Operativni sistemi I

Vežbe 3

1. KONTROLA PRISTUPA NA NIVOU FAJL SISTEMA

Pristup resursima pod mrežnim operativnim sistemima (kao što je i Linux) je strogo kontrolisan. Fajl sistem je fundamentalni resurs svake radne stanice ili servera, a kontrola pristupa fajlovima i direktorijumima (dodela ovlašćenja za pristup i zaštita od neovlašćenog pristupa) ključna komponenta ozbiljnih zaštitnih polisa u svakom višekorisničkom sistemu.

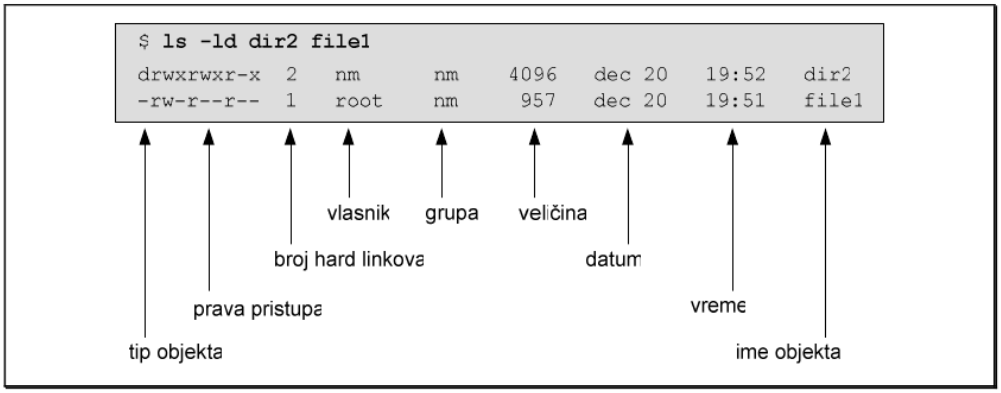
1.1 Vlasnički odnosi i prava pristupa

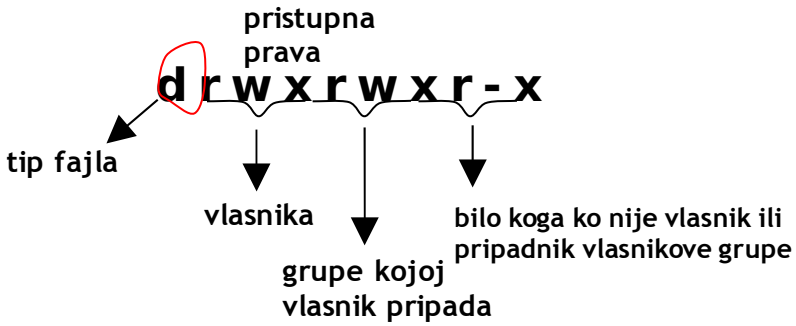
Jedna od najznačajnijih komponenti svake ozbiljne zaštitne politike je kontrola pristupa na nivou fajl sistema. Kontrolom pristupa na nivou fajl sistema određuju se:

- **skup korisnika** koji mogu pristupiti objektima, odnosno fajlovima i direktorijumima,
- nivo pristupa, odnosno **skup akcija** koje autorizovani korisnici mogu izvršiti nad tim objektima.

Kontrola pristupa na nivou fajl sistema zasniva se na vlasničkim odnosima, odnosno vlasništvu nad objektima (pripadnost objekta korisnicima i grupama) i pravima pristupa. Prava pristupa se dodeljuju svakom fajlu i direktorijumu.

Prava pristupa za datoteke i direktorijume najlakše se mogu odrediti pomću komande **ls (list)** sa parametrom **-l (long)**, kao što je prikazano u sledećem primeru:





1. **Tip fajla**

   Prvi karakter ukazuje na tip datoteke:

   - **– (dash)** - je reč o običnoj, regularnoj datoteci
   - **d** - reč je o direktorijumu
   - **b** - blok uređaj - block special file (npr. /dev/sda)
   - **c** - karakter uređaj - character special file (npr. /dev/tty1)
   - **l** - simbolički link
   - **p** - imenovani pipe
   - **s** - socket.

2. **Prava pristupa**

   Sledećih devet znakova predstavljaju prava pristupa objektu za tri vlasničke kategorije, a to su vlasnik, grupa i ostali. Prva tri karaktera definišu prava pristupa vlasnika, druga tri prava pristupa grupe kojoj fajl pripada i poslednja tri karaktera prava pristupa za ostale:

   - **Vlasnik (owner)** najčešće je korisnik koji je kreirao objekat, osim ukoliko superuser (root) ne promeni vlasništvo. U tom slučaju, vlasnik je korisnik kome je vlasništvo dodeljeno. Vlasnik objekta može biti bilo koji korisnik sistema, regularan ili sistemski.
   - **Grupa (group)** je korisnička grupa kojoj je fajl formalno priključen. Za razliku od korisnika koji mogu pripadati većem broju grupa, **objekti fajl sistema mogu pripadati samo jednoj grupi**, koja može biti regularna ili sistemska. Najčešće je to primarna grupa korisnika koji je objekat kreirao. Superuser naknadno može promeniti pripadnost objekta grupi.
   - **Ostali (others, public)** su svi korisnici koji nisu ni vlasnik objekta, niti pripadaju grupi kojoj objekat pripada. Prava pristupa za svaku vlasničku kategoriju eksplicitno se dodeljuju svakom objektu prilikom kreiranja, a kasnije se mogu promeniti.

   Pravo pristupa za svaku grupu se zadaje na isti način, sa istim rasporedom karaktera **rwx**:

   - **pravo čitanja (r - read),**
   - **pravo upisa (w - write),**
   - **pravo izvršavanja (x - execute)**.

   Ukoliko se na odgovarajućoj poziciji nalazi **crtica -, pravo je ukinuto**.

   Primeri:

```
$ ls -la ~/.bash_profile
-rw-r--r--      1      nm          nm       509        Mar  10        17:21
.bash_profile
$ ls -ld /bin /root
drwxr-xr-x    2    root    root  2048    Apr 1      20:16     /bin
drwxr-x---    7    root    root  1024    Apr 20     15:43     /root
```

Na primer, Sistemski direktorijum **/bin** sadrži najčešće korišćene UNIX komande. Svim korisnicima sistema dato je pravo korišćenja direktorijuma /bin. Svi korisnici sistema mogu da se

pozicioniraju na direktorijum, mogu da pročitaju sadržaj i pokrenu komande koje se u njemu nalaze. Pravo upisa dato je jedino superuser-u.

Drugo, sistemski direktorijum /root je home direktorijum superusera, koji nad njim ima sva prava. Članovi grupe root mogu da pročitaju sadržaj direktorijuma i da se pozicioniraju na njega, dok je ostalim korisnicima pristup direktorijumu zabranjen.

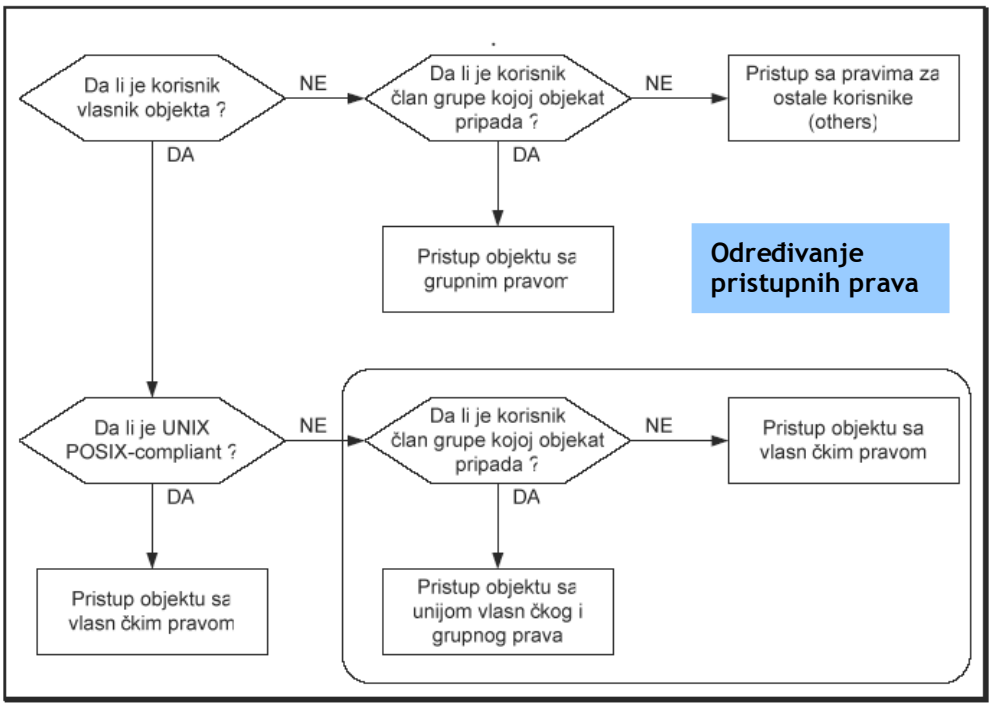
Značenje prava za fajlove i direktorijume bitno se razlikuje, što je prikazano u sledećoj tabeli:

	pristupna prava za fajlove	**pristupna prava za direktorijume**
read (r)	korisnik može pročitati sadržaj fajla, odnosno može prikazivati fajl na ekranu, štampati ga ili kopirati;	korisnik može pročitati sadržaj direktorijuma, što znači i da korisnik može da izvrši komandu ls. **Napomena:** za prikazivanje detaljnog listinga direktorijuma (ls -l) neophodno je i x pravo nad direktorijumom
write (w)	korisnik može modifikovati sadržaj fajla. **Napomena:** može obrisati fajl samo ako mu je dato pravo upisa nad roditeljskim direktorijumom;	korisnik može modifikovati sadržaj direktorijuma, odnosno dodavati nove fajlove i brisati postojeće, kreirati i brisati poddirektorijume. **Napomena:** može obrisati direktorijum samo ako mu je dato pravo upisa nad roditeljskim direktorijumom;
execute (x)	korisnik može izvršavati fajl, pod uslovom da se radi o shell programu ili o fajlu u binarnom izvršnom formatu;	korisnik se može pozicionirati na direktorijum (komandom cd), može prikazivati dugački listing (ls -l) sadržaja i pretraživati direktorijum (find).

Svim fajlovima i direktorijumima dodeljen je korisnički identifikator (UID) i grupni identifikator (GID) vlasnika. Kernel razrešava vlasničke odnose na osnovu ovih identifikatora.

```
$ ls -ln
-rw-rw-r--        1    859    861       20     dec 23     14:04    kyuss
-rw-rw-r--        1    859    861       20     dec 23     15:20    stoner
$ id
uid=859(nm) gid=861(nm) groups=861(nm),0(root)
```

Napomena: Opcija **-n** komande **ls** daje numeričke vrednosti za UID i GID.



1.2 Promena pristupnih prava

Prava pristupa mogu promeniti isključivo vlasnici fajlova i direktorijuma, dok root kao superuser može da promeni pristupna prava svakom objektu. Komanda **chmod** može se pokrenuti u **simboličkom (relative)** ili **oktalnom (absolute)** režimu.

1.2.1 Simbolički režim

Korisnik dodeljuje ili oduzima prava u odnosu na postojeća, dok se postojeća prava koja nisu specificirana argumentom komande ne menjaju. Format komande u simboličkom modu je:

```
$ chmod [-R] symbolic_mode[,...] objectname
```

Primer:

```
$ chmod u=rwx myscript
```

symbolic_mode sastoji se od tri komponente:

- **vlasnička kategorija** na koju se komanda odnosi: vlasnik (u), grupa (g), others (o), sve kategorije (a);
- **operator**: dodela prava (+), ukidanje prava (-), dodela tačno određenih prava (=);
- **prava pristupa** koja se dodeljuju ili oduzimaju: r, w i/ili x.

Primeri:

```
$ touch myfile
$ ls -l myfile
-rw-r--r--        1    nm      nm      0     dec 23     15:25     myfile

$ chmod go+w myfile
$ ls -l myfile
-rw-rw-rw-        1    nm      nm      0     dec 23     15:25     myfile
```

dodata prava upisa kategorijama group i others

```
$ chmod u-w myfile
$ ls -l myfile
-r--rw-rw-        1    nm      nm      0     dec 23     15:25     myfile
```

oduzeto pravo upisa vlasniku

```
$ chmod u=rw,go-w myfile
$ ls -l myfile
-rw-r--r--        1    nm      nm      0     dec 23     15:25     myfile
```

dodeljen skup prava rw vlasniku i ukinuto pravo upisa kategorijama group i others

```
$ chmod a= myfile
$ ls -l myfile
----------        1    nm      nm      0     dec 23     15:25     myfile
```

svima ukinuta sva prava.

Parametar -R se koristi za rekurzivnu promenu pristupnih prava direktorijuma i svih objekata (poddirektorijuma i fajlova) koji se u njemu nalaze. Ukoliko se navede parametar -R, argument *objectname* mora biti direktorijum.

Primer:

```
$ ls -ld parent_dir
drwxr-xr-x        2    nm      nm     4096       Apr 28    09:10     parent_dir
$ ls -l parent_dir
parent_dir:
total 0
-rw-r--r--        1    nm      nm        0    Apr  28    09:09     dir1
-rw-r--r--        1    nm      nm        0    Apr  28    09:10     dir2
-rw-r--r--        1    nm      nm        0    Apr  28    09:09     file1
-rw-r--r--        1    nm      nm        0    Apr  28    09:09     file2
$ chmod -R o-rx parent_dir
$ ls -ld parent_dir
drwxr-xr-x        2    nm      nm     4096       Apr 28    09:10     parent_dir
$ ls -l parent_dir
parent_dir:
total 0
-rw-r--r-- 1 nm  nm        0     Apr  28     09:09 dir1
-rw-r--r-- 1 nm  nm        0     Apr  28     09:10 dir2
-rw-r--r-- 1 nm  nm        0     Apr  28     09:09 file1
-rw-r--r-- 1 nm  nm        0     Apr  28     09:09 file2
```

1.2.2 Oktalni režim

Komandom chmod u oktalnom režimu dodeljuju se prava pristupa svim vlasničkim kategorijama istovremeno. Prava koja korisnik navede kao argument komande eksplicitno zamenjuju postojeća prava (prethodna prava se ne prolongiraju), tako da se ovaj režim naziva apsolutnim. Komanda zatheva da se u ovom režimu kao argument navedu tri oktalne cifre od kojih svaka predstavlja prava pristupa za jednu vlasničku kategoriju.

Moguće oktalne vrednosti sa odgovarajućim pravima opisane su sledećom tabelom:

Oktalna vrednost	Suma prava po binarnoj vrednosti	Odgovarajuća prava	Definicija
7	4 + 2 + 1	r w x	čitanje, izmena i izvršavanje
6	4 + 2 + 0	r w -	čitanje i izmena
5	4 + 0 + 2	r - x	čitanje i izvršavanje
4	4 + 0 + 0	r - -	samo čitanje
3	0 + 2 + 1	- w x	izmena i izvršavanje
2	0 + 2 + 0	- w -	samo izmena
1	0 + 0 + 2	- - x	samo izvršavanje
0	0 + 0 + 0	- - -	bez prava pristupa

Sintaksa komande chmod u oktalnom režimu je slična sintaksi komande u simboličkom režimu:

```
$ chmod [-R] absolute_mode objectname
```

Apsolutna prava formiraju se pomoću tri oktalne cifre kojima su predstavljena prava pristupa za vlasnika, grupu i ostatak sveta. Parametar -R se, kao i u simboličkom režimu, koristi za rekurzivnu promenu pristupnih prava direktorijuma i svih objekata koji se u njemu nalaze. U tom slučaju argument objectname mora biti direktorijum.

Napomena: Kada se koristi oktalni režim **moraju se navesti sve tri oktalne cifre** u tačnom redosledu (*vlasničko pravo -grupno pravo - pravo za ostatak sveta*).

```
$ ls -l betatest
-rw-rw-rw-   1   nm    nm 0 dec 23 15:25 betatest
$ chmod 555 betatest
$ ls -l betatest
-r-xr-xr-x   1   nm    nm 0 dec 23 15:25 betatest
$ ls -l denywrites
-rwxrwxrwx   1   nm    nm 0 dec 23 15:25 denywrites
$ chmod 755 denywrites
$ ls -l denywrites
-rwxr-xr-x   1   nm    nm 0 dec 23 15:25 denywrites
```

1.3 Promena vlasničkih odnosa

UNIX postavlja inicijalne vlasničke odnose prilikom kreiranja fajla ili direktorijuma. Korisnik koji kreira objekat postaje njegov vlasnik, a objekat se formalno pridružuje primarnoj grupi vlasnika.

1.3.1 Promena vlasnika

Komandom chown (change owner) root kao superuser može da promeni vlasnika objekta, a ukoliko konkretan sistem to dozvoljava, to može učiniti i vlasnik. Regularni korisnici Linux sistema mogu promeniti vlasničke odnose samo ako na sistemu nije aktiviran mehanizam disk kvote (*disk quota*), kojim se korisnicima ograničava iskorišćenje prostora na diskovima. Kada se za fajl promeni vlasništvo, prava pristupa starog vlasnika određena su kategorijama *group* i *others*. Sledeće komande prikazuju sintaksu za promenu vlasništva:

```
# chown [-R] new_owner objectname
$ whoami
nm
$ ls -l myfile
-rw-r--r--        1    nm           nm     0   Apr 28       12:07     myfile
$ chown jsmith myfile
chown: changing ownership of `myfile': Operation not permitted
$ su
Password: ******
# chown jsmith myfile
# exit
exit
$ ls -l myfile
-rw-r--r--        1    jsmith       nm     0   Apr 28       12:07     myfile
```

1.3.2 Sticky bit (t)

Postavljanjem sticky bita za direktorijum uvodi se sledeće ograničenje: **bez obzira na pravo upisa koje korisnik ima nad tim direktorijumom, on u njemu ne može obrisati tuđe datoteke (odnosno datoteke kojima on nije vlasnik)**. Tipičan primer je sistemski direktorijum **/tmp**. Sticky bit se postavlja i ukida komandom chmod tako što se u simboličkom režimu svim vlasničkim kategorijama dodeli pravo t (chmod +t directory), a ukida oduzimanjem prava t.

```
$ ls -l public_dir
-rwxrwxrwx        1    nm      nm    4096      dec 23       15:25   public_dir1
$ chmod +t public_dir1
$ ls -l public_dir1
-rwxrwxrwt        1    nm      nm    4096      dec 23       15:25   public_dir1
$ chmod -t public_dir1
-rwxrwxrwx        1    nm      nm    4096      dec 23       15:25   public_dir1
```

2. SHELL - Rad iz komandne linije

Grafičko radno okruženje opterećuje procesor i povećava rizik u smislu sigurnosti sistema, tako da se, po pravilu, ne instalira na serverima. Tada sistem administratorima na raspolaganju ostaje komandni interpreter (shell) i prateći skup alata za rad sa fajlovima.

2.1 KOMANDNI INTERPRETER (shell)

Shell je interfejs između korisnika i kernela, odnosno jezgra OS-a. Shell prihvata komande koje korisnik zadaje, zatim ih interpretira i potom ih izvršava, pri čemu po potrebi pokreće odgovarajuće programe. Na UNIX sistemima postoji više različitih komandnih interpretera, a korisnici u toku rada po potrebi mogu preći iz jednog u drugi.

Komandni interpreter je proces koji obavlja sledeće funkcije:

- interpretaciju komandne linije,
- pokretanje programa,
- redirekciju ulaza i izlaza,
- povezivanje komandi u pipeline,
- rad sa fajlovima iz komandne linije
- zamenu imena datoteka,
- rukovanje promenljivama i kontrolu okoline (*environment*),
- shell programiranje.

2.1.1 Interpretiranje komandne linije

Kad se korisnik prijavi na sistem u kontekstu tekućeg login procesa izvršava se proces shell, odnosno komandni interpreter. Na ekranu se prikazuje komandni prompt (shell prompt), a to je najčešće znak **$, ukoliko se na sistem prijavi običan korisnik, odnosno #, ukoliko se na sistem prijavi root**. Kada korisnik zada neku komandu (odnosno otkuca neki tekst i pritisne *Enter*), shell to pokušava da interpretira. Tekst unet u shell prompt naziva se komandna linija (command line), čiji je opšti oblik:

```
$ command [opcije] [argumenti]
```

Znak $ je odzivni znak komandnog interpretera (shell prompt). Komanda može biti **interna (ugrađena u shell) ili eksterna (realizovana kao poseban program koji se nalazi u sistemskoj putanji)**. Opcije i argumenti su parametri koje shell prenosi komandi, pri čemu su argumenti najčešće obavezni i predstavljaju ime nekog fajla, direktorijuma, korisnika ili, na primer, identifikator procesa.

Ime komande, opcije i argumenti razdvajaju se razmakom. Shell interpretira razmak kao graničnik i na osnovu toga razdvaja argumente i opcije od imena komande. U jednu komandnu liniju može se uneti **najviše 256 karaktera**. Imena većine UNIX komandi po pravilu se formiraju od malih slova. Više UNIX komandi mogu se navesti u istoj komandnoj liniji ukoliko su razdvojene znakom tačka-zarez.

```
$   cal                 #  samo komanda
```

```
$   df /dev/sda          #  komanda (fd) i argument (/dev/sda)
$   cp 1.txt 2.txt       #  komanda (cp) i dva argumenta (1.txt i 2.txt)
$   date –u              #  komanda (date) i opcija (-u)
$   ls –l /etc           #  komanda (ls), opcija (-l) i argument (/etc)
$   clear ; date         #  dve komande koje se izvršavaju jedna za drugom
```

Opcije su osetljive na velika i mala slova (case-sensitive) i mogu se navesti na dva načina:

-x znak minus (-) praćen jednim slovom,
--option dva znaka minus (--) praćena punim imenom opcije.

2.1.2 Inicijalizacija programa

Nakon interpretacije komandne linije shell inicira izvršenje zadate komande. Ukoliko komanda nije interna (ugrađena u shell, poput komande cd) shell traži izvršni fajl koji odgovara imenu komande u direktorijumima navedenim u sistemskoj putanji (koja se može dobiti komandom **echo $PATH**). Nakon toga shell pokreće program i prosleđuje mu argumente i opcije navedene u komandnoj liniji.

Napomena: Ukoliko se izvršni fajl nalazi u tekućem direktorijumu ili u nekom direktorijumu koji nije u sistemskoj putanji ($PATH), ime komande se mora zadati sa putanjom. Slede i primeri koji ilustruju pokretanje programa koji se nalaze u tekućem direktorijumu i direktorijumu /usr/sbin:

```
$ ./myscript
$ /usr/sbin/useradd
```

2.1.3 Redirekcija ulaza i izlaza

UNIX komande primaju podatke sa **standardnog ulaza (stdin)**, rezultate izvršenja šalju na standardni izlaz (stdout), a poruke o greškama na **standardni uređaj za greške (stderr)**. Većina UNIX komandi koristi tastaturu kao standardni ulaz, a monitor kao standardni izlaz i uređaj za greške.

Ulaz komande preusmerava se pomoću znaka < (manje od) na sledeći način:

```
$ command < inputdevice
```

Primer:

```
$ wc -l < /tmp/jsnmith.dat
```

Za redirekciju izlaza se koristi znak >. Ukoliko se redirekcija vrši u postojeću datoteku, datoteka se briše, a zatim se kreira nova u koju se smešta rezultat izvršenja komande. Za dodavanje izlaza na postojeću datoteku koristi se **znak >>**.

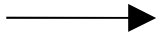
```
$  sort kyuss.txt > /dev/lp0
$  ls -l /home/jsmith > myfile
$  ls -l /tmp/jsmith >> myfile
$  >emptyfile
```

2.1.4 Povezivanje komandi u pipeline

Pipeline funkcioniše na sledeći način: standardni izlaz komande sa leve strane znaka pipe (|) postaje standardni ulaz komande sa desne strane znaka. Znak pipe zahteva komande i sa leve i sa desne strane, a razmaci između znaka i komande su proizvoljni.

Primer:

```
$ ls -l /etc > /tmp/files_in_etc
$ wc -l < /tmp/files_in_etc
145
```



```
$ ls -l /etc | wc -l
145
```

2.1.5 Zamena imena fajlova – JOKER znaci

Džoker karakteri: ***, ? i []**. Argument komande koji sadrži džoker karakter zamenjuje se odgovarajućom listom datoteka shodno pravilima zamene. Komandni interpreter izvršava ovu zamenu pre izvršavanja same komande, odnosno pre pokretanja programa.

```
$ echo *
myfile1 kyuss.txt file3 anotherfile3 file4
```

- **karakter *** menja bilo koji niz znakova proizvoljne dužine
- **karakter ?** menja bilo koji znak (tačno jedan znak)
- **opseg [poc-kraj]** menja tačno jedan znak koji pripada datom opsegu.

Opseg se ne sme zadati u opadajućem redosledu.

```
# ls -d /etc/[a-d][a-d]*
/etc/adduser.conf       /etc/bash.bashrc            /etc/bash_completion.d
/etc/adjtime            /etc/bash_completion         /etc/calendar
```

2.1.6 Rukovanje promenljivama i kontrola okruženja

Da bi komandni interpreter bio fleksibilniji i lakši za korišćenje, u shell je uveden koncept okruženja. Okruženje je skup promenljivih (kao što je, na primer, sistemska putanja) čije vrednosti korisnici mogu menjati i na taj način prilagoditi radno okruženje svojim potrebama. Dodatno, korisnici mogu definisati nove promenljive i brisati postojeće. Jedan od primera je ispis trenutnog PATH-a, spiska lokacija po kojima UNIX traži izvršne fajlove:

```
echo $PATH
```

2.1.7 Shell programiranje

Komandni interpreteri nude specifičan jezik za pisanje shell programa (*shell script*), koji se mogu koristit za automatizovanje raznih administrativnih zadataka.

2.1.8 Korišćenje kontrolinih karaktera

Kontrolni karakteri se zadaju: <Ctrl> + karakter (<Ctrl> se na ekranu prikazuje kao simbol ^ (carret)). Kontrolni karakteri Bourne-again shella koji se najčešće koriste su:

- **<Ctrl-c>** prekida izvršenje procesa koji radi u prvom planu;
- **<Ctrl-d>** označava kraj fajla; napuštanje programa koji podatke čitaju sa standardnog ulaza
- **<Ctrl-u>** briše celu komandnu liniju;
- **<Ctrl-w>** briše zadnju reč u komandnoj liniji;

- **<Ctrl-s>** privremeno zaustavlja izvršenje procesa u prvom planu. Može se koristiti prilikom pregledanja sadržine nekog velikog direktorijuma komandom ls ili ukoliko se neka datoteka prikazuje na ekranu programom cat;
- **<Ctrl-g>** nastavlja se izvršenje procesa u prvom planu.

Primer: bc (basic calculator)

```
$ bc            # pokreće bc
100/5           # inicira operaciju deljenja
20              # program bc prikazuje rezultat prethodne operacije
<Ctrl-d>        # napuštanje programa i povratak u shell
$
```

2.1.9 Alternativno ime komande (alias)

Alias je način dodele kraćeg imena pomoću kog se određena komanda, ili niz komandi, može pozvati iz komandnog interpretera. Na primer, može se dodeliti alias **ll** (long listing) koji izvršava komandu **ls –l**. Alias je aktivan samo u komandnom interpreteru za koji je napravljen. Za korn i bash alias se dodeljuje na sledeći način:

```
$ alias aliasname=value
```

Primeri:

komandi se može dodeliti kraće alternativno ime

```
$ alias h=history
$ alias c=clear
```

jednom komandom se može zameniti sekvenca komandi

```
$ alias home="cd;ls"
```

može se kreirati jednostavno ime za izvršavanje komandi sa određenim parametrima

```
$ alias ls="ls -l"
$ alias copy="cp -i"
```

2.1.10 Ponavljanje komandne linije (history)

Komandni interpreter bash upisuje svaku komandnu liniju u history fajl. Ovo omogućava da se prethodne komande ponove, pri čemu se pre ponovnog izvršavanja mogu i izmeniti. Komande se takođe mogu ponavljati na osnovu rednog broja koji im je pridružen u history datoteci. Bash shell history datoteku smešta u home direktorijum korisnika (~/.bash_history), i u njoj podrazumevano čuva 1000 prethodno izvršenih komandi.

Broj komandi koje se mogu smestiti u ovu datoteku može se promeniti pomoću promenljive HISTSIZE - na primer, ako je HISTSIZE=500, to znači da se u datoteku ~/.bash_history mogu smestiti 500 prethodno izvršenih komandi. Komanda history u bash shellu prikazuje prethodno izvršene komande:

```
$  history 3
   331 finger
   332 mail
   333 history 5
```

2.1.11 Kompletiranje imena fajlova

```
$ ls -l /etc/pas<Tab>
$ ls -l /etc/passwd
```

Ukoliko shell u tekućem direktorijumu pronađe više od jednog fajla čije ime počinje tim karakterima, korisnik će morati da unese još nekoliko karaktera u imenu datoteke, a zatim da ponovo pritisne taster <Tab>. Dodatno, ako korisnik dva puta pritisne <Tab>, shell će prikazati listu fajlova čija imena odgovaraju početku imena koje je korisnik uneo.

2.1.12 Poređenje poznatih komandnih interpretera

Sedamdesetih godina pojavili su se Bourne Shell, Korn Shell i C Shell, na osnovu kojih su kasnije formirane dve klase komandnih interpretera: klasa bazirana na Bourne shell-u i klasa bazirana na C shell-u.

Bourne shell (sh)

Stephen Bourne je razvio Bourne shell za AT&T UNIX okruženje. Bourne shell (sh) se smatra za originalni UNIX komandni interpreter. Postoji na svim UNIX/Linux sistemima, ali se svi noviji komandni interpreteri koriste kao podrazumevani, jer su osetno bolji. Bourne shell je poznat po tome što je uveo mnogo suštinskih ideja, kao što je, na primer, izlazni status izvršenih komandi, koji je praktično omogućio pisanje shell script programa.

C shell (csh)

C shell (csh) je razvijen s ciljem da pruži okruženje za pisanje skriptova i izvršavanje naredbi izvedenih iz sintakse popularnog jezika C. Kod osnovnog C shella ne postoji mogućnost modifikacije komandne linije, ali postoji mogućnost ponavljanja komandi, kao i mogućnost kreiranja aliasa. Većina Linux sistema, nudi poboljšanu varijantu C shella, koja se naziva Enhanced C shell (tcsh) koji omogućava modifikaciju komandi. Pored sličnosti sa C sintaksom, C shell ima ugrađenu aritmetiku i funkciju poređenja, dok interpreteri bazirani na Bourne shell-u u te svrhe moraju pozivati eksterne komande (expr, bc).

Bourne-again shell (bash)

Bourne-again shell (bash) je najčešće korišćeni komandni interpreter pod Linux sistemima i prestavlja poboljšanu verziju Bourne shella, koja pruža mnoge dodatne mogućnosti kao što je ponavljanje i modifikovanje komandi i kompletiranje imena datoteka.

2.2 OSNOVNE KOMANDE ZA RAD SA FAJLOVIMA

2.2.1 Dobijanje pomoći

Navođenje opcije --help u samoj komandi.

Na primer:

```
$ mkdir --help
Usage: mkdir [OPTION] DIRECTORY...
Create the DIRECTORY(ies), if they do not already exist.
   -m, --mode=MODE        set permission mode (as in chmod), not rwxrwxrwx - umask
   -p, --parents           no error if existing, make parent directories
as needed
   -v, --verbose           print a message for each created directory
         --help            display this help and exit
         --version         output version information and exit
Report bugs to <bug-fileutils@gnu.org>.
```

Ispisuje na ekranu sintaksu i objašnjenja za odgovarajuće argumente i opcije, bez detaljnijeg opisa same komande. Ukoliko objašnjenje ne može stati na jedan ekran - pipeline sa komandom less (command **--help | less**).

Man stranice

Jedan od najkompletnijih izvora pomoći (ponekad i jako komplikovan i nejasan) su stranice uputstva za korišćenje komande (manual page, odnosno man page).

```
$ man command
```

2.2.2 Lokatori komandi

whereis

prikazuje lokaciju izvršnih datoteka, izvornog koda i prateće dokumentacije programa

```
$ whereis [-bms] command
```

bez parametara prikazuje lokacije svih elemenata programa

-b izvršne datoteke

-m uputstva

-s izvorni kôd

Primer:

```
$ whereis insmod
insmod: /sbin/insmod /usr/share/man/man8/insmod.8.gz
```

upotreba komande whereis za pronalaženje lokacije programa insmod (koji se koristi za dodavanje modula u aktivno Linux jezgro)

which

prikazuje samo lokaciju izvršnih datoteka; traži izvršnu datoteku u direktorijumima navedenim u sistemskoj putanji i ukoliko je nađe, prikazuje putanju i ime prve pronađene komande

```
$ which [-a] command
```

a primeri komande kada je poziva root korisnik (jer navedene komande nisu dostupne običnom korisniku) su:

```
# which insmod
/sbin/insmod
# which fdisk
/sbin/fdisk
```

which insmod može pozvati samo root korisnik, jer samo on i ima pravo da izvši insmod

apropos

na ekranu prikazuje ime i opis svih komandi koje u opisu imaju zadati string.

```
$ apropos whoami
ldapwhoami           (1)  - LDAP who am i? tool
whoami               (1)  - print effective userid
```

2.2.4 Određivanje tipa fajla

Programi koji rade u UNIX komandnoj liniji ne prepoznaju datoteke na osnovu ekstenzija.

```
$ file kk.c
kk.c: ASCII C program text
```

Na UNIX sistemima postoji nekoliko osnovnih tipova fajlova:

- **tekstualni fajlovi** - ASCII (neformatiran tekst), English text (tekst sa interpunkcijskim karakterima) i izvršni shell programi.
- **izvršni (binarni) fajlovi**
- **fajlovi u koje su smešteni podaci** (na primer, Open Office Writer dokument).

2.2.5 Kopiranje, pomeranje i brisanje fajlova

Komanda	*Opis*
pwd	Print working directory: prikazuje punu putanju trenutnog direktorijuma
ls	List: prikaz sadržaja specificiranog direktorijuma
cd	Change directory: promena tekućeg direktorijuma
mkdir	Make directory: kreiranje specificiranog direktorijuma
rmdir	Remove directory: brisanje direktorijuma
cp	Copy: kopiranje fajla/direktorijuma na specificiranu lokaciju

Primeri:

```
$ cp /home/a.a /tmp/b.b
$ cp a* /tmp
$ cp /etc/[a-d][1-5]* .
$ cp –r /etc /tmp/oldconfig
```

kopiranje direktorijuma /etc sa svim poddirektorijumima i datotekama u direktorijum /tmp/oldconfig/etc (datoteka /etc/passwd kopira se u /tmp/oldconfig/etc/passwd),

```
$ cp –r /etc/* /tmp/oldconfig
```

kopiranje kompletnog sadržaja direktorijuma /etc u direktorijum /tmp/oldconfig (datoteka /etc/passwd kopira se u /tmp/oldconfig/passwd),

```
$ cp –r a* /tmp/mybackup
```

kopiranje datoteka čije ime počinje sa a iz tekućeg direktorijuma i svih poddirektorijuma u direktorijum /tmp/mybackup.

- Vlasnik kopije je korisnik koji je pokrenuo komandu cp,
- Datoteka se dodeljuje primarnoj grupi korisnika koji je pokrenuo komandu cp,
- Pristupna prava kopije se dobijaju se logičkim množenjem bitova pristupnih prava originala i vrednosti promenljive umask. Na primer: ako su pristupna prava originalne datoteke 666, a vrednost promenljive **umask** 002, pristupna prava kopije biće 664, tj, samo za *other* permisije kao

  6_8 *and not* $(2_8) = (0110)_2$ *and not* $(0010)_2 = (0100)_2 = 4_8$
- maska se takođe može setovati komandom **umask** ***maska***
- Sva tri vremena kopije (vreme kreiranja, poslednjeg pristupa i poslednje modifikacije) jednaka su vremenu pokretanja komande **cp**. Vreme poslednjeg pristupa originalne datoteke se takoće menja i jednako je vremenu pokretanja komande cp.

file Identifikacija tipa fajla (binary, text, itd).

cat Concatenate: prikaz fajla

head prikaz početka fajla

tail prikaz kraja fajla

less pretraga kroz fajl od kraja ka početku

more pretraga kroz fajl od početka prema kraju

touch kreira prazan fajl ili modifikuje atribute postojećeg

mv Move: pomeranje fajla na drugu lokaciju ili promena imena

rm Remove: brisanje fajla

wc brojanje reči karaktera i linija

```
$ wc [-cwl] filename
$ wc -l /etc/protocols
```

find Traži fajl ili direktorijum

traži fajlove čiji atributi zadovoljavaju kriterijume pretrage u direktorijumu koji je naveden kao početna tačka pretrage i svim poddirektorijumima, rekurzivno; ukoliko korisnik ne naznači komandi šta da uradi sa datotekama koje pronađe, komanda neće izvršiti nikakvu akciju.

```
$ find / –name urgent.txt –print
$ find /tmp -user jsmith -size +50 - print
$ find /home/jsmith -name "*.old" -print
```

Ostali kriterijumi pretrage su:

- **username uname**
- **groupname gname**
- **atime n** traže se datoteke kojima niko nije pristupio tačno n dana (n mora biti ceo broj, a dozvoljeni su i oblici -n i +n);
- **mtime n** traže se datoteke koje niko nije modifikovao -//-
- **perm mode** prava pristupa zadata u oktalnom obliku
- **links n** traže se sve datoteke sa n hard linkova (n mora biti ceo broj, a dozvoljeni su i -n i +n);
- **type x** traže se sve datoteke koje su tipa x, pri čemu x može biti b (blok uređaj), c (karakter uređaj), d (direktorijum), p (imenovani pipe);
- **inode n** traže se sve datoteke čiji je i-node n;
- **newer fname** traže se sve datoteke koje su modifikovane pre datoteke fname;
- **local** traže se sve datoteke koje se nalaze na lokalnim diskovima.

```
$ find /usr/home -name list.txt -exec rm {} \;
```

2.2.6 Traženje teksta u fajlu

```
grep
$ grep only myfile
$ grep 'w.r' myfile
```

Komanda grep može koristiti tzv. regularne izraze, o kojima će biti reči u posebnom poglavlju jer su veoma značajni prilikom procesiranja teksta.

2.2.7 Linkovi

Na UNIX sistemima postoje dve vrste linkova, i to hard link i simbolički link (symbolic link).

Hard linkovi

Kada korisnik pozove datoteku po imenu (na primer: cat tekst1), UNIX prevodi simboličko ime datoteke koje je naveo korisnik u interno ime, koje koristi operativni sistem. Zbog posebne interne reprezentacije, korisnici mogu datotekama dodeliti veći broj imena. Hard link je jedno od tih imena, odnosno alternativno ime datoteke.

```
$ ln file1 file2
$ ls file*
file1 file2
```

Ukoliko korisnik obriše datoteku file1, **file2 se ne briše**. Osobine:

- link i original imaju isti i-node, tako da se moraju nalaziti na fizički istom sistemu datoteka (hard link se ne sme nalaziti na drugoj particiji ili na drugom disku). Ne mogu se linkovati datoteke sa mrežnog sistema datoteka (NFS);
- ne može se linkovati direktorijum niti nepostojeća datoteka;
- vlasnik, grupa i prava pristupa su isti za link i za original;
- slobodan prostor na disku neznatno se umanjuje (jedna dir-info struktura više za alternativno

ime datoteke);

- broj linkova originalne datoteke uvećava se za jedan nakon linkovanja;
- datoteka sa hard linkovima se ne može obrisati sa diska sve dok se ne uklone svi hard linkovi koji upućuju na tu datoteku.

Simbolički linkovi

Simbolički linkovi se mogu kreirati na dva načina:

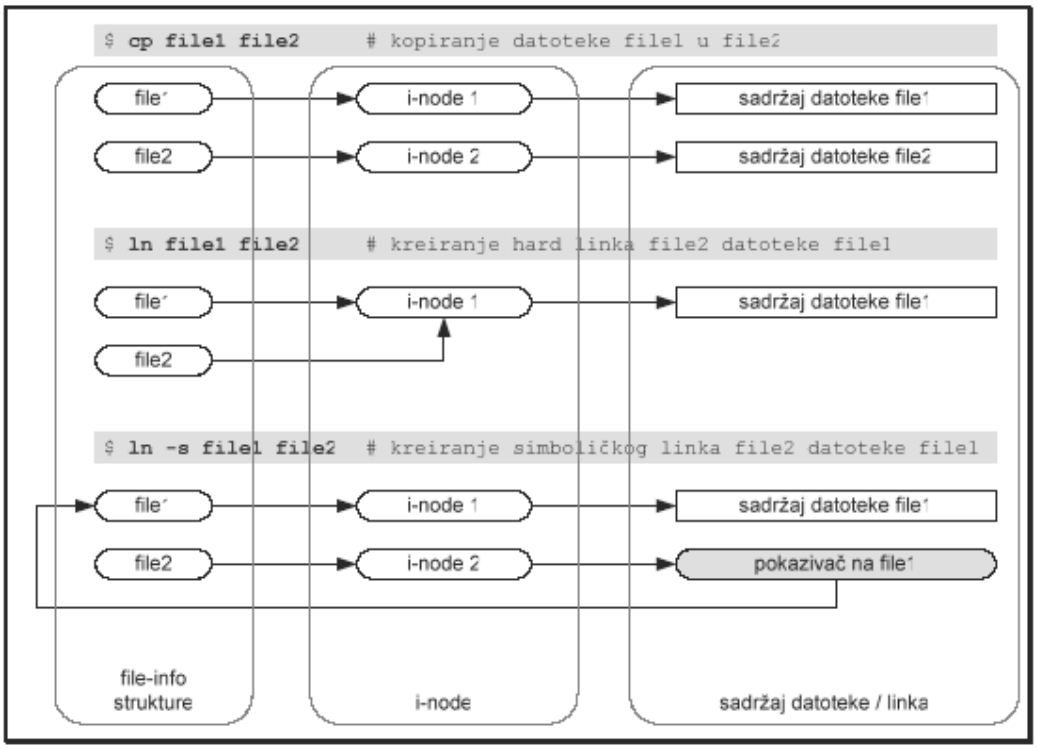
```
$ ln -s original linkname
$ cp -s original linkname
```

Osobine:

- svaki simbolički link koristi poseban i-node i jedan blok podataka u sistemu datoteka; mogu se kreirati nalaziti na fizički istom ili različitom sistemu datoteka, odnosno na istoj ili drugoj particiji (disku). Takođe, mogu se linkovati datoteke sa mrežnog sistema datoteka (NFS);
- može se linkovati direktorijum, kao i nepostojeća datoteka;
- u odnosu na original, link može imati različitog vlasnika, grupu i prava pristupa. Na korisnika koji datoteci ili direktorijumu pristupa putem simboličkog linka primenjuje se unija restrikcija (presek dozvola) linka i datoteke. Na primer, neka je korisnik **user2** vlasnik linka **link1** koji ukazuje na datoteku **file1**, i nek **pripada grupi** kojoj je ta datoteka formalno dodeljena. Ukoliko su pristupna prava za link i datoteku 777 i 640 respektivno, korisnik će imati samo pravo čitanja te datoteke, dakle, za grupu

  (rwx) *and* (r--) = (r--)
- slobodan prostor na disku se umanjuje (za jedan blok podataka). Takođe, simbolički link troši jedan i-node iz i-node tabele;
- broj linkova originalne datoteke se ne uvećava za jedan nakon linkovanja, već ostaje isti kao pre linkovanja;
- s obzirom da simbolički link može ukazivati na nepostojeći objekat, originalna datoteka se može obrisati sa diska bez obzira na broj simboličkih linkova koji upućuju na nju.

```
$ ln -s /etc dir_etc
$ ls -l dir_etc
lrwxrwxrwx 1 root root 4 Sep 5 14:40 dir_etc -> /etc
$ ls -l unexist
ls: unexis: No such file or directory
$ ln -s unexist junk
$ ls -l junk
lrwxrwxrwx 1 root root 15 Sep 5 14:40 junk -> unexist
```


$ cp file1 file2 # kopiranje datoteke file1 u file2
file1
file2
i-node 1
i-node 2
sadržaj datoteke file1
sadržaj datoteke file2
$ ln file1 file2 # kreiranje hard linka file2 datoteke file1
file1
file2
i-node 1
sadržaj datoteke file1
$ ln -s file1 file2 # kreiranje simboličkog linka file2 datoteke file1
file1
file2
i-node 1
i-node 2
sadržaj datoteke file1
pokazivač na file1
file-info strukture
i-node
sadržaj datoteke / linka