Corso di Laboratorio di Sistemi Operativi

A.A. 2009-2010

-

Fabio Buttussi

Informazioni generali

- Orario: Mercoledì dalle 8:30 alle 10:15
- Docente: Fabio Buttussi
- Home page del corso:
  `http://users.dimi.uniud.it/~fabio.buttussi/labso0910/index.html`
- Orario di ricevimento:
  Su appuntamento (`fabio.buttussi@dimi.uniud.it`)

Programma

Il programma del corso si suddivide in tre parti:

- **la shell UNIX**;
- **basi di linguaggio C**;
- **programmazione di sistema**;

Sono inoltre disponibili online 4 approfondimenti facoltativi sulla **scelta, installazione (su macchina reale o virtuale) e configurazione di un sistema GNU/Linux**.

Bibliografia

Testi adottati:

- **B.W. Kernighan, D.M. Ritchie. "Linguaggio C", Jackson, 2a edizione, 1989.**

- K. Haviland, D. Gray, B. Salama, "UNIX System Programming", Addison Wesley, 2a edizione, 1999.

Testi di consultazione o approfondimento:

- P. Cornes, The Linux A-Z, Prentice Hall, 1997.

- G. Glass, K. Ables. "UNIX for Programmers and Users". Prentice Hall, 2a edizione, 1999.

Modalità di superamento del corso

- svolgimento di **tre progetti** (uno per ogni parte del corso);
- gli approfondimenti facoltativi non sono oggetto di verifica;
- i progetti vanno svolti in gruppi di 2 persone o individualmente (i gruppi non devono necessariamente essere gli stessi per ogni progetto);
- i progetti vanno inviati al docente via e-mail entro il **24 settembre 2010** (entro il 31 agosto 2010 se si desidera registrare l'esame nella sessione di settembre);
- ad ogni progetto verrà assegnato un voto in trentesimi;
- i voti concorreranno alla formulazione del voto finale di Sistemi Operativi.

Organizzazione delle lezioni

- mini-riassunto delle lezioni precedenti (5-10 min);
- introduzione e spiegazione dei nuovi argomenti (30-40 min);
- consegna degli esercizi (5-10 min);
- svolgimento individuale degli esercizi;
- discussione collettiva degli esercizi (al completamento di ciascun esercizio da parte di quasi tutti).

La Shell di Unix

- La parte del sistema operativo Unix dedita alla gestione dell'interazione con l'utente è la **shell**, ovvero, un'**interfaccia a carattere**:
  - l'utente impartisce i comandi al sistema digitandoli ad un apposito **prompt**;
  - il sistema stampa sullo schermo del terminale eventuali messaggi all'utente in seguito all'esecuzione dei comandi, facendo poi riapparire il prompt, in modo da continuare l'interazione.

- Versioni moderne di Unix forniscono **X-Windows**, un'interfaccia grafica (a finestre), che consente di inviare comandi tramite menu, utilizzando un mouse.

- **X-Term** è un emulatore di terminale che gira sotto X-Windows, fornendo localmente un'interfaccia a carattere. **Konsole** è un emulatore di terminale per KDE.

Tipi di Shell

sh	Bourne shell
bash	Bourne again shell
csh	C shell
tcsh	Teach C shell
ksh	Korn shell

Quando viene invocata una shell, automaticamente al login o esplicitamente:

1. viene letto un file speciale nella home directory dello user, contenente informazioni per l'inizializzazione;
2. viene visualizzato un **prompt**, in attesa che l'utente invii un comando;
3. se l'utente invia un comando, la shell lo esegue e ritorna al punto 2; ad esempio, `echo $SHELL` stampa sullo schermo del terminale il percorso della shell di login, mentre il comando `bash` invoca la shell bash.

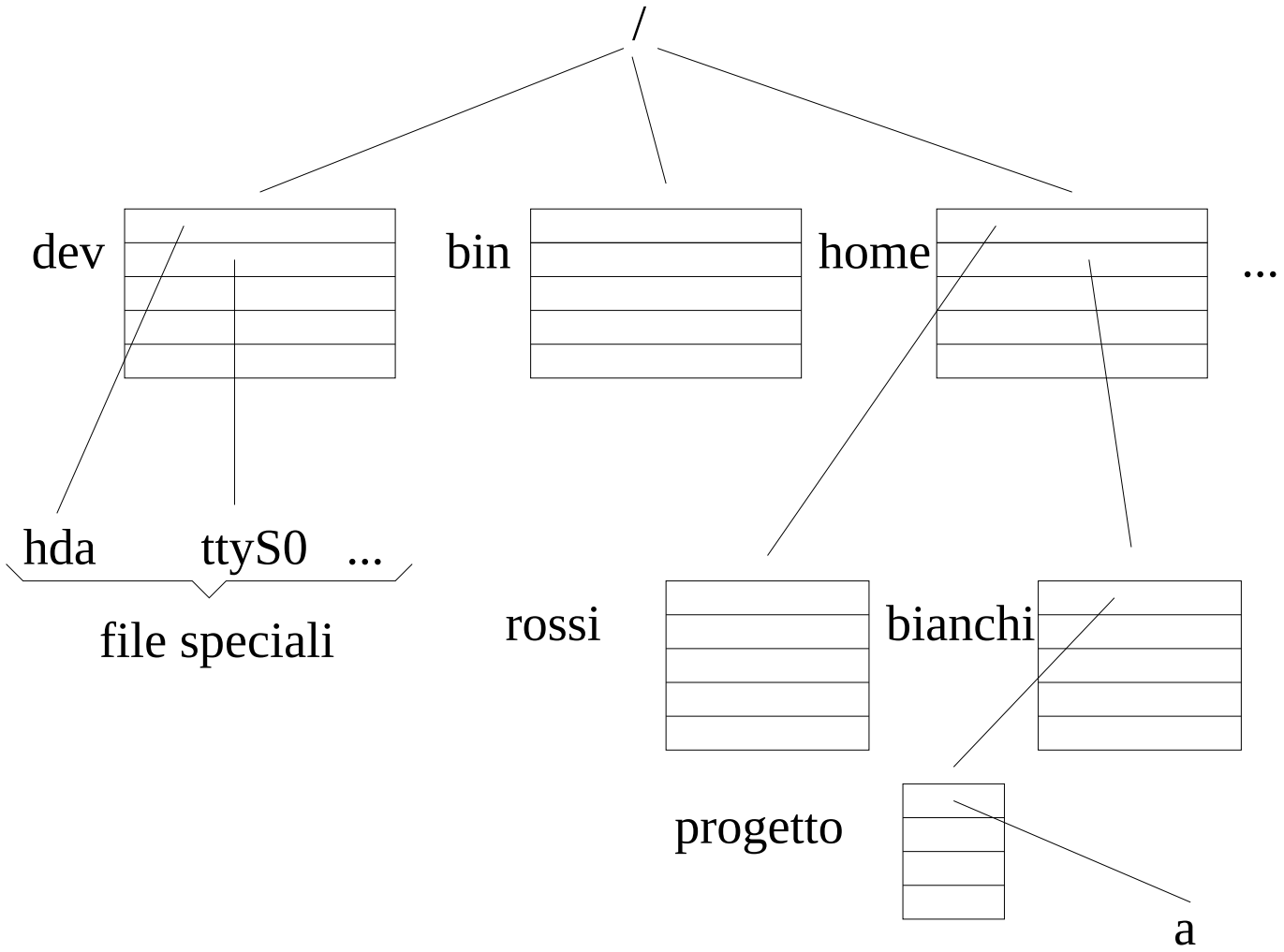
Per terminare la shell si possono usare i seguenti metodi:

- premere Ctrl-D;
- digitare i comandi `logout` o `exit`.

File in Unix

- Ordinari
- Directory
- Speciali

I file sono organizzati in una struttura gerarchica ad albero:



Il pathname

Ci si riferisce ai file tramite il

pathname { assoluto (rispetto a root /)
relativo (rispetto alla directory corrente)

Esempio:

(assoluto) `/home/bianchi/progetto/a`

(relativo) `progetto/a` (supponendo di trovarsi nella directory `/home/bianchi`)

- Present working directory:
  `> pwd`
  `/home/bianchi`
- Change directory:
  `> cd /bin` (`cd` senza argomenti sposta l'utente nella sua home directory)
- Per spostarsi nella directory "madre":
  `> cd ..`
  dove `..` è l'alias per la directory "madre".
- `> pwd`
  `/home/bianchi`
  `> cd ./progetto` (dove `.` è l'alias per la directory corrente)
  `> pwd`
  `/home/bianchi/progetto`

Comandi per manipolare file e directory

- Listing dei file:
  `> ls`
  `> ls -l`
  `> ls -a`
  `> ls -al`
  `> ls -l /bin`
  `> ...`

- Creazione/rimozione di directory:
  `> mkdir d1`
  `> rmdir d1`

- Copia il file `f1` in `f2`:
  `> cp f1 f2`

- Sposta/rinomina il file `f1` in `f2`:
  `> mv f1 f2`

- `cp` e `mv` come primo argomento possono prendere una lista di file; in tal caso il secondo argomento deve essere una directory:
  `> cp f1 f2 f3 d1` (copia `f1`, `f2`, `f3` nella directory `d1`)

Un esempio d'uso del comando `ls`

Eseguendo il comando `ls -l /bin` si ottiene il seguente output:

```
...
lrwxrwxrwx    1 root     root            4 Dec  5  2000 awk -> gawk
-rwxr-xr-x    1 root     root         5780 Jul 13  2000 basename
-rwxr-xr-x    1 root     root       512540 Aug 22  2000 bash
...
```

da sinistra a destra abbiamo:

1. tipo di file (- file normale, `d` directory, `l` link, `b` block device, `c` character device),
2. permessi,
3. numero di hard link al file,
4. proprietario del file,
5. gruppo del proprietario del file,
6. grandezza del file in byte,
7. data di ultima modifica,
8. nome del file.

I permessi dei file

Unix è un sistema **multiutente**. Per ogni file ci sono 4 categorie di utenti: **root**, **owner**, **group**, **world**
L'amministratore del sistema (root) ha tutti i permessi (lettura, scrittura, esecuzione) su tutti i file. Per le altre categorie di utenti l'accesso ai file è regolato dai permessi:

```
> ls -l /etc/passwd
-rw-r--r--    1 root     root          981 Sep 20 16:32 /etc/passwd
```

Il blocco di caratteri `rw-r--r--` rappresenta i permessi di accesso al file. I primi 3 (`rw-`) sono riferiti all'owner. Il secondo blocco di 3 caratteri (`r--`) è riferito al group e l'ultimo blocco (`r--`) è riferito alla categoria world.

La prima posizione di ogni blocco rappresenta il permesso di **lettura** (`r`), la seconda il permesso di **scrittura** (`w`) e la terza il permesso di **esecuzione** (`x`). Un trattino (-) in una qualsiasi posizione indica l'assenza del permesso corrispondente.

N.B.: per "attraversare" una directory, bisogna avere il permesso di esecuzione su di essa.

Il comando `chmod`

L'owner di un file può cambiarne i permessi tramite il comando `chmod`:

- `> chmod 744 f1` (imposta i permessi del file `f1` a `rwxr--r--`)
  Infatti: `rwxr--r--` $\rightsquigarrow$ `111 100 100` $\rightsquigarrow$ `7 4 4` (leggendo ogni gruppo in ottale)

- `> chmod u=rwx,go=r f1` (produce lo stesso effetto del comando precedente)
  dove `u` rappresenta l'owner, `g` il gruppo e `o` il resto degli utenti (world)
  Inoltre:
  `+` aggiunge i permessi che lo seguono,
  `-` toglie i permessi che lo seguono,
  `=` imposta esattamente i permessi che lo seguono.
  Quindi l'effetto di `chmod g+r f1` è in generale diverso da `chmod g=r f1`.

Ulteriori comandi

- Visualizzazione del contenuto di un file:

  ```
  > cat f1
  > more f1
  > less f1
  > tail f1
  > head f1
  ```

- Consultazione del manuale on-line:

  * Sezione 1 : comandi
  * Sezione 2 : system call
  * Sezione 3 : funzioni di libreria

  ...

  ```
  > man passwd
  > man -a passwd
  > man -s2 mkdir
  > man man
  ```

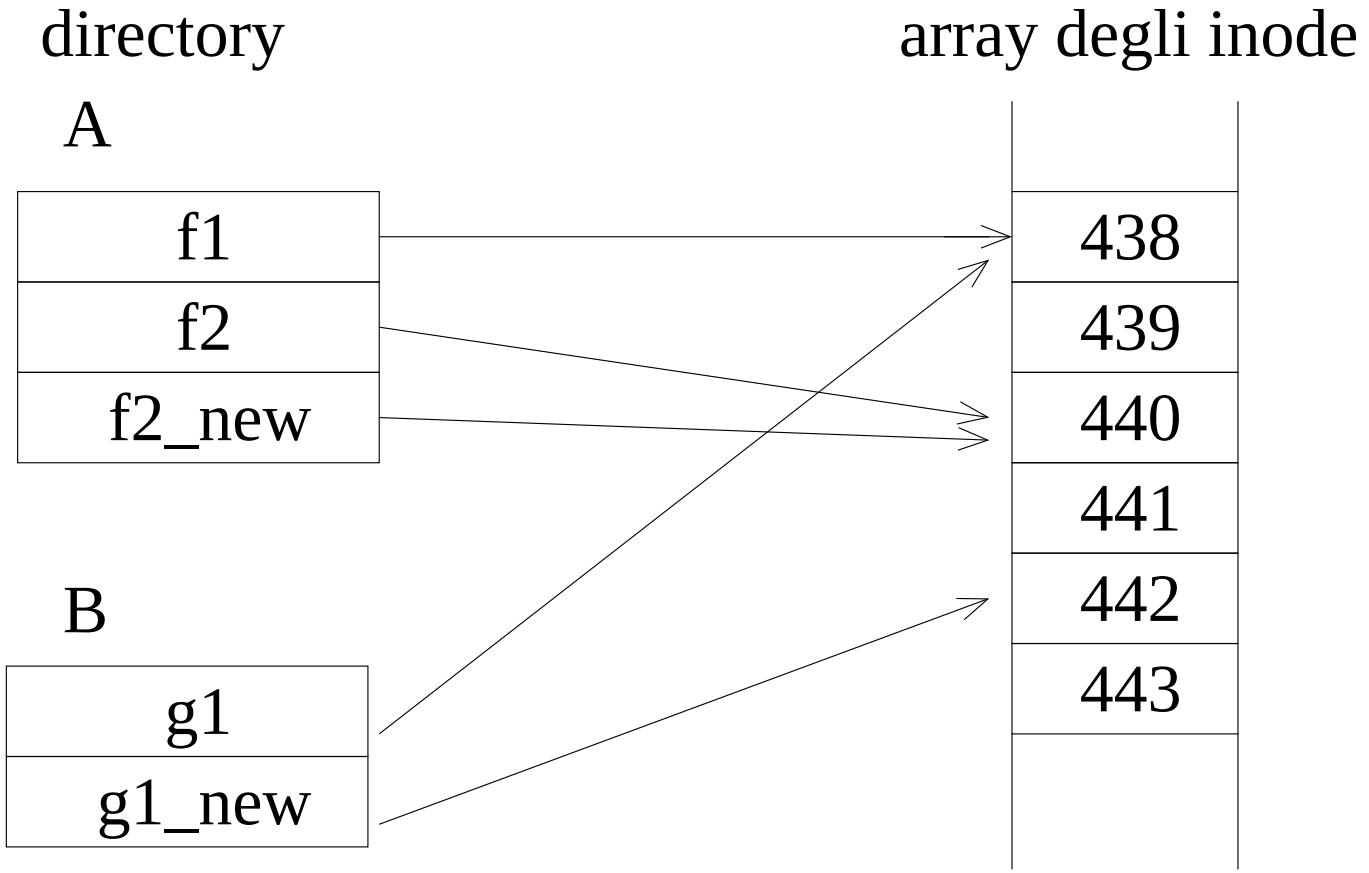
Inode e link

In Unix, ad ogni file corrisponde un numero di **inode**, che è l'indice in un array memorizzato su disco.

Ogni elemento dell'array contiene le informazioni relative al file (data di creazione, proprietario, dove si trova il contenuto del file su disco, . . .).

Le directory sono tabelle che associano nomi di files a numeri di inode.

Ogni entry di una directory è un **link**.



Link e link simbolici

- Creazione di **link (hard)**:
  `> ln f2 f2_new`
  il file f2_new ha lo stesso inode di f2
  `> ln f1 g1`

- Creazione di un **link simbolico**:
  `> ln -s g1 g1_new`
  un link simbolico è un tipo di file speciale in Unix; g1_new è un file di testo che contiene il pathname di g1

Esercizi

- Esplorate il vostro file system. Qual è il pathname della vostra home directory?
- Visualizzate i file della vostra home directory ordinati in base alla data di ultima modifica.
- Che differenza c'è tra i comandi `cat`, `more`, `tail`?
- Un link simbolico può puntare ad un altro link che a sua volta punta ad un file?
  Se è possibile, c'è un limite al numero di link simbolici che si possono avere in catena?
  Qual è lo svantaggio dei link simbolici rispetto ai link hard?
- Trovate un modo per ottenere l'elenco delle subdirectory contenute ricorsivamente nella vostra home.
- Trovate due modi diversi per creare un file.
- I seguenti comandi che effetto producono? Perché?

```
> cd
> mkdir d1
> chmod 444 d1
> cd d1
```