Gestione dei processi

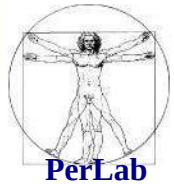
Insegnamento di Sistemi Operativi di Rete
Master Universitario in Tecnologie Internet

Domenico De Guglielmo

E-mail: domenicodegu@virgilio.it Telefono: 050 2217 468

Pervasive Computing & Networking Lab (PerLab) http://www.perlab.it
Dipartimento di Ingegneria dell'Informazione, Università di Pisa

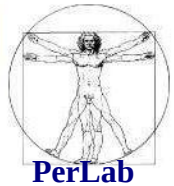
Sommario



- Gestione dei processi
  - informazioni associate ai processi
  - real/effective UID (GID)
- Comandi per la gestione dei processi
- Segnali
  - descrizione dei segnali
  - comandi per l'invio di segnali
- Shell e processi
  - esecuzione in background
  - invio segnali da tastiera (^C, ^D)
- Priorità dei processi
  - priorità assegnata dal SO
  - livello di nice

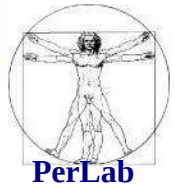
I Processi

Processo



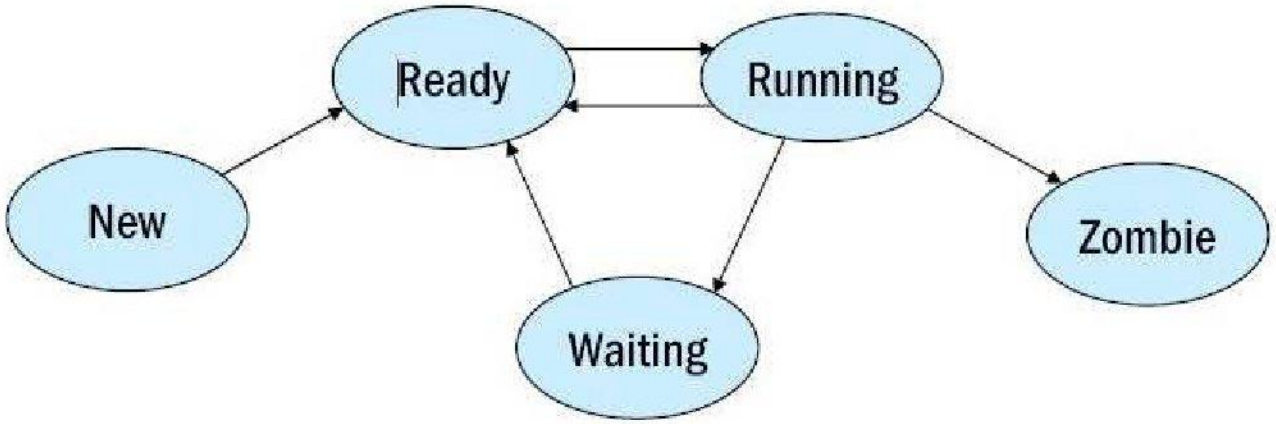
- Programma in esecuzione
- Generabile solo da un processo già esistente
  - si crea una gerarchia di processi (albero)
- Identificabile univocamente grazie ad un numero chiamato PID
- I processi di dividono in:
  - Utente
  - Sistema

Vita di un processo



- Creazione di un processo
  - tramite la chiamata di sistema fork()
- Terminazione di un processo
  - tramite la chiamata di sistema exit()

  diventa **“zombie”**: terminato, ma ha ancora associato un PID
- Stato di un processo



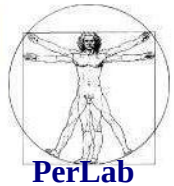
Informazioni associate ai processi

- Il kernel gestisce la **Tabella dei processi** dove tiene traccia dello stato di tutti i processi

- Informazioni associate:
  - nome del programma eseguibile
  - PID: identificatore univoco
  - PGID: identificatore di gruppo-processi
  - PPID: PID del processo da cui e' stato generato
  - UID: identificatore di utente
  - GID: identificatore di gruppo-utenti

  determinano i privilegi

Tabella dei processi

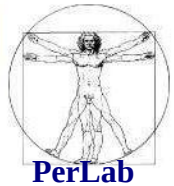


- Visibile nella directory /proc/.
- La directory /proc/ contiene tante cartelle quanti sono i processi in esecuzione, indicati per PID.
- Ogni cartella contiene dei file che rappresentano lo stato del processo.

init

- Fase di avvio del sistema operativo: il kernel avvia il processo speciale init (/sbin/init)
  - genitore di tutti gli altri processi (radice dell'albero) e
  - non ha processo genitore
  - PID = 1
  - effettua una serie di azioni per portare il sistema in un certo stato di partenza
  - eredita i figli dei processi che terminano

Esempio



- Visualizzare lo stato del processo con PID = 1:
  - Spostarsi nella cartella /proc/

    cd /proc/
  - Visualizzarne il contenuto

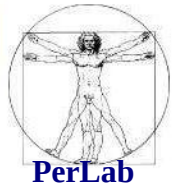
    ls
  - Spostarsi nella cartella relativa al processo 1

    cd 1/
  - Visualizzare lo stato del processo 1 contenuto nel file status

    less status

(tornare alla propria home cd)

Privilegi di esecuzione



- Ogni processo ha associati l'utente ed il gruppo a cui appartiene per stabilire quali operazioni può compiere.

- Ad ogni processo sono associate 2 coppie di User ID e Group ID:
  - RUID e RGID: Real UID e Real GID dell'utente che ha mandato in esecuzione il processo
    - non cambiano durante la sessione di login
  - EUID, EGID: Effective UID e Effective GID dell'utente e del gruppo che il kernel considera per determinare i privilegi per l'accesso ai file
    - possono variare durante l'esecuzione del processo

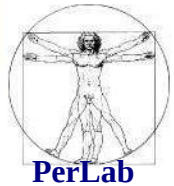
Meccanismo

- *RUID* e *EUID* e *RGID* e *EGID* nella maggior parte dei casi coincidono.

- Ogni file ha un *owner* e un *group owner*: se il file di un programma ha attivo il bit dei permessi *SUID* e *SGID*, allora, quando viene lanciato, *l'EUID* (EGID) diventerà quello dell'owner (group *owner)* del file e non quello dell'utente che lo ha lanciato.

- Esempio: se un programma appartiene al superuser e ha il bit SUID attivo, l'utente che lo lancia ottiene i privilegi di superuser durante l'esecuzione.

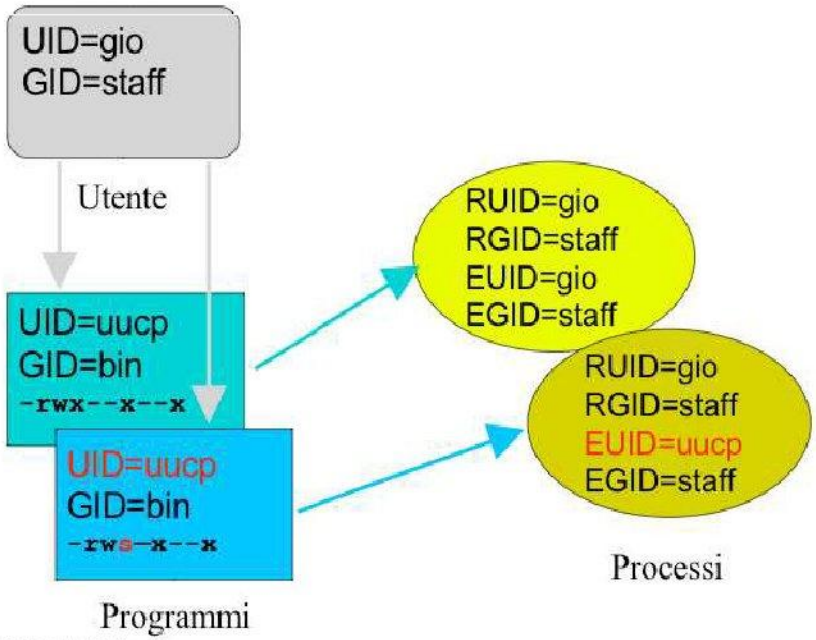
Esempio



- Il comando passwd consente a qualsiasi utente di cambiare la propria password.
  - Digitare il comando

    ls –l /usr/bin/passwd

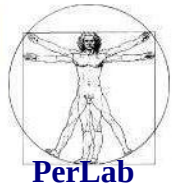
-r-s—x—x 1 root root 37084 Feb 4 /usr/bin/passwd



G. Schmid-Processi Unix

Monitoraggio dei Processi

Comandi di monitoraggio



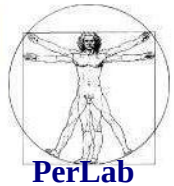
- Esistono 3 comandi per visualizzare le informazioni relative ai processi:
  - due rappresentano la situazione relativa ad un certo istante (statica)

    ps

    pstree
  - uno mostra costantemente l'aggiornamento della situazione, impegnando un terminale (dinamica)

    top

Comando ps



- Visualizza un elenco di processi in esecuzione

PID	TTY	STAT	TIME	COMMAND
32798	pts/0	S	0:00.03	-bash (bash)
32925	pts/0	R	0:00.00	ps

- PID: numero del processo
- TTY: terminale
- STAT: stato del processo

R : running	I : bloccato < 20s	l : multi-threaded
S : bloccato > 20s	D : pausa non interrompibile	+ : non foreground
T : sospeso	Z : zombie	s : session leader
W : in memoria virtuale	N : NICE > 0	

- TIME: tempo totale di utilizzo del processore
- COMMAND: comando utilizzato per avviare il processo

Comando ps

Sintassi:

ps [opzioni]

[opzioni] (senza -)

x : visualizza anche i processi che non provengono da terminali

u : formato utente, indica in particolare l'utente a cui appartiene ogni processo

a : visualizza i processi di tutti gli utenti

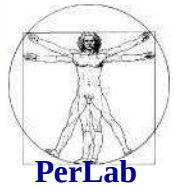
l : formato esteso, mostra un elenco lungo

r : non mostra i processi in pausa

o : precisa *solo* le colonne da mostrare, elencate di seguito

U nome_utente: mostra i processi di nome_utente

Comando ps



ps aux: indica tutti i processi in esecuzione

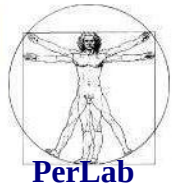
USER	PID	%CPU	%MEM	VSZ	RSS
User	32929	0.0	0.1	1484	944

TT	STAT	STARTED	TIME	COMMAND
p0	R+	6:40PM	0:00.00	ps aux

- %CPU: percentuale d'uso del processore
- VSZ : memoria virtuale usata
- RSS : memoria fisica usata
- TT : terminale virtuale a cui è associato

  TT = ?? Processi non associati a nessun terminale. Sono i DEMONI, cioè processi di sistema
- STARTED : orario di partenza del processo
- TIME : tempo totale di uso del processore
- COMMAND : comando che ha creato il processo

Comando pstree



- Visualizza lo schema ad albero dei processi in esecuzione

Sintassi:

pstree [opzioni] [PID | utente]

[opzioni]

-p : mostra i PID dei processi

-a : visualizza tutta la riga di comando e non solo il nome

[PID | utente]

PID : si può specificare un numero di processo

utente : si può specificare un nome utente

Comando top

- Offre una visione dinamica dei processi e dell'uso delle risorse

```
10:13pm  up 58 min,  5 users,  load average: 0.09, 0.03, 0.01
67 processes: 65 sleeping, 2 running, 0 zombie, 0 stopped
CPU states:  5.9% user,  0.7% system,  0.0% nice, 93.5% idle
Mem:   62296K av,  60752K used,   1544K free,  36856K shrd,  22024K buff
Swap: 104416K av,      8K used, 104408K free                 16656K cached

  PID USER     PRI  NI  SIZE  RSS SHARE STAT  LIB %CPU %MEM   TIME COMMAND
  588 root      16   0  6520 6520  1368 R       0  5.1 10.4   0:02 X
  613 daniele    6   0   736  736   560 R       0  1.3  1.1   0:00 top
  596 daniele    1   0  1108 1108   872 S       0  0.1  1.7   0:00 fvwm2
    1 root       0   0   388  388   336 S       0  0.0  0.6   0:08 init
    2 root       0   0     0    0     0 SW      0  0.0  0.0   0:00 kflushd
    3 root       0   0     0    0     0 SW      0  0.0  0.0   0:00 kswapd
   82 root       0   0   352  352   300 S       0  0.0  0.5   0:00 kerneld
  139 root       0   0   448  448   364 S       0  0.0  0.7   0:00 syslogd
  148 root       0   0   432  432   320 S       0  0.0  0.6   0:00 klogd
  159 daemon     0   0   416  416   340 S       0  0.0  0.6   0:00 atd
  170 root       0   0   484  484   400 S       0  0.0  0.7   0:00 crond
  181 bin        0   0   336  336   268 S       0  0.0  0.5   0:00 portmap
  204 root       0   0   404  404   336 S       0  0.0  0.6   0:00 inetd
```

Comando top

- load average : utilizzo della CPU nell'ultimo minuto, ultimi 5 e ultimi 15
- CPU(s)

  user : usata dagli utenti

  system : usata dal sistema

  nice : cresce quando cambio NICE
- Mem: uso della memoria (totale, usata e libera)
- Impegna un terminale
- Accetta comandi interattivi

Comando top

Sintassi:

top [opzioni]

[opzioni]

-d delta : tempo tra un aggiornamento e l'altro

-i : visualizza anche processi inattivi (zombie)

- Comandi interattivi:

h | ? : riassunto dei comandi

k : per inviare segnali ai processi

i : visualizza solo i processi inattivi e viceversa

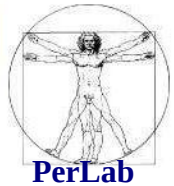
r : renice

n | # : cambia la quantità di processi da visualizzare

q : termina l'esecuzione di top

Segnali

Comunicazione tra processi

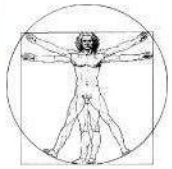


- Segnali
  - messaggi elementari inviabili ad un processo per informarlo del verificarsi di una condizione

  I programmi sono scritti in modo tale da

  - intercettare i segnali e comportarsi di conseguenza
  - intercettare i segnali e comportarsi in modo diverso da quello prevedibile
  - ignorare i segnali

Come generare un segnale



PerLab

- Digitare in un terminale delle particolari combinazioni di tasti

  Es. CTRL+C: interruzione di un processo.

- Usare la chiamata di sistema kill()
- Generati dal kernel per informare i processi quando succede qualcosa di particolare.
- Eccezioni hardware (divisione per zero, accesso non valido alla memoria, etc)

Elenco dei segnali

Nome segnale	Numero segnale	Descrizione segnale
SIGHUP	1	Interruzione di comunicazione col terminale
SIGINT	2	Interruzione del processo attraverso comando da tastiera
SIGQUIT	3	Chiude programma attraverso comando da tastiera
SIGILL	4	Istruzione illegale
SIGTRAP	5	L'esecuzione del processo ha raggiunto un breakpoint (trap), il debugger puo' informare di questo lo sviluppatore
SIGABRT	6	Interruzione anormale (abort) del processo.
SIGEMT	7	Emulare istruzionieseguite
SIGFPE	8	Eccezione in un numero in virgola mobile
SIGKILL	9	Interruzione immediata. Questo segnale non puo' essere ignorato ed il processo che lo riceve non puo' eseguire delle operazioni di chiusura "morbida".
SIGBUS	10	Errore di bus: "tentato accesso ad una porzione indefinita di memoria"

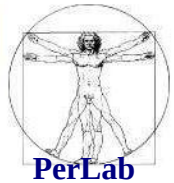
Elenco dei segnali

Nome segnale	Numero segnale	Descrizione segnale
SIGSEGV	11	Errore di segmentazione
SIGSYS	12	Chiamata di sistema errata
SIGPIPE	13	Se un processo che dovrebbe leggere da una pipe termina inaspettatamente, questo segnale viene inviato al programma che dovrebbe scrivere sulla pipe in questione.
SIGALRM	14	Il timer in tempo reale e' scaduto. Segnale sollevato da alarm().
SIGTERM	15	Terminazione del programma; il comando kill invia questo segnale se non diversamente specificato.
SIGURG	16	Sono disponibili dei dati urgenti per il processo su un socket.
SIGSTOP	17	Ferma temporaneamente l'esecuzione del processo: questo segnale non puo' essere ignorato
SIGTSTP	18	Ferma temporaneamente l'esecuzione del processo
SIGCONT	19	Il processo puo' continuare, se era stato fermato.
SIGCHLD	20	Processo figlio terminato o fermato

Elenco dei segnali

Nome segnale	Numero segnale	Descrizione segnale
SIGTTIN	21	Un processo in esecuzione in background tenta di leggere da terminale
SIGTTOU	22	Un processo in esecuzione in background tenta di scrivere sul terminale
SIGIO	23	I / O su un possibile Descrittore
SIGXCPU	24	Esaurito il tempo di CPU disponibile per il processo
SIGXFSZ	25	Superata la dimensione massima consentita per i file per il processo
SIGVTALRM	26	Un conto alla rovescia impostato per il processo e' terminato. Misura il tempo "virtuale" consumato dal solo processo.
SIGPROF	27	Un conto alla rovescia impostato per il processo e' terminato: misura il tempo di CPU usato dal processo e dal sistema per eseguire azioni istruite dal processo stesso.
SIGWINCH	28	Dimensione della finestra e' cambiato
Siginfo	29	Richiesta di informazioni
SIGUSR1	30	Definito dall'utente
SIGUSR2	31	Definito dall'utente
SIGTHR	32	Thread di interrupt

Segnali prodotti da tastiera



Ctrl+C *(SIGINT)*	interrompe il comando corrente
Ctrl+Z *(SIGSTOP)*	ferma il comando corrente, da continuare con fg in primo piano o in sottofondo con bg
Ctrl+D	esci dalla sessione corrente, simile a exit
Ctrl+W	cancella una parola nella linea corrente
Ctrl+U	cancella l'intera linea
Ctrl+/ *(SIGQUIT)*	uscita invocata da tastiera
Ctrl+R	cicla attraverso la lista dei comandi recenti
!!	ripete l'ultimo comando
exit	esci dalla sessione corrente

Comando kill

Sintassi:

kill –numero/stringa PID

- Il processo che riceve il segnale fa operazioni di default

- kill –l : mostra l'elenco dei segnali disponibili

Comando kill

- root può lanciare segnali a tutti i processi

- gli utenti possono lanciare segnali solo ai processi di cui sono proprietari

Comando killall

-Manda un segnale a tutti i processi specificati

Sintassi

killall [-Z,--context pattern]
[-e,--exact] [-g,--process-group]
[-i,--interactive] [-q,--quiet]
[-r,--regexp] [-s,--signalsignal]
[-u,--user user] [-v,--verbose]
[-w,--wait] [-I,--ignore-case]
[-V,--version] [--] name ...

killall: opzioni principali

- killall -l

  lista tutti i segnali

- killall -s USR1 proc

  invia il segnale USR1 al processo *proc*

- killall -9 proc

  - uccide un processo che non risponde

  (-9 = SIGKILL)

Comando killall

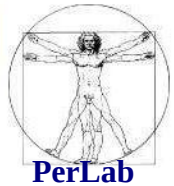
- Segnale di default: SIGTERM

Il segnale puo' essere specificato

- per nome: -HUP o –SIGHUP
- per numero: -1

Background e Foreground

Definizione



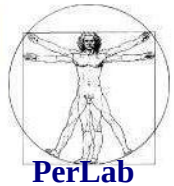
Processo in background:

- processo sullo sfondo
- non necessita di interazione con l'utente ed ha un output limitato
- sintassi: comando [argomenti] &

Processo in foreground:

- processo in primo piano
- l'utente deve attendere il completamento prima di lanciarne un altro
- sintassi: comando [argomenti]

Continuazione



- Un processo messo in pausa col comando [Ctrl-z] può ripartire:

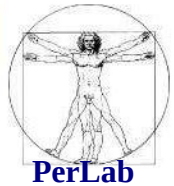
  in background

  Comando “bg” : esegue un processo sospeso in background

  Sintassi: bg

  Comando “fg” : manda l’ultimo processo in background in foreground

Problemi background



- Terminazione o continuazione nel momento in cui si opera una chiusura del terminale
- Automaticamente termina all'atto di chiusura del terminale

  segnale di hangup (SIGHUP)

- Utilizzando il comando “nohup” il processo continua a funzionare anche dopo la chiusura del terminale

Comando nohup

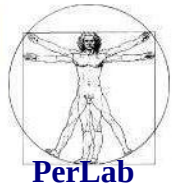
Sintassi:

$ nohup *comando* [> *outcomando]* &

- Se l'utente non provvede a ridirigere esplicitamente l'output di un comando preceduto da nohup, stdout e stderr vengono automaticamente rediretti insieme nel file nohup.out.

Priorità dei Processi

Priorità dei processi



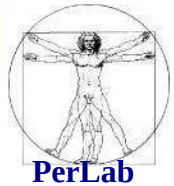
- I processi vengono eseguiti “contemporaneamente” nel senso che utilizzano il processore a turno, per un breve intervallo di tempo.

- “Scheduler" UNIX:

  parte del kernel che organizza tali turni in base al livello di priorità dei processi.

- Livello di priorità di un processo:
  - valore assegnato dal sistema operativo
  - priorità alta corrisponde ad un valore basso

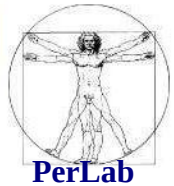
Modifica della priorità



- Si utilizza il valore di *NICE*
- L'utente (normale o super-user) non può intervenire sulla priorità ma solo sul valore NICE
- Il valore NICE
  - si somma alla priorità stabilita dal sistema operativo
  - permette di “rallentare” o “accelerare” un processo
  - valori assegnabili: da -20 a +19
  - solo root può assegnare valori negativi (può “accelerare” un processo)

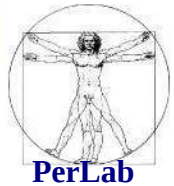
Quando viene creato un nuovo processo, alla variabile NICE viene assegnato un valore intermedio.

Valore NICE



- Se la variabile NICE aumenta il suo valore allora il processo esegue più lentamente alleggerendo il sistema.
- Il superuser
  - può operare nell'intero campo NICE e su tutti i processi di sistema
  - opportune precauzioni soprattutto per i valori fortemente negativi
- L'utente
  - puo' solo aumentare il valore NICE
  - non è possibile ridurre il valore NICE nemmeno per riportarlo al valore precedente

Comando nice



- E' possibile apportare una variazione di priorità di un processo con il comando nice.

Sintassi:

nice [opzioni] [comando [arg1 [arg2 ..]]]

- [opzioni]: opzioni del comando;
  - tipicamente –n delta, con delta il valore da sommare algebricamente al valore di nice
  - default = 10; super user = [-20; +19]; user = [0; +19]
- comando: comando da eseguire che, quando sarà in esecuzione, sarà il processo

Esempio: *nice –n 19 pico*

Comando renice

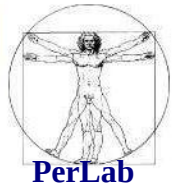
- Permette di variare il livello di NICE di un processo *in esecuzione*
- Un utente può applicare il renice solo ai propri processi
- Il comando renice accetta opzioni che consentono di specificare i processi non solo per PID ma anche per utente o gruppo di processi

Sintassi: renice priorità [[-p] pid ..] [[-g] pgrp ..] [[-u] user ..]

Esempio: renice 19 –p 1134

Altri comandi utili

Comando fuser



- Elenca i processi che utilizzano i file indicati come argomento

Sintassi:

fuser [opzioni] file

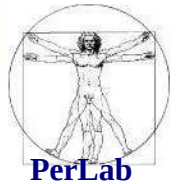
Opzioni principali:

-k : invia un segnale ai processi che usano il file; se non specificato è SIGKILL

-segnale : specifica il segnale da inviare con –k

-v : mostra i risultati in forma tabellare

Comando uptime



- Indica da quanto tempo è in funzione il sistema senza interruzioni

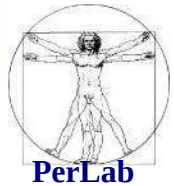
Sintassi:

uptime [opzioni]

Formato risultato:

- orario attuale
- da quanto tempo è in funzione il sistema
- carico medio

Comando free



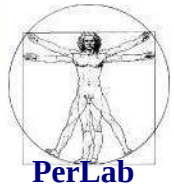
- Fornisce informazioni relative alla memoria reale e virtuale

Sintassi:

free [opzioni]

- Opzioni principali:
  - -s delta : aggiornamento continuo a intervalli regolari, ogni delta secondi
  - -b : valori in byte (altrimenti in Kilobyte)

Risorse e riferimenti



Riferimenti su dispensa:

'Amministrazione di un Sistema UNIX in Rete‘

Cap. 5