Laboratorul 5

1. Legături simbolice

Sub sistemele Linux, conceptul de partajare a fişierelor de către mai mulţi utilizatori este mult mai dezvoltat decât sub alte sisteme de operare. El se sprijină pe conceptele de legătură şi drepturi de acces la fişiere, pe care le vom dezvolta în continuare.

Fiecare utilizator îşi poate construi propria sa structură de directoare şi fişiere în directorul său gazdă. Pentru ca un fişier să poată fi partajat de mai mulţi utilizatori, el trebuie să aparţină structurilor corespunzătoare acestora. În acest sens, apare problema ca un fişier să fie fiul mai multor directoare. Pentru a se evita apariţia de informaţii redundante despre acel fişier, informaţiile sunt păstrate într-un director care aparţine structurii utilizatorului care a creat acel fişier şi în celelalte directoare se crează câte o legătură către fişierul respectiv. De asemenea, în cadrul structurii de fişiere a unui utilizator, atunci când o aplicaţie lansată dintr-un director, foloseşte un fişier dintr-un alt director, poate utiliza în locul căii către acel fişier o legătură simbolică la acesta.

Există două tipuri de legături: `fixe` ("hard"), care se creează implicit şi `simbolice`. Legăturile fixe reprezintă un alt nume dat unui fişier; legăturile simbolice funcţionează ca scurtaturi ("shortcut") către un anumit fişier. Legăturile fixe pot funcţiona la nivelul unui singur sistem de fişiere local, pe când cele simbolice pot lucra peste mai multe sisteme de fişiere. Din aceste cauze, cele simbolice sunt cel mai des folosite.

Crearea unei noi legături pentru un fişier se face cu comanda `ln` ("link"), care are s**intaxa:**

```
ln [-fs] numecale [legătură]
```

sau:

```
ln [-fs] numecale... catalog
```

- Comanda este înrudită cu `cp` în sensul că produce noi intrări în catalog, dar nu produce şi o copiere fizică a fişierului.
- Dacă se foloseşte prima formă sintactică şi sunt prezente ambele argumente (fără opţiuni), atunci noua intrare de catalog apare sub numele `legătură`.
- Dacă al doilea argument lipseşte, legătura creată în catalogul curent va avea acelaşi nume ca şi ultima componentă din `nume_cale`.
- Dacă se foloseşte a doua formă sintactică, atunci în `catalog` se creează legături având ca nume ultimele componente din lista de `nume_cale`.

O legătură simbolică este de fapt un fişier text, al cărui conţinut este `nume_cale` şi al cărui nume este `legătură` (sau ultima componentă din `nume_cale`). Legăturile simbolice pot traversa sistemele de fişiere. Nu se pot crea legături fixe spre un catalog. Doar superutilizatorul poate forţa această operaţie prin opţiunea `-f` ("force").

Exemplu. Relativ la structura din figura 2.2.2. ; dacă se tastează:

```
$cd /home/ilflorea
$ln –s /usr/local/bin/fis myfis
$ls -l myfis
Lrwxrvxrwx 1 ilflorea prof 19 Oct 17 12:02
myfis ->/usr/local/bin/fis
```

În catalogul gazdă al utilizatorului s-a creat o legătură simbolică, `myfis`, spre fişierul `/usr/local/bin/fis`. Se observă că prin comanda `ls –l`, lungimea fişierului `myfis` este afişată a fi 18, ceea ce corespunde cu numărul de caractere din numele de cale dat.

1. Copiaţi recursiv un director în directorul gazdă.
2. Creaţi o legătură simbolică din directorul gazdă către un fişier dintr-un alt director

2. Drepturi de acces

Dacă se tastează comanda `ls -l cale`, pentru fiecare fişier, după primul caracter urmează un grup de 9 caractere, care definesc drepturile de acces la fişierul respectiv. În raport cu un fişier sau director, din punct de vedere al drepturilor, utilizatorii se împart în trei categorii:

- proprietarul fişierului, notat cu `u` (user).
- grupul de utilizatori, notat cu `g` (group), cu drepturi specifice, de obicei mai slabe decât ale proprietarului, dar mai puternice decât ale restului utilizatorilor.
- restul utilizatorilor, notat cu `o` (others), cei care nu sunt în primele două categorii.

Pentru toţi utilizatorii se foloseşte notaţia *a* (all), echivalentă cu combinaţia `ugo`.

Exemplu. Presupunem că avem un fişier cu notele studenţilor dintr-o anumită grupă la o disciplină. Proprietarul fişierului este numele de user al profesorului care a predat disciplina, grupul este format din numele de user ale studenţilor care au susţinut examen la disciplina respectivă.

Fiecare utilizator are asociat un identificator unic UID (**U**ser **Id**entification), care este un număr natural. De asemenea, grupurile de utilizatori se identifică prin numere numite GID-uri (**G**roup **ID**entification). Un utilizator aparte, cu drepturi depline asupra tuturor fişierelor este *root* sau superuserul. Pentru fiecare categorie de utilizatori, fişierul permite trei drepturi:

(i) dreptul de citire, notat cu `r` (*read*)

(ii) dreptul de scriere, notat cu `w` (*write*) care include crearea de subdirectori, ştergerea de subdirectori, adăugarea sau ştergerea de intrări în director, modificarea fişierului etc.

(iii)dreptul de execuţie, notat cu `x` (*execution*) care permite lansarea în execuţie a unui fişier. Acest drept, conferit unui director, permite accesul în directorul respectiv (`cd`) .

În consecinţă, pentru specificarea drepturilor de mai sus asupra unui fişier sau director sunt necesari 9 biţi. Reprezentarea externă a acestei configuraţii se face printr-un grup de 9 caractere `rwxrwxrwx`, în care absenţa unuia dintre drepturi la o categorie de utilizatori este indicată prin - (minus).

Modul de atribuire a acestor drepturi se poate face cu ajutorul comenzii `chmod`. Cea mai simplă formă a ei este:`chmod mod fisier`
unde `mod` poate fi **absolut** sau **simbolic.** În **cazul absolut,** este un întreg reprezentat în sistemul de numeraţie `8`. În reprezentare binară, prima grupă de `3` biţi indică drepturile proprietarului(user-ului), cea de-a doua indică drepturile grupului, iar cea de-a treia cifră indică drepturile restului utilizatorilor. Corespunzător unei anumite cifre octale, drepturile setate sunt prezentate în tabelul 1. (simbolul – înseamnă că dreptul respectiv nu este setat).

Număr	Permisiune
0	---
1	--x
2	-w-
3	-wx
4	r--
5	r-x
6	rw-
7	rwx

Tabelul 1. Codificarea drepturilor în cifre octale

Modul simbolic este o combinaţie între literele `u,g,o,r,w,x` şi semnele `+` (acordarea unui drept), - (ridicarea unui drept).

Exemple. Comanda:
`$chmod 755 fis1`
fixează pentru fişierul `fis1` din directorul curent, toate drepturile pentru user, de citire şi execuţie pentru grup şi ceilalţi utilizatorii. În urma acestei comenzi, drepturile asupra fişierului `fis1` vor deveni: `rwxr-xr-x` (în reprezentare binară `755` este `111101101`).
Pentru a se da grupului de utilizatori drepturi de scriere, se tastează:
`$chmod g+w fis1`
Pentru a se lua grupului de utilizatori drepturile de execuţie, se tastează:
`$chmod g-x fis1`
Dacă se dă comanda:
`$chmod a-r fis1`
se va lua dreptul de citire al tuturor utilizatorilor.

Drepturi implicite. În momentul intrării în sistem, unui utilizator i se vor acorda nişte drepturi implicite pentru fişiere nou create. Toate fişierele şi directoarele create de utilizator pe durata sesiunii de lucru îl vor avea ca proprietar, iar drepturile vor fi cele primite implicit. Fixarea drepturilor implicite sau aflarea valorii acestora se poate face folosind comanda `umask`. Drepturile implicite sunt stabilite scăzând (octal) masca definită prin `umask` din 777. Pentru a se afla valoarea măştii se lansează comanda `umask.` Rezultatul este afişarea măştii, de regulă 022. Deci drepturile implicite vor fi: 777-022=755, adică userul are toate drepturile, iar grupul şi restul vor avea doar drepturi de citire şi de execuţie.

Exemplu. Dacă se doreşte schimbarea acestei măşti pentru a da userului toate drepturile, grupului drept de citire şi execuţie iar restului nici un drept, (corespunzătoare constantei octale 750), se va da comanda
`$umask 027 (777-027=750).`
Efectul ei va rămâne valabil până la un nou `umask` sau până la încheierea sesiunii.

1.Scrieţi o comandă `chmod` prin care, relativ la un fişier să se seteze: toate drepturile pentru proprietar, dreptul de citire pentru grupul de utilizatori şi nici un drept pentru restul utilizatorilor; se va folosi reprezentarea numerică a drepturilor.
2. Folosind reprezentarea simbolică a drepturilor, să se scrie comenzile `chmod` care să seteze toate drepturile pentru grup şi dreptul de citire pentru restul utilizatorilor.
3. Să se scrie o comandă `umask,` prin care să se seteze drepturile implicite următoarele: toate drepturile pentru proprietar, drepturi de citire pentru grup, nici un drepr pentru restul utilizatorilor.

3. Căutarea fişierelor într-o structură de directoare

Căutarea se face cu comanda `find` a cărei sintaxă generală este:

```
find cale criterii [actiune]
```

`cale` specifică numele directorului ce reprezintă rădăcina structurii în care se face căutarea.
`criterii` reprezintă specificaţiile după care se face căutarea.

`actiune` specifică eventuale prelucrări care se fac asupra fişierelor găsite.

Căutarea dupa nume se face cu opţiunea `–name.` Opţiunea `-name este` "case sensitive"; se foloseşte opţiunea `-iname` dacă nu se doreşte să se facă distinţie între literele mari şi cele mici. În căutarea fişierelor, se pot folosi şabloane şi în acest caz numele de fişier este specificat între apostroafe.

Se foloseşte `-regex` în locul lui `-name` pentru a face căutarea pentru fişiere al căror nume trebuie să îndeplinească un şablon definit printr`-o expresie regulată, concept care îl vom discuta într-un modul următor.`

Exemple.

`$find / -name fisier`

va lista toate fişierele dintr-un sistem de fisiere (de pe un disc) al căror nume este `fisier` ; această comandă va căuta în toate directoarele la care este permis accesul; dacă nu este permis accesul se va semnala acest lucru.

Dacă nu se doreşte să se facă distinţie între literele mari şi cele mici, se tastează:

`$find / -iname fisier`

Această comandă va include în rezultatul căutării şi `Fisier` sau `fIsier` , de exemplu.

În căutarea fişierelor se pot folosi şi şabloane şi în acest caz numele de fişier este specificat între apostroafe.

`$find / -name 'fis*'`

listează toate fişierele din sistem al căror nume începe cu caracterele `` `fis' ``.

`$find / -name 'fis???'`

listează toate fişierele din sistem al căror nume începe cu caracterele `` `fis', `` urmate de trei caractere oarecare.

`$ find /home/ilflorea -name '*fis*'`

listează toate fişierele din din structura care porneşte din directorul `/home/ilflorea` cu textul `` `fis' `` inclus în nume.

`$find . -regex `.*\(a1\|a2\).*``

listează toate fişierele(inclusiv calea) din structura care porneşte din directorul curent al căror nume contine şirurile `` `a1` `` sau `` `a2` `` oriunde în numele lor.

Căutarea fişierelor după dimensiune se face cu opţiunea `-size`, urmată de un număr natural, opţional precedat de un semn. Avem trei cazuri:

- când este precedat de semnul `` `+` ``, `rezultatul căutării va fi toate fişierele care au dimensiunea mai mare decât numărul dat;`
- când este precedat de semnul `` `-` ``,`vor fi selectate toate fişierele a căror dimensiune este mai mică decât numărul dat;`
- `când nu este specificat nici un semn rezultatul va consta din toate fişierele care au exact aceeaşi dimensiune.`

`Unitatea implicită este blocul de 512` baiţi; dacă numărul specificat anterior este urmat de caracterul `k, respectiv b` `atunci unitatea de măsură este` **kilobaitul**, respectiv **baitul**. Se foloseşte opţiunea `-empty` pentru a se găsi toate fişierele de dimensiune nulă.

Exemple.

`$find ~ -size +1000k`

va lista toate fişierele din structura care porneşte din directorul gazdă care au mai mult de `1000` kilobaiţi.

`$find ~ -size -500b`

va lista toate fişierele din structura care porneşte din directorul gazdă care au o

dimensiune mai mică de `500` baiţi

```
$find / -size 42
```

va lista toate fişierele din sistem care au o dimensiune de 42 de blocuri a câte 512 baiţi.

```
$find ~ -empty
```

va afişa toate fişierele vide din structura care porneşte din directorul gazdă.

Căutarea fişierelor după momentul modificării. Pentru a găsi fişiere modificate într-un interval de timp specificat, se folosesc opţiunile `-mtime` sau `-mmin`; argumentul folosit cu `-mtime` specifică numărul de zile, pe când cel cu `–mmin` specifică numărul de minute.

Exemple.

```
$find /usr/local -mtime -1
```

va lista toate fişierele din structura care porneşte din directorul `/usr/local` care au fost modificate în ultimele 24 de ore.

```
$find /usr -mmin 5
```

va lista toate fişierele din structura care porneşte din directorul `/usr` care au fost modificate în ultimele 5 minute.

Pentru a specifica un interval de timp, numărul utilizat ca argument, va fi precedat de semnul `+`, respectiv `-`, având semnificaţia selectării tuturor fişierelor pentru care timpul de la ultima modificare este mai mare sau egal, respectiv mai mic sau egal decât argumentul specificat.

Exemple.

```
$find /usr/local -mtime +1 -2
```

va afişa toate fişierele din structura care porneşte din directorul`/usr/local` care au fost modificate în ultimele 24 de ore.

```
$find /usr -mmin -5
```

va lista toate fişierele din structura care porneşte din directorul `/usr` care au fost modificate în ultimele 5 minute.

Opţiunea `-daystart` va specifica faptul că contorizarea timpului se va face începând din ziua curentă, în loc de ziua trecută.

Exemple.

```
$find ~ -mtime 2 -daystart
```

va lista toate fişierele din structura care porneşte din directorul gazdă care au fost modificate de acum două zile.

```
$find /usr -mtime +356 -daystart
```

va lista toate fişierele din structura care porneşte din `/usr` care au fost modificate de cel mult un an.

```
$find ~ -mtime 2 -mtime -4 -daystart
```

va lista toate fişierele din structura care porneşte din directorul gazdă, care au fost modificate acum două până la patru zile. În exemplul precedent, am combinat opţiunile `-mtime 2` şi `-mtime -4.`

Opţiunea `-newer` va fi folosită pentru a găsi fişierele mai noi decât un fişier dat.

Exemplu.

```
$find ~ -newer /etc/fis
```

va lista toate fişierele din structura care porneşte din `~` care au au data de creare mai recentă decât fişierul `/etc/fis.`

Pentru a căuta fişiere a căror dată de creare este ulterioară unei date specificate, se va folosi comanda `touch` pentru a introduce acea dată într-un fişier, care va fi argument pentu opţiunea `-newer`.

Exemplu.

```
$touch -t 04012010 /tmp/timestamp
$find ~ -newer /tmp/timestamp
```

va căuta toate fişierele din structura care porneşte din directorul gazdă care au fost modificate după 1 aprilie 2010.

Opţiunea `-used` este folosită pentru a găsi fişiere accesate după un număr de zile de la ultima modificare.

Exemplu.

```
$find ~  -used +100
```

va căuta fişierele din structura care porneşte din directorul gazdă care au fost accesate după cel puţin 100 de zile de la data ultimei modificări.

Căutarea fişierelor după proprietar. Cu opţiunea `-user, respectiv -group` se pot căuta fişiere care aparţin unui anumit utilizator, respectiv unui grup de utilizatori.

Exemplu.

```
$find /usr/local/fonts -user florea
```

va lista toate fişierele din structura care porneşte din `/usr/local/fonts` care aparţin utilizatorului `florea`.

```
$find ~ -group info
```

va lista toate fişierele din structura care porneşte din directorul ~ al căror proprietar este grupul `info`.

Execuţia de comenzi asupra fişierelor căutate se realizează cu opţiunea `-exec` care are ca argument o anumită comandă. Sfârşitul comenzii este marcat de caracterele `';'`. Dacă se foloseşte şirul `'{}'` în cadrul comenzii, acesta este înlocuit cu numele de fişier care a fost găsit

Exemple.

```
$find . -name ''*fis*'' -exec rm '{}'  ';';
```

va şterge toate fişierele care în numele lor conţin subşirul `fis`, situate în directorul curent sau în subdirectoarele acestuia.

```
$find ~/html/ -name '*.html' -exec grep linux `{}` `;`
```

va găsi toate fişierele din structura care porneşte din directorul `~/html/` cu extensia `.html` şi va afişa liniile din aceste fişiere care conţin textul `linux' .

Pentru a se realiza comanda respectivă cu confirmare, se foloseşte `-ok` în locul lui `-exec`.

```
$find ~ -used +365 -ok rm `{}` `;`
```

va găsi toate fişierele din structura care porneşte din directorul gazdă, care au fost accesate la mai mult de un an de când au fost modificate şi va aştepta confirmarea înainte de ştergere.

Căutarea fişierelor după tip se face cu opţiunea `-type c` ; rezultatul este "true" dacă tipul fişierului este `c`, un caracter care poate fi: `f` - fişier simplu; `d` - catalog; `b` - fişier special bloc; `c` - fişier special caracter; `p` - "pipe" (canal) cu nume; `s` - "socket" (pentru comunicarea în reţea).

Exemplu. Pentru a lista numărul fişierelor din structura care porneşte din directorul `gazda se tastează:`

```
$find ~ \! -type d|wc -l
```

Observăm legarea în pipe a comenzilor şi utilizarea operatorului de negare.

```
$ find /usr/share  -type d | wc –l
```

va lista numărul directoarelor din structura care porneşte din directorul `/usr/share.`

Căutarea fişierelor după mai multe criterii. Se pot combina mai multe opţiuni pentru a se găsi fişierele care îndeplinesc criterii multiple.

Exemple.

```
$find ~ -name 'fis*' -newer /etc/ultmod
```

va lista fişierele din structura care porneşte din directorul gazdă al căror nume începe cu şirul `` `fis' `` şi care sunt mai noi decât fişierul `/etc/ultmod`

```
$find ~ -size +2000000c -regex '.*.[^gz]' -exec gzip
'{}' ';'
```

va comprima toate fişierele din structura care porneşte din directorul gazdă şi care au o dimensiune de cel puţin 2 megabaiţi şi care nu au fost comprimate folosind comanda `gzip` .

Alte optiuni:

a) `-perm onum` - "true" dacă drepturile de acces la fişier corespund exact numărului octal `onum`.

b) `-print` - întotdeauna adevărată, are ca efect afişarea numelui fişierului curent (care corespunde unui criteriu de căutare anterior specificat);

c) `-ls` - întotdeauna adevărată, are ca efect afişarea numelui de cale al fişierului curent şi informaţii suplimentare despre acesta (analog comenzii `ls -li`);

d) `-prune` - întotdeauna adevărată, are ca efect oprirea căutării la numele de cale curent, dacă acesta este catalog (nu se caută recursiv).