Appunti sull'interfaccia a riga di comando dei sistemi Linux-based

Filippo Maria LAURIA
filippo.lauria@iit.cnr.it

Istituto di Informatica e Telematica — Consiglio Nazionale delle Ricerche
via G. Moruzzi, 1 — 56124 Pisa

Introduzione

I sistemi Linux-based sono sistemi operativi basati sul kernel Linux. Essi sono sistemi Unix-like, ovvero molto simili a UNIX, un sistema operativo proprietario nato tra il 1960 ed il 1970. Il sistema operativo GNU/Linux è un sistema operativo Unix-like, nato dalla combinazione del kernel Linux, creato negli anni '90 da Linus Torvald, e del sistema operativo GNU, ideato da Richard Stallman negli anni '80. GNU/Linux è diffuso attraverso tante sue diverse distribuzioni (distro).

Le distro possono essere free o non-free a seconda se esse sono costituite solamente da software completamente libero e non. Secondo questa classificazione, ad esempio, Ubuntu è considerata una distribuzione GNU/Linux non-free poiché opzionalmente permette di installare software proprietario ed inoltre la versione del kernel Linux incluso in Ubuntu contiene firmware anch'essi proprietari.

Per semplicità di trattazione, in questi appunti, quando ci si riferisce a sistemi Linux in realtà ci si riferisce a sistemi GNU/Linux.

Caratteristiche rilevanti dei sistemi Linux

1. **Sistema multiutente**
   in contrapposizione al sistema "personal", più utenti, individuati da un username univoco, possono interagire con il sistema collegandosi da diversi terminali (TTYs nel gergo Unix).

2. **Sistema multiprogrammato**
   più processi possono eseguire "contemporaneamente".

3. Gestione della memoria virtuale utilizzando meccanismi di paginazione e segmentazione della memoria fisica.

Nelle maggiori distribuzioni "desktop" odierne, è generalmente presente un'interfaccia utente grafica. Al fine di interagire con l'interfaccia a riga di comando, alla quale ci riferiremo d'ora in avanti con il termine shell (il programma che prende in ingresso comandi provenienti dalla tastiera e li passa al sistema operativo per il loro espletamento), queste distribuzioni, di default, mettono a disposizione un emulatore di terminale.

Il filesystem

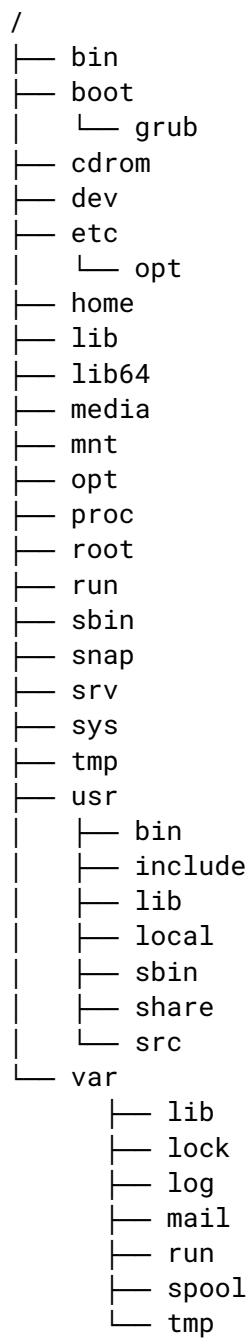

Fig. 1: Ubuntu Filesystem layout generato con il comando tree

Il filesystem di un sistema operativo è un meccanismo usato per l'organizzazione gerarchica, la lettura, la scrittura, l'accesso, la manipolazione e la navigazione dei file all'interno di una memoria di massa (ad es. un disco, per la persistenza dei dati) o anche in RAM.
Il layout del filesystem di Ubuntu è riportato in Fig. 1, che come si può osservare ha la struttura di un albero generico.

La directory radice è "/" dentro la quale sono contenute (logicamente) tutte le altre directory. Da notare che, a differenza del sistema Microsoft Windows dove ogni disco fisico è una radice del filesystem, nei sistemi Unix-like la radice è unica anche con la presenza di più dischi, i quali vengono montati logicamente in punti diversi dell'albero.

Le distribuzioni Debian e Ubuntu aderiscono allo standard per la gerarchia del file system, FHS (Filesystem Hierarchy Standard) il quale definisce l'albero delle directory (di cui una versione semplificata, tratta da una distribuzione Ubuntu, è riportata in Fig. 1). Per lo FHS ogni directory (o cartella nel gergo Windows) ha un preciso scopo:

- **/bin**: adibita a contenere file eseguibili (binaries) per tutti gli utenti;
- **/boot**: adibita a contenere i files statici utilizzati dal boot loader;
- **/dev**: adibita a contenere i files che identificano i dispositivi (devices);
- **/etc**: adibita a contenere i files di configurazione globale del sistema;
- **/etc/opt**: adibita a contenere i files di configurazione per la gli eseguibili contenuti in /opt;
- **/home**: adibita a contenere le home directories di ciascun utente del sistema;
- **/lib** e **/lib64**: adibite a contenere librerie condivise (tra i binari contenuti in /bin) e moduli del kernel;
- **/media** e **/mnt**: mount point per dispositivi removibili e/o filesystem temporanei;
- **/opt**: adibita a contenere pacchetti software aggiuntivi (opzionali);
- **/proc**: adibita a contenere informazioni (sotto forma di files "virtuali") sullo stato del kernel e dei processi;
- **/root**: home directory dell'utente root;
- **/sbin**: adibita a contenere binari di sistema (system binaries);
- **/tmp**: adibita a contenere files temporanei;
- **/usr**: (user system resources) adibita a contenere una sotto-gerarchia di dati condivisibili (tra sistemi FHS compliant) in sola lettura;
- **/var**: adibita a contenere dati variabili (ad es. log di sistema e spool temporanei).

Lo shell prompt

Questi appunti faranno riferimento alla shell Bash (Bourne Again SHell), una tra le shell oggigiorno più diffuse.

```
user@MasterBox:~$
```

Il riquadro precedente, rappresenta il prompt della shell che appare ad es. quando l'emulatore del terminale viene lanciato. In una distribuzione Ubuntu questo può essere fatto utilizzando la combinazione di tasti **Ctrl+Alt+T**.

È il modo che Bash utilizza per dirci che è in attesa di nostri comandi. Il formato della stringa di prompt ha una struttura ben definita, dipendente dalla distribuzione usata, ma comunque configurabile.

- **user** e **MasterBox** sono rispettivamente l'username e l'hostname scelti ed impiegati per la creazione degli esempi mostrati in queste pagine;
- "**@**" e "**:**" sono due separatori
- **~** è un simbolo che rappresenta la home directory, ovvero esso indica che la nostra **current working directory** è la nostra home directory.
  Tipicamente la home directory di un utente è contenuta all'interno della directory **/home** (eccetto la home directory di root che è **/root**).
- **$** indica che l'utente user non è un utente amministratore. Al contrario, il simbolo che compare in presenza di un utente amministratore è **#**.

Le prossime sezioni saranno suddivise a seconda delle azioni che si possono intraprendere, attraverso la shell Bash, all'interno di un sistema operativo Unix-like, presentando gruppi di comandi specifici per compiere tali azioni.

Di ciascun comando verrà fornita una sintetica descrizione insieme ad alcuni esempi di utilizzo. Per avere una visione esaustiva del comando si faccia riferimento al manuale dei sistemi operativi Unix-like. Per visualizzare il manuale di un determinato "comando" si utilizza un altro comando: **man <comando>**.

Qualora il comando non fosse un binario "as is" (ad es. uno tra quelli contenuti nelle directory **/bin** o **/usr/bin**) è possibile che esso sia un comando "built in" all'interno di Bash. In tal caso è possibile visualizzare informazioni dettagliate sul suo utilizzo il comando **help <comando>**.

Si aggiungono alla classificazione dei tipi di comando anche gli **alias**, ovvero ridefinizione di comandi (o loro sequenze) utilizzando un nome (i.e. l'alias). Infine, per conoscere se un comando è un programma "as is", o "built in" in Bash o un alias è possibile utilizzare il comando **type <comando>**.

```
user@MasterBox:~$ type cd
cd is a shell builtin
user@MasterBox:~$ type cat
cat is /bin/cat
user@MasterBox:~$ type ls
ls is aliased to `ls --color=auto'
```

Navigare nel filesystem

- **pwd** è un'abbreviazione dalla lingua inglese di print working directory, ovvero stampa la directory corrente. Il comando mostra il percorso assoluto della directory corrente;

- **cd** è anch'esso un'abbreviazione dalla lingua inglese di change directory, ovvero cambia directory. È un comando built in che consente di cambiare la directory corrente;

- **ls** è un comando utilizzato per elencare i file e le directory presenti nella directory corrente o in una directory specificata. Fornisce un elenco dei nomi dei file e delle directory, consentendo di visualizzare rapidamente i contenuti di una directory specifica. È possibile personalizzare l'output utilizzando opzioni come la visualizzazione di informazioni aggiuntive sui file, l'ordine dell'elenco e il filtraggio dei risultati.

Esplorare il filesystem

- **file** è un comando utilizzato per determinare il tipo di file. Esso accetta come argomento uno o più file e restituisce informazioni sul tipo di file, come il formato, il contenuto o l'estensione associata. Il comando si basa su una serie di criteri per riconoscere il tipo di file, come la struttura interna dei dati o le firme caratteristiche. Queste informazioni possono essere utili per comprendere la natura di un file, ad esempio se si tratta di un file di testo, un'immagine, un file eseguibile o un archivio compresso;

- **less** è un comando utilizzato per visualizzare il contenuto di un file di testo in modo interattivo. Esso consente di scorrere il testo in avanti e indietro, cercare parole specifiche, saltare tra le pagine e fornire una visualizzazione più comoda per la lettura di file di grandi dimensioni. A differenza del comando **cat**, che visualizza l'intero contenuto di un file in una volta sola, less permette di visualizzare il testo in modo graduale, consentendo all'utente di esplorare il contenuto senza dover scorrere l'intero file. Il comando less è particolarmente utile quando si desidera visualizzare e leggere file di grandi dimensioni, come registri di sistema o file di log, in modo efficiente e senza sovraccaricare la memoria del sistema.

Manipolazione di files e directories

- **cp** è un comando utilizzato per copiare file e directory. Consente di creare una copia identica di un file o di un insieme di file all'interno della stessa directory o in una directory di destinazione specificata. Il comando cp può anche essere utilizzato per copiare file in altre directory o rinominarli;

- **mv** è un comando utilizzato per spostare o rinominare file e directory. Esso consente di spostare un file o una directory da una posizione all'altra all'interno del sistema file. Può anche essere utilizzato per rinominare un file o una directory, cambiandone il nome all'interno della stessa posizione;

- **mkdir** è un comando utilizzato per creare una nuova directory. Consente di creare una directory vuota con un nome specificato. Se necessario, è possibile specificare anche il percorso completo della directory da creare. Il comando mkdir è particolarmente utile quando si desidera organizzare i file in diverse cartelle o creare una struttura di directory gerarchica;

- **rm** è un comando utilizzato per rimuovere file e directory. Esso consente di eliminare file o directory specificate dal sistema file. Il comando rm può essere utilizzato per eliminare file in modo permanente, senza la possibilità di recupero. Quando si utilizza il comando rm per rimuovere una directory, è necessario specificare l'opzione **-r** o **-rf** per eliminare la directory e il suo contenuto in modo ricorsivo.

Redirezione

- **cat** è un comando che concatena e visualizza il contenuto di uno o più file di testo. Può essere utilizzato anche per creare nuovi file o aggiungere contenuto a file esistenti;

- **sort** è un comando che ordina le linee di testo di un file in base all'ordine alfabetico o numerico. È possibile specificare opzioni per personalizzare l'ordine di ordinamento;

- **uniq** è un comando che filtra le linee duplicate consecutive all'interno di un file di testo e restituisce solo le linee uniche. Può essere utilizzato in combinazione con il comando sort per rimuovere le linee duplicate non consecutive;

- **grep** è un comando che cerca e filtra le righe di testo che corrispondono a un pattern specificato. Può essere utilizzato per cercare parole o stringhe all'interno di uno o più file di testo;

- **wc** è un comando che conta il numero di linee, parole e caratteri presenti in un file di testo. Può essere utilizzato anche per contare il numero di byte o il valore massimo di una lunghezza di linea;

- **head** è un comando che visualizza le prime N linee di un file di testo. Il numero predefinito di linee mostrate è 10, ma può essere specificato un diverso numero di linee da visualizzare;

- **tail** è un comando che visualizza le ultime N linee di un file di testo. Il numero predefinito di linee mostrate è 10, ma può essere specificato un diverso numero di linee da visualizzare;

- **tee** è un comando che legge l'input da uno o più file o dalla pipeline e lo scrive contemporaneamente sia sulla console che su uno o più file di output. È utile quando si desidera visualizzare l'output mentre si salva anche una copia in un file.

Permessi

- **chown** è un comando che consente di cambiare il proprietario di uno o più file o directory nel sistema. È possibile specificare il nuovo proprietario utilizzando il nome utente o l'identificatore numerico dell'utente. Questo comando richiede privilegi di amministratore (root) per essere eseguito;

- **chgrp** è un comando che consente di cambiare il gruppo di uno o più file o directory nel sistema. È possibile specificare il nuovo gruppo utilizzando il nome del gruppo o l'identificatore numerico del gruppo. Anche in questo caso, il comando richiede privilegi di amministratore per essere eseguito.

Identità

- **sudo** è un comando che consente agli utenti di eseguire altri comandi con i privilegi di amministratore (root) temporaneamente. È ampiamente utilizzato per eseguire operazioni amministrative che richiedono autorizzazioni elevate. L'utente deve inserire la propria password per confermare l'identità prima di poter utilizzare sudo. Questo permette di limitare l'accesso alle funzionalità di amministrazione solo agli utenti autorizzati;

- **su** è un comando che consente di passare ad un altro utente o diventare l'utente root (amministratore) nel sistema. Richiede la password dell'account dell'utente di destinazione o la password di root per autenticarsi. Una volta autenticati, gli utenti possono eseguire comandi con i privilegi dell'utente specificato;

Ricerche nel filesystem

- **updatedb** è un comando che aggiorna il database dei file utilizzato dal comando **locate**. Questo comando esegue una scansione del file system e crea o aggiorna un database contenente informazioni sui file e sulle directory presenti nel sistema. È comunemente utilizzato per migliorare le prestazioni del comando locate, consentendo ricerche rapide di file e directory basate su nomi;

- **locate** è un comando che consente di cercare file e directory nel sistema utilizzando il database creato o aggiornato dal comando **updatedb**. Permette di trovare rapidamente i file corrispondenti a un determinato pattern di ricerca, fornendo il percorso completo dei file trovati.

- **find** è un comando molto versatile che consente di cercare file e directory nel sistema in base a diversi criteri, come il nome del file, il tipo, la dimensione, il timestamp e altro ancora. Può essere utilizzato per eseguire ricerche avanzate e complesse nel file system.

Processi

- **kill** è un comando che consente di inviare un segnale specifico a un processo in esecuzione nel sistema. Di solito viene utilizzato per terminare un processo in modo forzato inviando il segnale **SIGKILL** (-9). È possibile specificare l'ID del processo o il nome del processo per terminarlo;

- **killall** è un comando che termina tutti i processi che corrispondono a un determinato nome di processo. A differenza di kill, che richiede l'ID del processo, killall consente di terminare i processi utilizzando il nome del processo;

- **bg** è un comando che mette in background un processo in pausa o sospeso. Viene utilizzato principalmente con i processi in esecuzione in modalità foreground. Una volta messo in background, il processo continua a eseguire, ma non occupa più la console;

- **fg** è un comando che ripristina un processo in esecuzione in background e lo riporta in modalità foreground. Può essere utilizzato per riprendere l'esecuzione di un processo in background e riportarlo alla console interattiva;

- **jobs** è un comando che elenca i processi in esecuzione in background nell'attuale sessione della shell. Mostra l'elenco dei processi con il loro ID e lo stato (attivo o inattivo);

- **ps** è un comando che visualizza lo stato dei processi in esecuzione nel sistema. Fornisce informazioni dettagliate sui processi, come l'ID del processo (PID), il consumo di risorse, il proprietario del processo e altro ancora;

- **top** è un comando interattivo che fornisce una visualizzazione in tempo reale dei processi in esecuzione nel sistema. Mostra una lista dei processi con informazioni sul loro utilizzo della CPU, della memoria e di altre risorse. È spesso utilizzato per monitorare le prestazioni del sistema e identificare i processi che consumano molte risorse.

L'ambiente

- **printenv** è un comando che stampa l'elenco delle variabili d'ambiente correnti. Le variabili d'ambiente sono variabili speciali utilizzate dal sistema operativo e dalle applicazioni per memorizzare informazioni come il percorso di ricerca degli eseguibili, le impostazioni di configurazione e altro ancora;

- **set** è un comando che visualizza l'elenco di tutte le variabili d'ambiente e le variabili locali definite nella sessione corrente della shell. Oltre alle variabili d'ambiente, set mostra anche altre informazioni di configurazione, come le opzioni della shell e le funzioni definite dall'utente;

- **export** è un comando utilizzato per creare o modificare variabili d'ambiente. Consente di rendere una variabile d'ambiente disponibile a tutti i processi figli della shell corrente. Quando viene utilizzato con un'assegnazione di variabile (ad esempio, **export VAR=valore**), crea una nuova variabile d'ambiente con il nome VAR e il valore specificato;

- **alias** è un comando che consente di creare abbreviazioni o alias per i comandi. Con alias, è possibile assegnare un nome diverso a un comando lungo o complesso, semplificandone l'utilizzo. Gli alias sono specifici della shell in cui vengono definiti e sono validi solo per la sessione corrente della shell. Possono essere utili per creare abbreviazioni personalizzate o per sostituire opzioni predefinite di un comando.

Gestione dei pacchetti

- **apt-get** è un comando che consente di gestire i pacchetti. Fornisce funzionalità per l'installazione, la rimozione, l'aggiornamento e la gestione delle dipendenze dei pacchetti. È in grado di scaricare i pacchetti dai repository di Debian e installarli sul sistema;

- **apt-cache** è un comando utilizzato per visualizzare informazioni sui pacchetti disponibili nei repository. Consente di cercare pacchetti, visualizzare le dipendenze, le descrizioni, le versioni e altre informazioni correlate;

- **dpkg** è un comando di basso livello utilizzato per gestire i pacchetti direttamente nel sistema Debian. È responsabile dell'installazione, rimozione e gestione dei pacchetti Debian e dei file di configurazione associati. Può essere utilizzato per lavorare con pacchetti in formato .deb sia localmente che attraverso repository.

Gestione dei dispositivi

- **mount** è un comando che consente di collegare un sistema di file o un dispositivo di archiviazione alla struttura dei directory del sistema operativo, consentendo l'accesso ai file e alle directory contenuti. Attraverso il comando mount, è possibile specificare il punto di montaggio desiderato e il percorso del dispositivo o del sistema di file da montare;

- **umount** è un comando utilizzato per smontare un sistema di file o un dispositivo di archiviazione precedentemente montato nel sistema. Indicando il percorso del punto di montaggio, umount disconnette il dispositivo o il sistema di file dalla struttura dei directory, rendendo i file inaccessibili fino al successivo montaggio;

- **fsck** è un comando che esegue il controllo e la riparazione dei file system danneggiati o incoerenti. Viene utilizzato per verificare l'integrità dei file system e correggere eventuali errori o incongruenze rilevati. fsck può essere eseguito su diversi tipi di file system e dispone di opzioni specifiche per la gestione delle riparazioni;

- **fdisk** è un comando che consente la gestione delle partizioni di un disco rigido. Attraverso fdisk, è possibile creare, eliminare, ridimensionare e modificare le partizioni del disco. Questo comando offre un'interfaccia testuale per la manipolazione delle partizioni, consentendo di organizzare lo spazio su disco in modo appropriato per l'utilizzo previsto;

- **mkfs** è un comando utilizzato per creare un file system su un dispositivo di archiviazione. Può essere utilizzato per formattare una partizione o un'unità di archiviazione specifica, preparandola per l'utilizzo con un sistema di file specifico. Il comando mkfs offre opzioni per specificare il tipo di file system da creare, nonché altre configurazioni relative al layout del file system;

- **dd** è un comando che consente di copiare dati da un'origine a una destinazione specifica. Può essere utilizzato per creare immagini di disco, clonare dischi, copiare file o blocchi di dati specifici e altro ancora. dd opera a basso livello, copiando i dati esattamente come sono, senza interpretarli o modificarli. Questo lo rende un comando potente ma anche potenzialmente rischioso se utilizzato in modo errato.

Networking

- **ping** è un comando che consente di verificare la connettività di rete tra un host locale e un host remoto utilizzando il protocollo ICMP. Invia pacchetti di controllo al destinatario specificato e riceve le risposte, fornendo informazioni sulla latenza di rete e sulla perdita di pacchetti;

- **traceroute** è un comando che traccia il percorso dei pacchetti di rete da un host locale a un host di destinazione. Utilizzando il protocollo ICMP o UDP, traceroute invia pacchetti con un valore di TTL incrementale, rilevando gli hop intermedi attraverso i quali i pacchetti transitano e misurando i tempi di latenza;

- **ip** è un comando di rete versatile utilizzato per la configurazione e la gestione delle interfacce di rete su un sistema. Fornisce funzionalità per visualizzare e modificare l'indirizzo IP, la configurazione di routing, le tabelle ARP, le interfacce di rete e altro ancora;

- **ss** è un comando che fornisce informazioni dettagliate sulle connessioni di rete attive, i socket e altre statistiche di rete. Può essere utilizzato per visualizzare le connessioni TCP e UDP, le porte aperte, le interfacce di rete, le statistiche di trasferimento e altre informazioni rilevanti per la configurazione e la gestione della rete;

- **netstat** è un comando utilizzato per visualizzare le connessioni di rete attive, le tabelle di routing, le statistiche delle interfacce di rete e altre informazioni sullo stato della rete. Può mostrare le connessioni TCP e UDP, i socket aperti, i processi che utilizzano le porte di rete e altre informazioni di monitoraggio di rete;

- **ftp** è un protocollo di trasferimento di file che consente di copiare file da un host remoto a un host locale o viceversa utilizzando una connessione TCP. Il comando ftp fornisce un'interfaccia interattiva per accedere a server FTP e gestire il trasferimento di file tra i due sistemi;

- **wget** è un comando utilizzato per scaricare file da server Web utilizzando il protocollo HTTP, HTTPS o FTP. Può essere utilizzato per scaricare file singoli o intere directory da un URL specificato. wget supporta il download ricorsivo, la ripresa dei download interrotti, l'autenticazione e altre opzioni avanzate;

- **curl** è un comando che consente di effettuare richieste verso server Web e di visualizzare o elaborare le risposte. Supporta una vasta gamma di protocolli, inclusi HTTP, HTTPS, FTP, SMTP e molti altri. curl può essere utilizzato per scaricare file, inviare dati, eseguire test di connettività e interagire con API Web;

- **ssh** è un protocollo di rete crittografato che consente di accedere e gestire in modo sicuro un host remoto su una rete. Il comando ssh viene utilizzato per stabilire una connessione sicura tramite crittografia e autenticazione, consentendo agli utenti di eseguire comandi e trasferire file in modo sicuro tra host locali e remoti.

Riferimenti

- Linux e il sistema GNU - https://www.gnu.org/gnu/linux-and-gnu.it.html
- The Linux Command Line - http://linuxcommand.org/tlcl.php
- Filesystem Hierarchy Standard - http://www.pathname.com/fhs/pub/fhs-2.3.pdf
- LinuxFilesystemTreeOverview - https://help.ubuntu.com/community/LinuxFilesystemTreeOverview
- Linux man pages - https://linux.die.net/man/