ΤΕΙ ΚΕΝΤΡΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

ΑΝΟΙΚΤΑ ΑΚΑΔΗΜΑΙΚΑ ΜΑΘΗΜΑΤΑ

ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ ΙΙ (Θ)

Ενότητα 7: ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ ΙΙ

- Νικολαΐδης Αθανάσιος
- Διδάκτορας Ανάπτυξης Τεχνικών Προστασίας Πληροφορίας Εικόνας
- ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΤΕ

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

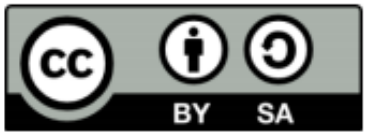
ΕΣΠΑ 2007-2013
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.

- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.



Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «Ανοικτά Ακαδημαϊκά Μαθήματα στο ΤΕΙ Κεντρικής Μακεδονίας» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Ενότητα 7

ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ Ι

Νικολαΐδης Αθανάσιος
Διδάκτορας Ανάπτυξης Τεχνικών Προστασίας Πληροφορίας Εικόνας

Περιεχόμενα ενότητας

Σκοποί ενότητας

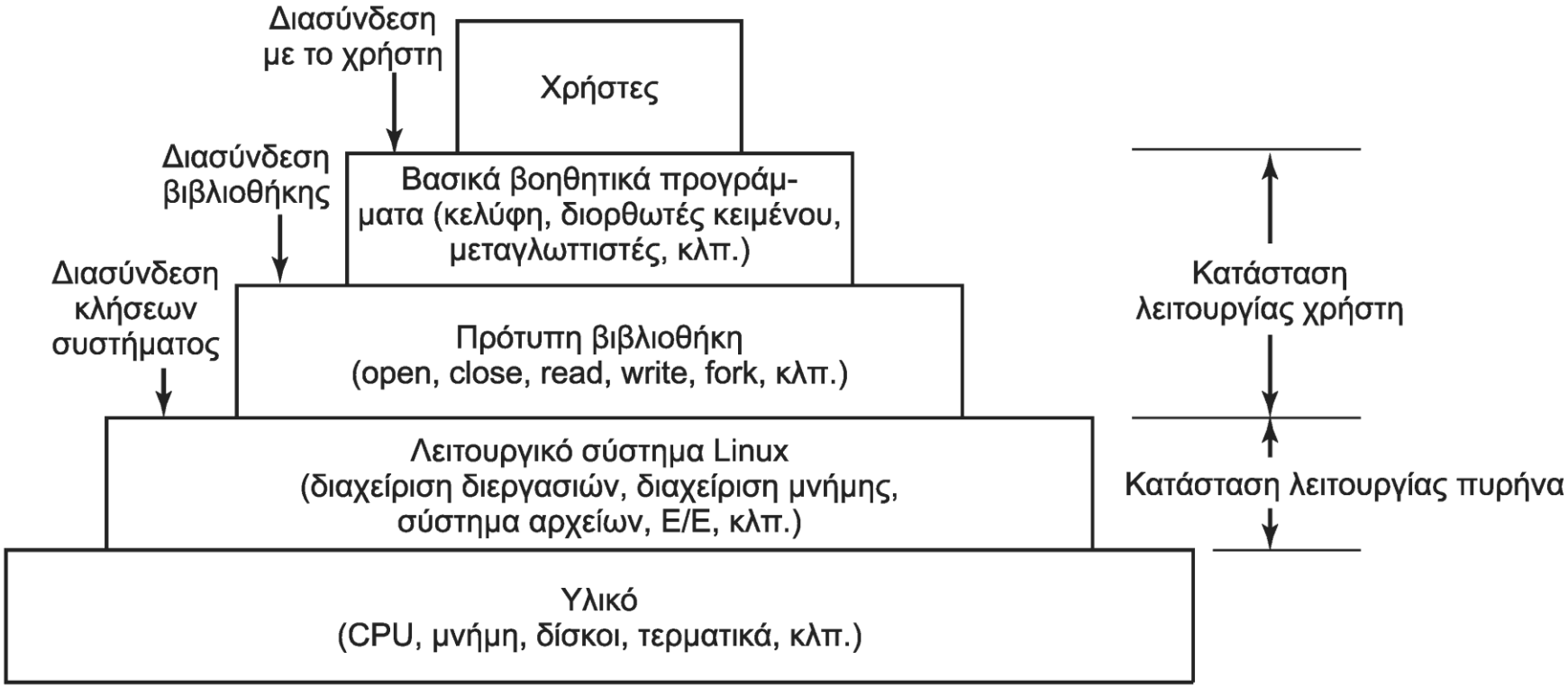
Ιστορία του UNIX

- Προήλθε από το MULTICS (Multiplexed Information and Computing Service) των M.I.T., Bell Labs και GE.
- Δύο βασικές εκδόσεις:
  - AT&T System V Release 4 (SVR4)
    - Αρχικά αναπτύχθηκε από την AT&T, τώρα SCO
  - BSD (Berkeley Software Distribution)

Σύνοψη του UNIX

- Υποστηρίζει διάφορες αρχιτεκτονικές
- Η δομή του ποικίλει
- Υποστηρίζει προεκτοπιστική πολυεπεξεργασία
- Πολυχρηστικό περιβάλλον – γενικά ασφαλές
- Υποστηρίζει πολυνηματικές εφαρμογές
- Προστασία/ασφάλεια υψηλή σε σύγχρονες εκδόσεις
- Υποστηρίζει συμμετρική πολυεπεξεργασία
- Υψηλά κλιμακώσιμο/φορητό σε διάφορα συστήματα
- Πολλοί τύποι UNIX υπάρχουν

Επίπεδα του UNIX

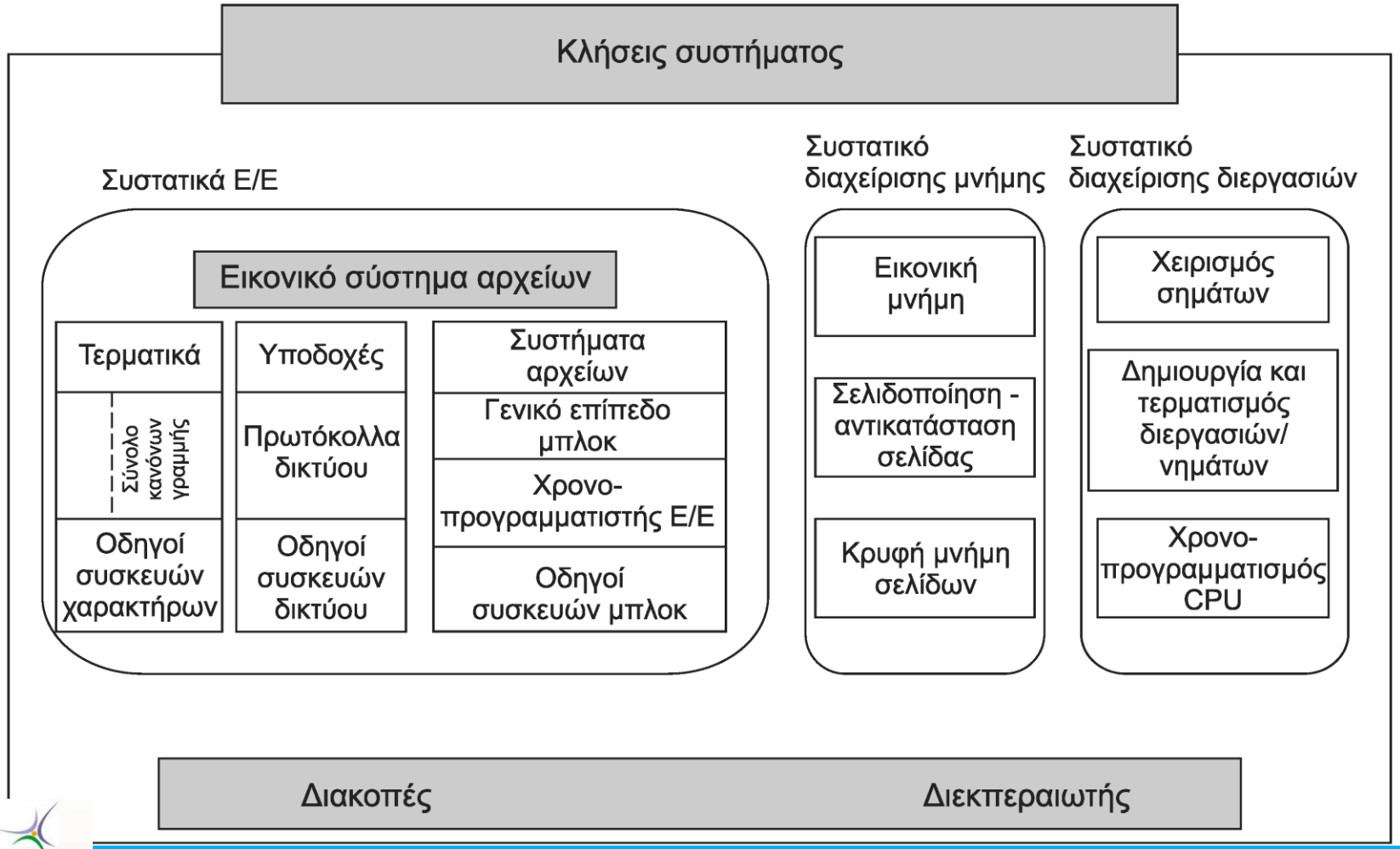


Βοηθητικά προγράμματα του UNIX

Πρόγραμμα	Τυπική χρήση
cat	Συνένωση αρχείων και αποστολή του αποτελέσματος στην καθιερωμένη έξοδο
chmod	Αλλαγή των δικαιωμάτων πρόσβασης σε αρχεία
cp	Δημιουργία αντιγράφου ενός η περισσότερων αρχείων
cut	Αποκοπή στηλών κειμένου από ένα αρχείο
grep	Αναζήτηση σε ένα αρχείο για κάποιο μοτίβο
head	Εξαγωγή των αρχικών γραμμών ενός αρχείου
ls	Εμφάνιση πληροφοριών για έναν κατάλογο
make	Μεταγλώττιση αρχείων για τη δημιουργία δυαδικού αρχείου
mkdir	Δημιουργία καταλόγου
od	Εμφάνιση περιεχομένων αρχείου στο οκταδικό σύστημα
paste	Επικόλληση στηλών κειμένου σε ένα αρχείο
pr	Μορφοποίηση ενός αρχείου για αποστολή στον εκτυπωτή
ps	Εμφάνιση των εκτελούμενων διεργασιών
rm	Διαγραφή ενός ή περισσότερων αρχείων
rmdir	Διαγραφή καταλόγου
sort	Αλφαβητική ταξινόμηση των γραμμών ενός αρχείου
tail	Εξαγωγή των τελευταίων γραμμών ενός αρχείου
tr	Μετάφραση μεταξύ συνόλων χαρακτήρων

Μερικά από τα πιο συνηθισμένα βοηθητικά προγράμματα του UNIX που απαιτεί το πρότυπο POSIX.

Δομή πυρήνα του UNIX



Δημιουργία διεργασίας στο UNIX

```
pid = fork();                   /* αν η fork επιτύχει, ένα PID > 0 επιστρέφεται στη μητρική διεργασία */
if (pid < 0) {
    handle_error();             /* η fork απέτυχε (π.χ. λόγω έλλειψης μνήμης ή γιατί κάποιος */
                                /* πίνακας του συστήματος ήταν γεμάτος */

} else if (pid > 0) {
                                /* ο κώδικας της μητρικής διεργασίας τοποθετείται εδώ */
} else {
                                /* ο κώδικας της θυγατρικής διεργασίας τοποθετείται εδώ */
}
```

Απλοποιημένο κέλυφος POSIX

```
while (TRUE) {                          /* επανάληψη συνεχώς */
    type_prompt();                      /* εκτύπωση του προτρεπτικού μηνύματος στην οθόνη */
    read_command(command, parameters);  /* ανάγνωση της εισόδου από το τερματικό */

    pid = fork();                       /* δημιουργία θυγατρικής διεργασίας */
    if (pid < 0) {
            printf("Unable to fork");   /* συνθήκη σφάλματος */
            continue;                   /* επανάληψη βρόχου */
    }

    if (pid!= 0) {
        waitpid(–1, &status, 0);        /* η μητρική περιμένει τη θυγατρική */
    } else {
        execve (command, parameters, 0); /* η θυγατρική κάνει τη δουλειά */
    }
}
```

Πίνακας διεργασιών στο UNIX

- Κάθε εγγραφή αφορά μια διεργασία και περιλαμβάνονται:
- 1. Παράμετροι χρονοπρογραμματισμού.
- 2. Εικόνα μνήμης.
- 3. Σήματα.
- 4. Καταχωρητές μηχανής.
- 5. Κατάσταση κλήσης συστήματος.
- 6. Πίνακας περιγραφέων αρχείων.
- 7. Λογιστικές πληροφορίες.
- 8. Στοίβα πυρήνα.
- 9. Διάφορα.

Θέματα διεργασιών UNIX

- ***Δαίμονες*** (διεργασίες παρασκηνίου)
  - ***Δαίμονας Cron*** που ενεργοποιείται κάθε λεπτό για να ελέγξει δρομολογημένα συμβάντα (π.χ. αντίγραφο ασφαλείας δίσκου)
- ***Σωληνώσεις*** – συγχρονισμένα κανάλια μεταξύ διεργασιών για να περνούνε ρεύματα από bytes
- ***Σήματα*** – διακοπές λογισμικού που χρησιμοποιούνται για διαδιεργασιακή επικοινωνία. Επιλογές: συνέλαβε, αγνόησε ή σκότωσε διεργασία.

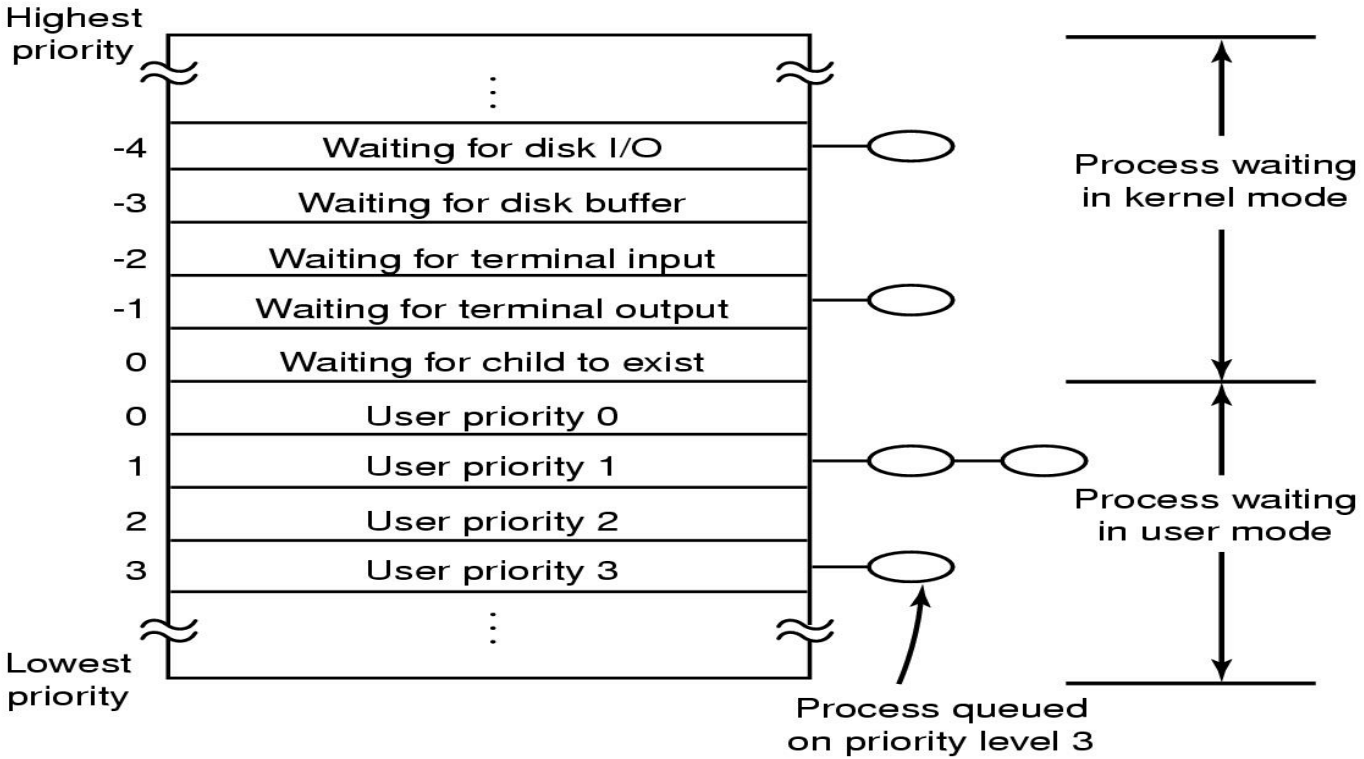
Σήματα που απαιτούνται από το POSIX

Σήμα	Αιτία
SIGABRT	Ανώμαλος τερματισμός μιας διεργασίας και αποτύπωση πυρήνων σε ένα αρχείο (core dump)
SIGALRM	Το χρονόμετρο συναγερμού έχει χτυπήσει
SIGFPE	Συνέβη κάποιο σφάλμα σε πράξη κινητής υποδιαστολής (π.χ. διαίρεση με το 0)
SIGHUP	Η τηλεφωνική γραμμή επικοινωνίας της διεργασίας διακόπηκε
SIGILL	Ο χρήστης πάτησε το πλήκτρο DEL για να διακόψει τη διεργασία
SIGQUIT	Ο χρήστης πάτησε το πλήκτρο που επιβάλλει αποτύπωση πυρήνων σε αρχείο
SIGKILL	Τερματισμός της διεργασίας (δεν μπορεί να αγνοηθεί ή να συλληφθεί)
SIGPIPE	Η διεργασία έγραψε δεδομένα σε έναν αγωγό από τον οποίο δε διαβάζει καμία άλλη διεργασία
SIGSEGV	Η διεργασία αναφέρθηκε σε άκυρη διεύθυνση μνήμης
SIGTERM	Αίτηση για διακοπή της διεργασίας με ομαλό τρόπο
SIGUSR1	Διατίθεται για χρήση από την εκάστοτε εφαρμογή
SIGUSR2	Διατίθεται για χρήση από την εκάστοτε εφαρμογή

Κλήσεις συστήματος για διαχείριση διεργασιών

Κλήση	Περιγραφή
pid = fork()	Δημιουργία θυγατρικής διεργασίας, πανομοιότυπης με τη μητρική
pid = waitpid(pid, &statloc, options)	Αναμονή μέχρι τον τερματισμό της θυγατρικής διεργασίας
s = execve(name, argv, environp)	Αντικατάσταση της εικόνας πυρήνα μιας διεργασίας
exit(status)	Τερματισμός εκτέλεσης της διεργασίας και επιστροφή κωδικού κατάστασης
s = sigaction(sig, &act, &oldact)	Καθορισμός της ενέργειας του σήματος
s = sigreturn(&context)	Επιστροφή από ένα σήμα
s = sigprocmask(how, &set, &old)	Εξέταση ή αλλαγή της μάσκας του σήματος
s = sigpending(set)	Λήψη του συνόλου των μπλοκαρισμένων σημάτων
s = sigsuspend(sigmask)	Αντικατάσταση μάσκας σήματος και μπλοκάρισμα της διεργασίας
s = kill(pid, sig)	Αποστολή του σήματος σε μια διεργασία
residual = alarm(seconds)	Ανάθεση τιμής στο χρονόμετρο επαγρύπνισης
pause()	Αναστολή καλούντος μέχρι το επόμενο σήμα

Χρονοδρομολόγηση στο UNIX



- Βασίζεται σε δομή ουρών πολλαπλών επιπέδων (με πρώτη ουρά αυτή της ψηλότερης προτεραιότητας, round-robin σε κάθε ουρά)
- Μια διεργασία που μπλοκαρίστηκε και περίμενε για ένα συμβάν μπαίνει στην κατάλληλη ουρά όταν ξεμπλοκαριστεί

Χρονοδρομολόγηση στο UNIX

- Μια φορά το δευτερόλεπτο η προτεραιότητα όλων των διεργασιών υπολογίζεται ξανά για την αποφυγή λιμοκτονίας:

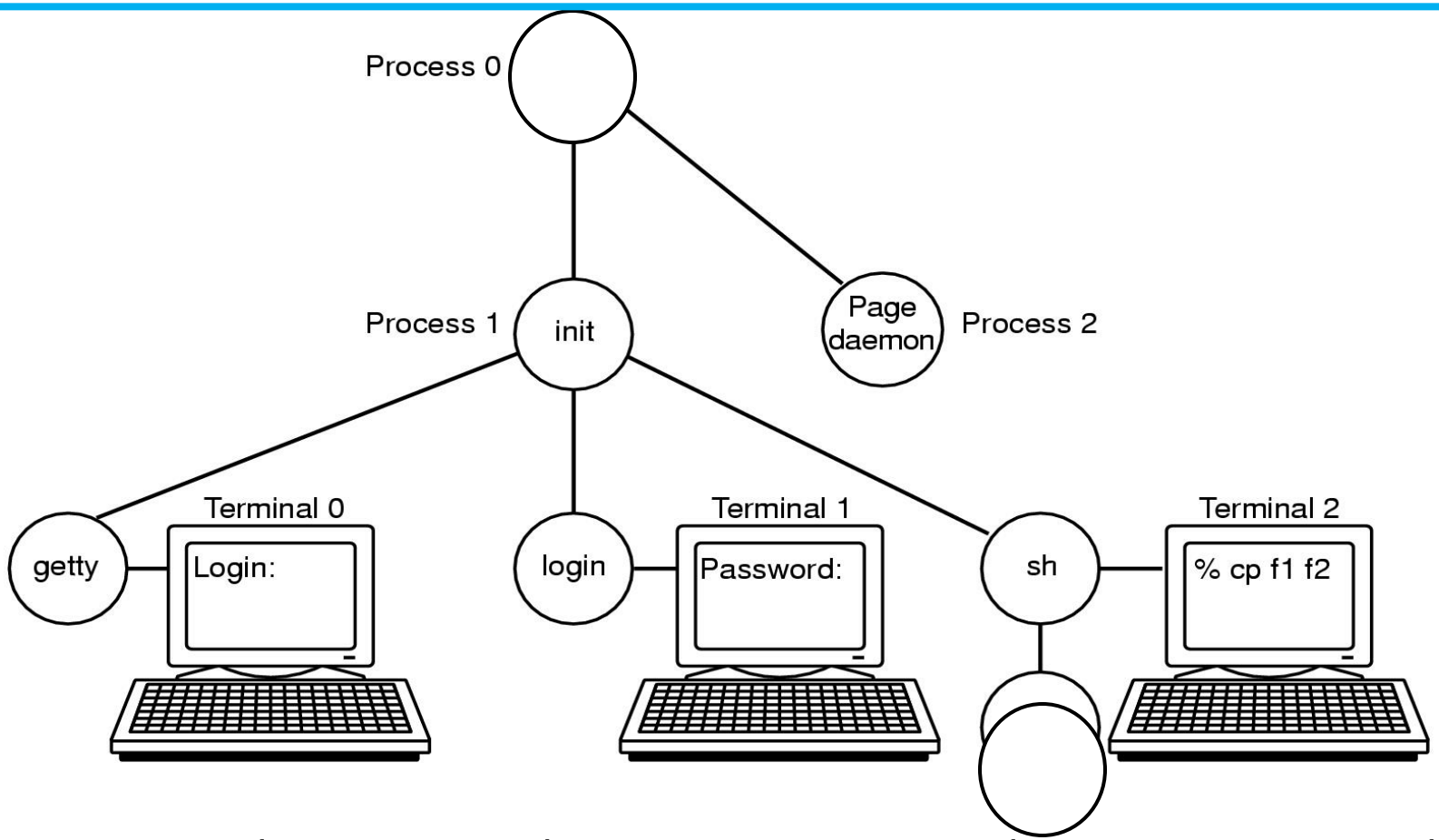
  Προτεραιότητα = Χρήση_ΚΜΕ + nice + βάση

- Η χρήση ΚΜΕ αντιπροσωπεύει το μέσο αριθμό χτύπων ρολογιού το δευτερόλεπτο που είχε η διεργασία στη διάρκεια των τελευταίων δευτερολέπτων
- Το nice είναι μια τιμή μεταξύ -20 και 20 (προκαθορισμένη 0). Η κλήση συστήματος nice μπορεί να χρησιμοποιηθεί για να θέσει την τιμή σε 0-20
- Η βάση είναι μια παράμετρος συστήματος στον πηγαίο κώδικα του UNIX
- Ο δρομολογητής επιβάλλει να εξυπηρετηθούν οι περιορισμένες από ΚΜΕ διεργασίες (θετικές ουρές) όταν οι περιορισμένες από Ε/Ε και διαδραστικές διεργασίες είναι μπλοκαρισμένες

Εκκίνηση του UNIX

- Ο πρώτος τομέας του δίσκου εκκίνησης (master boot record) διαβάζεται και εκτελείται
- Αυτός ο τομέας φορτώνει το πρόγραμμα εκκίνησης
- Το πρόγραμμα εκκίνησης διαβάζει τον ριζικό κατάλογο, φορτώνει τον πυρήνα και αρχίζει την εκτέλεσή του
- Ο πυρήνας διαβάζει το υπόλοιπο του Λ.Σ. (κύριο τμήμα κώδικα C)
- Ο κώδικας C κάνει κάποιες αρχικοποιήσεις, ορίζει δομές δεδομένων του συστήματος, φορτώνει οδηγούς συσκευών και δημιουργεί την πρώτη διεργασία

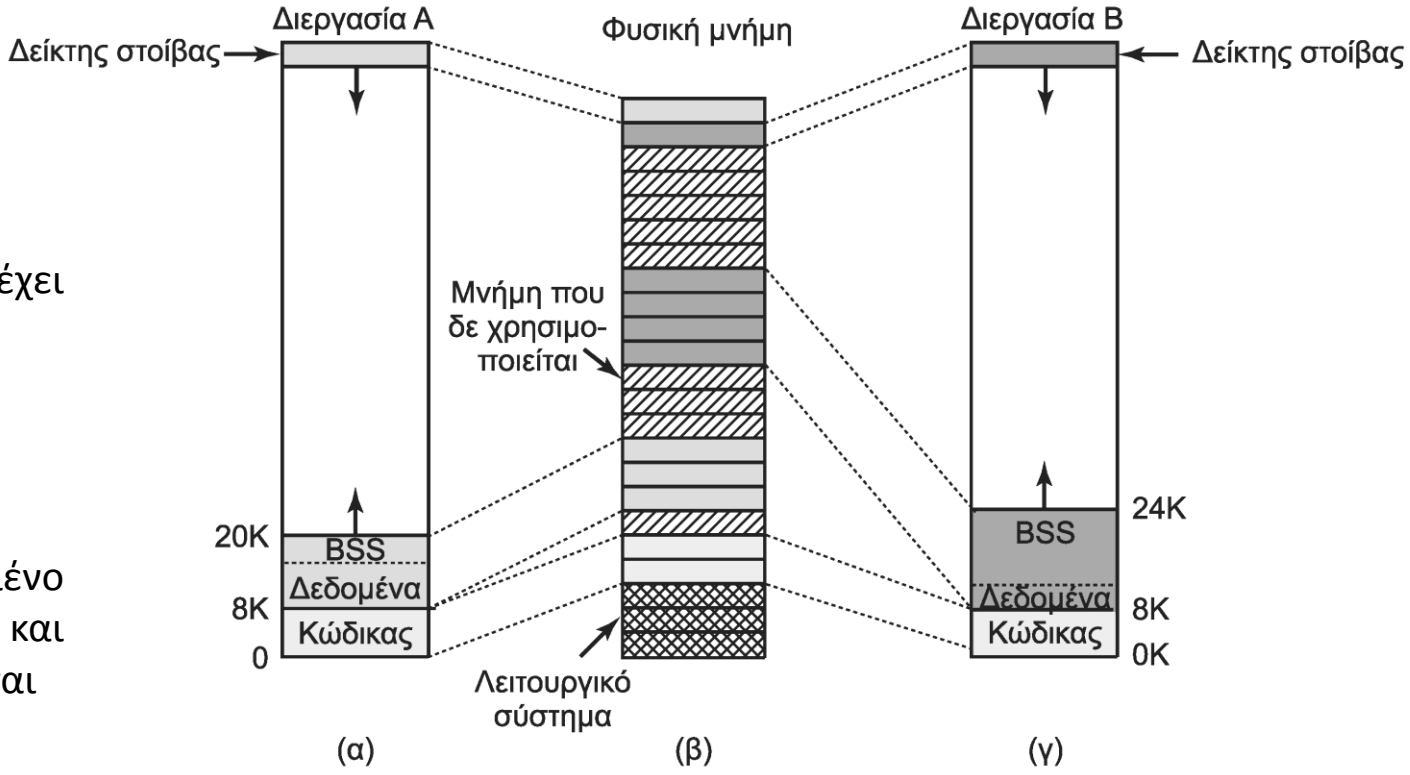
Εκκίνηση του UNIX



Οι ακολουθίες διεργασιών που χρησιμοποιούνται για να εκκινήσουν κάποια συστήματα

Διαχείριση μνήμης στο UNIX

- Κάθε διεργασία έχει τρία τμήματα:
  - Κώδικα
  - Δεδομένων
  - Στοίβας
- Κώδικας έχει ορισμένο μήκος, δεδομένα και στοίβα μεταβάλλονται

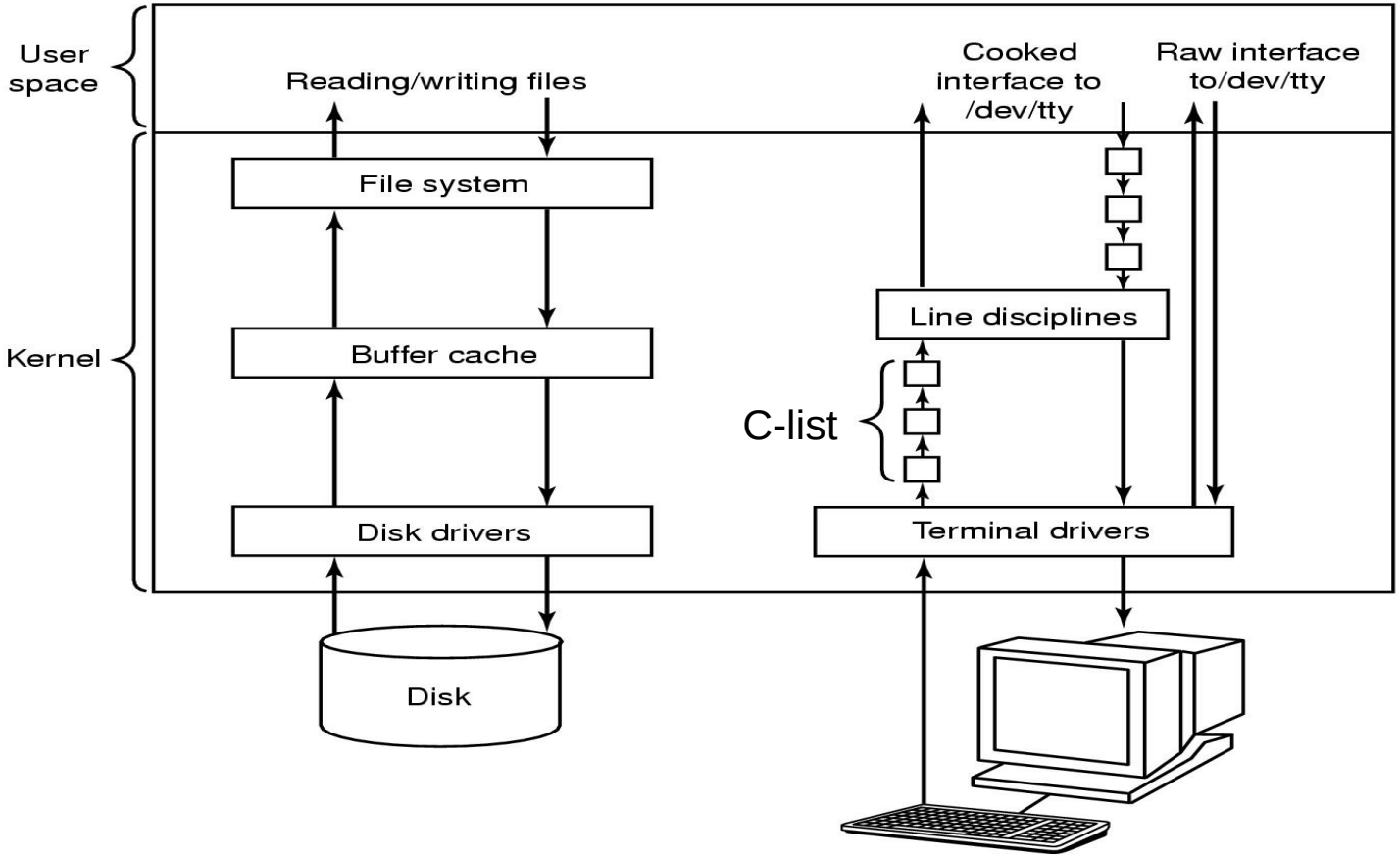


Ε/Ε στο UNIX

- Οι συσκευές αντιστοιχούνται σε ***ειδικά αρχεία***
- Προσπελαύνονται ως απλά αρχεία (π.χ. read, write, open κλπ.)
- Όταν προσπελαύνεται ειδικό αρχείο, το σύστημα αρχείων καθορίζει το μείζονα και ελάσσονα αριθμό συσκευής και κατά πόσο είναι ειδικό αρχείο μπλοκ ή χαρακτήρων
- Ο μείζων αριθμός συσκευής χρησιμοποιείται για να δείξει είτε στον πίνακα ***bdevsw*** είτε στον ***cdevsw***.
- Αυτοί οι πίνακες περιέχουν δείκτες σε διαδικασίες για να ανοίξει η συσκευή, να διαβαστεί, να γραφτεί κλπ.

Device	Open	Close	Read	Write	Ioctl	Other
Null	null	null	null	null	null	...
Memory	null	null	mem_read	mem_write	null	...
Keyboard	k_open	k_close	k_read	error	k_ioctl	...
Tty	tty_open	tty_close	tty_read	tty_write	tty_ioctl	...
Printer	lp_open	lp_close	error	lp_write	lp_ioctl	...

Ε/Ε στο UNIX

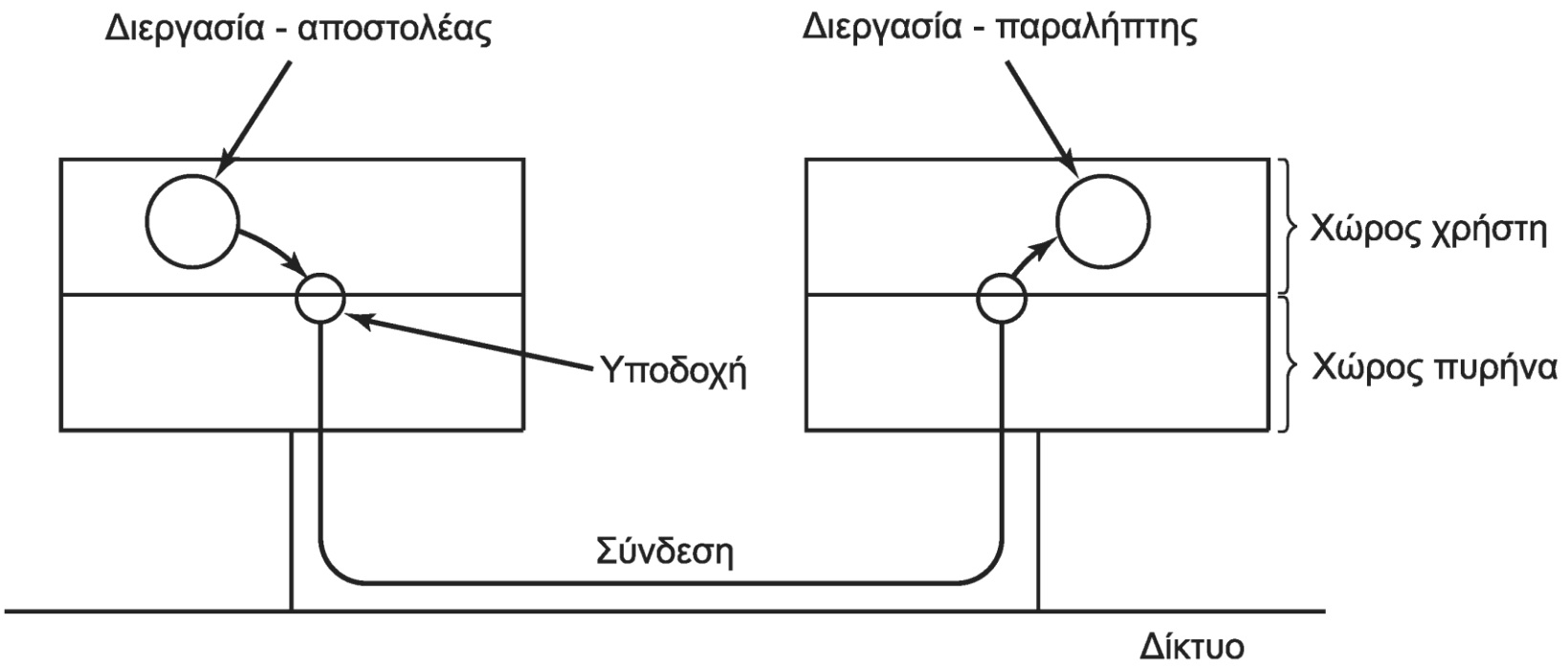


Το σύστημα Ε/Ε στο BSD

Ε/Ε στο UNIX

- Για ειδικά αρχεία μπλοκ, τα μπλοκ αποθηκεύονται σε κρυφή μνήμη απομονωτή, η οποία χρησιμοποιείται για εγγραφή και ανάγνωση.
- Κάθε μισό λεπτό, τα τροποποιημένα μπλοκ γράφονται στο δίσκο.
- Για ειδικές συσκευές χαρακτήρων, τα δεδομένα αποθηκεύονται προσωρινά σε μια αλυσίδα από ***C-lists***. Ένα μπλοκ C-list έχει μήκος 64 χαρακτήρες, συν ένα μετρητή και ένα δείκτη προς το επόμενο μπλοκ.

Δικτύωση στο UNIX

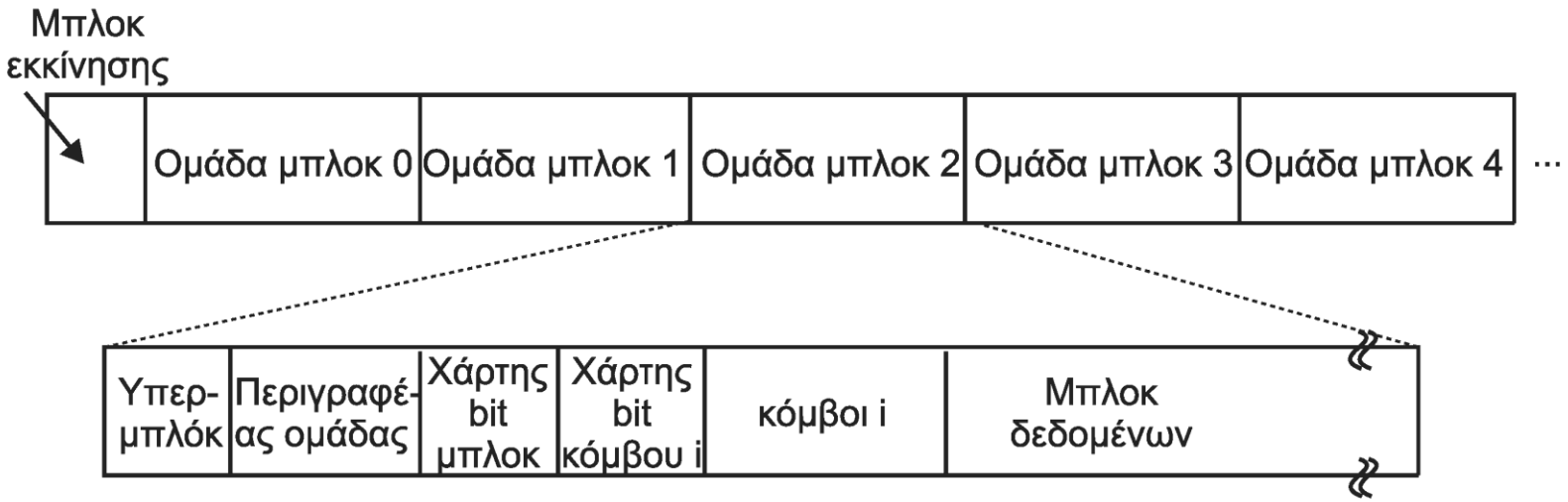


- Οι ***υποδοχές*** (sockets) χρησιμοποιούνται για την αποκατάσταση σύνδεσης μεταξύ δύο κόμβων του δικτύου.
- Οι υποδοχές δημιουργούνται και καταστρέφονται δυναμικά.

Δικτύωση στο UNIX

- Με τη δημιουργία υποδοχής επιστρέφεται περιγραφέας αρχείου (χρειάζεται για εγκαθίδρυση επικοινωνίας, ανάγνωση, εγγραφή και απελευθέρωση της σύνδεσης).
- Το ένα μέρος κάνει μια κλήση ***listen*** σε μια τοπική υποδοχή, η οποία δημιουργεί έναν απομονωτή και μπλοκάρεται μέχρι να φτάσουν δεδομένα.
- Το άλλο μέρος κάνει μια κλήση ***connect*** δίνοντας ως παραμέτρους το περιγραφέα αρχείου της τοπικής υποδοχής και τη διεύθυνση μιας απομακρυσμένης υποδοχής (κάθε υποδοχή έχει μια διεύθυνση στο δίκτυο).
- Αφού εγκαθιδρυθεί η σύνδεση, η υποδοχή λειτουργεί όπως μια σωλήνωση.
- Τύποι συνδέσεων: αξιόπιστο συνδεσμοστρεφές ρεύμα bytes, αξιόπιστο συνδεσμοστρεφές ρεύμα πακέτων, αναξιόπιστη μετάδοση πακέτων.

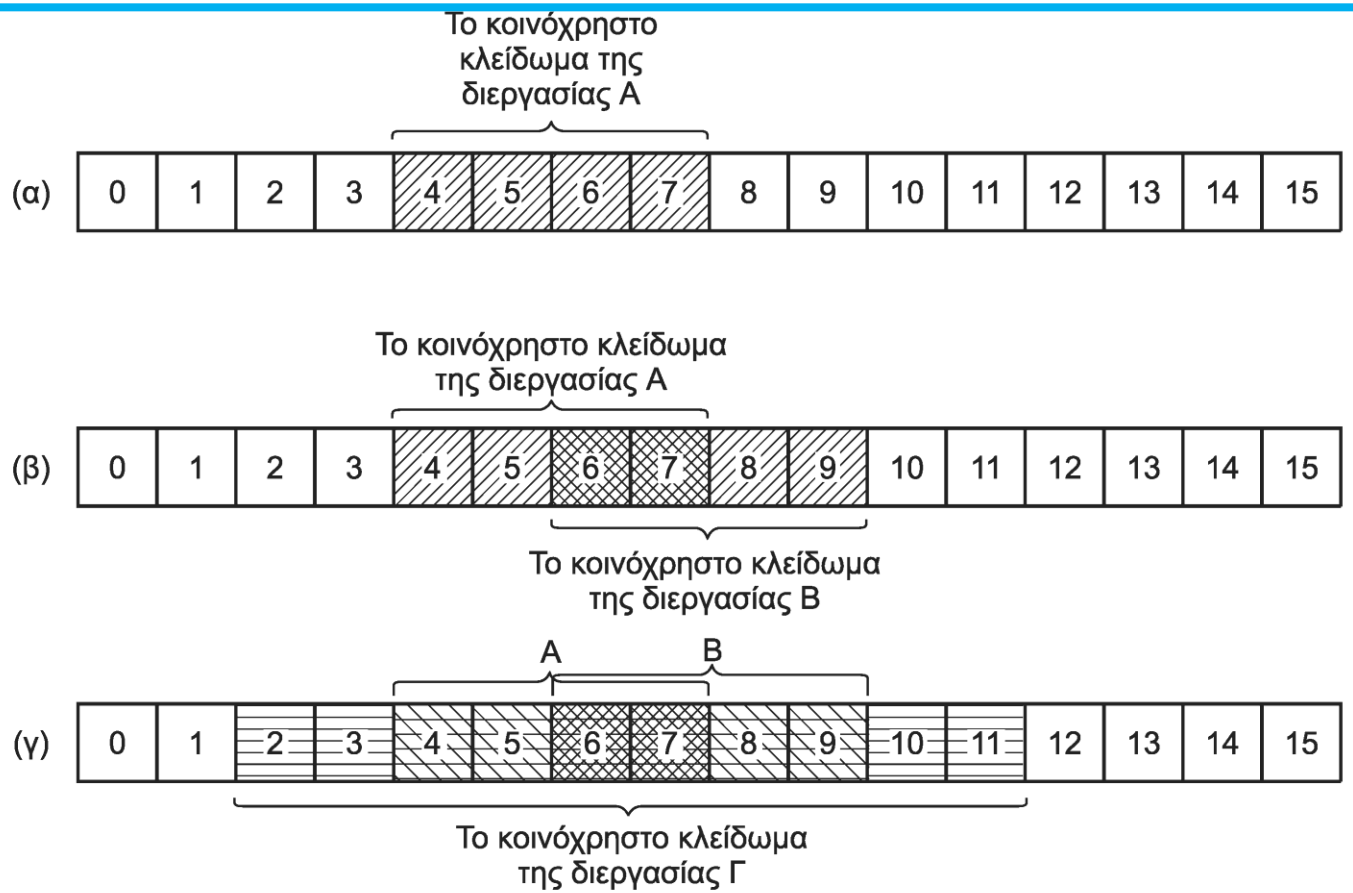
Διάταξη δίσκου στο ext2



Κλειδώματα αρχείων στο UNIX

- Η προσπέλαση ενός αρχείου από διαφορετικές διεργασίες χρειάζεται κάποια διαχείριση κρίσιμου τμήματος.
- Αυτό γίνεται με ***κλειδώματα***.
- Ένα κλείδωμα ορίζεται με ένα όνομα αρχείου, το αρχικό byte και το πλήθος των bytes.
- Όταν επιχειρεί ένα κλείδωμα, μια διεργασία μπορεί:
  - Να μπλοκαριστεί όσο δε μπορεί να τοποθετήσει το κλείδωμα, ή
  - Να μη μπλοκαριστεί, αν η κλήση συστήματος απλά επιστρέφει έναν κωδικό κατάστασης που λέει αν το κλείδωμα πέτυχε ή όχι.

Κλειδώματα αρχείων στο UNIX



(α) Ένα αρχείο με ένα κλείδωμα. (β) Προσθήκη ενός δεύτερου κλειδώματος. (γ) Ένα τρίτο κλείδωμα.

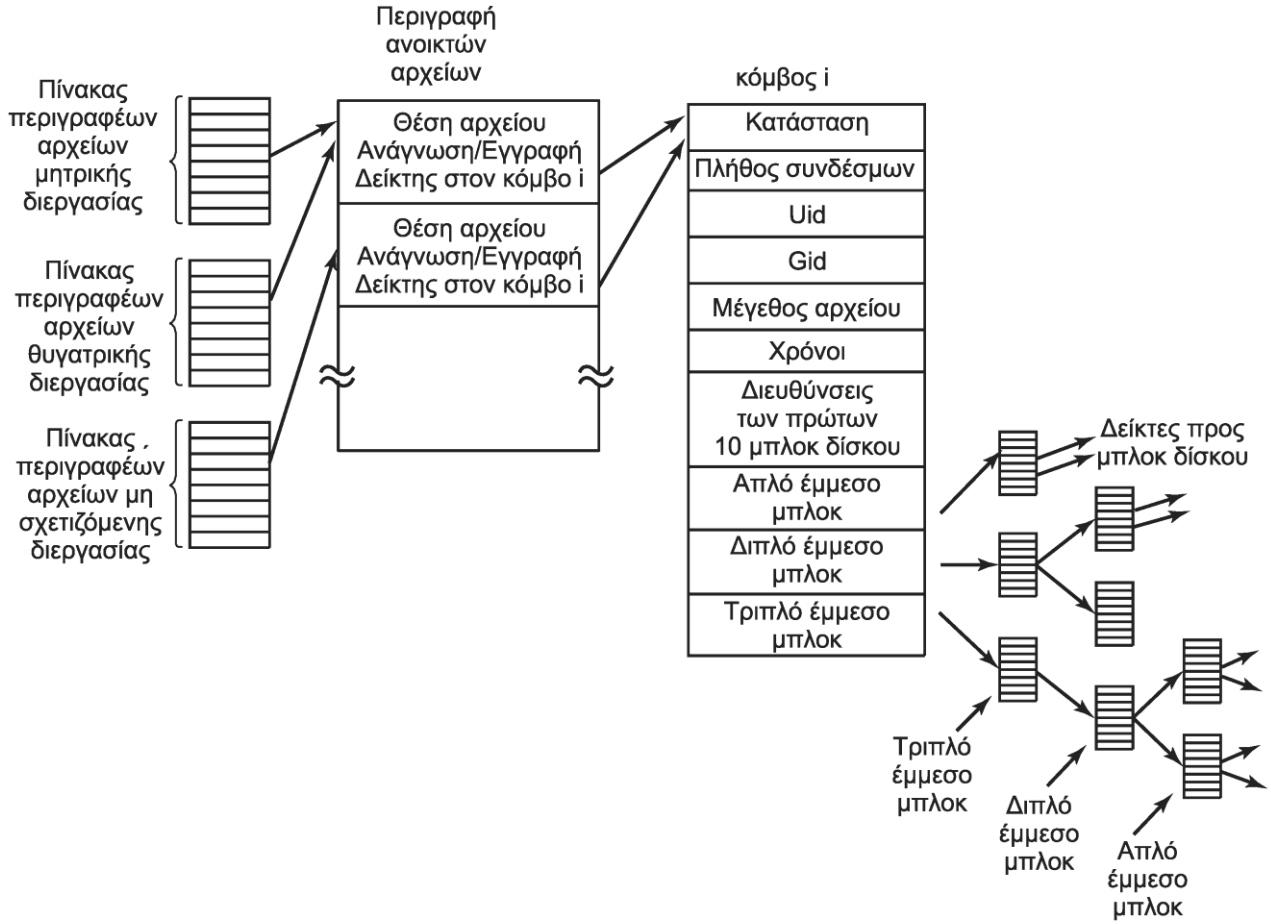
Κλήσεις συστήματος για αρχεία

Κλήση συστήματος	Περιγραφή
fd = creat(name, mode)	Ένας τρόπος δημιουργίας ενός νέου αρχείου
fd = open(file, how, ...)	Ανοίγει ένα αρχείο για ανάγνωση, εγγραφή, ή και τα δύο
s = close(fd)	Κλείνει ένα ανοιχτό αρχείο
n = read(fd, buffer, nbytes)	Διαβάζει δεδομένα από ένα αρχείο και τα τοποθετεί σε προσωρινή μνήμη
n = write(fd, buffer, nbytes)	Διαβάζει δεδομένα από μια προσωρινή μνήμη και τα αποθηκεύει σε ένα αρχείο
position = lseek(fd, offset, whence)	Μετακινεί το δείκτη αρχείου
s = stat(name, &buf)	Διαβάζει τις πληροφορίες κατάστασης ενός αρχείου
s = fstat(fd, &buf)	Διαβάζει τις πληροφορίες κατάστασης ενός αρχείου
s = pipe(&fd[0])	Δημιουργεί έναν αγωγό (pipe)
s = fcntl(fd, cmd, ...)	Κλείδωμα αρχείου και άλλες λειτουργίες

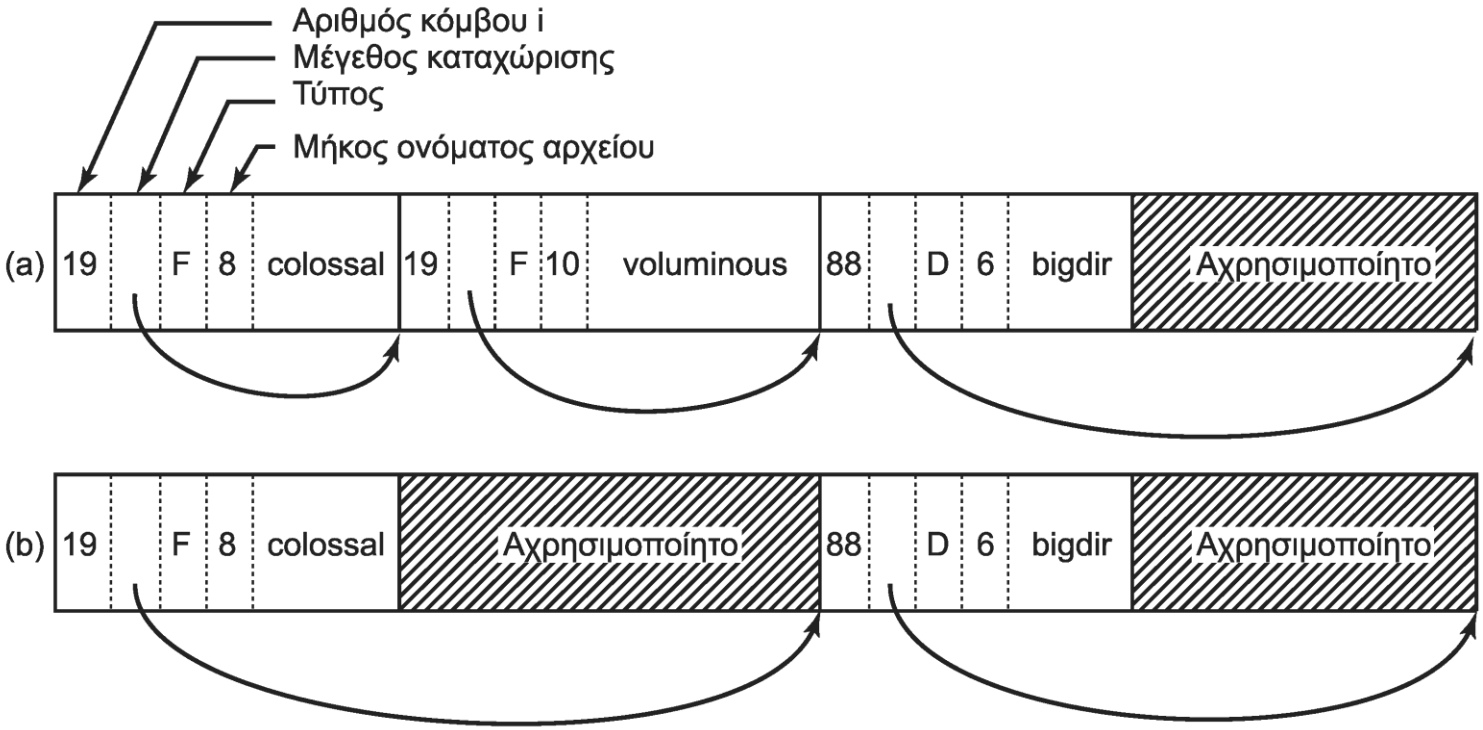
Κλήσεις συστήματος για καταλόγους

Κλήση συστήματος	Περιγραφή
s = mkdir(path, mode)	Δημιουργεί ένα νέο κατάλογο
s = rmdir(path)	Διαγράφει έναν κατάλογο
s = link(oldpath, newpath)	Δημιουργεί ένα σύνδεσμο προς ένα υπάρχον αρχείο
s = unlink(path)	Διαγράφει ένα αρχείο από έναν κατάλογο
s = chdir(path)	Αλλάζει τον κατάλογο εργασίας
dir = opendir(path)	Ανοίγει έναν κατάλογο για ανάγνωση
s = closedir(dir)	Κλείνει έναν κατάλογο
dirent = readdir(dir)	Διαβάζει μια καταχώριση καταλόγου
rewinddir(dir)	Επιστρέφει στην αρχή ενός καταλόγου για να τον ξαναδιαβάσει

Πίνακες περιγραφής αρχείων



Κατάλογοι στο UNIX



(α) Ένας κατάλογος με τρία αρχεία. (β) Ο ίδιος κατάλογος αφού διαγραφεί το αρχείο voluminous.

Ασφάλεια στο UNIX

- Κάθε χρήστης στο UNIX έχει ένα UID (ένας ακέραιος μεταξύ 0 και 65535). Αρχεία, διεργασίες και άλλοι πόροι σημειώνονται με το UID του ιδιοκτήτη τους.
- Ο χρήστης με UID 0 είναι ο ***superuser***.
- Οι χρήστες μπορούν να οργανωθούν σε ***ομάδες***, που επίσης αριθμούνται με GID των 16 bits.
- Παραδείγματα τρόπων προστασίας αρχείων:

Δυαδικό	Συμβολισμός	Επιτρεπόμενη πρόσβαση στο αρχείο
111000000	rwx------	Ο ιδιοκτήτης μπορεί να διαβάσει, να γράψει, και να εκτελέσει
111111000	rwxrwx---	Ο ιδιοκτήτης και η ομάδα του μπορούν να διαβάσουν, να γράψουν, και να εκτελέσουν
110100000	rw-r-----	Ο ιδιοκτήτης μπορεί να διαβάσει και να γράψει, η ομάδα μπορεί να διαβάσει
110100100	rw-r--r--	Ο ιδιοκτήτης μπορεί να διαβάσει και να γράψει, όλοι οι άλλοι μπορούν να διαβάσουν
000000000	---------	Κανένας δεν έχει το παραμικρό δικαίωμα πρόσβασης
000000111	------rwx	Μόνον οι άλλοι έχουν πρόσβαση (περίεργο, αλλά νόμιμο)

Κλήσεις συστήματος για προστασία αρχείων

Κλήση συστήματος	Περιγραφή
s = chmod(path, mode)	Αλλάζει τον τρόπο προστασίας ενός αρχείου
s = access(path, mode)	Έλεγχος πρόσβασης με τα πραγματικά UID και GID
uid = getuid()	Παίρνει το πραγματικό UID
uid = geteuid()	Παίρνει το ενεργό UID
gid = getgid()	Παίρνει το πραγματικό GID
gid = getegid()	Παίρνει το ενεργό GID
s = chown(path, owner, group)	Αλλάζει ιδιοκτήτη και ομάδα
s = setuid(uid)	Αναθέτει τιμή στο UID
s = setgid(gid)	Αναθέτει τιμή στο GID

Τέλος Ενότητας