



# Linux per neofiti

- Interazione shell-terminale
- Gestione della sicurezza
- Gestione della protezione
- Principali comandi Unix
- Redirezione di I/O
- Files di startup

....la prima volta... o quasi....

Welcome to Linux at test81

  
pingusa

Login:

Password:

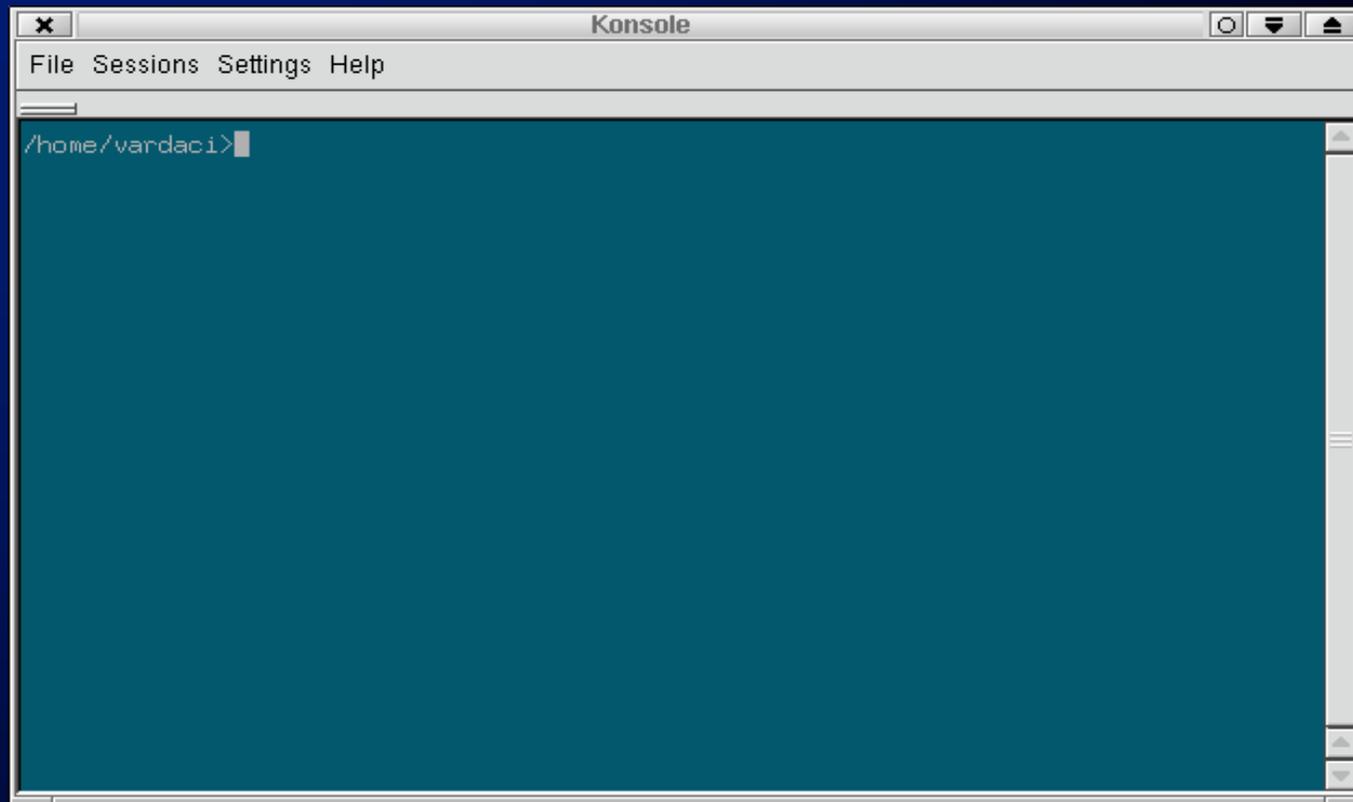
Session Type: default ▼



Login manager



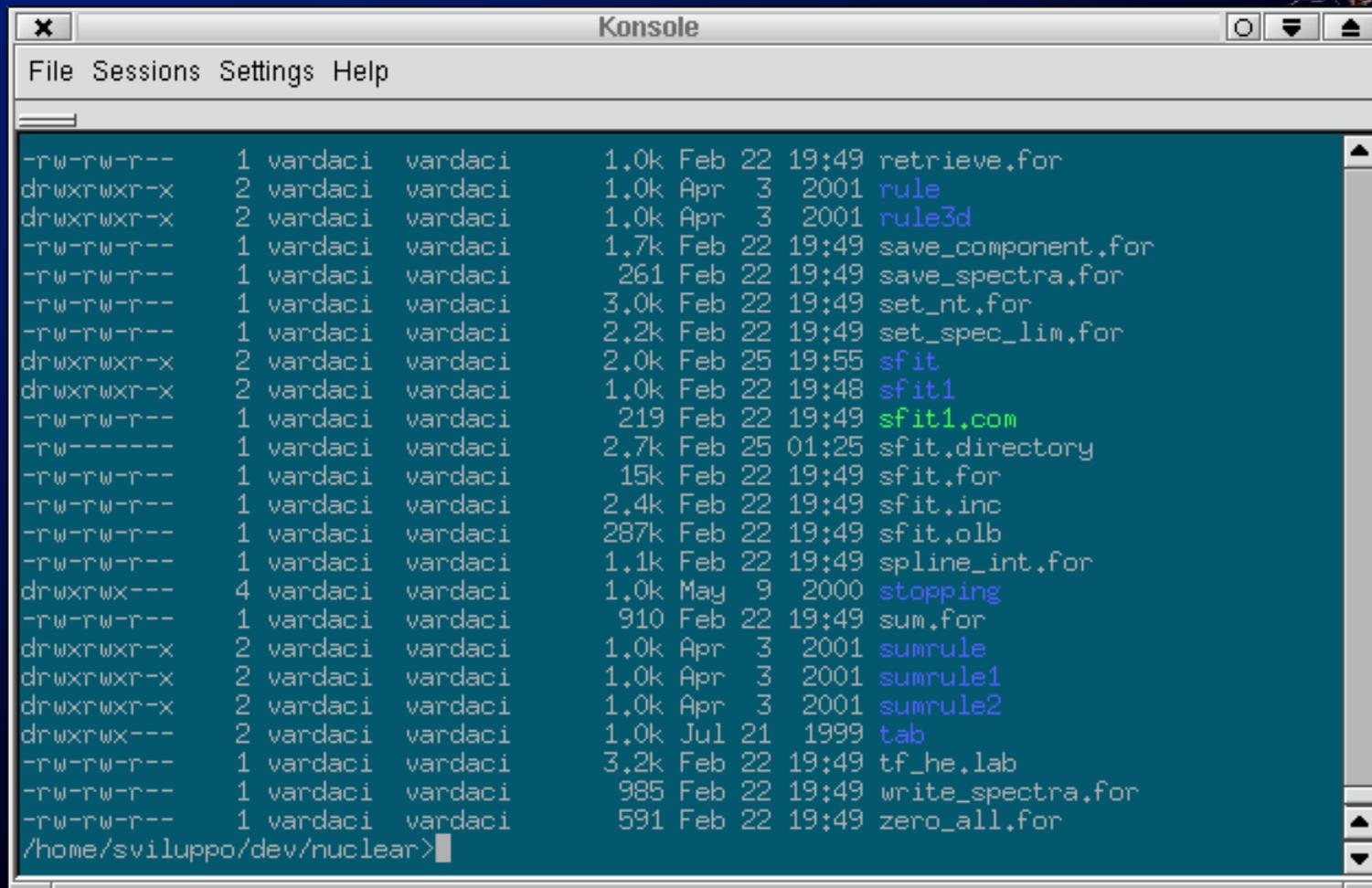
# Interazione shell-terminale (I)



Quando si apre una shell si entra in una directory di lavoro (default) che è assegnata all'utente dal system administrator. In questo esempio è /home/vardaci.

È possibile navigare nel file system con il comando `cd`.

# Interazione shell-terminale (II)



```
Konsole
File Sessions Settings Help

-rw-rw-r-- 1 vardaci vardaci 1.0k Feb 22 19:49 retrieve.for
drwxrwxr-x 2 vardaci vardaci 1.0k Apr 3 2001 rule
drwxrwxr-x 2 vardaci vardaci 1.0k Apr 3 2001 rule3d
-rw-rw-r-- 1 vardaci vardaci 1.7k Feb 22 19:49 save_component.for
-rw-rw-r-- 1 vardaci vardaci 261 Feb 22 19:49 save_spectra.for
-rw-rw-r-- 1 vardaci vardaci 3.0k Feb 22 19:49 set_nt.for
-rw-rw-r-- 1 vardaci vardaci 2.2k Feb 22 19:49 set_spec_lim.for
drwxrwxr-x 2 vardaci vardaci 2.0k Feb 25 19:55 sfit
drwxrwxr-x 2 vardaci vardaci 1.0k Feb 22 19:48 sfit1
-rw-rw-r-- 1 vardaci vardaci 219 Feb 22 19:49 sfit1.com
-rw-rw-r-- 1 vardaci vardaci 2.7k Feb 25 01:25 sfit.directory
-rw-rw-r-- 1 vardaci vardaci 15k Feb 22 19:49 sfit.for
-rw-rw-r-- 1 vardaci vardaci 2.4k Feb 22 19:49 sfit.inc
-rw-rw-r-- 1 vardaci vardaci 287k Feb 22 19:49 sfit.olb
-rw-rw-r-- 1 vardaci vardaci 1.1k Feb 22 19:49 spline_int.for
drwxrwx--- 4 vardaci vardaci 1.0k May 9 2000 stopping
-rw-rw-r-- 1 vardaci vardaci 910 Feb 22 19:49 sum.for
drwxrwxr-x 2 vardaci vardaci 1.0k Apr 3 2001 sumrule
drwxrwxr-x 2 vardaci vardaci 1.0k Apr 3 2001 sumrule1
drwxrwxr-x 2 vardaci vardaci 1.0k Apr 3 2001 sumrule2
drwxrwx--- 2 vardaci vardaci 1.0k Jul 21 1999 tab
-rw-rw-r-- 1 vardaci vardaci 3.2k Feb 22 19:49 tf_he.lab
-rw-rw-r-- 1 vardaci vardaci 985 Feb 22 19:49 write_spectra.for
-rw-rw-r-- 1 vardaci vardaci 591 Feb 22 19:49 zero_all.for
/home/sviluppo/dev/nuclear>
```

# Gestione della Sicurezza

- ogni utente e' identificato da un *account* cui corrisponde un User ID (UID) e un Group id (GID)
- ogni utente si deve far riconoscere dal sistema per mezzo della combinazione *username/password*

# Gestione della Protezione (I)

- Nei sistemi monoutente la protezione e' limitata ai singoli files i quali possono essere protetti con una password;
- Nel sistema multiutente UNIX si definiscono tre classi di protezione **user** (u), **group** (g) e **others** (o) ed ad ognuna di esse e' associata una tipologia di modi accesso

# Gestione della Protezione (II)

Le tipologie di modi di accesso sono tre:

- r** read
  - w** write
  - x** execute
- Il proprietario del file (user) puo' impostare queste tre tipologie per il file stesso.

Ad es, il proprietario puo' negare la lettura del file ai membri di un gruppo e a chiunque altro, oppure fare in modo che altri utenti accedano ad un directory:

```
$ chmod go-r expdata.dat  
$ chmod o+x /home/sviluppo
```

# Gestione della Protezione (III)

Osserviamo che quando si voglia fare eseguire alla shell dei comandi scritti in un file, ovvero un script, la tipologia `execute` del file deve essere attiva. Nel gergo si dice che "il file deve essere eseguibile":

```
$ chmod ugo+x commands
```

# Gestione della Protezione (IV)

```
Konsole
File Sessions Settings Help

-rw-rw-r-- 1 vardaci vardaci 1,0k Feb 22 19:49 retrieve.for
drwxrwxr-x 2 vardaci vardaci 1,0k Apr 3 2001 rule
drwxrwxr-x 2 vardaci vardaci 1,0k Apr 3 2001 rule3d
-rw-rw-r-- 1 vardaci vardaci 1,7k Feb 22 19:49 save_component.for
-rw-rw-r-- 1 vardaci vardaci 261 Feb 22 19:49 save_spectra.for
-rw-rw-r-- 1 vardaci vardaci 3,0k Feb 22 19:49 set_nt.for
-rw-rw-r-- 1 vardaci vardaci 2,2k Feb 22 19:49 set_spec_lim.for
drwxrwxr-x 2 vardaci vardaci 2,0k Feb 25 19:55 sfit
drwxrwxr-x 2 vardaci vardaci 1,0k Feb 22 19:48 sfit1
-rw-rw-r-- 1 vardaci vardaci 219 Feb 22 19:49 sfit1.com
-rw-rw-r-- 1 vardaci vardaci 2,7k Feb 25 01:25 sfit.directory
-rw-rw-r-- 1 vardaci vardaci 15k Feb 22 19:49 sfit.for
-rw-rw-r-- 1 vardaci vardaci 2,4k Feb 22 19:49 sfit.inc
-rw-rw-r-- 1 vardaci vardaci 287k Feb 22 19:49 sfit.olb
-rw-rw-r-- 1 vardaci vardaci 1,1k Feb 22 19:49 spline_int.for
drwxrwx--- 4 vardaci vardaci 1,0k May 9 2000 stopping
-rw-rw-r-- 1 vardaci vardaci 910 Feb 22 19:49 sum.for
drwxrwxr-x 2 vardaci vardaci 1,0k Apr 3 2001 sumrule
drwxrwxr-x 2 vardaci vardaci 1,0k Apr 3 2001 sumrule1
drwxrwxr-x 2 vardaci vardaci 1,0k Apr 3 2001 sumrule2
drwxrwx--- 2 vardaci vardaci 1,0k Jul 21 1999 tab
-rw-rw-r-- 1 vardaci vardaci 3,2k Feb 22 19:49 tf_he.lab
-rw-rw-r-- 1 vardaci vardaci 985 Feb 22 19:49 write_spectra.for
-rw-rw-r-- 1 vardaci vardaci 591 Feb 22 19:49 zero_all.for
/home/sviluppo/dev/nuclear>
```

# Gestione della Protezione (V)

le tre classi di protezione u,g ed o possono essere combinate con i tre tipi di accesso (read, write, execute) :

<b>rwX</b>	<b>rwX</b>	<b>rwX</b>
<b>user</b>	<b>group</b>	<b>other</b>

La presenza di un permesso e' indicata dalla lettera appropriata in una data locazione convenzionale.

# Gestione della Protezione (VI)

<b>Modo di Accesso</b>	<b>Files Ordinari</b>	<b>Directory Files</b>
<b>Read</b>	Permette l'esame del contenuto	Permette di accedere all'elenco dei files nella directory
<b>Write</b>	Permette la modifica del file	Permette di creare o rimuovere files
<b>Execute</b>	Permette l'esecuzione degli scripts	Permette l'accesso alla directory

# Navigazione nel file system

`cd` `nomedir`

Cambia la directory corrente

`pwd`

Mostra la directory corrente

`ls` `[-lh...]` `[nomedir]`

Elenca i files nella directory corrente o in `nomedir`

`df` `[-h]` `[mountpoint]`

Disk free: stato di utilizzo di un disco

`du` `[-ak]` `[nomefile]`

Dimensioni di files e directory

# Manipolazione di files (e directories)

cp

rm

mv

mkdir

rmdir

ln

file

more

touch

head

tail

chmod

chown

chgrp

cat

diff

sort

tar

gzip

bzip2

find

Fate riferimento ai testi per i dettagli...

# Manipolazione dei processi

ps

bg (oppure &)

fg

kill

top

nice



# Comandi “generali”

man

who

whoami

lpr

echo

alias

which

chsh

passwd

date

time

grep

uname



# Principali directory e sottodirectory

La directory root (=radice) contiene l'intera gerarchia del file system.

- /bin** : file eseguibili essenziali (bin = binary, programmi e scripts)
- /dev** : file dei dispositivi di sistema (dev = device)
- /etc** : file di configurazione caratteristici del sistema
- /home** : directory relativa alle aree associate ad ogni utente
- /lib** : librerie essenziali per la gestione del sistema ( es. system calls)
- /usr** : raccoglie la maggior parte dei programmi applicativi
- /tmp** : contiene file temporaneamente creati dalle applicazioni
- /sbin** : file eseguibili necessari all'avvio del sistema
- /mnt** : punti di mount per file systems temporanei (floppy,cdrom...)

# Ridirezione dell' I/O (I)

Un qualunque processo apre automaticamente tre tipi di files: **standard input, standard output e standard error**.

Tipicamente questi file sono identificati con i numeri **0, 1 e 2**, detti **descrittori**, e sono associati, rispettivamente, alla tastiera, al terminale video e ad una console di controllo.

Lo scopo della **ridirezione** e' quello di **ridirigere** questi tre canali di comunicazione verso altri sistemi di I/O, come ad es. un file di testo.

In UNIX, la ridirezione dell' I/O e' realizzata attraverso gli operatori **< e >**.

# Ridirezione dell' I/O (II)

Es. Ridirigere la lista dei file di una directory dal terminale ad un file.

```
$ls /etc/init.d 1> /home/peppe/my_inetd
```

Lo *standard output* viene ridiretto (>) verso un file. Se il file non esiste viene creato al momento (on the fly); se esiste viene sovrascritto.

```
$ls /etc/init.d > /home/peppe/my_inetd
```

In questo caso la shell assume che vogliate usare lo *standard output*, ma fornisce un messaggio di errore se il file già esiste.

# Ridirezione dell' I/O (III)

Es. Quante parole ci sono in un file?

```
$ wc -l 0< /home/peppe/my_inetd
```

```
$ wc -l < /home/peppe/my_inetd (default)
```

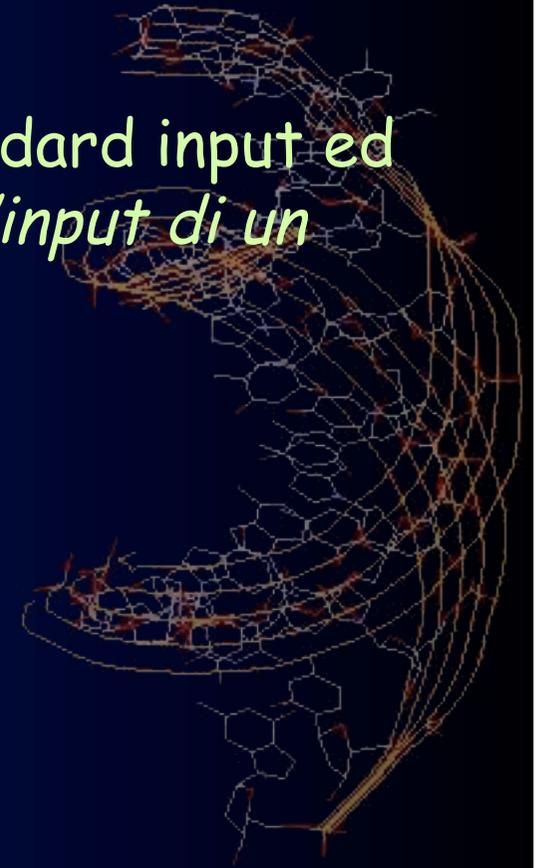
Spesso e' utile ridirigere lo *standard error* , ovvero il canale in cui i processi riversano i loro messaggi d'errore:

```
$ wc -l > /dev/null 2> \
    /home/peppe/my_error
```

# Pipe: comandi concatenati

La pipe permette di "concatenare" gli standard input ed output: *l'output di un programma diventa l'input di un altro programma con il simbolo |.*

```
$ cat my_file | lpr
```



# Caratteri speciali: Metacaratteri

Un metacarattere serve a sostituire un insieme contiguo di uno o piu' caratteri.

Es. i caratteri **\*** e **?**

```
$ls *.txt
```

Questo comando permette di elencare solo i files i cui nomi che terminano con “.txt”.

```
$ls g?.pdf
```

Questo comando permette di elencare solo i files i cui nomi sono di due lettere, iniziano con la lettera g e terminano con “.pdf”. (ad es. g1.pdf ma non g12.pdf)

Il compito di individuare i files che corrispondono all'espressione \*.txt non viene svolto dal comando ls ma dalla shell stessa.

# Path

I comandi Unix, come detto sono dei particolari files eseguibili. Come fa l'interprete dei comandi a sapere dove cercare questi files ?

La risposta è nel valore di una particolare variabile chiamata path che contiene un elenco di directories in cui cercare un comando o un generico eseguibile (script, programma etc.). La path conterrà in genere un certo numero di directories "standard" come /bin, /usr/bin e ./

Altre directories possono essere aggiunte dall'utente per eseguire direttamente comandi particolari.

Quando si invoca un comando il sistema "sfoglia" le directories elencate nella path (nell'ordine) ed esegue il primo comando corrispondente.

# Variabili di ambiente

La path è solo un esempio di una classe di variabili, dette variabili di ambiente, che vengono utilizzate dal sistema, o da particolari applicazioni per contenere informazioni specifiche della shell. Il comando echo, e il simbolo speciale \$ permettono di visualizzare il contenuto di tali variabili. Ad esempio:

```
echo $path
```

Visualizza la path

```
echo $home
```

Visualizza la home directory

```
echo $shell
```

Visualizza la shell corrente

# I file di startup

Quando si apre una shell, la shell stessa lancia un insieme di scripts che impostano l'ambiente di lavoro. Nella C shell questi scripts sono i seguenti:

- > /etc/csh.login
- > \$HOME/.cshrc
- > \$HOME/.login

Il primo script è gestito dal system manager, gli altri due sono modificabili dall'utente.