Prima Esercitazione

GNU/Linux e
linguaggio C

Stefano Monti
stefano.monti6@unibo.it

Agenda

- Introduzione a sistemi operativi GNU/Linux
  - Storia
  - Architettura ed elementi basilari
- Strumenti di Amministrazione
  - Locale: comandi shell comuni
  - Remoto: SSH/SCP, X Forwarding
- Esercitazione C

Unix e GNU/Linux

Unix: sviluppato negli anni '60-'70 presso Bell Labs di AT&T, attualmente sotto il controllo del consorzio The Open Group

- **UNIX certified**: sistemi fully-compliant (IBM AIX, Solaris 10, ...)
- **Unix-like**: sistemi NON fully-compliant (GNU/Linux, BSD, ...)

GNU (**G**NU's **N**ot **U**nix)

- progetto annunciato da Stallman '83; obiettivo la creazione di un sistema operativo **free Unix-like**
- obiettivo raggiunto nel '92, con l'**adozione del kernel Linux**

Linux

- primo kernel rilasciato nel 1991 ad opera di Torvalds
- successivamente, adozione di software dal progetto GNU
- attualmente **kernel ufficiale Linux** nome in codice **Vanilla**

Distribuzioni GNU/Linux

Attualmente varie **distribuzioni GNU/Linux** (comunemente *distro*):

- personalizzazione del **kernel** *vanilla*
- collezione di **pacchetti** (applicativi) **software**: archivi compressi usati per **automatizzare** e **semplificare** l'installazione di applicazioni (compilazione dei sorgenti, impostazione delle variabili di ambiente, configurazione di permessi, ecc...)
- alcuni esempi: Redhat/Fedora, Slackware, Debian, Gentoo, Ubuntu, SUSE, ecc...

Gestori di pacchetti: sono pacchetti a loro volta

- differenti per *famiglie* di distribuzioni (RPM, APT, Portage,...)
- operazioni di installazione, rimozione, aggiornamento di pacchetti software
- gestione dipendenze tra pacchetti

Ai fini del corso...

Necessità di utilizzare un sistema operativo **Unix-like**; varie possibilità:

- **Installazione** di una distribuzione Linux su una macchina fisica:
  - maggiore apprendimento ma complessità e problematiche maggiori (partizionamento del disco fisso, dual booting, ecc...)
- Uso distribuzioni **Linux Live CD**
  - nessuna installazione, ambiente di lavoro "stateless" e ripetibile caricato in RAM, elevati requisiti hardware, prestazioni penalizzate
- **Virtualizzazione**
  - installazione necessaria ma priva di implicazioni per la macchina fisica (configurazione dual boot, partizionamento del disco), prestazioni ragionevoli

Filesystem hierarchy standard (FHS)

http://www.pathname.com/fhs/

- **/** : Root
- **/bin** : File binari dei comandi essenziali
- **/sbin** : File binari dei comandi di sistema essenziali
- **/home** : Home degli utenti
- **/var** : Dati variabili
- **/boot** : File statici per operazioni di boot (avvio) della macchina
- **/dev** : File dispositivi
- **/etc** : File di configurazione
- **/lib** : Shared libraries e moduli del kernel
- **/media** : *Mount point* per *media* rimovibili
- **/mnt** : *Mount point* per operazioni di mount temporanee di FS
- **/opt** : Software applicativi
- **/tmp** : File temporanei
- ...

Operazioni di mount/unmount (1/2)

Gerarchia di filesystem organizzata ad albero (radice /).

File e directory possono risiedere fisicamente su dispositivi (device) differenti (es., hard disk, *media* rimovibili, cd, floppy, unità storage di rete, ...)

necessità di "agganciare" dinamicamente filesystem

su dispositivi eterogenei alla gerarchia di file in **/**

Comando **mount**

```
mount -t <fs_type> <device> <mount_point>
```

Es. `mount -t vfat /dev/sda1 /media/usbkey` monta il filesystem FAT32 (`vfat`) di una chiavetta USB (`/dev/sda1`) nella cartella `/media/usbkey`

Operazioni di mount/unmount (2/2)

Comando **`umount`** per "staccare" dalla gerarchia un filesystem precedentemente montato

```
umount <mount_point> | <device>
```

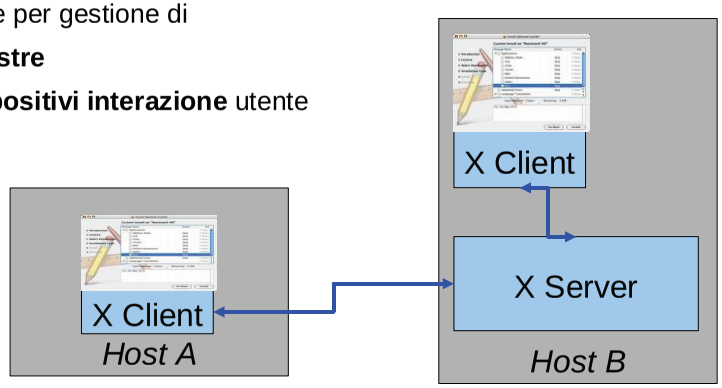
Alcune informazioni utili:

- comando **`dmesg`**: buffer output del kernel, utile per rilevare inserimento di nuove periferiche
- comando **`fdisk -l`** : tabella delle partizioni dei dischi (fissi e rimovibili) presenti nel sistema; utile per reperire device associati alle partizioni (es. /dev/hdaN per partizioni su HD primario IDE)
- comando **`mount`** senza argomenti: lista dei filesystem attualmente montati
- file **`/etc/fstab`** : informazioni statiche sui filesystem montati automaticamente durante la fase di avvio del S.O.; è possibile aggiungere nuove entry

X Window System

Strumento per la gestione di GUI (Graphical User Interface)

- approccio network-based **client/server**
- **cross-platform**: non legato ad un particolare S.O.
- sfruttato anche da Desktop Environment (es. GNOME, KDE)
- primitive per gestione di
  - **finestre**
  - **dispositivi interazione** utente
  - ...



Shell - comandi di base

- `man <nome_comando>`
  - manuale del comando e opzioni
- `cd <nome_directory>`
  - Permette di spostarsi all'interno del file system
  - Posizioni relative:
    - . (direttorio corrente) e .. (direttorio padre)
- `mkdir <nome_nuova_directory>`
  - Creazione di un nuovo direttorio
- `ls <parametri>`
  - Lista il contenuto del direttorio corrente
- `emacs (o vi) <nome_file>`
  - Programmi per l'editing; se il file non esiste, lo creano

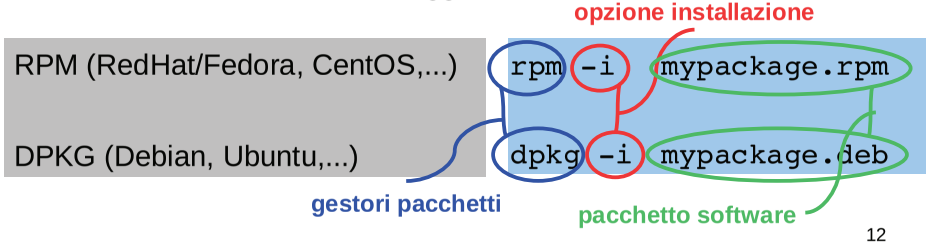
Shell – alcune utilità

- tar: gestione di archivi
  - creazione archivio: `tar -cf prova.tar prova/`
  - estrazione archivio: `tar -xf prova.tar`
  - opzione -z per la compressione dei file (formato .gz)
- top:
  - monitoring di risorse di sistem (memoria, CPU, processi)
  - interattivo

Gestione pacchetti software

Differenti modalità, più o meno automatizzate

- Forma basilare: scaricare i sorgenti e compilarli!!
- Pacchetti software
  - binari precompilati in un unico pacchetto (file)
  - gestione dipendenze tra pacchetti
  - installazione/rimozione/aggiornamento



Gestori pacchetti avanzati (1/2)

- Gestione di **repository** remoti di pacchetti
  - **reperimento** di pacchetti da **remoto**
  - installazione/aggiornamento **automatico**
- Risoluzione automatica delle **dipendenze**
- **Interattività**
- Gestione di **gruppi** di pacchetti (metapacchetti)

Gestori pacchetti avanzati (2/2)

	APT (Debian-based distros)	YUM (RedHat-based distros)
Gestione repository	*File:* `/etc/apt/sources.list`	*File in:* `/etc/yum.repos.d/`
Ricerca pacchetti	`apt-cache search <STRING>`	`yum search <STRING>`
Aggiornamento lista (locale) pacchetti	`apt-get update`	`--`
Aggiornamento di (tutti) i pacchetti installati	`apt-get upgrade`	`yum update <NOME_PKG>`
Installazione di un pacchetto (e relative dipendenze, se presenti)	`apt-get install <NOME_PKG>`	`yum install <NOME_PKG>`
Rimozione pacchetto	`apt-get remove <NOME_PKG>`	`yum remove <NOME_PKG>`

Gestione utenti (1/2)

- La maggior parte delle operazioni normalmente eseguite sulla macchina (editing di testo, utilizzo di applicativi, ecc...) non richiede particolari privilegi.
- Operazioni di amministrazione (configurazione, manutenzione, gestione, installazione software) possono essere eseguite solo da **super-utente** (utente **root**)
- Diritti di scrittura/lettura/esecuzione su file e direttori critici per la sicurezza/stabilità del sistema
- E' buona norma limitare l'utilizzo dell'account **root** per compiere le sole operazioni che lo richiedono, utilizzando un utente “normale” per le altre

Gestione utenti (2/2)

Strumenti per la gestione utente

- creazione di un utente: useradd <NOMEUTENTE>
- rimozione di un utente: userdel <NOMEUTENTE>
- modifica password: passwd <NOMEUTENTE>
- cambio identità : su <NOMEUTENTE> (senza <NOMEUTENTE>per utente root)
- username utente corrente: whoami
- Informazioni utente corrente : id
- Lista utenti correntemente loggati alla macchina : who

Amministrazione remota - SSH

Necessità di eseguire comandi shell su macchine

- fisicamente non accessibili ma **connesse** in rete
- mantenendo **riservatezza** dei comandi eseguiti

SSH : **S**ecure **SH**ell

- protocollo di rete per lo scambio sicuro (cifrato) di informazioni tra due host di rete
- uso tipico: login ad una shell remota
- se login (con credenziali del sistema remoto!) ha successo, si ottiene (in locale) una shell del sistema remoto

```
ssh stefano@192.168.1.1
```

nome utente remoto — indirizzo IP macchina remota

Protocollo SSH – copia remota di file

Comando **SCP**: copia di file tra macchine remote

```
scp [[user@]host1:]file1 ... [[user@]host2:]file2
```

- opzione `-r` per effettuare la copia di interi direttori

Es: `scp prova.txt stefano@192.168.1.1:/home/stefano`

copia il file locale `prova.txt` nella cartella `/home/stefano` della macchina 192.168.1.1, usando l'utente remoto `stefano`

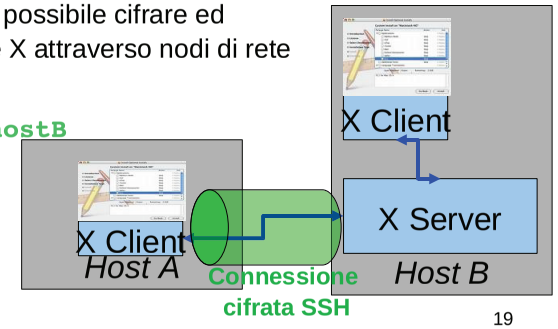
Protocollo SSH – X Forwarding

Necessità di eseguire accedere a programmi dotati di GUI (Graphic User Interface)

- fisicamente non accessibili ma **connesse** in rete
- mantenendo **riservatezza** delle operazioni eseguite

Tramite comando SSH, è possibile cifrare ed esportare una sessione X attraverso nodi di rete remoti

```
ssh –X nomeutente@hostB
```



Un esempio concreto

Installazione ambiente di sviluppo **tIDE** su macchina virtuale remota

- accedere tramite SSH alla macchina virtuale (con abilitazione X forwarding)

```
ssh -X nomeutente@indirizzo_ip
```

- lanciare browser web su macchina remota (GUI locale tramite X forwarding)

```
firefox
```

- scaricare tIDE da

```
http://snowmail.sn.funpic.de/tide/tide.jar
```

- avviare tIDE

```
cd <DIR_DI_DOWNLOAD>
java -jar tide.jar
```

Esercitazione 1- Obiettivi

Programmazione C

- Gestione dei parametri in ingresso
  - argomenti ***argc*** ed ***argv***
  - controllo di correttezza dei parametri in ingresso
- Gestione delle stringhe
  - libreria string.h

Compilazione sorgenti - C

- Comando `gcc <file>`
  - Compilatore C e C++
  - Compila `<file>` producendo il **file eseguibile** `a.out`
  - Per dare un nome diverso al file prodotto opzione `-o`
- Es: `gcc file_exec.c -o f_ex`
- Esecuzione: `./f_ex <parametri>`

Esercitazione 1 – Testo (1/2)

Si realizzi un programma C che abbia un'interfaccia del tipo

listaTreni treno1 .... trenoN

e che prenda in ingresso un numero arbitrario di stringhe rappresentanti il codice di un treno, nel formato:

<TIPOTRENO><NUMEROTRENO>

- <TIPOTRENO> stringa di due caratteri che rappresenta il tipo di treno (per semplicità, si assuma che abbia i valori "IC", "ES", "RG", rispettivamente per le tipologie Intercity, Eurostar e Regionale)
- <NUMEROTRENO> identificativo numerico univoco del treno di 4 cifre

Esercitazione 1 – Testo (2/2)

Il programma deve:

- controllare che sia stato passato ***almeno un treno***
- controllare che ogni codice passato sia ***conforme alle caratteristiche*** sopra indicate (in particolar modo, rispetti la lunghezza di 6 caratteri)
- stampare a video i soli identificativi dei treni, ***raggruppati per categoria***