



Programska podrška mjernih i procesnih sustava

vježba br. 8: **DATOTEČNI SUSTAV**

ZADACI ZA VJEŽBU

1. DEVICE DATOTEKE

Ulazno-izlaznim jedinicama (vanjskim uređajima) u Linux-u se pristupa čitanjem ili pisanjem u tzv. *device* datoteke. Sve se *device* datoteke nalaze u */dev* direktoriju. Tako je npr. prvi serijski port računala predstavljen *device* datotekom */dev/ttys0*. Npr., ukoliko bi htjeli modemu, koji je spojen na prvi serijski port poslati naredbu za reset, morali bi u odgovarajuću *device* datoteku upisati ATZ

```
# echo ATZ > /dev/ttys0
```

Da vidite kakve podatke šalje miš, ispište sadržaj datoteke */dev/mouse*:

```
# cat /dev/mouse
```

Pomicanje miša ili pritiskanje njegovih tipki bi trebalo na monitoru ispisivati znakove. Datoteka */dev/mouse* je zapravo link na datoteku */dev/psaux* koja predstavlja PS/2 port na kojeg je spojen miš.

Općenito, Linux razlikuje dvije vrste uređaja:

1. **blok uređaji** s nasumičnim pristupom (*random-access block devices*). Tipično su to diskovi. Podatke zapisujemo u blokovima određene veličine.
2. **znakovni uređaji** (*character devices*). Tipično trake i serijske linije. Podaci se direktno zapisuju znak po znak, kako koji pristigne, bez međupohranjivanja i čekanja da se skupi dovoljan broj znakova.

➤ Proučite sadržaj */dev* direktorija *ls -l* naredbom. Uočite da običan korisnik nema nikakva prava nad većim dijelom ovih datoteka.

Uz neke od *device* datoteka npr. piše: *crw-rw-rw-*. Prvo slovo *c* označava *character device*. Ukoliko je prvo slovo *b*, znači da se radi o *block device-u*.

IDE tvrdi diskovi (*block device*) označavaju se redom: */dev/hda*, */dev/hdb*, */dev/hdc*. Ukoliko želite vidjeti (na ne baš najbolji način) što se nalazi na vašem tvrdom disku, upotrijebite naredbu:

```
# less -f /dev/hda
```

Može se probati naći neki zapis na disku koji sadrži npr. riječ `student`. Traženje stringa unutar datoteke se pomoću `less` programa radi tako da se upiše znak / i nakon njega traženi niz znakova (npr. `student`). Sada će `less` prikazati prvo mjesto na tvrdom disku gdje je zapisana riječ `student`. Sa `n` i `N` se traži novo pojavljivanje tražene riječi prema naprijed odnosno prema nazad. Iz `less` programa izlazi se slovom `q`.

Uobičajeno je da na računalu budu prisutne sve `device` datoteke, makar dotični uređaji nisu instalirani. Npr. SCSI tvrdi disk je prikazan datotekom `/dev/sda` i možemo je lako naći u `/dev` direktoriju no to nije i dokaz da se SCSI tvrdi disk uistinu nalazi na našem računalu. Pokušajte pronaći sve `device` datoteke koje predstavljaju disketu jedinicu. To su sve one koje počinju sa `/dev/fd*`. Koliko ih ima? Kako vidite, pokriveni su razne mogućnosti. Tako npr. `/dev/fd0u1440` predstavlja prvu disketu jedinicu `fd0` fizičkih dimenzija 3.5“, *high density* (u), veličine 1440 kB. Da bi se izbjegle prevelike komplikacije za korisnika prilikom odabira tipa diskete, Linux ima posebne `device` datoteke koje automatski detektiraju vrstu diskete u disketnoj jedinici. Naravno, to nije moguće ukoliko disketa prethodno nije formatirana. Njihovi su nazivi, ovisno o rednom broju disketne jedinice, redom: `/dev/fd0`, `/dev/fd1`...

Dvije posebne datoteke su `/dev/zero` i `/dev/null`. Prva je beskonačni izvor nula, tj. može se iz nje čitati koliko se god dugo želi, a njen sadržaj su samo nule. Korisna je za generiranje datoteke određene veličine. Druga datoteka, `/dev/null` je trajno prazna datoteka. Može poslužiti kada se neki stream podataka želi pobrisati (npr. spam mail se šalje u `/dev/null`). Naime, bez obzira što se u nju upiše ona će uvijek ostati prazna. Provjerite to.

- U prilogu je C kod programa `hex` za čitanje hexadecimnalnog sadržaja datoteke. Iskopajte ga. Pored heksadecimalnog sadržaja datoteke, on prikazuje i ASCII znakove (ukoliko se radi o ASCII znakovima, u suprotnom ispisuje točkice). Probajte njime vidjeti sadržaj npr. datoteke `/etc/group` te `/dev/zero`:

```
$ ./hex /etc/group  
$ ./hex /dev/zero
```

2. PARTICIJE

Tvrdi disk može biti podijeljen u nekoliko particija. Svaka se particija korisniku čini kao posebni tvrdi disk. Na taj je način moguće na jednom računalu sa jednim tvrdim diskom imati više operacijskih sustava i više datotečnih sustava tako da se svaki nalazi na svojoj vlastitoj particiji. Maksimalno je moguće imati četiri tzv. primarne particije. Ukoliko nas samo četiri particije ne zadovoljavaju, moguće je jednu primarnu particiju (koja se tada naziva proširenom particijom - *extended partition*) podijeliti u veći broj tzv. logičkih particija.

Prvi sektor tvrdoga diska (512 By) naziva se *master boot record* (MBR) i u njemu je pohranjena informacija o broju i pozicijama primarnih particija. Kako je vaš prvi tvrdi disk podijeljen na particije, možete lako doznati naredbom:

```
# fdisk -l /dev/hda
```

Trebalo bi se dobiti nešto slično ovome:

Naziv datoteke: vjezba8.doc	Zagreb, 15.12.2005.	List 2 od 7
Ovaj dokument predstavlja intelektualno vlasništvo FER, ZESOI, LSS.		

```
Disk /dev/hda: 255 heads, 63 sectors, 4865 cylinders  
Units = cylinders of 930 * 512 bytes
```

Device	Boot	Start	End	Blocks	Id	System
/dev/hda1	*	1	4865	39078181	7	NTFS

Većina PC-a u labosu imaju samo jednu particiju, koja je bootabilna tj. na kojoj je operacijski sustav MS Windows sa NTFS datotečnim sustavom.

Partitioniranje tvrdoga diska obično se radi programom `fdisk`. Svaka particija ima svoju vlastitu *device* datoteku. Ime se daje na osnovu imena *device* datoteke tvrdoga diska na kojem se nalazi. Njemu se jednostavno pridodaje redni broj dotične particije. Dakle, `/dev/hda1` je prva primarna particija prvog IDE tvrdog diska. Logičke particije se označavaju sa brojem 5 pa na više. Tako bi se druga logička particija na trećem tvrdom disku označavala kao `/dev/hdc6`. Treća primarna particija drugog tvrdoga diska označavala bi se sa `/dev/hdb3`.

3. KREIRANJE DATOTEČNOG SUSTAVA

Formatiranje je proces dijeljenja magnetskog medija na koncentrične kružne staze (*tracks*) i dijeljenja staza u sektore. Da ne bi bilo zabune, treba naglasiti da se u MS Windows pod formatiranjem smatra, uz gore navedeno, i kreiranje datotečnog sustava. Ukoliko se želi naglasiti razlika, govorimo o formatiranju niže razine (pravo formatiranje) i formatiranju više razine (kreiranje datotečnog sustava). U Linuxu jednostavno govorimo o formatiranju i kreiranju datotečnog sustava. Tvrdi su diskovi obično već formatirani u tvornici. Nije čak ni preporučljivo formatirati (naravno, govorimo o formatiranju niže razine) tvrdi disk jer mu se time mogu smanjiti performanse.

Kada se na određenoj particiji kreira novi Linux datotečni sustav (`ext2`), stvara se određeni broj *inode*-ova koji adresiraju slobodne blokove u kojima su sami podatci. *Inode* je podatkovna struktura na disku koja u sebi sadrži informacije o određenoj datoteci. O broju *inode*-ova koji je na početku stvoren ovisi maksimalni mogući broj datoteka na našoj particiji. Tipično, imamo jedan *inode* za svaki blok (2 do 8 kB). Svakom je *inode*-u pridružen njegov jedinstveni broj. *Inode* u sebi sadrži: ID korisnika i grupe koji su vlasnici datoteke, tip datoteke, dopuštenja, vrijeme nastanka, pristupa i izmjene, broj linkova na datoteku, veličinu datoteke, adresu na disku gdje se nalazi sadržaj datoteke. Svi ti podatci se vide kada se sadržaj direktorija ispisuje sa `ls -l`.

Prva linija `ls -l` daje broj u kB koliko zauzimaju datoteke u trenutnom direktoriju. Uočite (kreirajte datoteku) da datoteka malog sadržaja (par byteova) mora zauzeti cijeli blok na disku. To je tipično 4kB.

inode sadrži sve podatke o datoteci osim njezinog imena i direktorija u kojem se nalazi. Direktorij je, pak, datoteka koja sadrži spisak drugih datoteka koje sadrži. U njemu nije ništa drugo nego lista s imenima datoteka i pripadajućih *inode*-ova.

Popis *inode*-ova u vašem trenutnom direktoriju možete dobiti naredbom:

```
$ ls -i -l
```

Ako vas zanima koliki je broj *inode*-ova unutar particije iskorišten, možete to dozнати naredbom:

```
$ df -i
```

Naredbom `df` pogledajte koliki je postotak prostora slobodan i da li postoji opasnost da nestane *inode*-ova prije nego li se iskoristi svo slobodno mjesto.

Pošto nemamo pristup disku mi ćemo napraviti datotečni sustav nad ramdiskom. Linux omogućava 16 ramdiskova (dio memorije), svaki je veličine 4MB. Datoteke su `/dev/ram1` do `/dev/ram16`.

Datotečni sustav se stvara, odnosno inicijalizira naredbom `mkfs`. `mkfs` je zapravo samo sučelje prema sličnom programu koji odgovara dotičnom tipu datotečnog sustava. Inicijalizirajmo datotečni sustav našeg ramdiska (koristimo `ram1` a ne prvi `ram0`).

```
# mkfs -t ext2 /dev/ram1
```

Evo rezultata izvođenja `mkfs` naredbe:

```
mke2fs 1.37 (21-Mar-2005)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
1024 inodes, 4096 blocks
204 blocks (4.98%) reserved for the super user
First data block=1
1 block group
8192 blocks per group, 8192 fragments per group
1024 inodes per group

Writing inode tables: done
Writing superblocks and filesystem accounting information: done
```

Koliko je *inode*-ova kreirano? Koliko je velik blok? Koliko je to ukupno prostora?

Informacije o cijelom datotečnom sustavu upisane su u tzv. *superblock* koji se prvi čita prilikom mounta.

Prekidač `-t` definira željeni tip datotečnog sustava. Općenito su podržani neki od slijedećih tipova datotečnih sustava:

```
minix    a local filesystem, supporting filenames of length
          14 or 30 characters.
ext2     a local filesystem with longer filenames, larger
          inodes, and lots of other features.
xiafs    a local filesystem with longer filenames, larger
          inodes, and lots of other features.
msdos    a local filesystem for MS-DOS partitions.
iso9660  a local filesystem used for CD-ROM drives.
nfs      a filesystem for mounting partitions from remote
          systems.
swap     a disk partition to be used for swapping.
```

Mogući je da su i drugi tipovi datotečnih sustava. Provjerite sadržaj datoteke `/proc/filesystems`.

Tek je sada naš ramdisk spremam za korištenje. Možemo ga *mountati* naredbom:

```
# mount /dev/ram1 /mnt/mojram
```

Naravno, direktorij `/mnt/mojram` potrebno je bilo prethodno kreirati. Probajte kreirati neki file u `mojram` direktoriju. Što piše pod total kada se napiše `ls -l`. Uočite kako je 1 blok velik 1KB tako da i datoteka od par byteova toliko zazuima.

Ako želimo napraviti datoteku točno određene veličine koristi nam `/dev/zero`. `dd` je naredba s kojom se sirovo kopira sadržaj datoteke (if = input file, of=output file).

```
# dd if=/dev/zero of=nova_datoteka bs=1025 count=1
```

Uočite (sa `ls -l`) kako sada ova datoteka zauzime 2 bloka tj. 2048 bytea na ramdisku.

Unmount radimo naredbom:

```
# umount /dev/ram1  
ili sa  
# umount /mnt/mojram
```

4. LINKOVI

Evo još jedne posebne vrste dototeka, pored već gore spomenutih *device* datoteka. To su linkovi. Linkovi omogućavaju da više različitih imena datoteka pokazuju na jednu te istu datoteku u datotečnom sustavu. Postoje dvije vrste linkova: *hard link* i simbolički tj. *soft link*. *Hard link* pridružuje dvama ili više datotekama jedan te isti *inode*. Kreirajte u vašem direktoriju tekstualnu datoteku `moja_datoteka`. Ovako ćemo napraviti *hard link* na nju:

```
$ ln moja_datoteka hlink
```

Probajte pogledati sadržaj datoteke `hlink`. Vidjet ćete da je identičan datoteci `moja_datoteka`. Ukoliko bi se izbrisala datoteka `moja_datoteka`, `hlink` datoteka bi i dalje normalno postojala jer je vezana isključivo na *inode*. Ako se napravi `touch` (obnavlja vrijeme zadnje modifikacije datoteke) nad jednom datotekom, promjena će biti vidljiva i na drugoj.

```
$ touch moja_datoteka
```

Treba naglasiti da na jednoj particiji ne možemo napraviti *hard link* na datoteku na drugoj particiji, pošto su *inode*-ovi ograničeni na particiju na kojoj se nalaze.

Simbolički link ne pokazuje na *inode* već na put gdje se datoteka nalazi. Samim time možemo zaključiti da ovaj link nije ovisan o particiji na kojoj se nalazi. Kreirajte simbolički link na `moja_datoteka`.

```
$ ln -s moja_datoteka slink
```

Uočite da naredba `ls -l` simbolički link označava prvim slovom `l` u retku.

5. TIPIČNI RASPORED DIREKTORIJA KOD LINUXA

FSSTND ili Linux filesystem standard je standard koji preporučuje kako bi trebalo izgledati tipično datotečno stablo Linux sustava. Poštivanje ovoga standarda olakšava pisanje ili prenošenje softwarea za Linux. Isto tako, i administriranje Linux strojeva znatno je olakšano. Evo nekih najvažnijih direktorija:

/bin - naredbe koje trebaju prilikom podizanja sustava, ali koje su korištene i od običnih korisnika

/sbin - slično kao i **/bin**

/etc - (*everything to configure*) konfiguracijske datoteke specifične za dotično računalo. Kako biste doznali koje datotečne sustave određeno računalo automatski *mounta* prilikom podizanja, proučite sadržaj **/etc/fstab** datoteke.

/dev - device datoteke

/tmp - privremene datoteke

/usr - sadrži sve naredbe, biblioteke, man stranice. Uglavnom, sve datoteke koje se ne mijenjaju. Pošto se u ovom direktoriju ne nalaze datoteke specifične za dotično računalo moguće je ovaj direktorij dijeliti preko mreže s drugim računalima i time znatno uštedjeti na zauzeću ostalih tvrdih diskova.

/var - sadrži podatke koji se mijenjaju za vrijeme rada sistema. Razne poruke o radu sustava (program *syslogd*) pohranjene su u **/var/log/messages** datoteci. Pošto je ona obično jako velika, često je korisno vidjeti samo njezin kraj:

```
# tail /var/log/messages
```

/proc - na postoji na tvrdom disku. Nalazi se u radnoj memoriji. Donosi informacije o procesima. Uočite da svaki proces (označen sa brojem *pid*) ima svoj vlastiti poddirektorij. Verziju kernela koju trenutno koristimo možemo direktno doznati (pored korištenja naredbe *uname -a*) iz datoteke **/proc/version**.

```
$ cat /proc/version
```

Koliko dugo naš sistem radi piše u datoteci **/proc/uptime**.

```
$ cat /proc/uptime
```

Prvi broj je broj sekundi od kada je OS bootan a drugi broj je broj sekundi koji je proveo besposlen. Tu informaciju koristi aplikacija *uptime* koja to ljepešće ispisuje.

```
$ uptime
```

PRILOG A: PROGRAM ZA PRIKAZ HEKSADECIMALNOG SADRŽAJA DATOTEKE

hex.c

```
#include <stdio.h>

int main(int argc, char *argv[])
{
    int j,rd;
    long i;
    FILE *file;
    unsigned char buf[16];

    if (argc==1)
        file = stdin;
    else
    {
        file = fopen(argv[1],"rb");
        if (file==NULL)
        {
            fprintf(stderr,"No such file or access denied.\n");
            return 1;
        }
    }
    i = 0l;
    do
    {
        rd = fread(buf,1,16,file);
        printf("%08X ",i);
        i += 16;
        for (j=0;j<rd;j++)
        {
            printf("%02x ",(unsigned char)*(buf+j));
            if (j==7)
                putchar(' ');
        }
        putchar(' ');
        for (j=0;j<16-rd;j++)
            printf("   ");
        for (j=0;j<rd;j++)
            if ((unsigned)*(buf+j)<127 && *(buf+j)>31)
                putchar(*(buf+j));
            else
                putchar('.');
        printf("\n");
    } while (rd==16);
    fclose(file);
    return 0;
}
```