Laboratorio di sistemi operativi
A.A. 2010/2011
Gruppo 2
Gennaro Oliva
3
La shell di Unix
(parte seconda)

Canali di comunicazione

I programmi dispongono di 3 canali di comunicazione standard

0
Standard input

1
Standard output

2
Standard error

Canali di comunicazione

0
Standard
input

2
Standard
error

1
Standard
output

```
potter@pocho: ~ (as potter)
potter@pocho:~$ tail -n 2 /etc/passwd /etc/shadow
==> /etc/passwd <==
Debian-gdm:x:119:131:Gnome Display Manager:/var/lib/gdm3:/bin/false
postgres:x:1004:1004:PostgreSQL:/mnt/PostgresPlus/8.4SS:/bin/sh
tail: impossibile aprire "/etc/shadow" per la lettura: Permission denied
potter@pocho:~$
```

Reindirizzamento canali standard

- La shell consente il reindirizzamento dei 3 canali all'interno di file con la sintassi :

```
$ comando operatore file
```

- Operatori disponibili:
  - > per lo standard output in modalità truncate

    ```
    $ comando > file
    ```
  - >> per lo standard output in modalità append

    ```
    $ comando >> file
    ```
  - 2> per lo standard error in modalita truncate

    ```
    $ comando 2> file
    ```
  - 2>> Per lo standard error in modalità append

    ```
    $ comando 2>> file
    ```
  - < per lo standard input

    ```
    $ comando < file
    ```

Reindirizzamento canali standard

- Più reindirizzamenti possono essere effettuati nella stessa linea di comando

```
$ comando  < in > out 2> err
```

- Per reindirizzare standard output e standard error nello stesso file si può utilizzare l'operatore >&

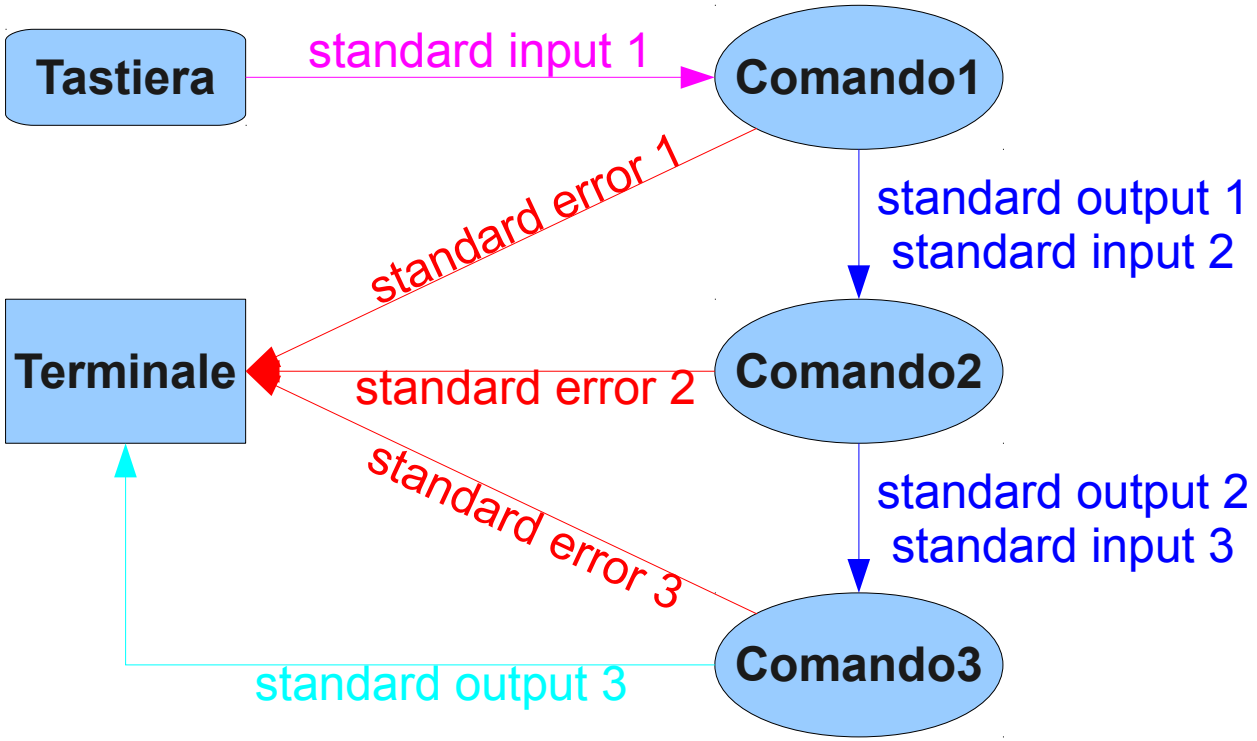
```
$ comando >& out-err
```

Programmi di utilità generica

- wc è un programma che permette di contare caratteri, parole e linee
- sort ordina le linee di un file di testo
- uniq elimina linee ripetute consecutive da un file
- diff confronta due file di testo
- find ricerca file
- wc, sort, uniq così come cat, head, tail, less e quasi tutti i comandi Unix in mancanza di specifiche diverse fornite a linea di comando, processano lo standard input

Pipeline

- Canale di redirezione che collega lo standard output di un programma allo standard input di un altro
- Utilizza il simbolo '|'
- Sintassi: comando1 | comando2 | ...



Esempi di utilizzo della pipeline

- Visualizza i file contenuti in /etc in modo interattivo

```
$ ls /etc | less
```

- Ordina i processi in esecuzione sul sistema in base al nome utente

```
$ ps -ef | sort | less
```

- Visualizza i file in /etc e le relative informazioni ordinandoli per dimensione

```
$ ls -la /etc | sort -k 5 -n | less
```

Esercizio 7

- Conta il numero di linee contenute nel file /etc/passwd
- Visualizza le righe del file /etc/passwd in ordine alfabetico
- Nella pagina di manuale di sort trova l'opzione per l'ordinamento numerico
- Utilizzare i comandi wc, sort, cat, uniq per processare lo standard input (per chiudere lo standard input si utilizzi ctrl+D)
- Conta i file con estensione .log nella directory /var/log
- Ordina i file nella tua directory in base al numero di link e visualizza l'elenco con less

Variabili

- Permettono di memorizzare stringhe ed interi
  - nomi case sensitive
  - possono contenere lettere cifre e underscore '_'
  - non possono iniziare con una cifra
- Definizione e assegnazione:

```
a=3

msg=”Benvenuti alla”
```

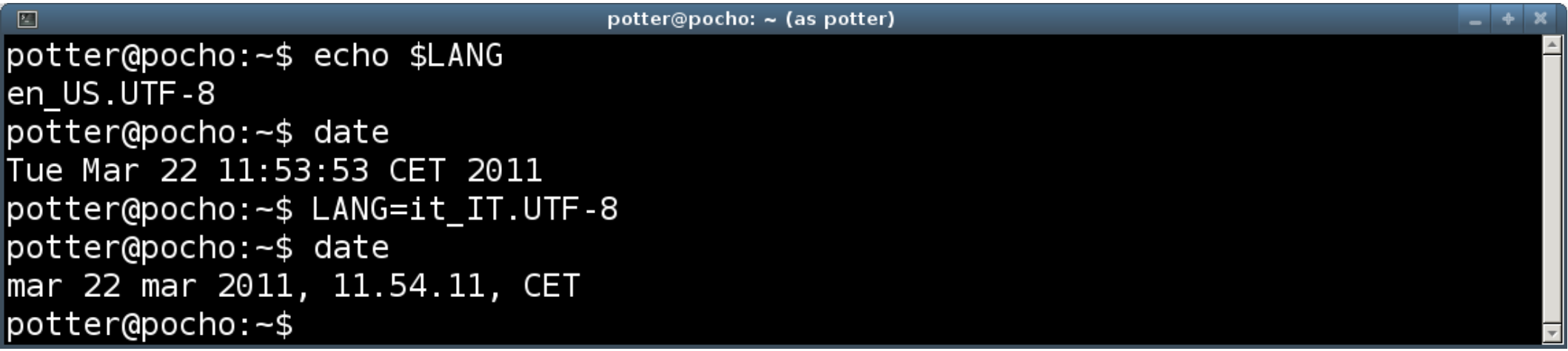
- In lettura si prepone il simbolo $, per delimitarne il nome si usano le parentesi graffe

```
$ echo $msg ${a}a lezione

Benvenuti alla 3a lezione
```

Variabili di ambiente

- Visibili a tutti i programmi eseguiti dalla shell
- Forniscono informazioni ai programmi ed eventualmente ne condizionano l'esecuzione
- Il comando env ne stampa l'elenco
- Per rendere d'ambiente una variabile dichiarata si usa il comando export



Variabili d'ambiente predefinite

- PATH le directory in cui la bash cerca i comandi
- USER l'utente che esegue la bash
- HOME la home directory dell'utente che la esegue
- PS1 il prompt
- HOSTNAME il nome del computer
- SHELL il pathname assoluto della shell in uso

- Il comando set visualizza tutte le variabili definite nella shell in esecuzione

La variabile PATH

- La variabile PATH memorizza i percorsi di ricerca dei comandi
- Contiene un elenco di directory separate da due punti
- Il valore di default per tutti gli utenti è impostato dall'amministratore
- L'utente può modificarla nel corso dell'esecuzione o fare in modo che venga modificata ad ogni avvio

La variabile PATH

- Esempio di PATH per l'utente root

```
# echo $PATH
/usr/local/sbin:/usr/local/bin:/usr
/sbin:/usr/bin:/sbin:/bin
```

- Esempio di PATH per l'utente regolare

```
$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/g
ames
```

Esecuzione di un Comando

- Distinguamo 2 casi:
- Quando il comando non contiene il carattere '/' la shell tenta di localizzarlo mediante i seguenti passi:

  1) verifica che il comando sia una funzione della shell

  2) verifica che sia un comando incorporato (built-in)

  3) cerca nelle directory specificate nell'ordine in cui sono elencate nella variabile PATH un file eseguibile con il nome specificato
- La shell esegue il primo file eseguibile con il nome specificato che trova
- Quando il comando contiene uno o più '/', la shell esegue il programma corrispondente al pathname specificato

PATH e directory corrente

- Quando si specifica il pathname per eseguire un comando è necessario utilizzare almeno un carattere '/'
- Per eseguire un file nella directory corrente sarà necessario utilizzare la sintassi ./nome_file
- Per ovviare a questo problema è possibile inserire la directory '.' nel PATH

Espansione della shell { }

- Prima di eseguire un comando la shell interpreta alcuni caratteri speciali
- Le parentesi graffe ci consentono di specificare un qualsiasi insieme di stringhe nella forma:

```
prefisso{stringa1,stringa2,striga3}suffisso
```

Esempio:

```
$ echo v{en,id,ic}i
veni vidi vici
```

- E' anche possibile innestarle

```
$ echo v{en,i{d,c}}i
veni vidi vici
```

Espansione della shell ~

- Il carattere '~' viene solitamente utilizzato per riferirsi alle home directory

  ~ identifica la home dir dell'utente della shell

  ~name identifica la home dir dell'utente name

- Esempio:

  ```
  $ mkdir ~/Mail
  ```

- Crea una directory Mail sotto la nostra home directory

Quoting

- L'inserimento di una stringa tra apici singoli ' ' protegge i caratteri speciali contenuti nella stringa dall'espansione da parte della shell.

```
$ echo '$HOME'=$HOME
$HOME=/home/oliva
```

- L'utilizzo di apici doppi “ ” consente l'espansione soltanto ai nomi di variabile mentre preserva gli altri caratteri speciali

```
$ echo "ls $HOME/* mostra il contenuto della homedir"
ls /home/oliva/* mostra il contenuto della homedir
```

- Il carattere \ escape preserva un singolo carattere dall'espansione

```
$ echo \*
*
```

Il tasto TAB

- Il tasto TAB può completare comandi, opzioni, argomenti, variabili
- Se c'è una sola possibilità completa la stringa
- Se c'è più di una possibilità non completa e con una seconda pressione del TAB vengono elencate tutte le possibilità disponibili

Esercizio 8

- Crea un file contenente tutte le variabili di ambiente denominato env-MATR e copialo sotto /tmp
- Copia un file di un altro tuo collega nella tua home directory (accertati che non sia vuoto)
- Concatena i due file e visualizza il contenuto ordinato senza mostrare le linee ripetute
- Visualizza prima le linee che non presentano ripetizioni poi quelle ripetute
- Copia il file /bin/echo nella tua home directory ed eseguilo con gli argomenti “Hello World!”
- Aggiungi alla variabile PATH la directory '.' facendo in modo che il comando which ls restituisca il path della copia nella tua directory

Esercizio 9

- Si utilizzi il comando echo per visualizzare la seguente frase:
  `All'username MATR contenuto nella variabile $USER corrisponde la home directory /home/INFOLAB/MATR contenuta nella variabile $HOME`

  dove MATR è il contenuto della variabile USER e /home/INFOLAB/MATR è il contenuto della variabile HOME
- Si creino tre directory “uno” “due” “tre” utilizzando l'espansione { }
- Si visualizzi con echo la home directory dell'utente root

Bibliografia

- http://www.freebsd.org/ports/shells.html
- http://appunti2.net/a228.htm#almltitle803