Concepte Unix/Linux

Evoluţia sistemului de operare UNIX/LINUX
Structura sistemului de operare Unix

Sistemul de fişiere Linux

Partiţii şi blocuri.

Structura unui disc Linux

Schema de alocare a blocurilor disc pentru un fişier

Sistemele de fişiere

Utilizarea fişierului fstab pentru definirea sistemelor de fişiere utilizate

Montarea şi demontarea sistemelor de fişiere.

Evoluţia sistemului de operare UNIX/LINUX

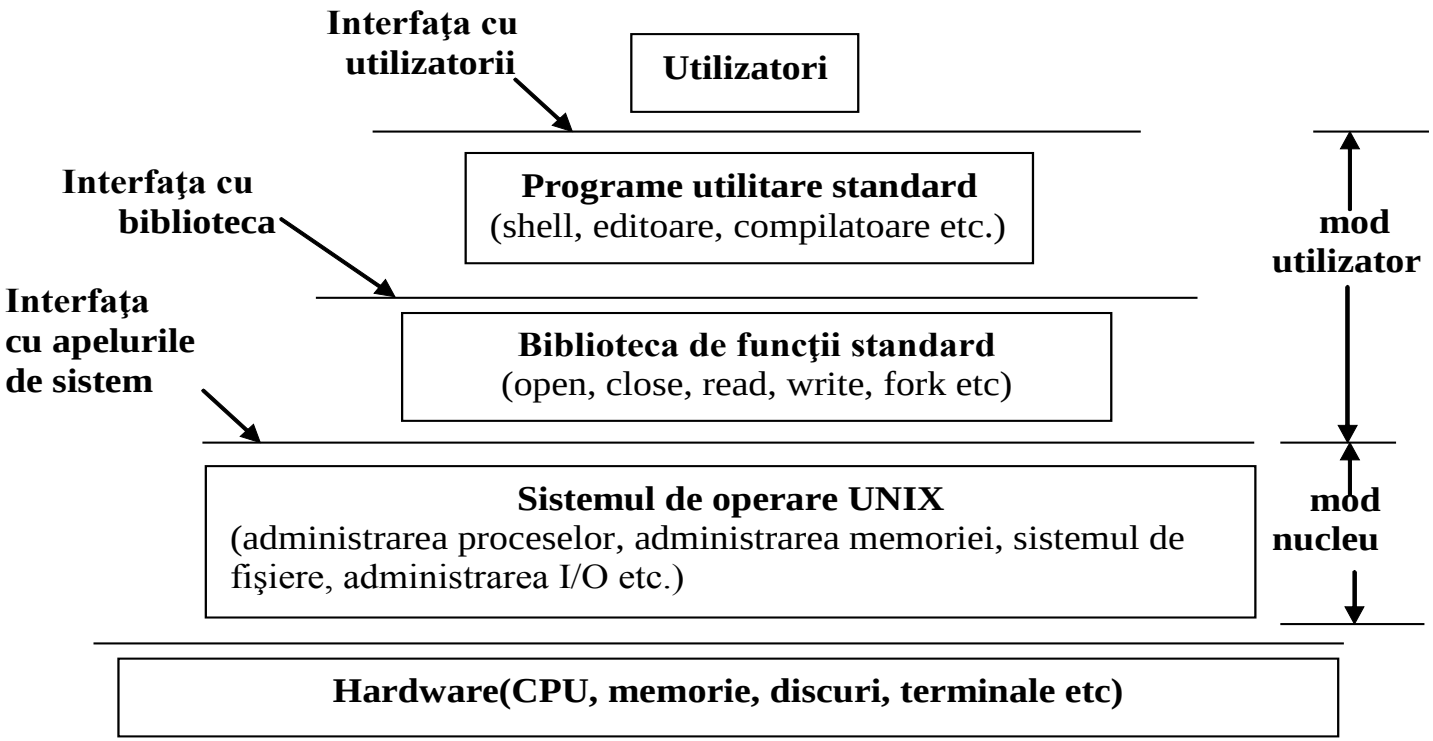
- În 1969 apare prima versiune a sistemului de operare UNIX. Anterior, a fost realizat de către MIT(Massachusets Institute of Technology) produsul MULTICS, care a lansat o serie de indei noi, printre care:
  - introducerea unui interpretor de comenzi care să fie un proces utilizator;
  - fiecare comandă să fie un proces;
  - introducere unor caractere de interpretare a liniilor;
  - sistem multi-utilizator şi interactiv.
- Ulterior, în cadrul firmei Bell Labs afost lansat un proiect de cercetare care a introdus un **sistem ierarhic** de organizare a fişierelor. Toate aceste concepte au stat la baza primei versiuni a sistemului UNIX, realizată în limbaj de asamblare şi implementată pe un sistem PDP-7.
- În 1971 apare următoarea versiune a sistemului de operare UNIX. Acesta a fost implementat pe un calculator PDP-11/20, mai puternic. Este rescris pentru noul sistem şi apare o documenraţie a sistemului. În 1974 sunt prezentate în *Communications of the ACM* ideile revoluţionare ale acestui nou sistem de operare. De asemenea, au fost dezvoltate încă trei versiuni ale sistemului şi autorii sistemului îl pun gratuit la dispoziţia universităţilor americane.
- În 1976 apare versiunea 6, ale cărei componente sunt majoritar scrise în limbajul C. În felul acesta, sistemul devine **portabil**, adică poate fi instalat cu eforturi minime pe platforme de calcul diferite.

- După această dată, sunt dezvoltate în paralel, de către Universitatea Berkeley şi Bel Labs(laboratoarele de cercetare ale firmei AT&T) diverse versiuni, în concordanţă cu evoluţia sistemelor de calcul. Versiunea 7, apărută în 1978 a devenit comercială, iar din punct de vedere conceptual a adăugat organizarea sistemului de I/O sub formă de fluxuri, care permite o configurare mai flexibilă a administrării comunicaţiei între procese, precum şi un sistem de fişiere care să permită apelul de proceduri la distanţă. Dintre următoarele implementări, cele mai semnificative sunt BSD 4.1, BSD 4.2, BSD 4.3 realizate la Universitatea Berkeley şi Sistem III, Sistem V ale Bell Labs.
- Următoarele relizări au încercat să standardizeze sistemul, ceea ce adus la formă general acceptată denumită UNIX System V Release 4 (prescurtat SVR4). Şi alte firme realizează proprii versiun de UNIX, care respectă standardele SVR4, dintre care cele mai semnificative sunt Ultrix şi OSF(firma DEC- Digital Equiment Corporation), HP/UX(Hewlett-Packard), IRIX(SGI-Silicon Graphics Incorporated), AIX(IBM), SunOS şi Solaris(SUN Microsystems) etc. Dintre facilităţile adăugate de aceste versiuni se remarcă introducere interfeţelor grafice cu utilizatorul şi cele legate de lucrul în reţele de calculatoare.

- Standardizarea diferitelor variante de UNIX s-a realizat sub auspiciile unui comitet de standarde al IEEE-POSIX(Portable Operating System UNIX) care a devenit o colecţie de standarde pe care le respectă toate sistemele de operare moderne. El este format din mai multe componente, dintre care cele mai importante sunt:

  POSIX.1 – defineşte procedurile care cer apeluri de sistem;
- POSIX.2 – se referă la interfaţa cu sistemul(shell);
- POSIX.3 – metode de testare a sistemului;
- POSIX.4 – componentele specifice ale sistemelor de operare ale calc. în timp real;
- POSIX.6 – elemente de securitate;
- POSIX.7 – elemente de administrare
- În 1987 a apărut sist. MINIX, lansat în scopuri didactice la Univ. Vrije din Amsterdam; este compatibil cu UNIX la nivel de comenzi şi apeluri de sistem, dar este original din punctual de vedere al proiect. şi implementării.
- În 1991, pornind de la MINIX, apare sistemul LINUX, creat de un student finlandez, Linus Torvalds. El afost creat pentru sistemele de calcul pe 32 de biţi, compatibile IBM PC. Sursele lui au fost puse la dispoziţia tuturor celor interesaţi, fiind libere pe Internet. Evoluţia sistemului LINUX a fost încurajată de majoritatea firmelor de software importante(cu excepţia Microsoft), o parte dintre ele oferind-ul la preţuri foarte mici. Urmând filozofia UNIX-ului, LINUX este mai degrabă un nucleu, care implementează funcţiile de bază, fiind deschis tuturor celor care doresc să adauge noi module care să extindă sistemul.

Structura sistemului de operare Unix



Organizarea sistemului sub UNIX

Sistemul de fişiere Linux

- **Tipuri de fişiere şi sisteme de fişiere.** Nucleul sistemului **Linux** priveşte un fişier ca o secvenţă de octeţi, lăsând în seama aplicaţiilor structurarea informaţiilor conţinute în acestea. In cadrul unui sistem de fişiere, apelurile sistem Linux gestionează urmatoarele tipuri principale de de fişiere: **Normale (obişnuite), Directori**, **Legături bard (hard links**), **Legături simbolice (symbolic links)**, **Periferice caracter, Periferice bloc**.
- *Fişierele obişnuite* sunt privite ca şiruri de octeţi, accesul la un octet putându-se face fie secvenţial, fie direct prin numărul de ordine al octetului.
- *Fişierele directori*. Un fişier director se deosebeşte de un fişier obişnuit numai prin informaţia conţinută în el. Un director conţine lista de nume şi adrese pentru fişierele subordonate lui. Uzual, fiecare utilizator are un director propriu care punctează la fişierele lui obişnuite, sau la alţi subdirectori definiţi de el.
- *Fişierele legături simbolice* sunt un fel de alias-uri pentru fişiere.
- *Fişierele fifo* sunt folosite pentru comunicaţia între procese care se execută pe acelaşi calculator.
- *Fişierele socket* sunt folosite pentru comunicaţia între procese care se execută într-o reţea de calculatoare.
- *Fişierele speciale*. Linux priveşte fiecare dispozitiv de I/O ca şi un fişier de tip special. Din punct de vedere al utilizatorului, nu există nici o deosebire între lucrul cu un fişier disc normal şi lucrul cu un fişier special. Se realizează astfel:
- • Simplitatea şi eleganţa comenzilor.
- • Numele unui dispozitiv poate fi transmis ca argument.

- **Partiţii şi blocuri.** Un sistem de fişiere Linux este găzduit fie pe un periferic oarecare (hard-disc, CD etc.), fie pe o *partiţie* a unui hard-disc.
- Partiţionarea unui bard-disc este o operaţie (relativ) independentă de sistemul de operare ce va fi găzduit în partiţia respectivă. De aceea, atât partiţiilor, cât şi suporturilor fizice reale le vom spune generic *discuri Linux*.
- O intrare într-un fişier director conţine *numele fişierului* şi *numărul nodului de index* al acestuia. Informaţiile conţinute în nodul de index sunt structurate pe câmpuri şi conţin;
  - informaţia de identificare a proprietarului fişierului;
  - identificarea grupului de utilizator care ar putea identifica acest fişier;
  - biţii de protecţie;
  - lungimea fişierului;
  - data ultimei actualizări;
  - numărul de legături la fişier;
  - un indicator care ne spune dacă este un fisier obişnuit sau director;
  - contine pointeri către structurile ataşate fisierului.

- **Structura unui disc Linux**
- `Blocul 0` - bloc de boot
- `Blocul 1` - Superbloc
- `Blocul 2` - inod
- `------------`
- `Blocul n` - inod
- `Blocul n+1` -zona fişier
- `-------------`
- `Blocul n+m` -zona fişier
- `Blocul 0` conţine programul de încărcare al **SO**. Acest program este dependent de maşina sub care se lucrează. Acţiunea lui se declanşează la pornirea sistemului. În acest moment, intră în lucru un mic program din memoria ROM, aşa-numitul *cod startup din BIOS*. Acesta ştie să citească primii 512 octeţi de pe disc, să îi depună undeva în memoria RAM şi să lanseze în execuţie secvenţa de octeţi citită. Evident, în primii 512 octeţi de pe disc va fi un program, *bootprogramul* sau *programul de pornire a încărcării sistemului de operare*. Principala sarcină a *bootprogramului* este aceea de a încărca în RAM partea din *kernel*-ul Linux (sau a altui sistem de operare) special destinată încărcării complete a sistemului de operare.

- Blocul 1 este numit şi *superbloc*. În el sunt trecute o serie de informaţii prin care se defineşte sistemul de fişiere de pe disc:
  - numărul n de *inoduri* (detaliem imediat);
  - numărul de zone definite pe disc;
  - pointeri spre harta de biţi a alocării inodurilor;
  - pointeri spre harta de biţi a spaţiului liber pe disc;
  - dimensiunile zonelor disc, etc.)
- Blocurile 2 la n(n este o constantă a formatării discului), este zona de *inoduri*. Un *inod* (sau *i-nod*) este numele *descriptorului* unui fişier. Inodurile sunt memorate pe disc sub forma unei liste (numită *i-listă*). Numărul de ordine al unui inod în cadrul i-listei se numeşte *i-număr*. Acest *i-număr* constituie legătura dintre fişier şi programele utilizator.
- Blocurile n+1 la n+m reprezintă *zona fişierelor*. Este partea cea mai mare din cadrul discului. Alocarea spaţiului pentru fişiere se face printr-o schema de indexare(prezentata mai tirziu). Informaţiile de plecare pentru alocare sunt fixate în inoduri.
- **Obs.** La discurile Linux actuale există, de regulă, mai multe zone de inoduri intercalate cu mai multe zone de fişiere.

- **Directori şi inoduri.**
- O intrare într-un fişier director conţine:
  - Numele fişierului
  - inumar
- Deci, în director se află numele fişierului şi referinţa spre inodul descriptor al fişierului.
- Un inod are, de regulă, între 64 şi 128 de octeţi şi el conţine informaţiile din tabelul următor:

- Drepturile de acces şi tipul fişierului.
- Numărul de legături spre acest fişier (contor de legare).
- Numărul (UID) de identificare al proprietarului(user ID).
- Numărul (GID) de identificare a grupului(group ID).
- Numărul de octeţi (lungimea) fişierului.
- Momentul ultimului acces la fişier.
- Momentul ultimei modificări a fişierului.
- Momentul ultimei modificări a structurii inodului.
- Lista adreselor disc pentru primele blocuri care aparţin fişierului.
- Referinţe către celelalte blocuri care aparţin fişierului.

Schema de alocare a blocurilor disc pentru un fişier

- Fiecare sistem de fişiere Linux are câteva constante proprii, printre care amintim: lungimea unui inod; lungimea unui bloc;lungimea unei adrese disc (implicit câte adrese disc încap într-un bloc);câte adrese de prime blocuri se înregistrează direct în inod;câte referinţe se trec în lista de referinţe indirecte.
- Indiferent de valorile acestor constante, principiile de înregistrare / regăsire sunt aceleaşi şi le vom prezenta în cele ce urmează. Pentru fixarea ideilor, vom alege aceste constante cu valorile întâlnite mai frecvent la sistemele de fişiere deja consacrate.
- Cu aceste constante, în fig. urm. este prezentată structura pointerilor spre blocurile ataşate unui fişier Linux. Aceste constante sunt:

  -un inod se reprezintă pe 64 octeţi,

  -un bloc are lungimea de 512 octeţi,

  -adresa disc se reprezintă pe 4 octeţi, deci încap 128 adrese disc într-un bloc,

  -în inod trec direct primele 10 adrese de blocuri,

  -lista de adrese indirecte are 3 elemente.

- In inodul fişierului se află o listă cu 13 intrări, care desemnează blocurile fizice aparţinând fişierului.
- Primele 10 intrări conţin *adresele primelor* 10 blocuri de câte 512 octeţi care aparţin fişierului.
- Intrarea nr. 11 conţine adresa unui bloc, numit *bloc de indirectare simplă*. El conţine adresele următoarelor 128 blocuri de câte 512 octeţi, care aparţin fişierului.
- Intrarea nr. 12 conţine adresa unui bloc, numit *bloc de indirectare dublă*. El conţine adresele a 128 blocuri de indirectare simplă, care la rândul lor conţin, fiecare, adresele a câte 128 blocuri, de 512 octeţi fiecare, cu informaţii aparţinând fişierului.
- Intrarea nr. 13 conţine adresa unui bloc, numit *bloc de indirectare triplă*. În acest bloc sunt conţinute adresele a 128 blocuri de indirectare dublă, fiecare dintre acestea conţinând adresele a câte 128 blocuri de indirectare simplă, iar fiecare dintre acestea conţine adresele a câte 128 blocuri, de câte 512 octeţi, cu informaţii ale fişierului.

- În fig. am ilustrat prin cercuri blocurile de informaţie care aparţin fişierului, iar prin dreptunghiuri blocurile de referinţe, în interiorul acestora marcând referinţele.
- Numărul de accese necesare pentru a obţine direct un octet oarecare este cel mult 4. Pentru fişiere mici acest număr este şi mai mic. Atât timp cât fişierul este deschis, inodul lui este prezent in memoria internă. Tabelul următor dă numărul maxim de accese la disc pentru a obţine, în acces direct orice octet dintr-un fişier, în funcţie de lungimea fişierului.

Lungime maxima (blocuri)	Lungime maxima (octeţi)	Accese indirecte	Accese la informaţie	Total accese
10	5120	-	1	1
10+128	70656	1	1	2
10+128+ +128^2=16522	8459264	2	1	3
10+128+128^2+ +128^3=2113674	1082201088	3	1	4

- La sistemele Linux actuale lungimea unui bloc este de 4096 octeţi care poate înregistra 1024 adrese, iar în inod se înregistrează direct adresele primelor 12 blocuri. In aceste condiţii, tabelul de mai sus se transformă în:

Lungime maxima (blocuri)	Lungime maxima (octeţi)	Accese indir.	Accese la inform.	Total acc.
12	49152	-	1	1
12+1024=1036	4243456	1	1	2
12+1024+1024^2= 1049612	4299210752	2	1	3
12+1024+1024^2+ +1024^3=1073741824	4398046511104 (peste 5000Go)	3	1	4

- **Exemplu.** Presupunem că avem un fişier de lungime 10 Mo. Ne propunem să determinăm ce tip de indirectare se foloseşte pentru alocarea de spaţiu pentru acest fişier, dacă se foloseşte prima schemă de alocare. Observăm că:

  10 Mo=10x1024x1024 octeţi=10x220 octeti

  1 bloc=512 octeti=29 octeţi

- După alocare directă mai rămân:

  10x220-10x29 octeţi = 10x29x2043 octeţi

După indirectare simplă mai ramân:

10x29x2043-128x512 octeţi = 29x20302 octeţi

- Prin redirectare dublă se mai poate aloca spaţiu pentru:

  128x128x512 octeţi=29x16384 octeţi.

- Deoarece 20302>16384, rezultă că este necesară indirectare triplă.

Administrarea sistemului de fişiere

- **Introducere.**
- În sistemele Linux, accesul la dispozitive precum discuri flexibile, cd-uri, partiţii ale unui hard-disc (în general memorii externe) se realizează în mod diferit de sistemele de tip Windows.
- Nu există volume separate de genul A:,C: etc ; orice astfel de dispozitv este integrat în sistemul de fişiere local printr-o operaţie numită **montare**.
- Montarea asociază unui director întregul sistem de fişiere aflat pe dispozitivul montat. Mecanismul este deosebit de puternic, deoarece oferă posibilitatea de a avea o structură de directoare unitară, care grupează fişiere de pe mai multe partiţii sau discuri. Dacă se adaugă şi sistemul de fişiere NFS (Network File System), această structură de directoare va putea conţine şi sisteme de fişiere de pe altă maşină.
- Mai explicit, structura în cauză este una arborescentă, iar la adaugarea unei noi parţitii montată sub un director, de fapt adaugăm un subarbore în cadrul arborelui deja existent, legâdu-l de un nod ales de noi (în acest caz directorul în care îl montăm).

- **Operaţia de montare.** Are rolul de a face disponibil conţinutul unui sistem de fişiere, asimilându-l în cadrul structurii de directoare a sistemului.
- Un sistem de fişiere poate fi montat/demontat la/de la ierarhia sistemului.
- Partiţia rădăcină este întotdeauna montată la pornirea sistemului. Este imposibilă demontarea partiţiei rădăcină în timpul funcţionării sistemului.
- Ierarhia de fişiere de pe o partiţie poate fi montată în orice director al sistemului rădacină, acesta numindu-se punct de montare.
- După montare, directorul radăcină al sistemului de fişiere montat, înlocuieşte conţinutul directorului unde a fost montat.
- Pentru identificarea dispozitivelor periferice, sistemele UNIX folosesc intrări speciale în directoare, numite "device files".
- Fisierele speciale care indică unitaţi de disc sau partiţii sunt folosite la montare. În general aceste fişiere se găsesc în directorul `/dev` şi au denumiri standardizate.

- **Nume de discuri**

Nume	Descriere
`/dev/hda`	Primul hard disc IDE din sistem (Integrated Drive Electronics) (omologul lui C: din DOS şi Windows), conectat la IDE ca master drive.
`/dev/hdb`	Al doilea hard disc IDE din sistem, conectat la IDE ca slave drive.
`/dev/hdc`	Primul hard disc, conectat la al doilea controller IDE ca master drive.
`/dev/hdd`	Al doilea hard disc, conectat la al doilea controller IDE ca slave drive.
`/dev/sda`	Primul disc SCSI (Small Computer System Interface).
`/dev/sdb`	Al doilea disc SCSI.
`/dev/fd0`	Primul floppy disc (A: din DOS).
`/dev/fd1`	Al doilea floppy disc (B:din DOS).

- Hard discurile pot fi partiţionate; prima partiţie din a discului `hda` este `hda1`, a doua este `hda2` ş.a.m.d.
- În general, avem următoarele denumiri standardizate:

`fd`**X**	unitatea floppy
`hd`**X**	unităţi HDD sau CDROM pe IDE
`cdrom`**X**	unităţi cdrom (în general legatură simbolică)
`scd`**X**	discuri SCSI sau unităţi CDROM emulate SCSI sau pe USB
`sda`**X**	unităţi de stocare pe USB (HDD-uri , ZIP-uri , FDD-uri, Card Readere, Flash-uri)

- **"X"**-urile de mai sus sunt de fapt numere corespunzătoare unităţii respective. Presupunem ca avem doua hard-disk-uri IDE, care vor fi identificate de Linux ca fiind **hda** si **hdb**. De exemplu, în Windows ne-am fi referit la o partiţie ca fiind, C:, D:, E: şi aşa mai departe. În Linux, dacă vrem să ne referim la partiţia a treia de pe hard-disk-ul IDE slave, de pe controler-ul IDE primar, vom folosi **hdb3**; **hd** se referă la tipul unitaţii, **b** se referă poziţia unităţii, iar **3** este numărul partiţiei.
- Pentru aflarea partiţiilor de pe sistem şi a tipului lor se foloseşte comanda `fdisk -l`

- **Utilizarea fişierului `fstab` pentru definirea sistemelor de fişiere utilizate**
- `fstab` este un fişier de configurare ce conţine informaţii despre toate partiţiile şi dispozitivele de stocare ale calculatorului, precum şi cele legate de locul unde acestea ar trebui montate. Acest fişier este localizat în directorul `/etc`.
- Dacă nu se poate accesa partiţia Windows din Linux, nu se poate monta unitatea CD-ROM sau cea de floppy, atunci cauza probabilă este că fişierul `/etc/fstab` este configurat necorespunzator.
- `/etc/fstab` este un fişier text, deci se poate deschide şi edita cu orice editor de text din Linux, numai de utilizatorul cu drepturi de root.
- De asemenea, fişierul `fstab` este folosit pentru a monta la pornirea sistemului toate partiţiile configurate.
- Pe fiecare sistem există un fişier `/etc/fstab` cu un conţinut specific, datorită partiţiilor, dispozitivelor de stocare şi proprietatilor lor, ce sunt diferite de la un sistem la altul. În schimb, structura de baza a fişierului `fstab` este tot timpul aceeaşi.

- **Structura fişierului fstab**
- Fiecare linie conţine informaţii despre un dispozitiv sau o partiţie (sistem de fisiere).
- **Prima coloană** conţine numele dispozitivului ce reprezintă sistemul de fişiere. Cuvântul **none** indică sisteme de fişiere (`/proc`,`/dev/shm`, `/dev/pts`) care nu sunt asociate cu dispozitive speciale. Cu opţiunea **label** se poate indica o etichetă de volum în locul unui nume de dispozitiv. În felul acesta, se poate muta un volum pe un alt nume de dispozitiv, fără a se face modificări în `fstab`.
- **A doua coloană** conţine punctul de montare în sistemul de fişiere. Trebuie ca punctul de montare să fie un director care există în sistem, altfel el trebuie creat manual. Unele partiţii şi dispozitive sunt montate automat la bootarea sistemului Linux.
- **A treia coloană** conţine tipul sistemului de fisiere (vor fi prez. imediat).

- **A patra coloană** conţine optiunile de montare (comanda **mount**).

  - `auto, noauto` Cu opţiunea `auto`, dispozitivul va fi montat automat (la bootare) sau când se lanseaza comanda `mount.auto` este opţiunea implicită. Cu `noauto`, dispozitivul poate fi montat doar în mod explicit. În cazul în care nu doriţi ca partiţia respectivă să fie montată la bootare, specificaţi opţiunea `noauto`.

  - `user, nouser` Opţiunea `user` permite utilizatorilor normali sa monteze dispozitivul, în timp ce `nouser` permite doar adm. sa monteze dispozitivul; `nouser` este implicită.

  - `exec, noexec` Opţiunea `exec` se permite execuţia fişierele binare, care sunt pe acea partiţie, în timp ce `noexec` nu permite acest lucru. `noexec` este utilă pentru o partiţie care conţine fişierele binare ce nu trebuie execute, sau care nu pot fi executate (de exemplu de pe o partiţie Windows).

  - `ro` Monteaza sistemul de fis. in modul read-only.

  - `rw` Monteaza sistemul de fis. in modul read-write.

  - `defaults` Foloseste opt. implicite: `rw, suid, dev, exec, auto, nouser, async`.

  - `owner` Permite proprietarului unitatii sa o monteze.

- **A cincea coloană** (un numar) este folosită de comanda **dump** pentru a determina care sistem de fişiere trebuie salvat.
- **A şasea coloană** (un numar) este folosită de comanda `fsck` pentru a determina ordinea în care va verifica sistemele de fişiere la rebootare.
  - Sistemul de fis. coresp. găseste dir. rad. (/) va avea valoarea `1`

  - celelalte vor avea în general valoare `2` (sau `0` în cazul în care nu se doreşte verificarea lor).

- **Sistemele de fişiere**
- Sistemele de fişiere cele mai întâlnite sunt:
- `ext3` este continuatorul lui `ext2`, care la rândul lui este o perfecţionare a lui `ext`.
- Sistemul `ext` a fost unul din primele sisteme fisiere din Linux; el recunoaşte sistemele de fişiere din UNIX.
- El a a fost înlocuit `ext2`. Acesta a fost unul dintre cele mai rapide sisteme de fişiere. El a introdus suportul pentru volume pâna la 4 TB, gestiunea numelor lungi si suportul complet al fisierelor UNIX. Pentru a putea fi compatibil cu viitoarele versiuni, utilizeaza `hook`, un fel de plug-in ce permite extinderea functionalităţilor, mentinând structura de bază a sistemelor de fişiere.
- Datorita acestei caracteristici a luat nastere `ext3`, care introduce caracteristici de journalizare pentru **metadate** (atributele fişierului) sau pentru copierea fişierului în timpul fazei iniţiale de actualizare a jurnalului; această ultima opţiune, garantează o mai bună recuperare a datelor.

- `ReiserFS` este un sistem de fisiere cu jurnalizare. Acesta lucrează folosind metadate particulare asociate fişierelor, ceea ce îi permite să recupereze fisierele, după eventualele blocaje de sistem, cu o rapiditate şi o fiabilitate superioară altor sisteme. ReiserFS este disponibil începând cu versiunea 2.4.1 a nucleuluilui şi este folosit de multe distribuţii (printre care şi Gentoo – www.gentoo.org) ca sistem de fisiere implicit în locul clasicului ext3. Avantajul acestui sistem de fiş.: nu este legat de tehnologii anterioare precum `ext3`.
- `iso9660` recunoaşte nume lungi de fişiere şi informaţii obţinute în stil UNIX (permisiuni pentru fişiere, proprietatea asupra fişierelor şi legături).
- `proc` nu este un sist. de fişiere adevărat, ci o interfaţă de sist. de fişiere la nucleul Linux. Multe utilitare Linux se bazează pe `/proc`.
- `swap` este utilizat pentru partiţiile de memorie virtuală.
- `nfs` (Network File System) este tipul de sist.de fişiere utilizat pentru montarea sistemelor de fişiere de pe alte calculatoare din reţea.

- **Exemplu** de fişier fstab :
- `$ more /etc/fstab`

```
LABEL=/         /               ext3    defaults           1 1
LABEL=/boot     /boot           ext3    defaults           1 2
none            /dev/pts        devpts  gid=5,mode=620     0 0
/dev/fd0        /mnt/floppy     auto    noauto,owner       0 0
none            /proc           proc    defaults           0 0
/dev/hda5       swap            swap    defaults           0 0
/dev/cdrom      /mnt/cdrom      iso9660 noauto,owner,ro    0 0
/dev/hda1       /mnt/win        vfat    noauto             0 0
```

- **Observaţie.** Partiţiile de hard-disc rădăcină (`/`), de încărcare (`/boot`), sunt montate la încărcariea sistemului de operare.

- **Montarea şi demontarea sistemelor de fişiere.** Partiţiile specificate în fstab devin stabile şi pot fi folosite în formele prescurtate ale comenzii **mount**, pe care o vom studia în continuare.
- Sintaxa comenzii este de forma:

```
mount [opţiuni] [<dispozitiv>] [director]
```

- Opţiunile sunt:

`-a` montează toate intrările din fstab;

`-t tip_sist_fis` specifică tipul sistemului de fişiere care este montat;

`-o optiuni` specifică opţiunile procesului de montare (de obicei sunt specifice tipului de sistem de fişiere)

- În această situaţie, opţiunile cele mai utilizate sunt:

`ro` montează sistemul de fişiere numai pentru citire (read-only);

`rw` montează sistemul de fişiere pentru citire şi scriere (read-write implicit);

`exec` permite execuţia fişierelor binare (implicit);

`loop` permite montarea unui fişier ca şi când ar fi un dispozitiv;

Exemplu. Este permisă montarea unei imagini montarea unei imagini ISO a unui CD fără a scrie pe un CD-ROM.

- În general, comanda `mount` este utilizată în două moduri:

  - pentru a afişa discurile, partiţiile şi sistemele de fişiere de la distanţă, care sunt montate curent (se tastează `mount` fără parametri);

  - pentru a monta temporar un sistem de fişiere.

- **Exemple**.

1. `#mount` , va afişa toate sistemele de fişiere montate în sistem.
2. `#mount –t ext3` , afişează toate sistemele de fişiere de tip `ext3`.
3. `#mount –t ext3 -l` , pentru listarea etichetelor partiţiilor montate.
4. La încărcarea sistemului de operare se execută automat:

`#mount –a` , prin care se mont. simultan toate sist. de fişiere config. în `fstab`.

5. Pentru montarea altor sisteme de fişiere este indicat să se creze câte un director separat pentru fiecare dintre ele. De **exemplu**, se poate crea directorul `/mnt` şi apoi în acest director să se creeze subdirectoare separate:

`/mnt/windows` - partitia pe care se afla Windows

`/mnt/cdrom` – cdrom

`/mnt/floppy` - discheta de 1.44

`# mount /dev/hda -t vfat /mnt/windows` , pentru a monta part. Windows

`# mount /dev/cdrom -r /mnt/cdrom` , pentru a monta un CD

`# mount /dev/fd0 -t /mnt/floppy` , pentru a monta o dischetă

- Dupa teminarea utilizării unui sistem de fişiere montat parţial sau pentru demontarea temporară a unui sistem de fişiere se foloseşte cda `umount`:

```
#umount [optiuni] director
```

În care director este directorul care urmează a fi demontat.

- **Exemple**.

1. Demontarea dischetei montată în exemplul anterior:

```
#umount /mnt/floppy
```

- Acelaşi efect se obţine dacă se tastează:

```
#umount /dev/fd0
```

- Este indicat să se folosească prima variantă (numele de director), deorece demontarea va eşua dacă unitatea este montată în mai multe locuri.
- Dacă este afişat mesajul **device is busy**, înseamnă că demontarea a eşuat datorită faptului că un proces are un fişier deschis pe directoul (unitatea) care este obiectul demontării sau există un fişier de comenzi(shell), al cărui director curent este cel care se doreşte a fi demontat.
- Dacă se foloseşte `umount -l`, demontarea se va realiza de îndată ce unitatea va deveni liberă. Pentru a se forţa demontarea unui sistem de fişiere care nu este disponibil se foloseşte `umount -f`.

- **Crearea unui sistem de fişiere** se realizează cu comanda mkfs, a cărei sintaxă este:

```
#mkfs –t <tip_sist_de_fis> <dispozitiv>
```

Exemple.

```
#mkfs –t ext3 /dev/fd0
```

Crează un sistem de fişiere `ext3` pe dischetă.

```
#mkfs –t ext3 /dev/hda2
```

Crează un sistem de fişiere ext3 pe a doua partiţie a discului `hda`.

- **Verificarea si repararea sistemelor de fisiere** se face cu utilitarul `fsck` (**F**ile **S**ystem **C**heck). Astfel de reparaţii sunt necesare, de exemplu după o cădere a sistemului, în care acesta nu a reuşit să salveze pe disc conţinutul tuturor zonelor tampon. `fsck` poate rula doar pentru sisteme nemontate (exceptând sistemul radacină); pentru aceasta, sistemul trebuie adus în mod singleuser.
- Comanda este apelată automat la pornire pentru un anumit sistem de fişiere, dacă acesta prezintă erori sau dacă nu au fost demontate corect.
- Administratorul sistemului poate el însuşi să lanseze în execuţie comanda, atunci când observă că un anumit sistem de fişiere se comportă necorespunzător.

- Sintaxa comenzii este:

`#fsck [optiuni] [-t tip] sist_fis`

- Optiuni:

`-A`, execută verificările pe toate sistemele specificate în `fstab`.

`-N`, nu execută nici o acţiune dar arată ce trebuie făcut.

`-t tip`, specifică tipul sistemului de fişiere care trebuie verificat; implicit este `ext2`.

- În cazul sistemelor de fişiere `ext2` şi `ext3` comanda utilizată este `e2fsck`. Parametrii acestei comenzi sunt:

`-b superbloc` , specifică numărul superblocului din care `e2fsck` ar trebui să citească informaţiile despre partiţia respectivă (de obicei se foloseşte `-b 8193`).

`-c`, cere să se execute mai întâi programul `badbloks`, care verifică, pentru fiecare bloc integritatea discului; este o verificare amănunţită a discului care presupune o durată mare.

`-f`, forţează verificarea, chiar dacă nu există informaţii că sistemul de fişiere nu este în regulă.

`-y`, comunică lui `e2fsck` să considere că la toate eventualele întrebări răspunsul este da.

- **Exemple.**

`#fsck /dev/hda2` , verif. Sist.de fiş. ext3 de pe a doua part.a discului hda.

`#e2fsck /dev/hda2`, acelaşi efect ca în exemplul ant.

`#e2fsck -f -y /dev/hda2` , forţează verificarea şi presupune că răspunsul este da, la toate eventualele întrebări.

- **Directorul lost+found** conţine segmente de fisier recuperate de `fsck` pe care nu le poate reunifica cu fişierul din care provin. Acest director este localizat acolo unde este montată şi partiţia. De **exemplu**, dacă `/dev/hda2` este montată în `/usr`, atunci `/usr /lost+found` corespunde partiţiei `/dev/hda2`