www.uom.gr
ΠΑΝΕΠΙΣΤΗΜΙΟ ΜΑΚΕΔΟΝΙΑΣ
ΟΙΚΟΝΟΜΙΚΩΝ ΚΑΙ ΚΟΙΝΩΝΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
τμήμα Εφαρμοσμένης Πληροφορικής

Προγραμματισμός Διαχείρισης Συστημάτων ΙΙ

Μάθημα 3ο

Έλεγχος διεργασιών και Χρονοπρογραμματισμός εργασιών

Μιχαηλίδης Παναγιώτης

Περιεχόμενα

- Έλεγχος διεργασιών
  - Συστατικά μιας διεργασίας
  - Κύκλος ζωής μιας διεργασίας
  - Σήματα
  - Αποστολή σημάτων χρησιμοποιώντας kill και killall
  - Καταστάσεις διεργασίας
  - Επίδραση της πολιτικής προτεραιοτήτων με τις nice και renice
  - Παρακολούθηση διεργασιών με τις ps και top
  - Διεργασίες runaway

Συστατικά μιας διεργασίας

- Μια διεργασία είναι στιγμιότυπο ενός προγράμματος
- Από την σκοπιά του πυρήνα, μια διεργασία είναι:
  - Ένας χώρος διευθύνσεων (σύνολο σελίδων μνήμης με κώδικα, βιβλιοθήκες και δεδομένα)
  - Σύνολο δομών δεδομένων (μέσα στον πυρήνα)
    - Αντιστοίχιση του χώρου διευθύνσεων των διεργασιών
    - Τρέχουσα κατάσταση
    - Εκτέλεση προτεραιότητας
    - Χρήση πόρων
    - Μοναδική μάσκα (ποια σήματα είναι μπλοκαρισμένα)
    - Ο ιδιοκτήτης
    - Ποιες εντολές εκτελούνται στην τρέχουσα στιγμή

Χαρακτηριστικά μιας διεργασίας

- Ταυτότητα διεργασία – PID
  - Μοναδική ταυτότητα, wraps around
- Ταυτότητα γονέας – PPID
  - Όταν δημιουργείται μια διεργασία, υπάρχει ένας γονέας και μια θυγατρική
- Πραγματική και αποτελεσματική ταυτότητα χρήστη - UID και EUID
  - EUID χρησιμοποιείται για να προσδιορίζει τι άδειες έχει η διεργασία
  - Επίσης καταγράφει την αρχική EUID (αποθηκεύει UID)
    - Μπορεί να ξαναπροσπελαστεί αργότερα στο πρόγραμμα (ακόμα και μετά την αλλαγή EUID)
- Πραγματική και αποτελεσματική ταυτότητα ομάδας - GID και EGID

Χαρακτηριστικά μιας διεργασίας

- Niceness
  - Ο διαθέσιμος χρόνος CPU εξαρτάται από την πολιτική προτεραιοτήτων
  - Οι χρήστες μπορούν να κάνουν τις διεργασίες τους "nicer" για το υπόλοιπο του συστήματος

Κύκλος ζωής μιας διεργασίας

- Μια υπάρχουσα διεργασία καλεί την fork(2)
  - Parent is told PID of child
  - Child is told 0
- Η θυγατρική χρησιμοποιεί την exec για να ξεκινήσει ένα καινούργιο πρόγραμμα
- Όταν είναι έτοιμη να τερματίζει, η διεργασία καλεί exit(2) με τον κώδικα εξόδου
  - Η διεργασία γίνεται zombie
- Ο γονέας πρέπει να περιμένει wait(2) για να συλλέξει την κατάσταση της νεκρής θυγατρικής
  - Χρήση πόρων, γιατί τερματίστηκε
- Οι ορφανές αναλαμβάνονται από την init

Σήματα

- Τα σήματα είναι αιτήσεις διακοπής σε επίπεδο διεργασιών
- Χρήσεις
  - Διαδιεργασιακή επικοινωνία
  - Οδηγός τερματικού μπορεί να τερματίσει, να διακόψει ή να αναβάλει τις διεργασίες (Ctrl-C, Ctrl-Z)
  - Μπορούν να αποσταλούν από τον διαχειριστή συστήματος (με την kill) για διάφορους σκοπούς
  - Μπορούν να αποσταλούν από τον πυρήνα όταν μια διεργασία παραβιάζει ένα κανόνα
    - π.χ. Διαίρεση με το μηδέν

Χειρισμός σημάτων

- Η διεργασία μπορεί να προσδιορίζει ένα χειριστή σήματος για ένα συγκεκριμένο σήμα
- Αν δεν υπάρχει χειριστής, τότε ο πυρήνας παίρνει την εξ ορισμού δράση
- Όταν τελειώσει ο χειριστής να συλλάβει το σήμα, συνεχίζεται η εκτέλεση από το σημείο που έλαβε το σήμα
- Η διεργασία μπορεί να ζητήσει συγκεκριμένα σήματα να αγνοούνται ή να μπλοκαρίζονται
- Αν λάβει ένα σήμα ενώ είναι μπλοκαρισμένη, ένα στιγμιότυπο του σήματος αποθηκεύεται μέχρι να ξεμπλοκαριστεί

Σημαντικά σήματα

#	Name	Description	Default	Catch?	Block?	Dump?
1	HUP	Hangup	Terminate	Yes	Yes	No
	Reset request; clean up process on terminal (modem hangup)					
	**csh processes ignore HUP; bash users need nohup command*					
2	INT	Interrupt	Terminate	Yes	Yes	No
	Control-C, can catch and clean up before quiting.					
3	QUIT	Quit	Terminate	Yes	Yes	Yes
	Similar to TERM, but generates a core dump					
9	KILL	Kill	Terminate	No	No	No
	Never received by process; OS terminates process.					
*	BUS	Bus error	Terminate	Yes	Yes	Yes
	Error signal. Typically memory alignment problem.					
11	SEGV	Segmentation Fault	Terminate	Yes	Yes	Yes
	Error signal. Typically memory access to protected space.					

Περισσότερα σήματα

#	Name	Description	Default	Catch?	Block?	Dump?
15	TERM	Software termination	Terminate	Yes	Yes	No
	Request to terminate execution. Process can clean up, exit.					
*	STOP	Stop	Stop	No	No	No
	OS suspends execution of process until CONT received.					
*	TSTP	Keyboard stop	Stop	Yes	Yes	Yes
	Keyboard Ctrl-Z request to stop. Catchable.					
*	CONT	Continue after stop	Ignore	Yes	No	No
	Continue after STOP.					
*	WINCH	Window changed	Ignore	Yes	Yes	No
	Sent by terminal emulator when config changes (resize)					
*	USR1	User-defined	Terminate	Yes	Yes	No
	User defined. Apache restarts gracefully.					
*	USR2	User-defined	Terminate	Yes	Yes	No

Αποστολή σημάτων

- **`kill [-signal] pid`**
- **`kill`** στέλνει το σήμα TERM εξ ορισμού
- **`kill -9 pid === kill -KILL pid`**
  - "Εγγυάται" ότι η διεργασία θα τερματιστεί
- **`kill -USR1 910 3044`**
- **`killall`** δεν χρειάζεται το pid
- **`sudo killall -USR1 httpd`**

Καταστάσεις διεργασίας

- Η διεργασία βρίσκεται σε μια από τις τέσσερις καταστάσεις
  - Εκτελέσιμη – Μπορεί να εκτελεστεί
  - Αναμονή – Περιμένει για μερικούς πόρους
    - Δεν παίρνει χρόνο CPU μέχρι ο πόρος να είναι διαθέσιμος
  - Zombie – προσπαθεί να τερματιστεί
  - Τερματισμός - η διεργασία αναβάλεται (δεν επιτρέπεται να εκτελεστεί)
    - Όπως στην αναμονή, δεν μπορεί να αφυπνιστεί μέχρι να λάβει CONT

Πολιτική προτεραιοτήτων

- "Niceness" είναι συμβουλή στον πυρήνα σχετικά με το πως θα δρομολογήσει την διεργασία
- Το Linux κυμαίνεται από -20 (υψηλή προτεραιότητα, όχι nice) μέχρι +19 (χαμηλή προτεραιότητα, πολύ nice), 0 είναι εξ ορισμού
- Χρήστη/διεργασία μπορούν να αυξήσουν αλλά όχι χαμηλότερο niceness
  - Ο root μπορεί να είναι χαμηλότερο
- Παραδείγματα
  - `$ nice +5 ~/bin/longtask`
  - `$ renice -5 8829`
  - `$ sudo renice 5 -u boggs`

Παρακολούθηση διεργασιών: ps

- /bin/ps κύριο εργαλείο
- Δείχνει
  - PID, UID, προτεραιότητα, έλεγχος τερματικού
  - Χρήση μνήμης, χρόνος CPU, κατάσταση
- Πολλές παραλλαγές της ps
  - ps -aux (BSD, Linux)
  - ps -alf (Solaris)

Παράδειγμα εξόδου της ps

```
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0   1364    64 ?        S     2003   3:03 init [5]   --init
root         2  0.0  0.0      0     0 ?        SW    2003   1:35 [keventd]
root         3  0.0  0.0      0     0 ?        SWN   2003   0:27 [ksoftirqd_CPU0]
root         5  0.1  0.0      0     0 ?        SW    2003 465:05 [kswapd]
root         6  3.0  0.0      0     0 ?        SW    2003 7754:49 [kscand]
root         7  0.0  0.0      0     0 ?        SW    2003   1:16 [bdflush]
root         8  0.0  0.0      0     0 ?        SW    2003   4:06 [kupdated]
root         9  0.0  0.0      0     0 ?        SW<   2003   0:00 [mdrecoveryd]
root        13  0.0  0.0      0     0 ?        SW    2003  16:12 [kjournald]
root        92  0.0  0.0      0     0 ?        SW    2003   0:00 [khubd]
root       589  0.0  0.0      0     0 ?        SW    2003   0:01 [eth0]
root       761  0.0  0.0   1424   340 ?        S     2003   0:48 syslogd -m 0
root       766  0.0  0.0   1364   244 ?        S     2003   0:00 klogd -x
rpc        786  0.0  0.0   1524   360 ?        S     2003   0:22 portmap
rpcuser    814  0.0  0.0   1660   484 ?        S     2003   1:27 rpc.statd
ntp        933  0.0  0.0   1884  1880 ?        SL    2003  11:18 ntpd -U ntp -g
root      1045  0.0  0.0   2140   164 ?        S     2003   0:00 xinetd -stayalive
root      1092  0.0  0.0   1796   176 ?        S     2003   0:00 rpc.rquotad
root      1097  0.1  0.0      0     0 ?        SW    2003 267:24 [nfsd]
```

Παρακολούθηση διεργασιών: top

- /usr/bin/top είναι εναλλακτικό για μερικά ΛΣ
- Δείχνει τις πρώτες-n CPU χρησιμοποιούμενες διεργασίες
  - Συν άλλα στατιστικά όπως η χρήση μνήμης, φορτίο συστήματος
  - Μπορεί renice εντός top
  - Οι πληροφορίες ανανεώνονται αυτόματα κάθε 5 δευτερόλεπτα
  - Μπορεί να επικεντρωθεί σε ένα συγκεκριμένο χρήστη

Παράδειγμα εξόδου της top

```
 10:42pm  up 176 days,  1:23,  4 users,  load average: 0.04, 0.02, 0.00
117 processes: 116 sleeping, 1 running, 0 zombie, 0 stopped
CPU0 states:  0.0% user,  2.2% system,  0.0% nice, 97.2% idle
CPU1 states:  0.2% user,  0.1% system,  0.0% nice, 99.2% idle
Mem:  2064828K av, 2050848K used,   13980K free,       0K shrd,  131000K buff
Swap: 2047744K av, 1116880K used,  930864K free                 1744800K cached

  PID USER     PRI  NI  SIZE  RSS SHARE STAT %CPU %MEM   TIME COMMAND
    6 root      16   0     0    0     0 SW    2.5  0.0  7755m kscand
18460 root       9   0  1872 1828  1456 S     0.1  0.0   0:00 sshd
18533 kiran      9   0   508  504   408 S     0.1  0.0   0:00 muddleftpd
18605 brian     10   0  1092 1092   848 R     0.1  0.0   0:00 top
    1 root       9   0   100   68    48 S     0.0  0.0   3:03 init
    2 root       9   0     0    0     0 SW    0.0  0.0   1:35 keventd
    3 root      19  19     0    0     0 SWN   0.0  0.0   0:27 ksoftirqd_CPU0
    5 root       9   0     0    0     0 SW    0.0  0.0 465:05 kswapd
    7 root       9   0     0    0     0 SW    0.0  0.0   1:16 bdflush
    8 root       9   0     0    0     0 SW    0.0  0.0   4:06 kupdated
    9 root      -1 -20     0    0     0 SW<   0.0  0.0   0:00 mdrecoveryd
   13 root       9   0     0    0     0 SW    0.0  0.0  16:12 kjournald
   92 root       9   0     0    0     0 SW    0.0  0.0   0:00 khubd
  235 root       9   0     0    0     0 SW    0.0  0.0   0:00 kjournald
  589 root       9   0     0    0     0 SW    0.0  0.0   0:01 eth0
  680 root       9   0     0    0     0 SW    0.0  0.0   0:02 eth1
  761 root       9   0   384  340   300 S     0.0  0.0   0:48 syslogd
```

Διεργασίες runaway

- Τι μπορούμε να κάνουμε σχετικά με τις διεργασίες που χρησιμοποιούν μια ασυνήθιστη ποσότητα πόρων (μνήμη, CPU, χώρος δίσκου);
  - Αναγνωρίζουμε τις μεγάλες ποσότητες πόρων με τις εντολές ps και top
  - Ερχόμαστε σε επαφή με τον ιδιοκτήτη και το ρωτάμε σχετικά με την χρήση των πόρων
  - Αναβάλουμε χρησιμοποιώντας το σήμα STOP (ίσως τερματίζουμε την εργασία)
    - Ερχόμαστε σε επαφή με τον ιδιοκτήτη για επανεκκίνηση ή τερματισμός αργότερα
  - Renice τις ποσότητες CPU

Δημιουργία περιοδικών διεργασιών

- Ο αυτοματισμός είναι το κλειδί της απόδοσης
- Αντί να εκτελούμε εργασίες χειροκίνητα καθημερινά, εβδομαδιαία ή μηνιαία μπορούμε να τα χρονοδρομολογήσουμε
  - **cron**
  - **anacron**
- Περιλαμβάνει εργασίες όπως:
  - παρακολούθηση, καταγραφή, εφεδρικά αντίγραφα, διανομή αρχείων

cron

- Ο δαίμονας cron εκτελεί εργασίες στις συγκεκριμένες χρονικές στιγμές
- Εξετάζονται αρχεία crontab από τον cron για χρονοδρομολόγηση
  - /etc/crontab, /etc/cron.d/*, /var/spool/cron/*
- Ο cron αφυπνίζεται κάθε λεπτό και ελέγχει να εξετάζει αν υπάρχει κάτι για να εκτελεστεί
- Ο cron είναι ευαίσθητος σε αλλαγές στο χρόνο
- anacron δουλεύει καθημερινά
  - Καταγράφει πότε εκτελέστηκε τελευταία φορά η εργασία και συλλαμβάνει το χαμένο χρόνο

Αρχεία crontab

- Δομή:
  - λεπτά ώρα μέρα μήνα ημέρα_εβδομάδα [όνομα_χρήστη] εντολή
- Το όνομα χρήστη δεν αναφέρεται στα αρχεία /var/spool/cron (χρησιμοποιείται το όνομα αρχείου)
- Παραδείγματα εγγραφών crontab:

```
# run make at 2:30 each Monday morning
30 2 * * 1 (cd /home/joe4/project; make)
# remove files in /tmp not accessed in 3 days
20 1 * * * find /tmp -a atime +3 -exec rm -f {}
  ';'
# run system activity accounting tool every 10
  minutes
*/10 * * * * root /usr/lib/sa/sa1 1 1
```

Διαχείριση crontabs

- Χρησιμοποιούμε την crontab -e για επεξεργασία
  - Δημιουργεί ένα αντίγραφο
  - Χρησιμοποιεί την μεταβλητή περιβάλλοντος EDITOR
  - Ξαναυποβάλεται στο κατάλογο /var/spool/cron
- crontab -l θα εμφανίζει τα περιεχόμενα στην πρότυπη έξοδο (stdout)
- Τα αρχεία /etc/cron.allow και /etc/cron.deny μπορούν να ελέγξουν την πρόσβαση των υπηρεσιών cron

Χρήση cron

- Οι διανομές εγκαθιστούν εγγραφές crontab για αυτοματοποίηση εκτέλεση σεναρίων στις
  - /etc/cron.monthly/
  - /etc/cron.weekly/
  - /etc/cron.daily/
  - /etc/cron.hourly/
- Τυπικές εργασίες:
  - Εκκάθαριση του συστήματος αρχείων (αρχεία editor, αρχεία core) χρησιμοποιώντας την εντολή find
  - Διανομή αρχείων (mail aliases, sendmail config κλπ) χρησιμοποιώντας τις rsync, rdist
  - Καταγραφή