah                           # include file singoli
$ du -h | sort -h                  # ordina per dimensione
```

Top 10 Directory Più Grandi

```
$ du -h /home | sort -hr | head -10
```

Comando df - Spazio Filesystem

Disk Free - Mostra spazio disponibile su filesystem

```
$ df                # mostra tutti i filesystem
$ df -h            # formato human readable
$ df -h /home      # spazio su filesystem specifico
$ df -T            # mostra tipo filesystem
```

Output Tipico

Filesystem	Size	Used	Avail	Use%	Mounted
/dev/sda1	50G	35G	13G	73%	/
/dev/sdb1	200G	120G	72G	63%	/home

Comandi basename e dirname

basename - Estrae nome file da path:

```
$ basename /usr/local/bin/script.sh
script.sh
$ basename /home/user/
user
```

dirname - Estrae directory da path:

```
$ dirname /usr/local/bin/script.sh
/usr/local/bin
$ dirname /home/user/file.txt
/home/user
```

Comando ps - Processi in Esecuzione

Process Status - Mostra processi attivi

```
$ ps                # processi utente corrente
$ ps aux            # tutti i processi
$ ps axww           # con comandi completi
$ ps aux | grep firefox # cerca processo specifico
$ ps -u username    # processi di un utente
```

Colonne Importanti

- **PID**: Process ID (identificatore)
- **USER**: utente proprietario
- **%CPU**: utilizzo CPU
- **%MEM**: utilizzo memoria
- **STAT**: stato (R=running, S=sleeping, Z=zombie)
- **COMMAND**: comando eseguito

Comando top - Monitor Processi

Top - Monitor interattivo processi in tempo reale

```
$ top                                # avvia monitor
$ top -o mem                        # ordina per memoria
$ top -n 1                          # un refresh e poi esci
```

Comandi Interattivi in top

- q: esci
- k: kill processo (chiede PID)
- M: ordina per memoria
- P: ordina per CPU
- 1: mostra ogni CPU
- h: help

Comando kill - Terminare Processi

Kill - Invia segnali ai processi

```
$ kill PID           # termina processo (SIGTERM)
$ kill -9 PID        # termina forzatamente (SIGKILL)
$ kill -15 PID       # SIGTERM (default)
$ kill -1            # lista tutti i segnali
```

Segnali Principali

- **SIGTERM (15)**: terminazione gentile
- **SIGKILL (9)**: terminazione immediata
- **SIGHUP (1)**: ricarica configurazione
- **SIGINT (2)**: interruzione (come Ctrl+C)
- **SIGSTOP**: pausa processo
- **SIGCONT**: riprendi processo

Comando killall - Kill per Nome

Killall - Termina processi per nome

```
$ killall firefox          # termina tutti i Firefox
$ killall -9 chrome        # termina forzatamente Chrome
$ killall -u username      # termina processi di un utente
$ killall -i process       # chiede conferma
```

Attenzione!

killall termina TUTTI i processi con quel nome. Usare con cautela!

Gestione Job in Background e Foreground

```
$ command &           # esegui in background
$ jobs                 # lista job attivi
$ fg                   # porta ultimo job in foreground
$ fg %1                # porta job 1 in foreground
$ bg                   # riprendi job in background
$ bg %2                # riprendi job 2 in background
```

Control Keys

- Ctrl+Z: sospendi job corrente
- Ctrl+C: termina job corrente

Comandi type e which

type - Tipo di comando:

```
$ type ls
ls is aliased to 'ls --color=auto'
$ type cd
cd is a shell builtin
$ type python
python is /usr/bin/python
```

which - Percorso eseguibile:

```
$ which python
/usr/bin/python
$ which -a python           # mostra tutti i path
```

Comandi nohup e xargs

nohup - Esegui comando ignorando hangup:

```
$ nohup command &                # continua anche dopo logout
$ nohup ./script.sh &
# output in nohup.out
```

xargs - Costruisci ed esegui comandi:

```
$ find . -name "*.tmp" | xargs rm
$ echo "file1 file2" | xargs cat
$ ls | xargs -I {} echo "File: {}"
```

Editor di Testo in Linux

vim

```
$ vim file.txt
```

- Molto potente
- Curva apprendimento ripida
- Modalità comando/inserimento

nano

```
$ nano file.txt
```

- User-friendly
- Comandi visibili
- Ideale per principianti

emacs

```
$ emacs file.txt
```

- Estremamente potente
- Personalizzabile
- Curva apprendimento

Modalità

- **Normale:** navigazione e comandi
- **Inserimento:** digitare testo (tasto i)
- **Visuale:** selezione testo (tasto v)

Comandi Base:

- i: modalità inserimento
- ESC: modalità normale
- :w: salva
- :q: esci
- :wq: salva ed esci
- :q!: esci senza salvare

Navigazione:

- h,j,k,l: sinistra, giù, su, destra
- dd: cancella riga
- yy: copia riga
- p: incolla
- u: undo
- /text: cerca

Nano - Editor Semplice

```
$ nano filename.txt
```

Comandi Principali (=Ctrl)

- Ctrl+O: salva file (WriteOut)
- Ctrl+X: esci dall'editor
- Ctrl+K: taglia riga
- Ctrl+U: incolla
- Ctrl+W: cerca testo
- Ctrl+G: mostra help
- Alt+/: vai alla fine del file

I comandi sono sempre visibili in basso allo schermo!

whoami - Chi sono?

```
$ whoami  
username
```

who - Chi è connesso?

```
$ who  
user1      pts/0      2024-12-07 10:30  
user2      pts/1      2024-12-07 11:15
```

id - Informazioni utente

```
$ id  
uid=1000(user) gid=1000(user) groups=1000(user),27(sudo)
```

Comandi su e sudo

su - Switch User:

```
$ su                # diventa root (chiede password root)
$ su username       # diventa altro utente
$ su -              # diventa root con ambiente
```

sudo - Execute as SuperUser:

```
$ sudo command      # esegui comando come root
$ sudo -i           # shell root interattiva
$ sudo -u user command # esegui come altro utente
$ sudo !!           # riesegui ultimo comando come root
```

Sicurezza

Usa sempre sudo invece di su – quando possibile!

Comando passwd - Password

Password - Cambia password utente

```
$ passwd # cambia la tua password
$ sudo passwd username # cambia password altro utente
$ sudo passwd -l username # blocca account
$ sudo passwd -u username # sblocca account
$ sudo passwd -e username # forza cambio al prossimo login
```

Buone Pratiche

- Password lunghe (min 12 caratteri)
- Mix di maiuscole, minuscole, numeri, simboli
- Non riutilizzare password
- Cambiare periodicamente

Comando ping - Test Connettività

Ping - Verifica connessione di rete

```
$ ping google.com           # ping continuo
$ ping -c 4 google.com      # 4 pacchetti e stop
$ ping -i 2 host             # intervallo 2 secondi
$ ping -W 1 host             # timeout 1 secondo
```

Output Tipico

```
64 bytes from google.com: icmp_seq=1 ttl=117 time=12.4 ms
64 bytes from google.com: icmp_seq=2 ttl=117 time=11.8 ms
--- google.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss
round-trip min/avg/max = 11.8/12.1/12.4 ms
```

Traceroute - Traccia percorso pacchetti

```
$ traceroute google.com      # mostra tutti gli hop
$ traceroute -m 15 host      # max 15 hop
$ traceroute -n host         # non risolvere nomi
```

Utilizzo

- Diagnostica problemi di rete
- Identifica dove avviene il ritardo
- Mostra il percorso attraverso router
- Utile per troubleshooting

Su alcuni sistemi Linux: `tracert`

Altri Comandi di Rete

ifconfig/ip - Configurazione interfacce:

```
$ ifconfig          # mostra interfacce (deprecato)
$ ip addr show      # comando moderno
$ ip link show      # mostra solo link
```

netstat/ss - Connessioni:

```
$ netstat -tuln      # connessioni TCP/UDP
$ ss -tuln           # comando moderno
$ ss -t              # solo TCP
```

wget/curl - Download:

```
$ wget http://example.com/file.zip
$ curl -O http://example.com/file.zip
```

Comando export - Variabili Ambiente

Export - Esporta variabili all'ambiente

```
$ export VAR="value"          # crea/esporta variabile
$ export PATH=$PATH:/new/path # aggiunge a PATH
$ export                      # lista tutte le variabili
$ export -n VAR               # rimuove export
```

Variabili Comuni

- PATH: percorsi eseguibili
- HOME: directory home utente
- USER: nome utente
- SHELL: shell corrente
- PWD: directory corrente
- EDITOR: editor predefinito

Comandi env e printenv

env - Esegui con ambiente specifico:

```
$ env                # mostra tutte le variabili
$ env VAR=value command # esegui con variabile
$ env -i command     # esegui con ambiente pulito
$ env -u VAR command  # rimuovi variabile
```

printenv - Stampa variabili:

```
$ printenv          # tutte le variabili
$ printenv PATH     # solo PATH
$ printenv HOME USER # pi variabili
```

Variabili d'Ambiente - Esempi

Impostare Editor Predefinito:

```
$ export EDITOR=nano
$ export VISUAL=nano
```

Modificare PATH:

```
$ export PATH=$PATH:$HOME/bin
$ export PATH="/usr/local/bin:$PATH"
```

Variabili Personalizzate:

```
$ export API_KEY="abc123"
$ export DB_HOST="localhost"
$ echo $API_KEY
```

File di Configurazione Shell

Bash

- ~/.bashrc: configurazione shell interattiva
- ~/.bash_profile: configurazione login
- ~/.bash_aliases: alias personalizzati
- /etc/bash.bashrc: configurazione sistema

Zsh

- ~/.zshrc: configurazione principale
- ~/.zshenv: variabili d'ambiente
- ~/.zprofile: configurazione login

Ricaricare configurazione:

```
$ source ~/.bashrc
```

Comando crontab - Job Schedulati

Crontab - Pianifica esecuzione automatica comandi

```
$ crontab -l          # lista cron job
$ crontab -e          # modifica cron job
$ crontab -r          # rimuovi tutti i job
$ crontab -u user -e  # edita per altro utente
```

Sintassi Crontab

minuto ora giorno mese giorno_sett comando

- *: ogni valore
- */5: ogni 5 unità
- 1,5,9: valori specifici
- 1-5: range di valori

Crontab - Esempi Pratici

```
# Ogni giorno alle 2:30 AM
30 2 * * * /path/to/backup.sh

# Ogni ora
0 * * * * /path/to/script.sh

# Ogni 15 minuti
*/15 * * * * /path/to/check.sh

# Ogni lunedì alle 9:00
0 9 * * 1 /path/to/weekly.sh

# Primo giorno del mese alle 6:00
0 6 1 * * /path/to/monthly.sh

# Ogni 12 ore
0 */12 * * * /path/to/script.sh
```

Comando uname - Info Sistema

Unix Name - Informazioni sul sistema

```
$ uname                # nome kernel
$ uname -a             # tutte le info
$ uname -s             # nome sistema operativo
$ uname -r             # release kernel
$ uname -m             # architettura hardware
$ uname -p             # tipo processore
$ uname -n             # nome host
```

Output Esempio

```
$ uname -a
Linux hostname 5.15.0-56 x86_64 GNU/Linux
```

Comando history - Cronologia

History - Cronologia comandi eseguiti

```
$ history                # mostra cronologia
$ history 20             # ultimi 20 comandi
$ !123                   # esegui comando numero 123
$ !!                     # ripeti ultimo comando
$ !ping                  # ultimo comando che inizia con ping
$ history -c             # pulisci cronologia
```

Ricerca Incrementale

- Ctrl+R: ricerca nella cronologia
- Digita parte del comando
- Ctrl+R ancora per precedente
- Enter per eseguire

Comando `clear` - Pulire Schermo

Clear - Pulisce il terminale

```
$ clear
```

Scorciatoia da tastiera:

Ctrl + L

Altri Modi per "Pulire"

<pre>\$ reset</pre>	<pre># reset completo terminale</pre>
<pre>\$ tput clear</pre>	<pre># alternativa a clear</pre>

Output Redirection:

```
$ command > file.txt          # sovrascrive file
$ command >> file.txt          # aggiunge a file
$ command 2> error.log         # solo errori
$ command &> all.log           # output ed errori
$ command > /dev/null          # scarta output
```

Input Redirection:

```
$ command < input.txt          # legge da file
$ command << EOF               # heredoc
Testo multiplo
EOF
```

Pipe - Concatenare Comandi

Pipe (—) - Output di un comando come input di un altro

```
$ ls -l | grep ".txt"           # filtra output
$ cat file | grep "error" | wc -l
$ ps aux | grep firefox | awk '{print $2}'
$ history | tail -20 | head -10
```

Esempi Complessi

```
# Top 10 processi per memoria
$ ps aux | sort -nrk 4 | head -10

# Conta file per tipo
$ find . -type f | sed 's/.*\\.//' | sort | uniq -c

# Log errors nelle ultime 100 righe
$ tail -100 app.log | grep ERROR | wc -l
```

Wildcards - Pattern Matching

Wildcards Comuni

- *: qualsiasi sequenza di caratteri
- ?: un singolo carattere
- [abc]: uno tra a, b, o c
- [a-z]: range di caratteri
- [!abc]: qualsiasi tranne a, b, c

```
$ ls *.txt           # tutti i file .txt
$ ls file?.txt       # file1.txt, fileA.txt, etc.
$ ls [abc]*.txt      # iniziano con a, b, o c
$ rm temp*           # rimuove tutti temp...
$ cp project[1-5].doc backup/
```

Espansioni della Shell

Brace Expansion:

```
$ echo {1..10}          # 1 2 3 ... 10
$ echo file{1,2,3}.txt  # file1.txt file2.txt file3.txt
$ mkdir -p dir/{sub1,sub2,sub3}
$ echo {A..Z}           # A B C ... Z
```

Command Substitution:

```
$ echo "Oggi $(date)"
$ files=$(ls *.txt)
$ current_user='whoami' # vecchia sintassi
```

Tilde Expansion:

```
$ cd ~/documents      # $HOME/documents
```

Operatori Logici

```
# AND - esegui secondo solo se primo ha successo
$ mkdir newdir && cd newdir

# OR - esegui secondo solo se primo fallisce
$ command1 || command2

# Sequential - esegui entrambi indipendentemente
$ command1 ; command2

# Background
$ long_task &
```

Esempi Pratici

```
$ make && make install
$ ping -c 1 google.com && echo "Online" || echo "Offline"
$ cd /tmp && rm -f tempfile ; cd -
```

Creare Script Bash

```
#!/bin/bash
# Questo      un commento

echo "Hello World!"
echo "Utente: $USER"
echo "Home: $HOME"

# Variabili
NOME="Mario"
echo "Ciao $NOME"

# Parametri
echo "Primo argomento: $1"
echo "Tutti gli argomenti: $@"
echo "Numero argomenti: $#"
```

Rendere eseguibile:

Script Bash - Condizioni e Loop

If Statement:

```
if [ -f "file.txt" ]; then
    echo "File esiste"
else
    echo "File non trovato"
fi
```

For Loop:

```
for i in {1..5}; do
    echo "Iterazione $i"
done

for file in *.txt; do
    echo "Processing $file"
done
```

Non Usare mai:

- `rm -rf /` (cancella tutto!)
- `chmod 777` su file sensibili
- Eseguire script da fonti non fidate
- Usare `sudo` senza capire il comando

Raccomandazioni

- Leggere `man` prima di usare comandi nuovi
- Fare backup prima di operazioni distruttive
- Usare `-i` (interactive) con `rm`, `mv`, `cp`
- Verificare con `ls` prima di `rm *`
- Non loggarsi come root se non necessario

Alias Utili

```
alias ll='ls -alh'
alias ..='cd ..'
alias ...='cd ../..'
alias grep='grep --color=auto'
alias df='df -h'
alias du='du -h'
```

Consigli

- Usa Tab per autocompletamento
- Usa Ctrl+R per cercare nella cronologia
- Impara le scorciatoie da tastiera
- Crea script per task ripetitivi
- Documenta i tuoi script

Controllo Processi:

- `Ctrl+C`: interrompi processo
- `Ctrl+Z`: sospendi processo
- `Ctrl+D`: EOF / logout

Navigazione:

- `Ctrl+A`: inizio riga
- `Ctrl+E`: fine riga
- `Ctrl+U`: cancella fino a inizio
- `Ctrl+K`: cancella fino a fine

Editing:

- `Ctrl+W`: cancella parola
- `Ctrl+L`: clear screen
- `Ctrl+R`: ricerca cronologia
- `Tab`: autocompletamento

Altro:

- `!!`: ripeti ultimo comando
- `!`: ultimo argomento
- `!*`: tutti gli argomenti

Documentazione

- `man` pages: documentazione locale
- `info` pages: documentazione GNU
- `help` comando: help builtin shell
- comando `--help`: opzioni comando

Online

- <https://tldr.sh/> - Esempi pratici comandi
- <https://explainshell.com/> - Spiega comandi
- <https://www.gnu.org/software/bash/manual/> - Manuale Bash
- <https://stackoverflow.com/> - Q&A community

Navigazione:

- ls, cd, pwd
- mkdir, rmdir
- find

File:

- cp, mv, rm
- touch, cat, less
- grep, diff

Permessi:

- chmod, chown
- sudo, su

Processi:

- ps, top, kill
- jobs, bg, fg

Sistema:

- df, du
- uname, history

Rete:

- ping, traceroute

Utility:

- tar, gzip
- export, env

Punti Chiave

- Linux è potente, flessibile e libero
- La shell è più efficiente di una GUI per molte operazioni
- Regola 80/20: pochi comandi coprono la maggior parte dei casi d'uso
- `man` è il tuo migliore amico
- Pratica costante è la chiave dell'apprendimento

**La linea di comando è uno strumento
che richiede tempo per essere padroneggiato,
ma ne vale assolutamente la pena!**

Grazie per l'attenzione!

Domande?

Basato su:

"The Linux Commands Handbook"
di Flavio Copes

Presentazione creata per IIS Fermi Sacconi Ceci