Poglavje 4

Datotečni sistem

4.1 Logična organizacija podatkov

4.1.1 Datoteke in imeniki

Računalniški podatki, ki zahtevajo dolgotrajno hranjenje, so fizično shranjeni na vhodno-izhodnih napravah oz. pomnilnih medijih, kot so npr. trdi disk, disketa, zgoščenka, magnetni trak, pomnilniška kartica itd. Metode fizičnega hranjenja podakov na različnih pomnilnih medijih se lahko med seboj zelo razlikujejo, ravno tako tudi načini dostopa do shranjenih podatkov. Z uporabniškega stališča fizična organizacija podatkov na mediju ni zanimiva, zato je nujno, da operacijski sistem nudi enotno abstrakcijo shranjenih podatkov in to ne glede na fizične omejitve pomnilnega medija. Takšni abstrakciji podatkov pravimo *logična organizacija podatkov*. Večina operacijskih sistemov podatke logično organizira s pomočjo datotek in imenikov, pri čemer tudi Linux ni izjema. Oglejmo si logično organizacijo podatkov v Linuxu in njemu podobnih sistemih.

Datoteke

Osnovna zaključena zbirka podatkov se imenuje *datoteka*[1]. Podatkom, ki jih hrani neka datoteka, pravimo *vsebina datoteke*. Primer vsebine je izvorna koda, slika, seminarska naloga, preglednica itd.

[1] angl. file

Vsaka datoteka ima svoje *ime*. Ime datoteke lahko vsebuje male in velike črke abecede, številke in tudi večino simbolov. V Linuxu je pomembno razlikovanje med malimi in velikimi črkami. Zaradi posebnega pomena nekaterih simbolov je potrebno biti pri njihovi uporabi posebej pazljiv.

Poleg imena so pogosto zanimivi še drugi podatki o datoteki[2], kot so:

- datum in čas nastanka,
- datum in čas zadnje spremembe,
- dovoljenja za uporabo,
- lastnik (uporabnik) datoteke,
- skupina, kateri pripada datoteka,
- velikost datoteke,
- fizična lokacija vsebine.

Imeniki

Načinu hranjenja in organizacije datotek na pomnilnem mediju pravimo *datotečni sistem*[3]. Ta navadno vsebuje veliko število datotek, zato sorodne datoteke združujemo v skupine, ki jim pravimo *imeniki*[4]. Datotečni sistem lahko vsebuje veliko imenikov.

Imenik vsebuje datoteke oz. če smo natančnejši, *imenske vnose*[5]. Imenski vnos je poseben zapis, ki hrani pomembne meta-podatke datoteke, od katerih sta še posebej pomembna ime datoteke in lokacija njene vsebine.

Imenski vnos pove, kje se nahaja vsebina, ki sodi k ustreznemu imenu. Vsako ime določa natanko eno lokacijo vsebine, zato se v imeniku ne sme nahajati več imenskih vnosov z istim imenom. Vendar pa lahko različna imena določajo isto lokacijo, kot se to zgodi pri uporabi *trdih povezav*, o katerih bomo več povedali v nadaljevanju.

[2]Gre pravzaprav za podatke o podatkih oz. za meta-podatke.
[3]angl. file system
[4]angl. directory
[5]angl. directory entry

Struktura imenikov

Vsebina imenika je seznam imenskih vnosov. To je zbirka podatkov, zato je imenik pravzaprav datoteka, katere vsebina ustreza določenemu formatu. Potemtakem, ker imenik vsebuje datoteke, lahko vsebuje tudi imenike. Med imeniki torej obstaja hierarhičen odnos. Imeniku, ki je v nekem imeniku vsebovan, pravimo *podimenik*; imeniku, ki imenike vsebuje, pa *nadimenik*.

Vsak imenik je vsebovan v vsaj enem nadimeniku, z izjemo enega imenika, ki ni vsebovan v nobenem imeniku. Ta se imenuje *korenski imenik*[6]. Označimo ga s poševnico ('/'). Če potujemo od korenskega imenika v globino preko podimenikov, lahko dosežemo katerikoli datoteko. Vsi imeniki, s pričetkom v korenskem imeniku, tvorijo hierarhijo, ki ji pravimo *struktura imenikov*. To strukturo si večkrat ponazorimo z drevesnim diagramom, vendar ta ponazoritev ni vedno popolnoma na mestu, ker je (s pomočjo povezav) možno ustvariti tudi ne-drevesno hierarhijo.

Naslavljanje datotek

Seznam imenikov (vključno s ciljno datoteko), ki jih je potrebno obiskati, da prispemo do želene datoteke, se imenuje *pot*[7]. Ločilo med posameznimi imeniki v poti je poševnica ("/"). S potjo lahko naslovimo poljubno datoteko v datotečnem sistemu.

Glede na imenik, v katerem pričnemo obiskovanje, poznamo dve vrsti poti:

- *absolutno pot*,
- *relativno pot*.

Absolutna pot se prične s poševnico, prvi imenik, ki ga je potrebno obiskati, pa je korenski imenik datotečnega sistema. V nasprotnem primeru, t.j. ko se pot ne prične s poševnico, gre za relativno pot, z začetkom obiskovanja v trenutnem delovnem imeniku.

Trenutni delovni imenik[8] je eden izmed parametrov vsakega procesa v Linuxu, torej tudi ukazne lupine. Trenutni delovni imenik je prav-

[6] angl. root directory
[7] angl. pathname
[8] angl. current working directory

zaprav absolutna pot do nekega imenika in je vedno predpona vsakega relativnega naslavljanja.

Pot lahko naslavlja tudi neobstoječo datoteko. Pot, ki na zadnjem mestu vsebuje poševnico, naslavlja imenik. V poteh se uporabljajo naslednji simboli:

Oznaka	Opis
/	Korenski imenik in ločilo med imeniki v poti.
.	Trenutni imenik.
..	Nadimenik trenutnega imenika.
~	Domači imenik uporabnika.

Primer. Če je trenutni delovni imenik enak `/home/jure/`, potem vse naslednje poti naslavljajo isto datoteko:

```
/home/jure/finance/dohodnina.pdf
finance/dohodnina.pdf
~/finance/dohodnina.pdf
../jure/./finance/../finance/dohodnina.pdf
```

V praksi razlikovanje med izrazi datoteka, ime datoteke, vsebina datoteke in imenski vnos ni tako dosledno, kot je opisano zgoraj. Pogosto se uporablja le izraz datoteka in tako bo tudi v nadaljevanju tega besedila. Kadar bi zaradi tega lahko prišlo do dvoumnosti, se bomo zopet zatekli k bolj striktnemu poimenovanju.

4.1.2 Izpis vsebine imenika: `ls`

Najprej si oglejmo enega izmed najpogosteje uporabljanih ukazov, t.j. ukaz `ls`[9] za izpis vsebine imenika. Kratka navodila za njegovo uporabo so naslednja:

`ls` [`-aAdhlmFQR1`] *pot**

Ukaz pogosto uporabljamo brez argumentov, v tem primeru izpiše vsebino trenutnega delovnega imenika. Če pa podamo enega ali več imenikov, potem zaporedoma izpiše vsebino vseh podanih imenikov.

[9] angl. list

V privzetem načinu delovanja, t.j. brez uporabe dodatnih stikal, se izpišejo le imena datotek v izbranem imeniku. S stikali pa lahko spremenimo način delovanja ukaza. Naštejmo nekaj uporabnejših stikal:

- `-a` ... izpis naj vsebuje tudi skrite datoteke,
- `-A` ... izpis naj vsebuje tudi skrite datoteke, razen `.` in `..`,
- `-d` ... izpis podanih imenskih vnosov namesto vsebine imenikov,
- `-h` ... izpis velikosti datotek v lažje berljivi obliki,
- `-l` ... izpis več podatkov,
- `-m` ... izpis imen bo med seboj ločen z vejico,
- `-F` ... izpis tipa datoteke za imenom datoteke s posebnim znakom,
- `-Q` ... izpis imen v dvojnih narekovajih,
- `-R` ... rekurzivni izpis vsebine imenika in vseh njegovih podimenikov,
- `-1` ... izpis ene datoteke na vrstico.

Izhodni status ukaza je enak 0 ob uspešni izvedbi, 1 v primeru manjših težav in 2 v primeru večjih težav.

Primera. Oglejmo si naslednja primera uporabe ukaza `ls`. Prvi primer za enostaven izpis vsebine trenutnega imenika.

```
jure@ris:~>ls     ... enostaven izpis
Bash navodila.pdf  hello-bash.c     SlikaNamizja.jpeg
hello-bash         izpit-letos.txt
```

V treh stolpcih se je izpisalo pet (imen) datotek, ki jih vsebuje trenutni delovni imenik. Opazimo, da lahko imena vsebujejo tudi presledke.

Če želimo videti prav vse datoteke, ukazu dodamo stikalo `-a` ali `-A`. Prav tako želimo več podatkov o vsaki izmed datotek, kar sporočimo s stikalom `-l`.

```
jure@ris:~>ls -Al     ... vse datoteke, več podatkov
total 140
-rw-r--r-- 1 jure users 86420 2008-03-06 18:57 Bash navodila.pdf
-rwxr-xr-x 1 jure users 10763 2008-03-06 16:43 hello-bash
-rw-r--r-- 1 jure users   101 2008-03-06 16:42 hello-bash.c
-rw-r--r-- 1 jure users    10 2008-03-06 18:58 izpit-letos.txt
```

```
-rw-r--r-- 1 jure users     0 2008-03-06 16:53 .izpit-resitve.txt
-rw-r--r-- 1 jure users 32551 2008-03-06 16:40 SlikaNamizja.jpeg
```

Izpis vsebuje osem stolpcev. Prvi stolpec vsebuje tip datoteke in dovoljenja za njeno uporabo. Drugi stolpec vsebuje število trdih povezav na datoteko, ki je v tem primeru vedno enak 1, kar pomeni, da je vsebina datoteke dostopna samo pod enim imenom. Nato sledita stolpec z lastnikom in skupino, katerima pripada datoteka. Peti stolpec prikazuje dolžino datoteke, šesti datum in sedmi čas zadnje spremembe datoteke. V zadnjem stolpcu se nahaja ime datoteke. V izpisu opazimo tudi dodatno datoteko, ki je v prejšnjem primeru ostala skrita. Imena skritih datotek se namreč pričnejo z znakom ".".

4.1.3 Izpis strukture imenikov: `tree`

Naslednji ukaz izpiše drevesno strukturo imenikov.

`tree` [*imenik*]

Koren izpisanega drevesa je podani imenik, če le-ta ni podan se privzame trenutni delovni imenik. Oglejmo si naslednji primer uporabe ukaza (izpis ukaza je zaradi prostora izpuščen):

jure@ris:~>`tree -dL 2 /` *... samo imeniki, globina 2, začni v korenu*

4.1.4 Izpis in izbira delovnega imenika: `pwd` in `cd`

Ukaz `pwd`[10] izpiše trenutni delovni imenik lupine. Njegova uporaba je enostavna:

```
pwd
```

Vendar obstajata vsaj dve njegovi različici. Prva je program `/bin/pwd`, ki iz izpisane poti odstrani simbolične povezave. Druga je v lupino `bash` vgrajeni ukaz, ki v izpisu ohrani simbolične povezave.

Delovni imenik lupine lahko zamenjamo z naslednjim ukazom[11]:

`cd` [*imenik*]

[10] angl. print working directory
[11] angl. change directory

Če imenika ne podamo, se trenutni delovni imenik nastavi na domači imenik uporabnika. Ukaz je vgrajen v lupino.

4.1.5 Ustvarjanje in odstranjevanje imenika: `mkdir` in `rmdir`

Ustvarjanje in odstranjevanje imenikov izvedemo s pomočjo naslednjih dveh ukazov.

`mkdir` [`-p`] *imenik*$^+$

Ukaz `mkdir`[12] ustvari enega ali več novih imenikov. S stikalom `-p` zahtevamo, da se ustvarijo tudi imeniki, ki morebiti še ne obstajajo, na celotni podani poti.

`rmdir` [`-p`] *imenik*$^+$

Ukaz `rmdir`[13] odstrani podane imenike, ki morajo biti prazni. Z uporabo stikala `-p` se odstranijo vsi imeniki na podani poti.

Oglejmo si nekaj primerov ustvarjanja in odstranjevanja imenikov.

```
jure@ris:~>mkdir veselje     ... ustvarjanje imenika z imenom veselje
jure@ris:~>mkdir kam so sle     ... ustvarjanje več imenikov
jure@ris:~>mkdir 'vse rozice'     ... ustvarjanje enega imenika
jure@ris:~>mkdir -p vija/vaja/ven     ... ustvarjanje celotne poti
jure@ris:~>rmdir -p vija/vaja/ven     ... odstranjevanje celotne poti
```

4.1.6 Kopiranje, preimenovanje in odstranjevanje datoteke: `cp`, `mv` in `rm`

Ukaz `cp`[14] kopira datoteko *izvor* v datoteko *ponor*. Pri tem ustvari novo datoteko in vanjo prepiše vsebino izvorne datoteke. Njegova uporaba je naslednja:

`cp` *izvor ponor*

[12] angl. make directory
[13] angl. remove directory
[14] angl. copy

Ime datoteke lahko spremenimo z ukazom `mv`[15]. Ukaz pravzaprav omogoča več kot enostavno preimenovanje datoteke, saj lahko z njim datoteko premaknemo na popolnoma drugo mesto. Njegova uporaba je sledeča:

`mv` *izvor ponor*

Ukaz `rm`[16] odstrani podane datoteke. Uporabimo ga na naslednji način:

`rm` [`-fir`] *datoteka*$^+$

Zanimivejša stikala so:

- `-f` ... ignoriraj napake pri brisanju,
- `-i` ... zahtevaj uporabnikovo potrditev za vsako brisanje,
- `-r` ... rekurzivno brisanje podimenikov.

Pri uporabi stikal `-fr` priporočamo izredno previdnost!

Primeri. Nekaj primerov rokovanja z datotekami.

```
jure@ris:~>cp naloga.txt nadloga.txt     ... kopiranje
jure@ris:~>cp /etc/passwd  /hacks/passwd.lalg     ... kopiranje
jure@ris:~>mv skuta.jpg 'skuta in struca.jpg'     ... preimenovanje
jure@ris:~>mv ocene.txt  /arhiv/ocene-2008.txt     ... prestavljanje
```

4.1.7 Obdelava imena datoteke: `basename` in `dirname`

Pri obdelavi poti nam večkrat prideta prav naslednja dva ukaza. Uporaba prvega je sledeča:

`basename` *pot* [*pripona*]

Ta ukaz iz podane *poti* odstrani imenik, ki vsebuje naslovljeno datoteko (ohrani le ime datoteke). Poleg tega odstrani še *pripono* datoteke, če je le-ta podana.

[15] angl. move
[16] angl. remove

Nasprotno deluje naslednji ukaz:

`dirname` *pot*

ki iz podane poti odstrani datoteko, t.j. izpiše le imenik, ki vsebuje datoteko. Pri uporabi obeh ukazov ni pomembno ali naslovljena datoteka obstaja ali ne.

Primeri. Oglejmo si nekaj primerov uporabe obeh ukazov.

```
jure@ris:~>basename finance/dohodnina.pdf
dohodnina.pdf
jure@ris:~>basename finance/dohodnina.pdf .pdf
dohodnina
jure@ris:~>dirname finance/dohodnina.pdf
finance
jure@ris:~>dirname dohodnina.pdf
.
```

Pri tem bodimo še posebej pozorni na izpis zadnjega ukaza.

4.1.8 Nastavitev datuma datoteke: `touch`

Za vsako datoteko se vodita naslednja dva datuma:

- datum zadnje spremembe datoteke,
- datum zadnjega dostopa do datoteke.

Naslednji ukaz nam omogoča nastavitev obeh datumov.

`touch` [`-am`] [`-d` *datum*] *datoteka*$^+$

Če podana datoteka ne obstaja, potem se ustvari nova datoteka. Uporabimo lahko naslednja stikala:

- `-a` ... nastavitev datuma in časa dostopa,
- `-m` ... nastavitev datuma in časa zadnje spremembe,
- `-d` *datum* ... nastavitev na podani *datum*.

Če ne podamo stikal `-a` ali `-m`, potem se nastavita oba datuma. Brez stikala `-d` se privzame trenutni čas.

4.2 Posebne datoteke

4.2.1 Tip in vrsta datoteke

Poleg *navadnih datotek*[17] in imenikov poznamo še druge *tipe datotek*. V resnici ni nujno, da datoteka neposredno opisuje neke fizične podatke na pomnilnem mediju. Z datoteko je lahko predstavljena poljubna naprava, ki je zmožna delovati kot tok podatkov.

Tipi datotek so skupaj z njihovimi oznakami zbrani v naslednji tabeli:

Oznaka	Opis	Primer
-	navadna datoteka	`/etc/passwd`
d	imenik	`/dev`
b	bločno orientirana naprava	`/dev/hda` (trdi disk)
c	znakovno orientirana naprava	`/dev/tty` (terminal)
l	simbolična povezava	`/usr/src/linux` ()
p	imenovana pipa[18]	ustvarimo z `mkfifo`
s	vtičnica[19]	

Zgornje oznake se izpišejo pri uporabi ukaza `ls`.

Poleg tega da se datoteke medsebojno razlikujejo po tipu, se lahko razlikujejo tudi po vrsti podatkov, ki jih lahko datoteka vsebuje. V primeru razlikovanja datotek po vsebini, bomo raje govorili o *vrsti datotek*. Tako npr. poznamo ASCI tekstovne datoteke, datoteke s programsko kodo, izvajalne datoteke itd. Večkrat je vrsta datoteke podana tako, da je imenu datoteke dodana pripona, ki opisuje vrsto.

4.2.2 Izpis tipa in vrste datoteke: `file`

Ukaz, ki izpiše tip in vrsto datoteke, je naslednji:

```
file datoteka+
```

Primeri. Za boljše razumevanje si oglejmo naslednje primere.

```
jure@ris:~>file /etc/passwd      ... navadna tekstovna datoteka
```

[17] angl. regular file
[18] angl. named pipe
[19] angl. socket

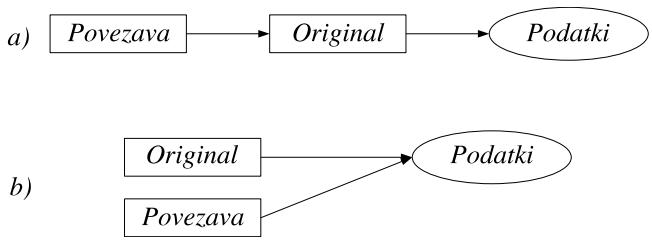
```
/etc/passwd:  UTF-8 Unicode text
jure@ris:~>file /dev/tty0     ... znakovno orientirana naprava
/dev/tty0:  character special
jure@ris:~>file init.sh     ... bash skripta
init.sh:  Bourne-Again shell script text
```

4.2.3 Simbolične in trde povezave

Simbolična povezava[20] je imenski vnos, ki kaže na nek drug imenski vnos. Predstavlja torej povezavo v dejanskem pomenu besede na ciljno datoteko. Brisanje simbolične povezave le-to izbriše, ciljna datoteka pa ostane nedotaknjena. Brisanje datoteke, na katero kaže neka simbolična povezava, izbriše datoteko, povezava pa ostane in v nadaljevanju kaže na neobstoječo datoteko.

Trda povezava[21] je imenski vnos, ki opisuje isto fizično datoteko kot nek drug imenski vnos. Predstavlja torej isto datoteko kot ciljna datoteka. Vsaka trda povezava je pravzaprav le dodaten imenski vnos poleg prvotnega. Po stvaritvi trde povezave ni več praktične razlike med ustvarjeno trdo povezavo in originalno datoteko. Če želimo datoteko dokončno odstraniti, moramo pobrisati prav vse trde povezave nanjo.

V praksi večkrat zasledimo in uporabljamo simbolične povezave kot trde. Razlika med simboličnimi in trdimi povezavami je prikazana na naslednji sliki.



Slika 4.1: a) Simbolične in b) trde povezave.

[20] angl. symbolic link, soft link
[21] angl. hard link

4.2.4 Ustvarjanje in branje povezave: `ln` in `readlink`

Ukaz `ln` omogoča ustvarjanje simboličnih in trdih povezav. Njegova uporaba je naslednja:

`ln` [`-s`] *cilj povezava*

Če podamo stikalo `-s`, se ustvari simbolična povezava, sicer pa trda. Pri tem je *cilj* pot do ciljne datoteke, *povezava* pa ime novo ustvarjene povezave.

Včasih želimo ugotoviti, kam kaže neka simbolična povezava. To nam omogoča naslednji ukaz:

`readlink` *datoteka*$^+$

Primeri. Simbolično povezavo `gesla` na datoteko `/etc/passwd` ustvarimo s prvim ukazom v naslednjem primeru, z drugim ukazom izvemo kam kaže ustvarjena povezava:

```
jure@ris:~>ln -s /etc/passwd gesla     ... simbolična povezava
jure@ris:~>readlink gesla     ... branje simbolične povezave
/etc/passwd
```

4.2.5 Informacije o datoteki: `stat`

Ukaz `stat` omogoča izpis informacij o podani datoteki. Uporabljamo ga na naslednji način:

`stat` *stikala datoteka*

Nekaj zanimivejših stikal:

- `-L` ... sledenje povezavam,
- `-c` *format* ... nastavitev formata izpisa,
- `-t` ... izpis v zgoščeni obliki.

Pri praktični uporabi ukaza je najbolje, da format izpisa poiščemo s pomočjo vgrajenega priročnika `man stat`.