Introduzione a Linux

Modulo 1: cos'e'?

Alessandro Brunengo
Mirko Corosu
INFN – Sezione di Genova

Parte I

Nozioni introduttive
(non specifiche per linux)

Componenti di un PC

- Il processore (CPU: Central Process Unit)
- La memoria volatile (RAM: Random Access Memory)
- La memoria permanente (NV-RAM, EPROM, HD, Floppy, CD-ROM, ...)
- I canali di comunicazione interna (BUS RAM, BUS PCI, BUS AGP, ...)
- I canali di I/O (seriale, parallela, USB, ...)
- I controllers (PCI, AGP, IDE, SCSI, ..)
- La scheda madre
- Le schede periferiche (scheda video, scheda audio, scheda di rete, ...)

Il BIOS

- Il BIOS e' costituito da codice eseguibile e dati, contenuti in una EPROM sulla scheda madre del computer
- Alla accensione la CPU esegue il codice del BIOS, che configura il computer per prepararlo al boot
- E' possibile accedere (usualmente tramite tasti DEL o F2) ad un configuratore che permette di modificare la configurazione del BIOS

Dischi rigidi

I dischi rigidi sono costituiti da

- piatti di metallo o plastica, sulle cui superfici e' depositato un materiale magnetizzabile
- testine di lettura e scrittura, posizionate su bracci ruotanti, sensibili alla magnetizzazione dei piatti
- elettronica integrata, che permette di ricevere comandi dal controller e di eseguirli facendo ruotare i piatti e posizionando le testine in modo da leggere o scrivere determinate porzioni della superficie dei piatti

Formattazione fisica dei dischi

E' la divisione del disco in **tracce**, **settori** e **cilindri.** L'operazaione e' usualmente effettuata dal costruttore.

- traccia: anello circolare su una superficie di un piatto
- settore: porzione di traccia in grado di ospitare 512 bytes
- cilindro: insieme delle tracce di tutti i piatti equidistanti dal centro di rotazione

Formattazione logica dei dischi

- La formattazione logica dei dischi consiste nella costruzione di un file system sul disco, che deve essere gia' stato formattato a livello fisico
- La formattazione logica viene eseguita tramite un programma opportuno (*diskdruid*, *mkfs*, ...), all'atto della installazione o successivamente

Partizioni

- Il partizionamento consiste nella suddivisione di un disco in parti che appaiono al sistema come dischi diversi
- Esistono tre tipi di partizioni: primarie, estese, logiche
- Il partizionamento viene eseguito tramite programmi (fdisk, partition magic, diskdruid, ...) disponibili anche all'atto della installazione del sistema operativo

Partizioni primarie

- Su un HD possono essere definite al massimo 4 partizioni primarie
- Molti sistemi operativi (DOS, W9x) possono partire solo se installati su partizioni primarie
- Solo una partizione primaria puo' essere visibile ed attiva (ma W2000/WXP e Linux possono accedere a tutte le partizioni primarie)

Partizione estesa

- La partizione estesa e' una partizione primaria capace di contenere altre partizioni (logiche)
- La partizione estesa non puo' contenere dati, ma solo partizioni estese
- In una partizione estesa possiamo definire un qualunque numero di partizioni logiche
- Solo una partizione primaria puo' essere estesa

Partizioni logiche

- Le partizioni logiche possono esistere solo all'interno di una partizione estesa
- Possono contenere dati per ogni sistema operativo, o possono essere partizioni di boot per i soli sistemi capaci di partire da partizioni logiche (WNT/W2000/WXP, Linux,...)

Boot sector e MBR

- Il primo settore (512 bytes) di ogni partizione e' detto boot-sector della partizione
- Il boot sector di una partizione non contiene dati; puo' contenere codice eseguibile se la partizione e' *bootable*
- Il primo settore dell'HD non appartiene a nessuna partizione, ed e' detto Master Boot Record
- l'MBR contiene informazioni sulle partizioni del disco, e puo' contenere codice eseguibile (il Master Boot Program)

Sistemi operativi

- Un sistema operativo e' un programma che ci permette di utilizzare l'hardware del computer
- E' dotato di una interfaccia utente che accetta comandi ed esegue operazioni in funzione di questi
- Interagisce con l'hardware attraverso parti di programma chiamati *drivers*

OS single e multi user

- Un sistema operativo *single user* e' un sistema che e' capace di eseguire comandi forniti da un unico utente (DOS, W9*)
- Un sistema operativo *multi user* e' capace di eseguire comandi dati da piu' utenti (WNT, W2000, linux)

OS single e multi task

- Un sistema operativo *single task* e' capace di eseguire una funzione (un programma) per volta (DOS)
- Un sistema operativo *multi task* e' capace di eseguire piu' funzioni "*contemporaneamente* " (W9*, W2000, linux)

File System

- Un file system e' una struttura sul disco grezzo che permette archiviare dati sull'HD raggruppati in unita' che chiamiamo files
- Generalmente un file system organizza il contenuto dei dati in una struttura piatta o gerarchica di files e directories (o folders, o cartelle)
- Il file system gestisce lo spazio disco occupato e disponibile, le informazioni sui nomi dei files e delle directories, le informazioni sulla collocazione fisica dei dati

FAT (File Allocation Table)

- FAT e' il file system utilizzato da DOS, Windows 3.x e Windows 95. Puo' essere utilizzato da molti altri sistemi operativi (anche linux)
- FAT alloca lo spazio disco in clusters costituiti da un fissato numero di settori, ed utilizza una FAT per registrare quali cluster sono utilizzati, e quali cluster appartengono a quali file
- Limiti: dischi da 2 GB, e 65525 clusters.

FAT32

- FAT32 e' una versione migliorata di FAT. E' utilizzato da W95 seconda release, W98, W2000, WXP (NON WNT!), e linux
- Estende i limiti di FAT fino a dischi di 2 TB, ed aumenta il numero di clusters utilizzabili, rendendo possibile utilizzare clusters piu' piccoli (migliore efficienza)

NTFS (New Technology File System)

- File system utilizzato da WNT e W2000 (non da linux)
- Utilizza una struttura interna grossa, replicata, e fa uso di clusters di dimensione qualunque. Capace di identificare *bad sectors*
- Efficiente per dischi di grosse dimensioni, inadatto sotto i 500 MB

EXT2 (Extended File System 2)

- Gestione gerarchica in files e directories
- Limiti: file system fino a 2 TB, files fino a 2 GB (limite superato nelle versioni recenti di linux)
- Utilizza una struttura per descrivere i files: *inode*, ed allocazione a blocchi (1-4 KB)
- l'inode contiene informazioni su nome del file, proprietario, data di ultimo accesso, modifica delle caratteristiche, modifica del contenuto, collocazione fisica del file, protezioni

EXT2 (cont.)

- Ext2 ha diversi tipi di files (file ordinario, directory, link, device, fifo, socket...)
- Il file di tipo directory contiene una lista di associazioni filename-inode relative ai file contenuti nella directory
- Il file di tipo link consiste in un sinonimo per un dato inode
- I files di tipo device permette alle applicazioni operazioni di I/O su device (schede, linea seriale, porta parallela, ...) con la stessa sintassi dell'I/O su files ordinari

EXT3

- Versione evoluta di Ext2, con l'aggiunta di *journaling*, cioe' di un meccanismo che registra lo stato delle operazioni in corso su un dato file
- Piu' affidabile in caso di system crash, piu' veloce nel controllo e ripristino della consistenza interna del file system

Parte II

Cos'e' linux

Linux

- Linux e' un sistema operativo
- E' nato come realizzazione di un sistema operativo di tipo Unix per processori Intel x86 (poi esteso ad altri processori: Alpha, PPC, Sun SPARC, ...)
- In senso esteso, Linux e' costituito da un programma centrale (il kernel), i drivers e gli applicativi
- Piu' propriamente, Linux e' il kernel

Il kernel

- E' il programma eseguito inizialmente
- Gestisce la memoria fisica e virtuale
- Gestisce le componenti hardware tramite i drivers
  Gestisce piu' utenti sul medesimo sistema
- Gestisce piu' processi attraverso lo scheduler
- E’ un sistema modulare, cioe’ e’ capace di caricare dinamicamente (al volo) parti di kernel o drivers che si rendano necessarie
- Fornisce una interfaccia (terminale alfanumerico o interfaccia grafica) per ricevere comandi e fornire risposte

I driver

- Il kernel utilizza pezzi di programma, detti driver, per l'accesso all'hardware, o a determinate funzioni
- I driver non fanno propriamente parte del kernel, ma vengono caricati staticamente o dinamicamente per estendere le funzionalita' del kernel
- L'utilizzo di un driver puo' essere deciso sia autonomamente dal kernel che, in caso di necessita', dall'amministratore del sistema

Il resto...

- Tutto quello che non e' kernel e drivers, sono applicativi
- Gli applicativi sono programmi che svolgono funzioni, utilizzando il kernel per interagire con l'hardware tramite i drivers

Utenti e gruppi

- Linux e' un sistema multi-utente
- Esiste un database che definisce il nome e le caratteristiche degli utenti del sistema (solitamente i file /etc/passwd e /etc/shadow)
- E' possibile definire gruppi ed assegnare ad ogni utente l'appartenenza ad uno o piu' gruppi
- Esiste un utente speciale, root, che puo' fare qualunque cosa (detto anche superuser)

Processi

- Un qualunque programma in esecuzione e' un processo
- Un processo e' costituito da un insieme di istruzioni (il programma) ed un insieme di caratteristiche (la priorita', il proprietario, il tempo di esecuzione, la current working directory, ...)
- Il kernel alloca una zona di memoria per ogni processo, e tiene una lista dei processi attivi
- Linux ha un meccanismo, lo scheduling, che permette di eseguire diversi processi *contemporaneamente*

Lo scheduler

- Quando il kernel parte, esegue un processo particolare chiamato scheduler
- Lo scheduler prende il controllo della CPU ad intervalli regolari, e decide quale processo deve essere eseguito, in base a criteri di priorita' e di tempo di attesa
- Il kernel di linux e' pre-emptive, cioe' lo scheduler interrompe sempre l'esecuzione di qualsiasi processo per decidere chi sara' il successivo ad essere eseguito

Memoria virtuale e swap

- Linux puo' utilizzare una porzione di spazio disco (swap) per aumentare lo spazio di memoria disponibile
- L'area di swap solitamente risiede in una o piu' partizioni dedicate allo scopo
- La somma di memoria fisica e swap si chiama memoria virtuale
- Quando la richiesta di memoria complessiva supera la memoria fisica disponibile, parte dei dati presenti in memoria viene spostata nella area di swap

Terminali

- Linux e' nato pensando ad una interazione tra utente e sistema attraverso linee di comando
- A questo scopo linux utilizza una applicazione che si chiama terminale che, attraverso un programma (la shell) e' capace di ricevere input e comunicare messaggi in output
- Il terminale puo' operare su un monitor alfanumerico o su una finestra di una interfaccia grafica

Console

- Un terminale speciale e' la console
- La console e' il terminale che viene utilizzato dal kernel al caricamento del sistema, per comunicare all'utente cosa accade, e rimane sempre attiva
- Linux rende disponibili molteplici console virtuali, anche in presenza di interfaccia grafica
- Le console virtuali, usualmente 6, sono attivabili tramite la combinazione di tasti ALT^F1... ALT^F6; la combinazione ALT^F7 riattiva l'interfaccia grafica (se disponibile)

Parte III

Il file system

I nomi di file e directory

- In linux (in unix) il sistema di directory e di file ha una forma gerarchica
- Il punto iniziale e' la “root directory”, indicata con il simbolo ***/***
- Ogni file o directory ivi contenute si chiamano /<flename> o /<dirname>
- La struttura gerarchica e' organizzata ad albero, cioe' in modo che ogni directory puo' contenere al suo interno sia file che altre directory

I nomi di file e directory (cont.)

- Un file viene identificato specificando l'elenco delle directory che si devono attraversare, a partire dalla root, per raggiungerlo; il carattere utilizzato per separare i nomi delle directory tra loro e' **/**
- In generale, un file in una directory arbitraria ha nome **/dir/subdir/filename**
- Il nome di una directory puo' opzionalmente essere specificato lasciando indicato lo **/** finale:

  /dir/subdir/ in luogo di **/dir/subdir**

Le directory . e ..

- Ogni directory creata contiene due directory, di nome . e ..
- La directory . identifica la directory stessa: /dir/subdir/. identifica la directory /dir/subdir
- La directory .. identifica la parent directory: /dir/subdir/.. identifica la directory /dir
- Nota: la directory /.. fa eccezione; poiche' la / non ha una parent directory, /.. identifica la /

Nomi completi e relativi

- Il nome di un file specificato a partire dalla root si dice nome completo
- E' possibile identificare un file mediante un nome relativo, cioe' utilizzando come "punto di partenza" il nome di un'altra directory: ad esempio, se ci troviamo in /dir1/dir2, il file /dir1/dir2/dir3/file puo' essere identificato semplicemente con il nome dir3/file
- I nomi relativi sono caratterizzati dal fatto che non iniziano con **/**

Nomi completi e relativi (cont.)

- Tramite l'utilizzo delle directory .., e' possibile utilizzare un nome relativo anche per risalire l'albero delle directory: se siamo in /dir1/dir2, il file /dir1/dir3/dir4/file si potra' identificare con il nome relativo ../dir3/dir4/file

Proprieta' dei file (ext2)

- Ogni file o directory appartenente a file system di tipo ext2 (o ext3) dispone delle seguenti caratteristiche:
  - utente e gruppo proprietario del file
  - 3 livelli di permessi
  - data di creazione, modifica, accesso
  - tipo del file

Ownership

- Ogni file (o directory) ha la caratteristica di appartenere ad un utente ed a un gruppo
- Queste caratteristiche vengono definite al momento della creazione del file, ma possono essere cambiate in seguito
- Normalmente il proprietario e' l'utente che ha creato il file, il gruppo e' quello a cui appartiene la directory in cui il file viene creato.

Permission

In Linux sono definiti tre permessi:

- lettura (r): il file puo' essere letto (o copiato), la directory puo' essere listata
- scrittura (w): il file puo' essere modificato (non cancellato), nella directory e' possibile creare o cancellare file o subdirectory
- esecuzione (x): il file puo' essere eseguito come programma, la directory puo' essere attraversata (per navigare nel sottoalbero a valle di questa)

Permission (cont.)

I tre permessi (rwx) possono essere dati o negati a tre diversi insiemi di utenti:

- all'utente proprietario
- agli utenti appartenenti al gruppo a cui appartiene il file (o la directory)
- a tutti gli altri utenti conosciuti dal sistema

Permission speciali

- E' possibile definire un file eseguibile in modo che, alla sua esecuzione, il processo relativo venga eseguito non come l'utente (o il gruppo) che ha eseguito il programma, ma come l'utente (o il gruppo) proprietario del programma (permission set-UID-bit): esempio tipico e' il comando passwd
- E' possibile definire le permission di una directory in modo che tutti abbiano accesso in scrittura, ma solo chi il proprietario di un file contenuto nella directory avra' l'accesso per rimuoverli (permission stiky-bit): esempio tipico e' la directory /tmp

Informazioni di data

Per ogni file o directory viene conservata l'informazione di tempo relativa a:

- ultima modifica delle caratteristiche
- ultima modifica del contenuto
- ultimo accesso al contenuto

File speciali (ext2)

Oltre ai file ed alle directory, esistono altri tipi di file:

- link
- link simbolici
- device file
- ...

Link e link simbolici

- Un (hard) link e' un sinonimo di un file, cioe' un "nome" che viene associato ad un file gia' esistente; la reale rimozione del file si ha solo quando vengono rimossi tutti i link a quel file
- Un link simbolico (soft link) e' un file di tipo speciale, che punta ad un altro file esistente; la rimozione del file originale provoca la cancellazione dei dati, ed il link puntera' verso un file non piu' esistente
- A differenza degli hard link, i soft link possono puntare anche a directory, o a file e directory di altri file system

Device file

- Linux gestisce le comunicazioni con l'hardware in modo da rendere uniforme l'accesso ai vari device; realizza questo simulando l'accesso ai device come accesso a file ordinari
- I files che in realta' corrispondono a devices permettono la realizzazione di questo meccanismo
- Usualmente i device files risiedono in /dev

Accesso ad altri file system

- Ogni partizione su cui e' definito un file system viene visto come una struttura gerarchica che parte dalla root del file system
- Al boot, linux vede solo il file system su cui risiede il sistema
- L'accesso ad altri file system (su altre partizioni) viene fatto mediante un mount point

Il mount point

- Un mount point e' una directory ordinaria, vuota, che il kenrel associa alla root di un altro file system
- Se ad esempio, abbiamo un file system la cui root contiene il file **file1** e la directory **dir1**, e montiamo questo file system nella directory **/mnt/tmp**, il contenuto del nuovo file system sara' accessibile con i nomi:

  /mnt/tmp/file1
  /mnt/tmp/dir1

- In questo modo, tutto lo spazio disco utilizzabile sara' visto come un unico file system virtuale che parte dalla root del file system in cui e' installato il sistema operativo

Directories di sistema

- / la root directory
- /boot contiene il kernel ed i boot files
- /bin e /sbin contengono eseguibili
- /lib contiene le librerie di sistema
- /etc contiene i files di configurazione
- /usr contiene il software standard
- /var contiene i log files e le spool directory
- /root la home directory di root
- /home le home directories degli utenti

Directories di sistema (cont.)

- /dev contiene i device files
- /proc contiene files virtuali che puntano alle strutture dati del kernel

Parte IV

Il boot

Il boot

- All'accensione la CPU esegue le istruzioni nella ROM del BIOS
- l'ultima parte delle istruzioni del BIOS e' la *boot routine*, che causa l'esecuzione del master boot program dell'MBR del disco definito come bootable (o, se non specificato, del primo disco fisico)

Il boot (cont.)

- Il MBP legge la partition table ed esegue il codice contenuto nel boot sector della partizione primaria attiva
- Esistono MBP (lilo, grub, NTLDR,...) capaci di gestire piu' sistemi operativi collocati in diverse partizioni; in questi casi e' possibile scegliere la partizione (ovvero il sistema operativo) da far partire: il MBP eseguira' il boot sector della partizione scelta

Il boot (cont.)

- Il codice nel boot sector a sua volta carica in memoria ed esegue il sistema operativo (il kernel)
- Diversi sistemi possono partire solo da una partizione attiva (DOS, W9*)
- Linux, W2K, WXP possono partire direttamente da una partizione logica

Il boot (cont.)

- Il boot loader carica in RAM il kernel e lo esegue
- Il kernel alloca in memoria le strutture di dati necessarie al funzionamento del sistema, identifica l'hardware e carica i drivers necessari per utilizzarlo, crea un processo che si chiama init e cede la mano ad una funzione che si chiama scheduler
- init prende la mano, legge i file di configurazione e fa partire tutti i processi che la configurazione stabilisce
- alla fine viene eseguito un processo che si chiama getty (o una interfaccia grafica) per attendere un login