

La shell, questa sconosciuta

ovvero: perché usare un'interfaccia "da veri uomini"

Eugenia Franzoni <eugenia@catnic.it>

Come sfruttare al massimo questo talk

- spunti per il lavoro
- man bash!
- bisogna provare, provare, provare..

Perché una interfaccia di testo

- velocità
- potenza
- sempre disponibile
- agisce facilmente sui file di testo... ma
in Unix molte cose sono file di testo!

Il prompt della shell

- Cosa è
- Cosa vuol dire
- Utente normale o root?
 - Attenti al prompt... come evitare di spengere i server

I comandi nella shell

- Invio
- Ctrl-C
- Tasti per spostarsi nella linea di comando

Il case-sensitive

PIPPO e **pippo** sono due file diversi...

Il completamento automatico

- Evviva il Tab.
- Evviva il TabTab.
- Quanti comandi ho a disposizione?

La storia (history)

Freccia su, freccia giù

Ctrl-R: cerchiamo nella storia

Le estensioni dei file

Che significa .doc?

Usiamo il comando *file*.

Alcuni "giochi"

- cal dà un calendario
 - cal -3
 - cal -y
- date: che giorno è?
- echo $\$(5+4)$ fa da calcolatrice (solo interi)

Il potere di Greyskull

- *sudo comando*
- **SU -**

Chi sono? Dove sono?

- whoami
- pwd

Per leggere i file di testo

less nomefile

- avanti e indietro
- z o spazio = avanti di una schermata,
 - w = indietro di una schermata
 - frecce su/giu = su/giu di una riga
 - per cercare del testo
- /parola = cerca parola nel testo (ma anche + complicato!)
 - n va alla successiva parola

Le pipe

Ovvero: il potere di Unix per unire insieme
più comandi

L'output di un comando nell'input di un altro



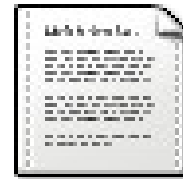
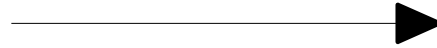
```
cat /etc/passwd | cut -d ":" -f 1 | sort
```

La redirectione

L'output di un comando in un file



Comando



File

```
ls /home/eugenia/divx/ > elencofilm.txt
```

Ancora redirectione, e viva i device

L'output, appendendo: >>

Gli errori: 2>

Es. `ls -R / >> elencofile.txt 2> /dev/null`

<< fino alla fine

`cat << EOF > appunti.txt`

prendo appunti su 'sta roba

boh funziona?

EOF

Sostituzioni a go-go e *

rinnoviamo i nostri file:

```
sed -i 's/vecchio/nuovo/g' *
```

Echo e le variabili

```
echo $USER
```

```
echo USER
```

```
pippo=pluto
```

```
echo $pippo
```

I processi in 7 semplici mosse

\$ xeyes

Ctrl-C

\$ xeyes &

\$ xeyes

Ctrl-Z

\$ bg

\$ fg

I processi: aiuto! :-)

\$ yes

Ctrl-C

\$ yes

Ctrl-Z

\$ bg

:-)

\$ fg

Ctrl-C

Top: che succede?

top

< e > : cambiando l'ordine degli addendi la somma non cambia

k e il PID: teniamo sotto controllo i nostri processi