Processi

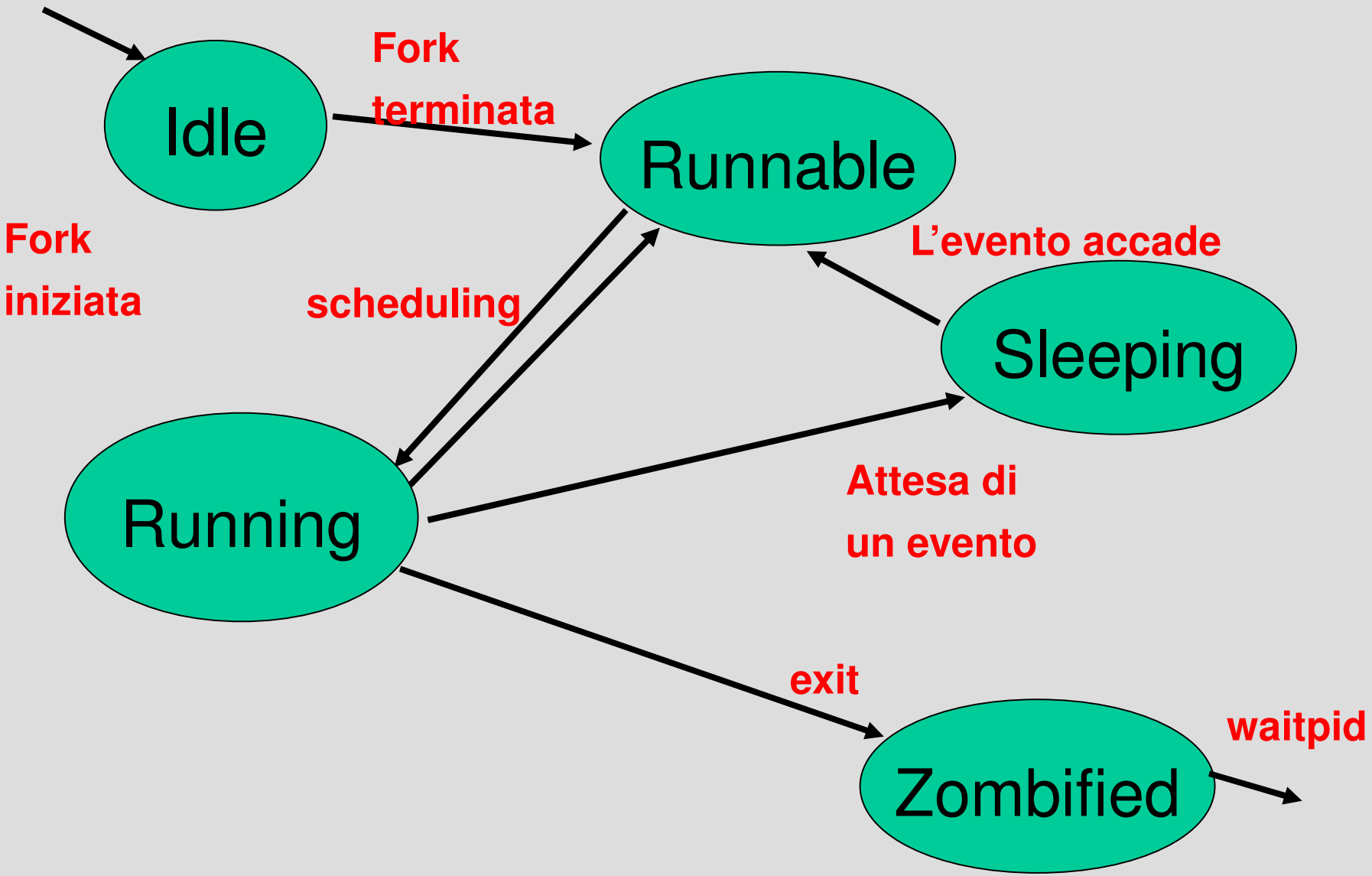
Proprietà di un processo

- Un processo è composto da
  - uno spazio di indirizzamento
    - pagine di memoria
  - un insieme di strutture dati
    - mappa dello spazio di indirizzamento
    - stato corrente (running, sleeping etc. etc.)
    - Priorità di esecuzione
    - Proprietario
    - Risorse utilizzate
    - Segnali bloccati

Proprietà di un processo

- PID
  - *Numero identificativo univoco*
- PPID
  - *PID del processo “padre”*
- UID, EUID
  - *ID dell'utente e ID dell'utente “effettivo” (in pratica solo per setuid)*
- GID, EGID
  - *ID del gruppo e ID del gruppo “effettivo”*
- Nice
  - *Priorità del processo*
- Terminale di controllo
  - *Terminale per l'input/output*

Processi & fork



Processi e Demoni

- Un demone è un processo particolare (ovvero non tutti i processi sono demoni. Un daemon gira in background e offre un servizio su richiesta.
  - Esempi: sendmail (spedizione posta), lpd (gestione stampanti), crond (attività periodiche)
- Un processo è un attività in esecuzione, in genere per un tempo limitato. Può offrire un servizio, quando richiamato da un demone. In genere è attivato da una shell.

Gestione Processi e Demoni

- Daemon:
  - /etc/init.d /etc/rc0.d /etc/rc1.d
- Gestione daemon per Solaris
  - *svcadm* gestione dei processi
  - *svcs* lista dei servizi attivi
- Processi:
  - ctrl+c, ctrl+z, lanciare un processo in background da terminale...
  - *fg*, *bg*, *ps*, *nohup, nice, renice, kill, pkill, fuser, pfiles, pgrep, pstop, prun*...

Segnali

- I segnali sono richieste di interruzione per un processo
  - (ctrl+c, ctrl+z, kill, segnali tra processi, dal kernel)

HUP	1	Riaggancia
INT	2	Interrompe
QUIT	3	Esce
KILL	9	Kill
BUS		Errore di bus
SEGV		Segmentation fault
TERM	15	Terminazione software
STOP		Stop
TSTOP		Stop “soft”
CONT		Continua un processo fermo

Gestire i processi

- Inviare i segnali da tastiera: ctrl+c e ctrl+z
- Inviare i segnali: kill
- Monitorare i processi del terminale: *jobs, bg, fg*
- Modificare la priorità dei processi: *nice* e *renice*
- Monitorare i processi: *ps*
- Monitorare i daemon con SMF

Jobs, i processi da terminale

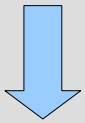
- Un processo lanciato da un terminale viene indicato spesso con “jobs”
  - Il comando *jobs* mostra solo i processi su quel terminale
- Altre operazioni
  - ctrl+c uccide il processo
  - ctrl+z ferma temporaneamente il processo
- Per far ripartire un processo interrotto con ctrl+z
  - *bg %numerojobs* riparte in background
  - *fg %numerojobs* riparte in foreground

Caratteri speciali per il controllo dei processi

- ; esegue più comandi da una sola riga
- & esegue in background un comando
- > ridirigere l'output
  - *ls -l > ppp* l'output viene scritto nel file ppp
- >> ridirigere l'output
  - *ls -l >> ppp* l'output viene aggiunto in coda a ppp
- >& (2>) ridirigere output e errori
  - *ls -l >& ppp* scrive output ed errori su ppp
- | “pipe” - comandi in serie

Processi e terminale: *nohup*

- Quando si lancia un processo da un terminale il processo rimane “agganciato” al terminale stesso, anche se, per esempio lo si mette in foreground



- Uccidere il terminale significa uccidere il processo, perchè il terminale, all'atto di chiusura lancia il segnale SIGHUP al processo figlio:
  - *nohup comando argomenti* quando si lancia il comando
  - *nohup -p numeroprocesso* per un comando già avviato

Visualizzare processi: *ps*

- *ps* è il comando più importante per la visualizzazione dei processi
  - Sfortunatamente le molte opzioni sono spesso differenti tra i vari sistemi UNIX
- Alcuni esempi:
  - *ps* — elenco processi sul terminale
  - *ps -A* — elenco di tutti i processi
  - *ps -elf* — elenco dettagliato
  - *ps -u nomeutente* — cerca i processi di un utente

Visualizzare processi: *ps*

- *ps -e (-A)* lista tutti i processi
- *ps -l* riporta tutti i dettagli sul processo
- *ps -f* riporta tutti i dettagli sul comando

Differenze?

- *ps -a* lista processi più “importanti”
- *ps -o* output personalizzato
- *ps -p* informazioni su un processo specifico

ps -elf

- Esempio di *ps -elf* | *more*

```
bash-3.00$ ps -elf | more
 F S      UID   PID  PPID   C PRI NI     ADDR     SZ    WCHAN    STIME TTY          TIME CMD
 1 T     root     0     0   0   0 SY fec1fac0      0           16:19:51 ?          1:18 sched
 0 S     root     1     0   0  40 20 d2ad6488    623 d21ccc72 16:19:52 ?          0:14 /sbin/init
 1 S     root     2     0   0   0 SY d2ad5bf8      0 fec72db0 16:19:52 ?          0:00 pageout
 1 S     root     3     0   0   0 SY d2ad5368      0 fec83cd4 16:19:52 ?          0:19 fsflush
 0 S   daemon   121     1   0  40 20 d2ad4ad8   1057 d6d0ed16 16:20:34 ?          0:00 /usr/lib/crypto/kcfd
 0 S     root     7     1   0  40 20 d2ad4248   2866 d2926316 16:19:55 ?          0:17 /lib/svc/bin/svc.startd
 0 S     root     9     1   0  40 20 d2ad39b8   2262 d2926716 16:19:55 ?          0:39 /lib/svc/bin/svc.configd
 0 S     root   333     7   0  40 20 d2ad3128    519 d32e4e60 16:21:08 ?          0:00 /usr/lib/saf/sac -t 300
 0 S     root   142     1   0  40 20 d2ad2898   1345 d34d4716 16:20:35 ?          0:00 /usr/lib/sysevent/syseventd
 0 S     root   402     1   0  40 20 d2ad2008    517 d9a901b2 16:21:12 ?          0:00 /usr/sadm/lib/smc/bin/smcboot
 0 S     root    61     1   0  40 20 d2ad1490   1076 d6bba732 16:20:14 ?          0:00 /sbin/dhcpagent
 0 S     root   338   333   0  40 20 d2ad0c00    610 d3422316 16:21:08 ?          0:00 /usr/lib/saf/ttymon
 0 S     root   408   402   0  40 20 d2ad0370    517 d9a902b2 16:21:13 ?          0:00 /usr/sadm/lib/smc/bin/smcboot
 0 S     root   356     1   0  40 20 d2acfae0   3590 d6d0e116 16:21:10 ?          0:02 /usr/lib/fm/fmd/fmd
 0 S     root   154     1   0  40 20 d2acf250    739 d8403f16 16:20:37 ?          0:00 /usr/lib/picl/picld
 0 S     root   310   309   0  40 20 d2ace9c0    709 d9a90df2 16:21:07 ?          0:00 /usr/lib/autofs/automountd
 0 S     root   136     1   0  40 20 d2ace130   1814 d6cc9f16 16:20:35 ?          0:03 /usr/sbin/nscd
 0 S     root   139     1   0  40 20 d2acd8a0    431 d3413316 16:20:35 ?          0:00 /usr/lib/power/powerd
 0 S   daemon   304     1   0  40 20 d2acd010    691 d9a90fb2 16:21:06 ?          0:00 /usr/sbin/rpcbind
 0 S     root   309     1   0  40 20 d2acc498    638 d2acc504 16:21:07 ?          0:00 /usr/lib/autofs/automountd
 0 S     root   611     1   0  40 20 d2acbc08   1342 d9d3a316 16:21:37 ?          0:00 devfsadmd
-- Continua --
```

ps -elf

- S stato del processo
  - R running
  - T stopped
  - S sleeping
  - W waiting
  - Z zombie

- UID, PID, PPID numeri identificativi (utente, processo etc.)
- PRI Priorità (numeri alti, bassa priorità)
- ADDR Indirizzo di memoria
- SZ Dimensione totale del processo
- WCHAN L'indirizzo di memoria per il quale il processo è S
- STIME Starting time del processo
- TTY Il terminale dal quale il processo è stato lanciato
- TIME Tempo totale di esecuzione
- CMD la riga di comando del processo

Visualizzare processi: *ps*

- *ps -G GID*
  - Elenco dei processi del gruppo avente numero di gruppo GID
- *ps -U UID*
  - Elenco dei processi del'utente avente numero identificativo UID
- *ps -p numero_processo*
  - Elenco di processi specifici
- *ps -t numero_terminale*
  - Elenco dei processi su di un terminale specifico

Terminare un processo: *kill*

- *kill* invia un segnale al processo. Di default invia un TERM
  - *kill -s segnale numeroprocesso*

    oppure
  - *kill -numsegnale numeroprocesso*

- Altra opzione utile
  - *kill -l* mostra i segnali che kill può inviare

Priorità di un processo: *nice* e *renice*

- Per attribuire una priorità differente da quella di default ad un processo lanciato da riga di comando si utilizza il comando *nice*
  - *nice ±num nomecomando*
- Il proprietario del processo può diminuire la priorità in ogni istante con *renice*. L'unico che può alzare la priorità è root.
  - *nice ±num numerocomando*
- Purtroppo, di volta in volta, *num* indica il valore assoluto della priorità oppure una differenza dalla priorità differente (anche tra *nice* e *renice* in uno stesso sistema ci possono essere differenze)

Monitorare i processi: *top*, *prstat*

```
top - 16:50:53 up  6:43,  1 user,  load average: 0.12, 0.19, 0.17
Tasks: 140 total,   5 running, 134 sleeping,   1 stopped,   0 zombie
Cpu(s):  8.7%us,  1.1%sy,  0.0%ni, 89.9%id,  0.3%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:   2824244k total,  2797140k used,    27104k free,   139736k buffers
Swap:  1028120k total,       44k used,  1028076k free,  1646264k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 8594 root      20   0  262m 154m  12m R    7  5.6  12:29.23 X
12107 ale       20   0  923m 263m  40m S    5  9.6  33:04.04 firefox
13785 ale       20   0  396m  64m  22m S    3  2.4  13:20.48 amule
15211 ale       20   0  158m  23m  15m S    2  0.8   0:00.41 ksnapshot
12092 ale       20   0  152m  15m  10m S    0  0.6   0:00.32 korgac
    1 root      20   0  3748  600  504 S    0  0.0   0:01.01 init
    2 root      15  -5     0    0    0 S    0  0.0   0:00.00 kthreadd
    3 root      RT  -5     0    0    0 S    0  0.0   0:00.00 migration/0
    4 root      15  -5     0    0    0 S    0  0.0   0:00.03 ksoftirqd/0
    5 root      RT  -5     0    0    0 S    0  0.0   0:00.00 migration/1
    6 root      15  -5     0    0    0 S    0  0.0   0:00.04 ksoftirqd/1
    7 root      15  -5     0    0    0 S    0  0.0   0:00.15 events/0
    8 root      15  -5     0    0    0 S    0  0.0   0:00.27 events/1
    9 root      15  -5     0    0    0 S    0  0.0   0:00.00 khelper
  119 root      15  -5     0    0    0 S    0  0.0   0:00.00 kintegrityd/0
```

pgrep, pkill

- *pgrep* process grep, ricerca i processi
- *pkill* pgrep+kill, cerca e uccide processi
  - -u (-g) cerca per utenti (gruppi)
  - -l output esteso
  - -n (-o) solo i processi più nuovi, più vecchi
  - -v inverte il significato della ricerca
  - -signal specifica quale segnale inviare

fuser, pfiles, ptree, prun

- *fuser* identifica processo e utente utilizzatore del file o di un dispositivo montato
  - -u mostra l'utente
- *pfiles* Trova i file che utilizza il processo
- *ptree* Elenca i processi gerarchicamente
- *prun, pstop, pwait, psig...*
  - Vari proc tools per gestione dei processi

SMF
(Solaris10)

- *svcs* stato del servizio
- *svcadm* per gestire i servizi (enable/disable)
- *svcprop* lista le proprietà di un servizio
- *svccfg* gestione servizi
- *inetadm* gestisce il servizio inetd (gestore dei servizi di rete)

SMF: *scvs*

- -a mostra tutti i servizi, attivi e non attivi
- -d lista le dipendenze
- -D lista le dipendenze
- -l mostra informazioni dettagliate
- -p lista i PID associati al servizio
- -x spiegazioni dello stato dei servizi
- -v ulteriori informazioni

Che comando sto usando? which

- *which nomecomando*
  - Scrive in output il percorso completo del nomecomando che viene eseguito. Talvolta, infatti ci possono essere vari comandi con lo stesso nome localizzati in più percorsi.
  - Es: *fssnap* (comando per backup) è localizzato in /usr/sbin e in /usr/lib/fs/ufs/fssnap

Processi: cartella /proc

- Come detto in UNIX tutto è un file, anche i processi, che infatti si trovano in una cartella che in genere è /proc

```
bash-3.00# ls /proc
0     1102  146   3     329   362   394   490   578   640   718   768   777   827   870   9     929
1     128   152   304   337   378   418   505   581   7     743   770   797   829   880   905   937
1000  134   2     326   345   390   419   506   61    714   747   772   815   836   887   915
1091  139   253   327   347   391   426   544   624   717   765   775   817   868   889   919
```

- In questo caso ogni processo ha una sua cartella con il nome uguale al suo PID.
- E' possibile uccidere un processo cancellandolo da /proc!

Scheduling dei processi

- Daemon che gestisce i processi periodici: crond
- Comandi per la gestione
  - *crontab*
  - *at*
- File
  - /var/spool/cron/crontab
  - /var/spool/cron/atjobs
  - /etc/crontab
  - /etc/cron.allow — default: nessuno può utilizzare crontab
  - /etc/cron.deny
  - /etc/at.allow (/etc/at.deny)

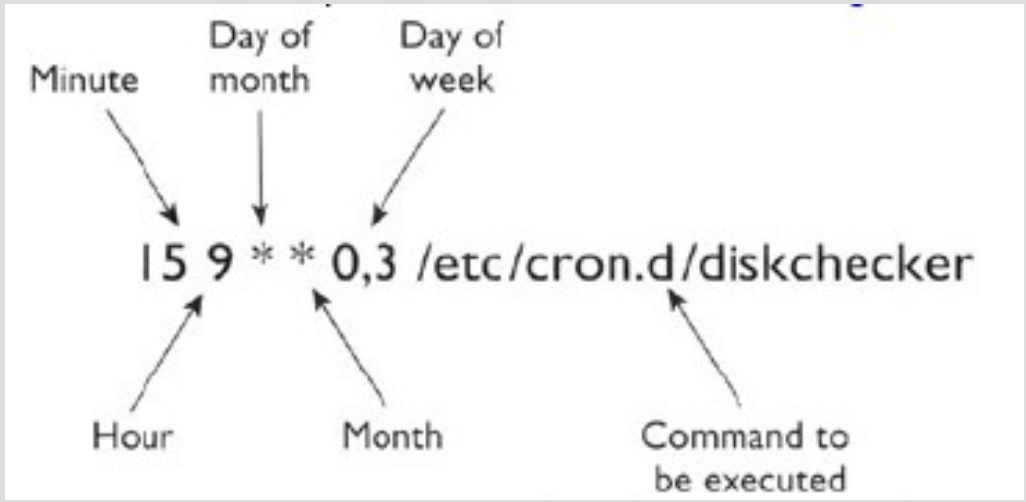
crond

- Il demone crond compie le seguenti operazioni
  - Controlla i nuovi file crontab e atjobs
  - Legge i comandi e gli orari all'interno dei file
  - Esegue i comandi all'orario stabilito
  - Attende nuove notifiche dai comandi *crontab* e *at*

Scheduling dei processi

- *crontab* gestione dei file crontab
  - *crontab -l* lista i processi
  - *crontab -e* crea un file crontab
  - *crontab -r* rimuove un processo

- Formato di un file crontab



at

- *at [-m] <time> [<date>]*
  - Esegue un comando solo per una volta
  - Dopo il comando *at*, si entra in un prompt dove vanno immessi i comandi da eseguirsi. Si termina con CTRL+D
  - time ora di esecuzione, nel formato hhmm, oppure “nidnigh”, “noon”, “now”
  - date data, 3 lettere del mese, giorno del mese oppure “today”, “tomorrow”