1. Le basi di Linux

Amministrare un sistema Linux

1.1. Aspetti generali

Unix

Unix è un sistema operativo, nato nel 1969 presso i Bell Labs ad opera di Dennis Ritchie, Ken Thompson, Brian Kernighan ed altri programmatori. Inizialmente scritto in assembly e poi riscritto in C.

Unix divenne un prodotto commerciale, con costo elevato ed il codice sorgente non era incluso. Anche acquistando separatamente una copia dei sorgenti, non era più possibile modificarli e condividere le migliorie apportate con altri programmatori. Altre società commerciali adottarono la modalità di distribuzione del software senza sorgenti, ponendo le basi di un nuovo modello di sviluppo proprietario.

Nel 1984 Richard Stallman decise di dare vita ad un nuovo sistema operativo di tipo Unix il cui codice sorgente potesse essere liberamente copiato e modificato. Nacque il progetto GNU (GNU is Not Unix). Il nuovo modello di sviluppo prese il nome di Software Libero (free software). Venne scritta una licenza specifica GNU General Public License (GPL) che aggirasse i limiti imposti dai diritti d'autore e consentisse a chiunque di copiare e modificare un lavoro, seppur nel rispetto di condizioni e termini rigorosi.

Linux

Agli inizi degli anni '90, Linus Torvald, uno studente finlandese in scienze dell'informazione iniziò ad apportare variazioni a Minix, un sistema operativo di tipo Unix per personal computer allora utilizzato nei corsi universitari sui sistemi operativi. Torvald decise di migliorare il componente principale del software alla base di Minix, chiamato kernel, e di scriverne uno proprio.

Alla fine del 1991, Torvald pubblicò la prima versione di questo kernel su Internet e la battezzò “Linux”. La forza di questo progetto fu l'adozione della licenza GNU GPL, in questo modo. Linux risultava un software che poteva essere liberamente utilizzato, copiato e modificato da chiunque, a condizione che le stesse libertà fossero estese a tutte le copie e le varianti.

Nel tempo e da tutto il mondo migliaia di programmatori sparsi sull'intero pianeta contribuirono al suo progetto e Linux è diventato un sistema operativo completo, moderno e che può essere utilizzato sia da programmatori che da non addetti ai lavori.

La struttura di Linux

Il concetto base dell'architettura di ogni sistema Unix come GNU/Linux è quello di una rigida separazione fra il kernel (il nucleo del sistema), cui si demanda la gestione delle risorse hardware (come la CPU, la memoria, le periferiche) ed i processi, (le unità di esecuzione dei programmi), che nel caso vanno dai comandi base di sistema, agli applicativi, alle interfacce per l'interazione con gli utenti.

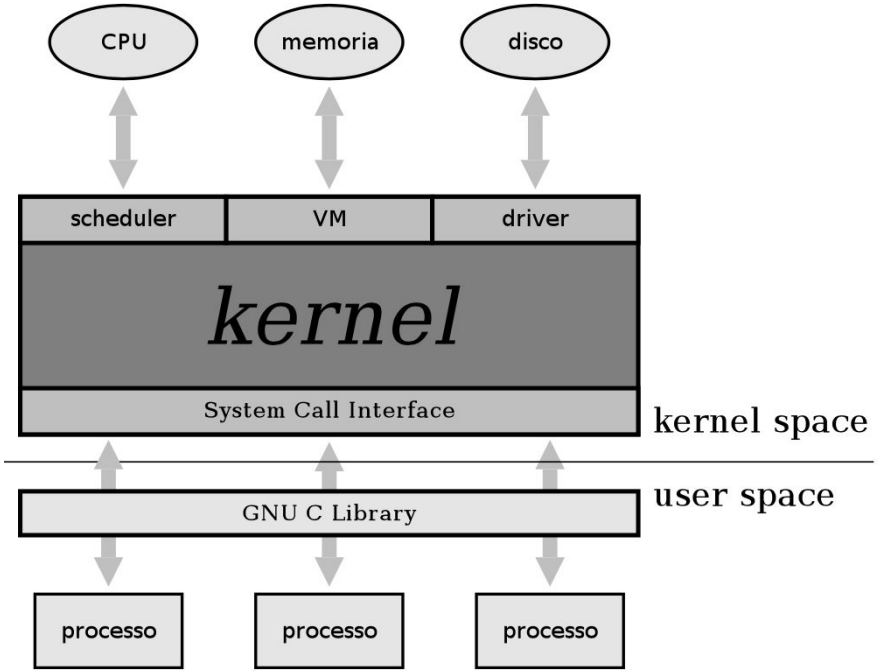
Lo scopo del kernel infatti è solo quello di essere in grado di eseguire contemporaneamente molti processi in maniera efficiente, garantendo una corretta distribuzione fra gli stessi della memoria e del tempo di CPU, e quello di fornire le adeguate interfacce software per l'accesso alle periferiche della macchina e le infrastrutture di base necessarie per costruire i servizi. Tutto il resto, dall'autenticazione all'interfaccia utente, viene realizzato usando processi che eseguono gli opportuni programmi.

Questo si traduce in una delle caratteristiche essenziali su cui si basa l'architettura dei sistemi Unix: la distinzione fra il cosiddetto user space, che è l'ambiente a disposizione degli utenti, in cui vengono eseguiti i processi, e il kernel space, che è l'ambiente in cui viene eseguito il kernel.

La struttura di Linux

I due ambienti comunicano attraverso un insieme di interfacce ben definite e standardizzate; secondo una struttura come quella mostrata in figura.

Per **kernel** si intende il cuore di un sistema operativo: il codice che gestisce le risorse presenti sul sistema e le rende disponibili alle applicazioni.



Distro

Benché il kernel costituisca il cuore del sistema, da solo sarebbe assolutamente inutile. Per avere un sistema funzionante dal punto di vista di un utente normale infatti occorre avere, oltre al kernel, anche tutti i programmi che gli permettano di eseguire le varie operazioni con i dischi, i file, le periferiche.

Per questo al kernel vengono sempre uniti degli opportuni programmi di gestione per il sistema e tutta una serie di programmi applicativi, ed è l'insieme di questi e del kernel che costituisce un sistema funzionante. Di solito i rivenditori, o anche gruppi di volontari, come nel caso di Debian, si preoccupano di raccogliere in forma coerente i programmi necessari, per andare a costruire quella che viene chiamata una distribuzione. Alcuni esempi di distribuzioni sono Debian, RedHat e Ubuntu. Spesso ci si riferisce a una distribuzione semplicemente chiamandola "Linux".

Il gruppo principale di questi programmi, e le librerie di base che essi e tutti gli altri programmi usano, derivano dal progetto GNU della Free Software Foundation: è su di essi che ogni altro programma è basato, ed è per questo che è più corretto riferirsi all'intero sistema come a GNU/Linux, dato che Linux indica solo una parte, il kernel, che benché fondamentale non costituisce da solo un sistema operativo.

Ubuntu

Ubuntu è una distribuzione GNU/Linux basata sull'ambiente desktop GNOME. È progettata per fornire un'interfaccia semplice, intuitiva e allo stesso tempo completa e potente. I punti di forza di questa distribuzione sono l'estrema semplicità di utilizzo, l'ottimo riconoscimento e supporto dell'hardware, il vasto parco software costantemente aggiornato e una serie di strumenti di gestione grafici che la rendono improntata verso l'ambiente desktop. È corredata da un'ampia gamma di applicazioni libere. La versione desktop è stata realizzata per rispondere alle più frequenti necessità di un utente medio, quali navigazione in Internet, gestione dei documenti e delle immagini, svago e comunicazione.

Informazioni su OS: `cat /etc/os-release`

Informazioni sul kernel: `uname`

Comandi utili

- Documentazione dei comandi: `man <nome_comando>`
- Mostra gli ultimi comandi eseguiti: `history`
- Pulisce la schermata del terminale: `clear`
- Visualizza il percorso completo di dove si trova il comando: `which <programma>`
- Filtra le righe che contengono la parola “nome”: `grep nome`
- Visualizza i file che si trovano sotto “percorso” aventi “nome_file”:
  `find <percorso> -name <nome_file>`
- Mostrare la shell attualmente in uso: `echo $SHELL`
- Modificare un file: `nano <path_del_file>`

1.2. Gestione di file e directory

Gestione dei file

Un aspetto fondamentale dell'architettura di GNU/Linux è quello della gestione dei file. Esso deriva direttamente da uno dei criteri base della progettazione di tutti i sistemi Unix, quello espresso dalla frase "everything is a file" (cioè tutto è un file), per cui l'accesso ai file e alle periferiche è gestito attraverso una interfaccia identica. Inoltre, essendo in presenza di un sistema multiutente e multitasking, il kernel deve anche essere in grado di gestire l'accesso contemporaneo allo stesso file da parte di più processi.

In un sistema Unix tutti i file di dati sono uguali: non esiste cioè la differenza tra file di testo o binari che c' è in Windows. Inoltre le estensioni sono solo convenzioni, e non significano nulla per il kernel, che legge tutti i file di dati alla stessa maniera, indipendentemente dal nome e dal contenuto.

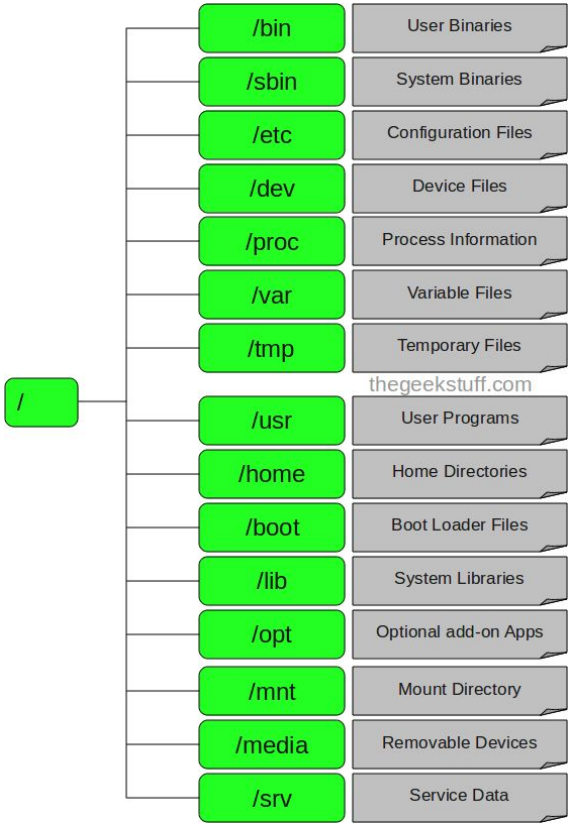
Linux (ed ogni sistema unix-like) organizza i dati che tiene su disco attraverso l'uso di un **file system**. Un **file system** è l'insieme dei tipi di dati astratti necessari per la memorizzazione (scrittura), l'organizzazione gerarchica, la manipolazione, la navigazione, l'accesso e la lettura dei dati.

File system

Le informazioni riguardanti un oggetto (file o directory) di un file system sono contenute in un **inode**, che viene identificato da un numero progressivo e descrive le caratteristiche base di un oggetto: permessi, data di modifica, tipo, posizione ecc.

Un sistema Linux, come ogni sistema Unix, ha una directory principale, chiamata **root** ed indicata con / sotto la quale si trovano tutte le altre directory e tutti gli altri file system eventualmente montati sul sistema.

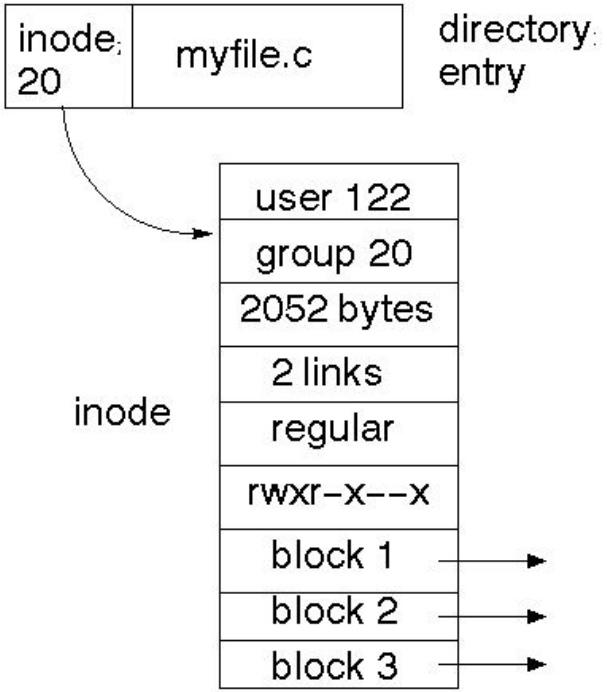
Il principio è radicalmente diverso da quello presente nel mondo Windows, dove ogni device o risorsa ha un suo nome o lettera identificativa al cui interno si trovano le directory del relativo file system.



inode

La struttura che identifica univocamente un singolo file all'interno di un filesystem è il cosiddetto inode: ciascun file è associato ad un inode in cui sono mantenute tutte le informazioni che lo riguardano, come il tipo, i permessi di accesso, utente e gruppo proprietario, le dimensioni, i tempi, ed anche tutti i riferimenti ai settori del disco (i blocchi fisici) che contengono i dati..

L'unica informazione non contenuta nell'inode è il suo nome; infatti il nome di un file non è una proprietà del file, ma semplicemente un'etichetta associata ad un inode. Le directory infatti non contengono i file, ma sono dei file speciali (di tipo directory, così che il kernel possa trattarle in maniera diversa) il cui contenuto è semplicemente una lista di nomi a ciascuno dei quali viene associato un numero di inode che identifica il file cui il nome fa riferimento.



Hard link

Si possono avere più voci in directory diverse che puntano allo stesso inode. Questo introduce il concetto di **hard link** (detto anche collegamento diretto): due file che puntano allo stesso inode sono fisicamente lo stesso file, nessuna proprietà specifica, come permessi, tempi di accesso o contenuto permette di distinguerli, in quanto l'accesso avviene per entrambi attraverso lo stesso inode. Siccome uno stesso inode può essere referenziato in più directory, un file può avere più nomi, anche completamente scorrelati fra loro. Per questo ogni inode mantiene un contatore che indica il numero di riferimenti che gli sono stati fatti, il cosiddetto **link count**.

Questo numero viene mostrato ad esempio nel risultato di `ls -l`. Il valore riportato nella seconda colonna della stampa dei risultati ci permette di dire se un file ha degli hard link, anche se non indica dove sono.

Per creare un hard link basta usare direttamente il comando ln, (il cui nome deriva da link file) :

- Crea un hard link a un file: `ln <nome_file> <hardlink>`

Soft link

Un limite che si ha con gli hard link è dovuto al fatto che si può fare riferimento solo ad un inode presente nello stesso filesystem della directory. Il comando **ln** darà un errore se si cerca di creare un hard link ad un file posto in un altro filesystem. Per superare questa limitazione sono stati introdotti i cosiddetti **collegamenti simbolici** o **symbolic link**, che vengono creati usando **l'opzione -s del comando ln**;

- Creato un soft link a un file: `ln -s <nome_file> <softlink>`

In questo caso viene creato un nuovo file, di tipo symbolic link, che avrà un suo **diverso inode** ed il cui contenuto è il percorso da fare per arrivare al file a cui esso fa riferimento, che a questo punto può essere in qualsiasi altro filesystem. E compito del kernel far sì che quando si usa un link simbolico si vada poi ad operare sul file che questo ci indica, ed è questo il motivo per cui per i link simbolici esiste **un apposito tipo di file**. Una seconda caratteristica dei link simbolici è la possibilità di creare dei collegamenti anche per delle directory. Questa capacità infatti, sebbene teoricamente possibile anche per gli hard link, in Linux non è supportata per la sua pericolosità.

Comandi per navigare le directory

- Elenca file e sotto-directory della directory corrente: `ls`
- Elenca file e sotto-directory di una cartella: `ls <percorso_relativo_o_assoluto_cartella>`
- Alias per la directory home (ls /home/utente/videos): `ls ~/videos`
- Carattere jolly: `ls ~/videos/*.mp4`
- Visualizzare contenuto delle sotto-directory: `ls -R`
- Elenca file e sotto-directory in colonna: `ls -l`
- Visualizza anche i file nascosti: `ls -a`
- Visualizzare la directory corrente: `pwd`
- Spostarsi in una directory: `cd ~/videos`
- Spostarsi nella directory precedente: `c -`

Linux non distingue tra file e cartelle. Una directory non può contenere due file e/o cartelle con lo stesso nome.

Comandi per gestire i file

- Modifica ora di accesso e modifica: `touch test.txt`
- Crea un nuovo file: `touch new_file.txt`
- Crea una nuova cartella: `mkdir  videos`
- Crea una nuova cartella e tutte le sottodirectory: `mkdir  -p videos/2022/12/`
- Copia un file: `cp test.txt videos/2022/12`
- Copia una directory: `cp -R videos/2022/12 videos/2023/12`
- Spostare un file o cartella: `mv test.txt videos/2022/12`
- Rinomina un file o cartella: `mv test.txt test1.txt`
- Elimina file: `rm test.txt`
- Elimina cartella: `rmdir videos/2022/12`
- Elimina ricorsivamente file e cartelle: `rm -Rf videos/`
- Stampare il contenuto di un file: `cat test.txt`

1.3. Gli utenti

Gli utenti

Su Linux esistono differenze fra i vari utenti, definite dai permessi e dall'accesso ai file e comandi che un utente può lanciare. È convenzione che i semplici utenti possano scrivere, leggere e modificare file solo all'interno del loro ambiente (home) e lanciare semplici comandi che non influiscono sulla configurazione del sistema. Per poter accedere completamente alle risorse del sistema bisogna accedere al sistema come amministratore ovvero come utente **root**.

In fase di installazione di una macchina Linux si consiglia di scegliere una password di root piuttosto complicata e di creare immediatamente un normale utente con il quale operare per tutte le attività di tipo non amministrativo. Nei server viene anche disattivato il login come amministratore da remoto e vengono utilizzati metodi di autenticazione più sicuri e complessi rispetto all'uso della password.

All'interno del sistema il kernel identifica ogni utente con un numero identificativo, chiamato user ID o UID. Questo è nullo per l'amministratore e diverso da zero per tutti gli altri utenti.

/etc/passwd

Le informazioni relative agli utenti sono memorizzate nel file **/etc/passwd**.

Comando: *less /etc/passwd*

Ogni riga rappresenta le informazioni di login per un utente.

- User name
- Password criptata (x significa che la password è memorizzata nel file /etc/shadow)
- User ID (UID)
- User group ID (GID)
- Nome completo dell'utente
- Home directory dell'utente
- Shell di default

I gruppi

Gli utenti poi possono venire raggruppati in gruppi; come per gli utenti ogni gruppo ha un nome (detto groupname) ed un corrispondente identificatore numerico, il group ID o GID.

Inoltre ad ogni utente è sempre associato almeno un gruppo, detto gruppo primario o gruppo principale, che è quello che viene usato anche come group ID. Di norma si fa sì che questo gruppo contenga come membro solo l'utente in questione e abbia nome uguale all'username, per dare un gruppo specifico ai file personali dell'utente dato che questo è il gruppo che viene usato nella creazione di nuovi file.

In generale però un utente può appartenere ad un numero qualunque di ulteriori gruppi, detti gruppi ausiliari; diventa così possibile permettere l'accesso a delle risorse comuni a tutti gli utenti che fanno parte dello stesso gruppo.

/etc/group

Le informazioni relative a gruppi sono memorizzate nel file **/etc/group**.

Comando: *less /etc/group*

Ogni riga rappresenta le informazioni di un gruppo.

- Nome gruppo
- Password criptata (x significa che la password è memorizzata nel file /etc/shadow)
- Group ID (GID)
- Lista degli utenti nel gruppo

Gestione dei privilegi

- Impersonare un altro utente (**s**ubstitute **u**ser): `su richard`
- Impersonare l'utente root: `su`
- Impersonare un altro utente con le variabili d'ambiente: `su -l <nome_utente>`
- Stampa il nome utente dell'utente attualmente loggato: `whoami`
- Eseguire un comando impersonando un utente: `sudo -u <nome_utente> whoami`
- Eseguire un comando impersonando l'utente root: `sudo whoami`

Gestione degli utenti

- Stampa l'id dell'utente e i gruppi di appartenenza: `id <nome utente>`
- Crea un nuovo utente: `sudo useradd <nome_utente>`
- Imposta la password per un nuovo utente: `sudo passwd <nome_utente>`
- Crea un nuovo utente con la relativa home directory: `sudo useradd -m <nome_utente>`
- Crea un nuovo utente e assegna i gruppi: `sudo useradd -g <primary> -G <group, group> <nome_utente>`
- Restituisce le informazioni di default di useradd: `useradd -D`
- Crea un gruppo: `sudo groupadd <nome_gruppo>`
- Aggiunge un utente a un gruppo: `sudo usermod -aG <nome_gruppo> <nome_utente>`
- Aggiunge un utente al gruppo sudo: `sudo usermod -aG sudo <nome_utente>`
- Rimuove un utente da un gruppo: `sudo gpasswd -d <nome_utente> <nome_gruppo>`
- Elimina un utente con la sua directory e lo spool di posta: `sudo userdel -r <nome_utente>`

Utente e gruppo di un file

Ogni file appartiene ad un utente e a un gruppo. È possibile cambiare (avendo i permessi) l'utente ed il gruppo con i comandi: chown (change owner) e chgrp (change group).

Eseguendo il comando `ls -l` possiamo vedere l'utente proprietario e il gruppo di appartenenza del file:

`-rw-r--r--  1 **richard richard**  3771 Jan  6  2022 .bashrc`

I comandi per modificare l'utente proprietario e il gruppo:

- Cambiare l'utente: `chown <nome_utente> <nome_file>`
- Cambiare il gruppo: `chgrp <nome_gruppo> <nome_file>`
- Cambiare l'utente e il gruppo: `chown <nome_utente>:<nome_gruppo> <nome_file>`

Al posto del nome utente e del gruppo è possibile specificare i rispettivi UID e GID.

Permessi dei file

Ogni file dispone di permessi separati per tre categorie di utenti:

- Utente proprietario (**u**ser)
- Gruppo proprietario (**g**roup)
- Il resto degli utenti (**o**ther)

Per ogni categoria esistono tre tipi di permessi:

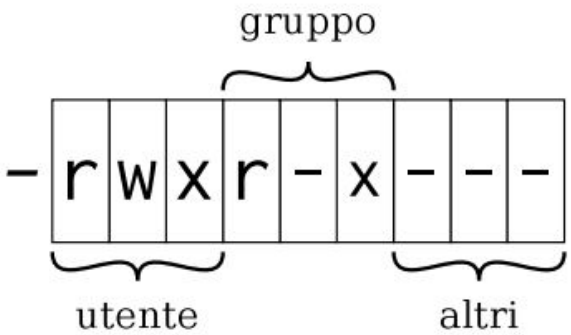
- Lettura definito dal flag **r** (**r**ead)
- Scrittura definito dal flag **w** (**w**rite)
- Esecuzione definito dal flag **x** (e**x**ecute)

Risultato del comando `ls -l /etc/passwd`: `-rw-r--r-- 1 root root 1719 Nov  4 21:03 /etc/passwd`

Permessi delle directory

Per le directory questi flag hanno un significato diverso:

- Lettura (**read**): poter visualizzare la lista dei file contenuti nella directory
- Scrittura (**write**): poter creare, copiare o spostare i file contenuti nella/dalla directory
- Esecuzione (**execute**): poter accedere e leggere i file contenuti nella directory



Modifica permessi dei file e directory

Per modificare le modalità di accesso ad un file si usa il comando: `chmod <operazione> <nome_file>`
L'operazione a sua volta è composta da:

Categoria:

- **u** utente proprietario del file
- **g** gruppo proprietario del file
- **o** tutti gli altri utenti del sistema
- **a** tutti gli utenti (equivale a 'ugo')

Azione:

- **+** Aggiunge permessi ai permessi esistenti
- **-** Rimuove permessi dai permessi esistenti
- **=** Assegna i permessi al file

Permesso: r,w,x hanno il significato visto in precedenza

Note sui permessi

- Il superutente, root, può sempre accedere a qualsiasi file presente sul sistema, indipendentemente dai suoi permessi di accesso.
- Di fatto **rwx**, sono la rappresentazione dello stato di tre bit, partendo dal più significativo si ha: r = 4, w=2, x= 1 pertanto i comandi **chmod** possono essere impartiti anche in forma numerica.
- Fino ad adesso si sono trattati soltanto i permessi ordinari, a questi però in tutti i sistemi unix-like sono stati aggiunti tre permessi speciali che permettono di rendere più flessibile una infrastruttura che, se fosse stata limitata soltanto a quegli ordinari, sarebbe stata troppo limitata. Dal punto di vista del kernel ciascun permesso corrisponde ad un bit in una apposita variabile mantenuta nell'inode del file. I bit usati per i permessi sono in tutto 12; i primi nove sono quelli già illustrati, gli altri tre vengono chiamati col nome assegnato al rispettivo bit, e cioè: **suid bit**, **sgid bit** e **sticky bit**.

1.4. Programmi e processi

Package manager

L'utilizzo dei Package Manager permette di installare, aggiornare, verificare o rimuovere i programmi con molta facilità. I programmi vengono raccolti all'interno di un singolo file che contiene anche le istruzioni per l'installazione e la disinstallazione necessarie al Package Manager.

L'**Advanced Packaging Tool** (conosciuto con l'acronimo **APT**), in informatica, è il gestore standard di pacchetti software della distribuzione GNU/Linux Debian.

La lista delle sorgenti software da cui attingere i pacchetti è contenuta nei file:

- /etc/apt/sources.list
- /etc/apt/sources.list.d

Per utilizzarlo basta digitare il comando: `apt` o `apt-get`

APT

- Installazione di un pacchetto: `apt install <nome_pacchetto>`
- Rimozione di un pacchetto: `apt remove <nome_pacchetto>`
- Rimozione di un pacchetto e configurazioni: `apt remove-purge <nome_pacchetto>`
- Rimozione di un pacchetto e dipendenze non più utilizzate: `apt autoremove <nome_pacchetto>`
- Aggiornare un pacchetto: `apt upgrade <nome_pacchetto>`
- Aggiornare tutti i pacchetti: `apt upgrade`
- Scarica la lista aggiornata dei pacchetti: `apt update`
- Scarica la nuova versione dei pacchetti e del kernel: `apt dist-upgrade -y`
- Esegue l'avanzamento di versione: `apt do-release-upgrade`

Monitorare le risorse del sistema

- Lista dei processi: `ps aux`
- Albero dei processi: `ps axjf`
- Terminare un processo: `kill <pid_processo>`
- Elenco dinamico dei processi: `top`
- Visualizzare informazioni sulla ram: `free -m`
- Informazioni sullo spazio sul disco (disk free): `df -h`
- Spazio occupato da una cartella: `du -hs folder/`
- Spazio occupato dal contenuto di una directory: `du -h folder/`

1.5. Connettività

Comandi utili per la connettività

- Visualizza informazioni sulle interfacce di rete: `ifconfig -a`
- Verificare la connettività con un host (pacchetto ICMP): `ping <indirizzo_host_remoto>`
- Traccia il percorso dei pacchetti sulla rete: `traceroute <indirizzo_host_remoto>`
- Risoluzione di un dominio con DNS (DNS lookup): `host <dominio>`
- Effettuare richieste di rete con diversi protocolli: `curl <indirizzo>`
- Scaricare un file: `wget <indirizzo>`

Risorse interessanti

- Olicyber (olimpiadi italiane di cybersicurezza): https://olicyber.it/
- CyberChallengeIT: https://cyberchallenge.it/
- PortSwigger (Web Security Academy): https://portswigger.net/web-security
- PwnFunction (Web Security): https://youtube.com/playlist?list=PLI_rLWXMqpSl_TqX9bbisW-d7tDqcVvOJ
- HackLog: https://hacklog.net/

```
richard@ubuntu:~$ echo Grazie per l’attenzione!
```