Osnove administracije operacijskog sustava 1 (Linux)

Debian

L101

OBRAZOVNI PROGRAMI srca

priručnik za polaznike

srce
Sveučilište u Zagrebu
Sveučilišni računski centar

Ovu su inačicu priručnika izradili:

Autor: mr. sc. Branimir Radić

Recenzent: Darko Culej

Urednik: Dominik Kenđel

Lektor: dr. sc. Jasna Novak Milić

Sveučilište u Zagrebu

Sveučilišni računski centar

Josipa Marohnića 5, 10000 Zagreb

edu@srce.hr

ISBN 978-953-8172-95-3 (meki uvez)

ISBN 978-953-8172-96-0 (PDF)

Verzija priručnika L101-20221114

Sadržaj

Uvod

Trajanje poglavlja:
10 min

Ovaj je tečaj koji polaznike uvodi u korištenje Linuxa. Tečaj služi da bi se stekla osnovna znanja i obrađuje najosnovnije pojmove. Zajedno s tečajem L102 predstavlja osnovu za početak rada na bilo kojem Linux operacijskom sustavu s naglaskom na Debian, konkretno Debian 11 za koji su izrađene vježbe.

Nakon pohađanja tečaja polaznici će znati osnovne principe rada u Linux administraciji, osnovne problematike CLI-a na Linuxu kao i osnove mrežne povezanosti komunikacije i nadzora aktivnosti na poslužiteljima.

Ovaj se tečaj sastoji od jedanaest poglavlja.

Nakon pohađanja ovog tečaja moći ćete:

- pripremiti i provoditi instalaciju distribucije Debian
- upravljati procesima, programima i komponentama operacijskog sustava Debian na osnovnoj razini
- izvoditi odabrane naredbe u naredbenoj liniji i razumjeti njihovu primjenu
- provoditi osnovne radnje s datotekama i direktorijima
- koristiti se naredbeno-linijskim uređivačem teksta vi
- upravljati sklopovljem, diskovima i particijama
- provoditi postupak instalacije dodatnog softvera i rada s paketnim sustavom (DPKG, RPM).

1. Općenito o Linuxu

Trajanje poglavlja:
35 min

Po završetku ovoga poglavlja moći ćete:

- odrediti što je Linux
- prepoznati značajnija događanja iz povijesti Linuxa
- prepoznati elemente filozofije otvorenog koda
- imenovati elemente distribucija Linux
- prepoznati najznačajnije distribucije Debian, RedHat i Slackware i distribucije nastale na temelju njih
- prepoznati vodeće projekte otvorenog kôda iz ovih skupina: uredski alati (OpenOffice, LibreOffice), web-preglednici, web-poslužitelji (Apache HTTP Server, Nginx), baze podataka (PostgreSQL i MySQL), LDAP, poslužitelji elektroničke pošte (Sendmail i Postfix), DNS, DHCP, programski jezici.

U ovoj se cjelini govori o tome što je to Linux i koji su najznačajniji događaji iz njegove povijesti. Upoznajemo se s najznačajnijim Linuxovim distribucijama: Debian, RedHat i Slackware, a spominju se i druge distribucije koje su nastale na temelju njih. Na kraju cjeline nalazi se pregled vodećih projekata otvorenog kôda.

1.1. O Linuxu

1.1.1. Što je Linux?

Linux je ime za jezgru (*kernel*) računalnog operacijskog sustava sličnog *Unix*u, ali najčešće i za cijeli operacijski sustav utemeljen na toj jezgri. *Linux* je dobio ime po svojem izvornom autoru **Linusu Torvaldsu**.

Unix je komercijalno ime za komercijalne operacijske sustave kao što su *IBM AIX, Sun Solaris* ili *HP-UX*. Samo se velike tvrtke smiju koristiti imenom *Unix*. *Linux* je *Unix*ov klon koji poštuje sve standarde koji ga karakteriziraju kao operacijski sustav utemeljen na *Unix*u. Više o tome može se pronaći na ovoj poveznici.

Prije nego što je Linus Torvalds napravio novu jezgru za *Intel*ov mikroprocesor 80386, operacijski sustavi slični *Unix*u u pravilu nisu bili primjenjivi niti korišteni u kućnoj upotrebi nego samo za istraživačke i uredske poslove. Linus Torvalds napravio je svoju jezgru prema uzoru na ***SunOS*** (operacijski sustav utemeljen na *Unix*u tvrtke *Sun Microsystems*) kojim se koristio na fakultetu.

Nakon što se neko vrijeme sam koristio jezgrom, Linus je **1991. godine** objavio izvorni kôd na Internetu te pozvao sve zainteresirane da sudjeluju u njegovu daljnjem razvoju. Mnogi su programeri prihvatili taj poziv, tako da je danas jezgra *Linux* zajedničko djelo programera i hakera diljem svijeta. Važno je naglasiti da se termin haker odnosi na entuzijaste koji odlično poznaju računalne sustave, za razliku od danas češćeg značenja tog termina koji ima negativne konotacije i odnosi se na osobe koje obavljaju kriminalne radnje preko računalnih sustava.

Linux je **slobodan softver**. Za njegov spontani razvoj zaslužni su brzi razvoj globalne komunikacijske mreže i licenca za korištenje GPL.

To je omogućilo stvaranje i rast globalne zajednice suradnika – korisnika i programera, koji su omogućili da *Linux* postane prepoznatljiv.

U prvim godinama ovog tisućljeća započeo je streloviti rast *Linux*a. Među **500 najjačih računala na svijetu** (superračunala) *Linux* je dominantan operacijski sustav. Više o tome može se pročitati na ovoj poveznici. Na ovoj je poveznici vidljivo da je u lipnju 2016. godine na 497 superračunala (od 500 najjačih superračunala) ili 99,44 % bio instaliran operacijski sustav *Linux*.

1.1.2. Kratka povijest Linuxa

Linux je nastao **5. listopada 1991. godine**. Tog je dana izašla njegova prva službena inačica – 0.02. Od tada broj ljudi koji se koristi *Linux*om, bilo kao programeri (razvijatelji jezgre ili aplikacija) bilo kao krajnji korisnici, stalno raste. No sama jezgra ne čini cjelokupan operacijski sustav. Pojedinci s raznih sveučilišta i programeri diljem svijeta spojili su *Linux*ovu jezgru s programima iz projekta GNU i tako dobili funkcionalan operacijski sustav.

U tim ranim danima u *Linux*u se uglavnom radilo u tekstnom sučelju, ali već 1996. utemeljen je **projekt KDE** (skraćenica od eng. *K Desktop Enviroment*) koji je *Linux*u (ali i drugim sustavima sličnim *Unix*u) dao vrhunsko grafičko sučelje. Činjenica da KDE u početku nije bio slobodan softver, potaknula je godinu kasnije razvoj grafičkog sučelja ***GNOME***.

U ranim danima *Linux* je služio kao eksperimentalni sustav kojim su se koristili studenti, hakeri, programeri i općenito ljudi usmjereni na rad s računalima. Nije bilo šire komercijalne upotrebe. To se promijenilo nastankom *web*-poslužitelja ***Apache***, koji je zajedno s *Linux*om pružio pouzdano i besplatno rješenje za pogonjenje velikog broja *web*-stranica. Tako je *Linux* u nekoliko godina istisnuo mnoge druge sustave temeljene na *Unix*u te u velikoj mjeri i *Windows NT* s tržišta poslužitelja.

Napredovanje *Linux*a na stolnim računalima znatno je sporije pa je *Linux* još uvijek rijetka pojava na kućnim i uredskim računalima. S vremenom je nastao velik broj novih programa za *Linux* (i druge *Unix*e – važno je naglasiti da aplikacije napisane za *Linux* mogu raditi na velikom broju drugih *Unix*a i obrnuto) za razne svrhe: uredski paketi, sve vrste programa za internet, PDF, gledanje i uređivanje slika, multimedija, snimanje CD/DVD-a i mnogi specijalizirani programi. Može se reći da danas za *Linux* postoje svi potrebni programi za prosječnog kućnog i uredskog korisnika.

Prednosti su uporabe *Linuxa*:

- **sigurnost** - za sada postoji samo neznatan broj virusa, a alati za uklanjanje špijunskog (*spyware)* i reklamnog (*adware)* softvera koji se mogu pokrenuti na *Linux*u su u začecima (*proof-of-concept*). Osnovni dizajn *Linux*a i pratećeg softvera otežava ozbiljne upade u sustav.

- **stabilnost** - stabilnosti sustava pridonosi modularan dizajn jezgre operacijskog sustava *Linux* koja omogućava da se pojedini dijelovi sustava zaustavljaju i ponovno pokreću prema potrebi, što kod npr. instalacije grafičkog pogonskog programa znači da se računalo ne mora ponovno pokrenuti nego je dovoljno učitati novi modul i ponovno pokrenuti grafički podsustav. Isto vrijedi i za druge pogonske programe.

- **posjedovanje više grafičkih sučelja i mogućnost prilagodbe potrebama korisnika** - *Linux* se s nekim vizualno siromašnijim sučeljem može instalirati i na sporijim računalima, koja bi za operacijski sustav *Windows* bila preslaba.

Prodor *Linux*a na kućna računala i u poslovni svijet usporava činjenica da se igre uglavnom izrađuju za operacijski sustav *Windows*, a nedostaju i mnogi profesionalni programi. Poseban su problem i pogonski programi (*drivers*) koje tvrtke rijetko izdaju ili se korisnici odbijaju njima koristiti, što zbog nesuglasica oko stavova po pitanju slobodnog sofvera, što zbog brzog i pomalo divljeg razvoja jezgre *Linux* koji otežava pisanje pogonskih programa. Zbog toga *Linux*ova zajednica teško surađuje s velikim komercijalnim tvrtkama koje bi mogle pomoći u podršci i njegovu širenju. Konačno, tu je i problem postojanja velikog broja distribucija, to jest specifičnih razlika među njima.

1.1.3. Filozofija slobodnog softvera i otvorenog izvornog koda

Slobodna programska podrška (ili slobodni softver) je softver koji se može rabiti, proučavati i mijenjati bez ograničenja te presnimavati i distribuirati bez ograničenja odnosno uz ograničenje da se daljnjim korisnicima moraju osigurati ista navedena prava, a u nekim slučajevima i da im u tu svrhu proizvođači hardvera moraju dopustiti pristup hardveru i njegovo mijenjanje.

Da bi se softver mogao distribuirati kao slobodan, mora biti dostupan u obliku koji je čovjeku razumljiv (u izvornom kôdu) uz naznaku gore navedenih povlastica. Ta je naznaka ili licenca za **slobodan softver** ili izjava da je izvorni kôd predan u **javno vlasništvo**.

U ranim danima informatike softver se slobodno dijelio i mijenjao među malobrobrojnim korisnicima računala na sveučilištima, istraživačkim laboratorijima, institutima i vladinim organizacijama. U tim ranim danima sav je softver bio slobodan. Tek krajem 70-ih godina 20. stoljeća pojedine su tvrtke (među kojima prednjači *Microsoft*), bojeći se konkurencije, počele zatvarati izvorni kôd i licencirati svoj softver tako da ograničava slobodu korisnika. Taj model je vrlo brzo prihvatila većina informatičke industrije. Nasuprot tom modelu, 80-tih godina 20. stoljeća nastao je pokret koji se zalaže za ponovno uvođenje slobodnog softvera u svakodnevni rad. Taj je pokret utemeljio **Richard Stallman**, iako je slobodni softver (npr. BSD) postojao i prije njega.

Stallmanova definicija slobodnog softvera, koju je objavio *Free Software Foundation* u veljači 1986. godine, određuje da je softver slobodan ako ljudi koji dobiju primjerak tog softvera imaju ove četiri slobode:

- **sloboda 0**: Sloboda pokretanja programa u bilo koje svrhe.
- **sloboda 1**: Sloboda proučavanja rada programa i njegove prilagodbe svojim potrebama (preduvjet za to je pristup izvornom kodu).
- **sloboda 2**: Sloboda distribuiranja presnimaka da bi se pomoglo bližnjemu.
- **sloboda 3**: Sloboda poboljšavanja programa i ustupanja izmijenjenih inačica javnosti za blagodat cijele zajednice (preduvjet za to je pristup izvornom kôdu).

Program je slobodan softver ako njegovi korisnici imaju **sve te slobode**. Prema tome, svatko može slobodno distribuirati presnimke, s preinakama ili bez njih, bez naplate ili s naplatom troškova distribucije, svakome i svugdje. Biti slobodan znači (između ostalog) da se ne mora tražiti dopuštenje niti platiti za softver.

1.1.4. Dodatni sadržaji

Više o *Linuxu*:

http://hr.wikipedia.org/wiki/Linux

http://en.wikipedia.org/wiki/Linux

Zašto otvoreni izvorni kod promašuje smisao slobodnog softvera?

https://www.gnu.org/philosophy/open-source-misses-the-point.hr.html

1.2. Najpopularnije distribucije

1.2.1. Linuxove distribucije

***Linux*ova distribucija** je operacijski sustav sastavljen od:

- *Linux*ove jezgre (s pogonskim programima)
- sistemskih i aplikacijskih programa *GNU*
- grafičkog servera *Xorg*
- grafičkog okruženja.

Osim tih osnovnih dijelova, različite distribucije uključuju veći ili manji broj drugih korisničkih programa specifične namjene. Svaka je distribucija podešena prema željama autora i korisnika za određenu namjenu. Nemoguće je utvrditi točan broj distribucija, a **ne postoji niti jasan kriterij** što čini *Linux*ovu distribuciju. Veliki broj distribucija i nepostojanje standarda (poput jedinstvenog načina instaliranja programa) mnogim korisnicima računala otežavaju prelazak na *Linux* i njegovu komercijalnu upotrebu.

S obzirom na to kako se distribuira softver uz pojedinu distribuciju, one se mogu podijeliti u **tri osnovne skupine**.

Softver se može distribuirati:

- u **izvornom kôdu** (kao kod distribucije *Gentoo*)
- u za to predviđenim **paketima** (koji sadrže izvršne inačice softvera)
- kao **izvršni programi ili skripte** koje same instaliraju softver (također u izvršnom obliku).

Dva su najčešća sustava za upravljanje paketima **RPM** i **DPKG**. Distribucije koje se koriste **RPM**-om često se nazivaju **RPM-distribucije**, a distribucije koje se koriste **DPKG**-om, **distribucije utemeljene na *Debian*u**.

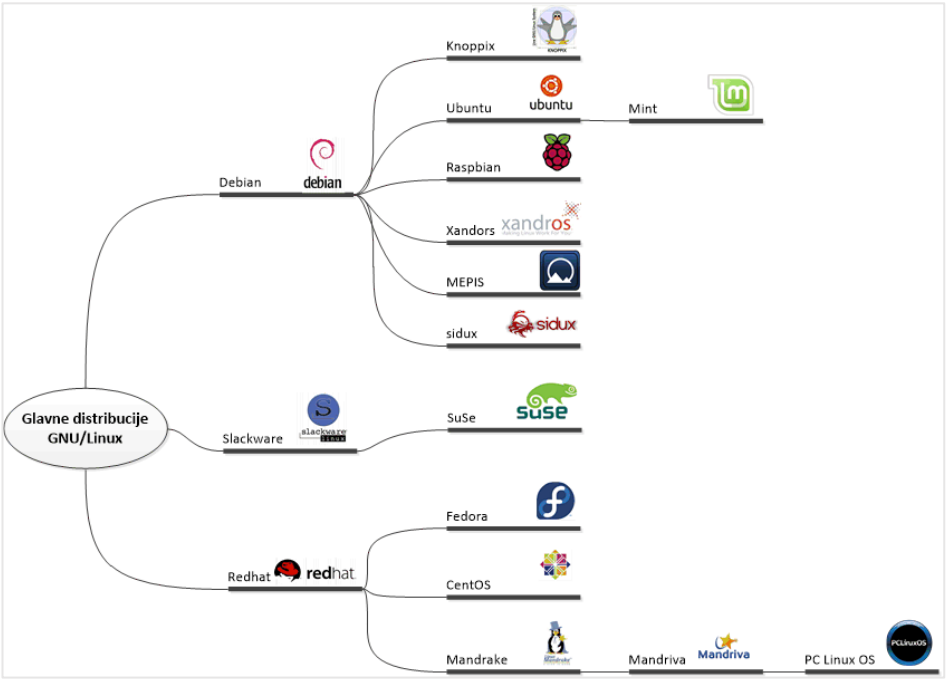
Primjeri su RPM-distribucija *RedHat Enterprise Linux, Fedora, Mandriva, PCLinuxOS, OpenSuse,* a distribucija temeljenih na Debianu (osim samog *Debiana*) *Ubuntu, Xandros, Mepis, Knoppix, Sidux, SteamOS i Raspbian*.

1.2.2. Linuxove distribucije

Naziv distribucije	Logotip	Opis
Slackware		Najstarija aktivna distribucija namijenjena iskusnim korisnicima, s nekim zastarjelim mehanizmima funkcioniranja, ali s ugledom stabilnog, sigurnog i pouzdanog sustava.
Debian		Veliki međunarodni projekt s filozofijom slobodnog softvera u osnovi. Osnova za najveći broj drugih distribucija.
Ubuntu		Distribucija temeljena na *Debian*u iza koje stoji veliki kapital, što je pomoglo naglom širenju popularnosti.
Mint		Po mnogima najjednostavnija distribucija temeljena na *Ubuntu*.
Fedora		Temelji se na bivšoj distribuciji *RedHat* i služi kao osnova tvrtci *RedHat* za izradu komercijalnog *Linux*a (*RedHat Enterprise Linux*).
OpenSUSE		Besplatna inačica komercijalne *Novell*ove distribucije *Suse*. U velikoj je mjeri prilagođena početnicima.
Mandriva		Izvorno nastao kao klon distribucije *RedHat*, razvio se u zasebnu distribuciju i u velikoj je mjeri prilagođen početnicima.
Gentoo		Distribucija namijenjena isključivo onima koji žele u potpunosti ući u svijet *Linux*a. Instalacija zahtijeva kompajliranje cijelog sustava iz izvornog kôda, što može trajati danima.
Knoppix		Također inačica Debiana, poznat kao prvi LiveCD.
Red Hat		Jedna od najstarijih komercijalnih distribucija.

1.2.3. Prikaz grana distribucija Linuxa

Sljedeća slika prikazuje razvoj *Linux*ovih distribucija. Kao što je već spomenuto, dva su najčešća sustava za upravljanje paketima RPM i DPKG. Distribucije koje se koriste RPM-om često se nazivaju RPM-distribucije i utemeljene su na distribuciji *RedHat*, a distribucije koje se koriste DPKG-distribucijama, utemeljene na *Debian*u. Iz *Debian*a je proizašlo mnogo distribucija koje se koriste paketnim sustavom DPKG (npr. *Knoppix*, *Ubuntu* itd.), a iz *RedHat*a su proizašle distribucije koje se koriste paketnim sustavom RPM (npr. *Fedora*, *CentOS*, *Mandrake*).



Potpun prikaz svih *Linux*ovih distribucija nalazi se na ovoj poveznici

1.2.4. Dodatni sadržaj

Poveznice za najraširenije distribucije:

- *Debian GNU/Linux*
- *RedHat*
- *CentOS*
- *Ubuntu*.

1.3. Pregled vodećih projekata otvorenog koda

1.3.1. Uredski alati

OpenOffice.org	***OpenOffice.org*** je skup uredskih alata koji se može koristiti slobodno i besplatno. Mogućnostima je i načinom rada *OpenOffice.org* usporediv s poznatim (i skupim) uredskim paketom *MS Office*. *OpenOffice* može bez poteškoća učitati većinu dokumenata nastalih u uredskom paketu *MS Office* (*Word*, *Excel*, *PowerPoint*), uređivati ih i spremati u formatu *OpenOffice* ili u orginalnom formatu.
LibreOffice	***LibreOffice*** je programski paket namijenjen uredskoj obradi podataka. Razvila ga je Zaklada dokumenata (*The Document Foundation*) kao novi slobodni paket umjesto postojećeg *OpenOffice.org.* Kompatibilan je s mnoštvom drugih programskih paketa poput *MS Office*a, a dostupan je na nekoliko platformi.

Uredski paketi *LibreOffice* i *OpenOffice* sastoje se od ovih modula:

Naziv	Ikona – OpenOffice	Ikona – LibreOffice	Namjena
Writer			obrada teksta i uređivanje HTML-a
Calc			izrada proračunskih tablica
Draw			crtanje vektorske grafike
Impress			izrada prezentacija
Math			izrada i uređivanje matematičkih formula
Base			upravljanje bazom podataka

1.3.2. Web-poslužitelji

	***Apache HTTP Server*** je besplatni *web*-poslužitelj otvorenog kôda za operacijske sustave utemeljene na *Unix*u, *Microsoft Windows*, *Novell NetWare* i druge platforme. *Apache* je najčešće korišteni *web*-poslužitelj na Internetu s udjelom višim od 50%. Više o zastupljenosti *web*-poslužitelja možete pronaći na sljedećoj poveznici.
	***Nginx***, je treći najpopularniji *web*-poslužitelj, iza *Apache*a i *Microsoft*ova *IIS*-a. Kao i *Apache*, otvorenog je kôda. Uz to što je *web*-poslužitelj, može odrađivati zadaće reverznog *proxy*-poslužitelja za mnoge protokole (HTTP, HTTPS, SMTP, POP3 i IMAP). Projekt *Nginx* pokrenut je sa snažnim fokusom na visoku konkurentnost, visoke performanse i malu potrošnju memorije.

1.3.3. Sustavi za upravljanje bazama podataka

	***PostgreSQL*** je robustan, objektno-relacijski sustav za upravljanje bazama podataka otvorenog koda, proizveden na temelju Berkeleyeva sustava za upravljanje bazama podataka *Postgres*. *PostgreSQL* sadrži moćan objektno-relacijski model podataka, bogat izbor vrsta podataka, laku nadogradivost i nadograđeni skup naredbi jezika SQL.
	***MySQL*** je također sustav za upravljanje bazom podataka otvorenog kôda. Uz *PostgreSQL, MySQL* je čest izbor baze za projekte otvorenog kôda, a distribuira se kao sastavni dio poslužiteljskih distribucija, no također postoje inačice i za druge operacijske sustave poput *Mac OS*-a, *Windows* itd.

U svojem se razvoju baza podataka *MySQL* suočila s raznim protivnicima svojeg sustava organiziranja podataka, jer su joj nedostajale neke osnovne funkcije definirane standardom *SQL* (*Structured Query Language*). Naime, baza *MySQL* optimizirana je da bude brza, nauštrb funkcionalnosti. Nasuprot tome, vrlo je stabilna i ima dobro dokumentirane module i ekstenzije te podršku brojnih programskih jezika: *PHP, Java , Perl, Python*.

1.3.4. Poslužitelji elektroničke pošte

U svijetu otvorenog kôda postoji nekoliko raširenih poslužitelja MTA (*Mail Transport Agent*), odnosno SMTP (*Simple Mail Transfer Protocol*). Najčešće se rabe *Sendmail* i *Postfix.*

sendmail.org	***Sendmail*** je praktično najčešći i najrašireniji, a ujedno jedan od najstarijih programa za razmjenu elektroničke pošte. Jedan je od prvih i najpoznatijih projekata otvorenog kôda koji svoje začetke vuče s početka osamdesetih godina prošlog stoljeća. *Sendmail* glasi kao brz, skalabilan i potpun MTA (u smislu podržavanja najvećeg broja mogućnosti i proširenja protokola). Riječ je o jednom od najpotpunijih i vjerojatno najsloženijih MTA-ova na tržištu. Prilično je loše sigurnosne prošlosti te mu je konfiguracijska datoteka nepotrebno nerazumljiva.
POSTFIX	***Postfix*** je također program za razmjenu elektroničke pošte, napisan kao alternativa *Sendmail*u. *Postfix* je prilično sigurna implementacija SMTP-poslužitelja: arhitekturalno je poslužitelj podijeljen na niz minimalnih jednostavnih servisa od kojih svaki obavlja samo svoj posao, a pri tome to radi s minimalno potrebnim dozvolama, prema potrebi čak i ne znajući za ostatak sustava i druge procese. *Postfix* ima laku i jednostavnu konfiguracijsku datoteku koja se sastoji od parametara i njihovih vrijednosti. Pri tome su od nezamjenjive kvalitete popratni alati (*postsuper*, *postqueue*), koji omogućavaju pregled svih parametara, njihovih trenutačnih i standardnih vrijednosti, te iznimno korisna manipulacija svim međuspremnicima i porukama u njima. *Postfix* se jednostavno postavlja i rekonfigurira, jednostavno se koristi i k tome je brz, učinkovit i siguran.

1.3.5. Web-preglednici

	***Google Chrome*** je *web*-preglednik otvorenog kôda koji razvija američka tvrtka *Google*. Stabilna inačica dostupna je za operacijske sustave *Microsoft Windows, Mac OS X* i *Linux*. Preglednik se koristi *Apple*ovim *WebKit layout engine* za prikazivanje *web*-stranica. Beta-inačica aplikacije izdana je 2. rujna 2008. godine, a konačna stabilna inačica je uslijedila 11. prosinca 2008. Naziv dolazi od imena okvira grafičkog korisničkog sučelja (tzv. *chrome*) prisutnog kod *web*-preglednika. U rujnu 2008. *Google* je učinio dostupnim cjeloviti izvorni kôd aplikacije, uključujući i *V8 JavaScript Engine* pod nazivom *Chromium*. To omogućuje drugim programerima da pregledaju sav kôd *Chrome*a te da doprinesu razvoju *Mac OS X*-ovih i *Linux*ovih inačica. Dio kôda koji je razvio *Google* dostupan je pod licencom BSD, što znači da se može inkorporirati u aplikacije otvorenog, ali i zatvorenog koda. *Chromium* sadrži istu funckionalnost kao i *Chrome*, no bez *Google*ova imena, automatskog ažuriranja te s drugačijim logotipom. Od svibnja 2012. *Google Chrome* je *web*-preglednik s najvećim udjelom na tržištu. Više o tome pročitajte na ovoj poveznici.
	***Mozilla Firefox*** je besplatni internetski preglednik otvorenog kôda čiji razvoj koordiniraju *Mozilla Foundation* i *Mozilla Corporation.* Kao stroj za prikaz *web*-stranica *Firefox* se koristi softverom *Gecko* koji implementira većinu trenutačnih *web*-standarda. U siječnju 2013. godine *Firefox* je bio treći najpopularniji preglednik u svijetu s udjelom od oko 20%. Na *Firefox*ovim službenim stranicama nude se instalacijski paketi za operacijske sustave *Android, Linux, Mac OS X* i *Microsoft Windows*, a *Firefox* je također uključen u distribucije mnogih operacijskih sustava temeljenih na *Linux*u i drugim *Unix*ima.

1.3.6. OpenLDAP

	***OpenLDAP*** je slobodna implementacija otvorenog kôda protokola LDAP (*Lightweight Directory Access Protocol*) koju je razvio projekt *OpenLDAP*. *OpenLDAP* je objavljen pod licencom *OpenLDAP Public License*.

LDAP aplikacijski protokol za čitanje i pisanje imenika preko mreže. Imenik je u LDAP-u datoteka ili skupina podataka koji su organizirani slično kao telefonski imenik, koji sadrže podatke o korisnicima, datotekama i aplikacijama, kao i njihove sigurnosne postavke. Posljednja inačica LDAP-a je 3. Opisi protokola sadržani su u IETF RFC 4510.

1.3.7. DNS BIND

DNS (*Domain Name System*)

DNS (*Domain Name System*) je hijerarhijsko raspoređeni sustav imenovanja računala, servisa ili bilo kojeg sredstva spojenog na Internet ili na privatnu mrežu. On povezuje različite informacije s domenskim imenima pripisanim svakom od subjekata u domeni. Ponajprije, prevodi lako pamtljiva

domenska imena u numeričke IP-adrese koje su potrebne za lociranje računalnih servisa i uređaja širom svijeta.

Praksa korištenja imena jednostavnija je i lakše pamtljivija od korištenja brojčane adrese domaćina (*host*) poslužitelja na mreži, a datira iz vremena ARPANET-a. Prije no što je DNS izmišljen 1982. godine, svakom računalu na mreži računalo je dodjeljivalo datoteku zvanu **HOSTS.TXT**. Datoteka HOSTS.TXT mapirala je imena u brojčane vrijednosti. Ubrzani rast mreže tražio je središnje održavanje te su ručno izrađene datoteke HOSTS.TXT postale neodržive. Bilo je neophodno uvesti skalabilniji sustav koji automatski rasprostranjuje potrebne informacije.

U 1984. godini studenti Douglas Terry, Mark Painter, David Riggle i Songnian Zhou napisali su prvu implementaciju imeničkog poslužitelja nazvanu *The Berkeley Internet Name Domain* (**BIND**).

BIND se naširoko distribuirao, posebno na sustavima *Unix* i bio je dominantan imenički poslužitelj korišten na Internetu. Alternativni imenički poslužitelji razvijeni su najviše kako bi poboljšali *BIND* koji je bio ranjiv. *BIND*-ova inačica 9 napisana je od početka i ima sigurnost zapisa usporedivu s modernim imeničkim poslužiteljima.

BIND je otvorenog kôda i *de facto* standard za imeničke poslužitelje

1.3.8. ISC DHCP

DHCP (*Dynamic Host Configuration Protocol*)

DHCP (*Dynamic Host Configuration Protocol*) mrežni je protokol koji se rabi za dodjeljivanje IP-adresa i drugih mrežnih postavki kao što su pretpostavljeni *gateway*, *subnet* maska i IP-adrese DNS-poslužitelja. Te postavke dodjeljuje DHCP-poslužitelj. Olakšava konfiguraciju mreže, jer eliminira ručno dodavanje osnovnih postavki za jednu računalnu mrežu. DHCP-poslužitelj osigurava da su dodijeljene IP-adrese ispravne i da u mreži nema sukoba adresa.

Najpoznatija implementacija otvorenog kôda je **ISC DHCP** (*Internet Software Consortium Dynamic Host Configuration Protocol*), objavljena pod licencom ISC. DHCP-poslužitelj može se pokrenuti na *Linux*u, kao i na drugim inačicama *Unix*a (*Solaris, BSD*).

2. Instalacija

Trajanje poglavlja:
100 min

Po završetku ovoga poglavlja moći ćete:

- opisati strukturu datotečnog sustava
- prepoznati osnovne particije ***root***, ***home*** i ***swap***
- prepoznati raspoložive vrste instalacije operacijskog sustava Debian GNU/Linux (mrežna (netinst) i cjelovita instalacija s medija)
- pronaći web-mjesta na kojima se nalaze aktualne instalacijske datoteke za operacijski sustav Debian/GNU Linux
- pripremiti podatke za provedbu instalacije (mrežni parametri, raspored particija)
- provesti mrežnu instalaciju (netinst) operacijskog sustava Debian GNU/Linux.

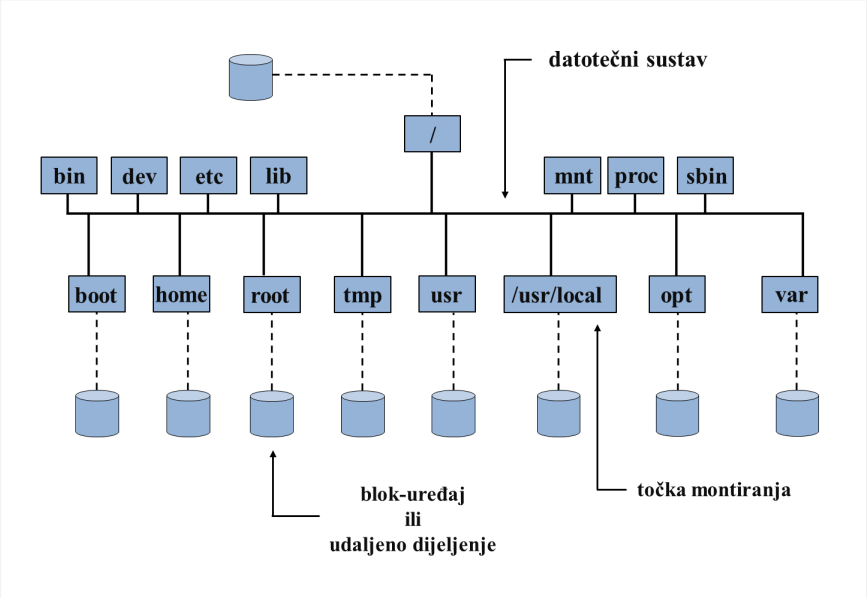
Ova cjelina obrađuje raspoložive vrste instalacije i provedbu instalacije operacijskog sustava Debian. Upoznajemo se sa strukturom Linuxova datotečnog sustava i particijama.

2.1. Primjeri particijskih shema

2.1.1. Struktura datotečnog sustava

Za pristupanje resursima na tvrdom disku operacijski sustav koristi se mehanizmom koji se zove **montiranje** (*mounting*). Za operacijske sustave kao što su *Unix* ili *Linux* to znači da se disk spaja (montira) na direktorij koji se zove **točka montiranja** (*mount point*).

Slika prikazuje strukturu datotečnog sustava na operacijskom sustavu *Linux*. Postoje mnogi resursi (ne moraju nužno biti samo lokalni tvrdi diskovi i particije, mogu biti i CD- ili DVD-mediji, udaljeni dijeljeni disk itd.) koji su spojeni na različite točke montiranja (*mount points)*.



Za korisnika je datotečni sustav jednostavno stablo s direktorijima i poddirektorijima. Korijen tog stabla zove se ***root*** i prikazuje se znakom **/**. To je prvi direktorij na koji operacijski sustav uključuje disk ili resurs, koji se zove ***root device***.

Važno je naglasiti da postoji i direktorij **/root** koji služi za korisničke podatke administratorskog korisnika *root*.

Proces dijeljenja diska na manje dijelove (particije) zove se **particioniranje diska**.

Potrebno je naglasiti da systemctl status httpd.service direktoriji **/bin**, **/dev/**, **/etc**, **/lib**, **/mnt**, **/proc** i **/sbin** moraju biti na datotečnom sustavu ***root*** (odnosno na particiji *root*). Svi drugi direktoriji mogu biti montirani na nekom drugom disku ili na drugoj particiji na istom disku. Npr. particija **/home** služi za korisničke podatke, tamo su smješteni svi korisnički računi koji su otvoreni na računalu. Preporuka je **/home** odvojiti od particije *root*, tako da korisnici ne bi zapunjenjem tog diska doveli u pitanje dostupnost cijelog sustava.

Direktorij	Opis namjene
/	Primarna hijerarhija*, root*-direktorij cijelokupne hijerarhije sustava, početak.
/bin	Izvršne datoteke važnih naredbi na razini tzv. *Single user mod*a, naredbe za sve korisnike (npr. cat, ls, cp).
/dev	Datoteke koje predstavljaju same fizičke ili virtualne uređaje (npr. diskovi, USB i drugi portovi).
/etc	Konfiguracijske datoteke sustava koje vrijede za cijeli sustav (ali ne i za korisničke programe i postavke koje su spremljene u korisničkom direktoriju **/home/ime/**).
/lib	Važne biblioteke za programe iz direktorija **/bin/** i **/sbin/**.
/mnt	Privremeno montirani datotečni sustavi. Nisu nužni za funkcioniranje sustava.
/proc	Virtualni datotečni sustav za prikaz rada kernela i procesa u obliku tekstnih i sličnih datoteka.
/sbin	Važni sistemski programi (npr. *init, route, ifconfig* itd.).

Kada je *root* montiran (priključen), direktoriji i poddirektoriji na tom uređaju (*root device*) mogu se koristiti kao točke montiranja za druge resurse (lokalni ili udaljeni disk, CD, DVD itd.), formirajući tako slijed direktorija uređen kao stablo.

Proces je ovakav:

- program za pokretanje operacijskog sustava *(bootloader)* prilikom pokretanja operacijskog sustava daje jezgri informaciju gdje se nalazi *root device*

drugi su uređaji montirani čitajući instrukcije iz datoteke **/etc/fstab**.

2.1.2. SWAP

Prostor za **SWAP** na *Debian*u i drugim distribucijama *Linux*a je **oblik virtualne memorije**. To znači da ako računalo ostane bez fizičke memorije (RAM), neke će podatke prenijeti iz RAM-a u taj prostor na disku. Particija SWAP je osnovna za procese suspendiranja i hibernacije računala.

Tijekom particioniranja diskova treba donijeti odluku koliko je prostora potrebno za **particiju SWAP**. Za to nema određenih pravila, a veličina prostora za SWAP ovisi o vrsti aplikacija koje se pokreću na računalu.

S novijim inačicama jezgre *Linux*a, ona može upravljati s najviše 32 odvojene particije ili datoteke SWAP u bilo kojem vremenu. Time je omogućeno dodavanje novog prostora za SWAP, prema potrebi.

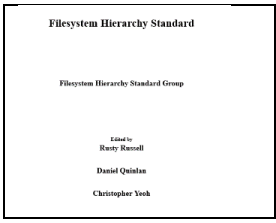
Preporučena vrijednost prostora SWAP tradicionalno je bila dvostruka od količine ugrađene fizičke memorije (RAM). To se s vremenom mijenjalo – rastom fizičke memorije smanjivala se potreba za prostorom za SWAP.

Trenutačne su preporuke:

Količina fizičke memorije	Minimalna preporučena količina prostora za SWAP
4 GB ili manje	2 GB
4 do 16 GB	4 GB
16 do 64 GB	8 GB
64 do 256 GB	16 GB
256 do 512 GB	32 GB

Ako se koristi hibernacija ili suspendiranje računala, tada minimalna potrebna količina prostora za SWAP mora biti veća od količine fizičke memorije u računalu.

2.1.3. Dodatni sadržaj

	Filesystem Hierarchy Standard Dokument koji detaljnije opisuje hijerarhiju datotečnog sustava.

2.2. Instalacija distribucije Debian GNU/Linux

2.2.1. Grafički elementi

Postoje dva načina instalacije distribucije *Debian GNU/Linux*:

- mrežna instalacija (*netinst*)
- cjelovita instalacija s medija.

Tijek je instalacije identičan za oba načina.

Podrazumno grafičko sučelje koje dolazi s distribucijom *Debian GNU/Linux* je *GNOME*. Prilikom instalacije moguće je odabrati i neko od još tri najčešće upotrebljavana grafička sučelja:

- *KDE (K Desktop Environment*)
- *LXDE*
- *Xfce*.

Mrežna instalacija (*netinst*)

Kod ovog načina instalacije na instalacijskom mediju nalaze se samo nužne datoteke za pokretanje instalacijske procedure. Svi se drugi paketi preuzimaju izravno s udaljenog poslužitelja na kojem se nalazi repozitorij *Debian*ovih paketa. Instalacijski medij je malen (oko 200MB) i može se brzo preuzeti na računalo. To je ujedno i najčešći način instalacije operacijskog sustava *Debian GNU/Linux* pa će taj način instalacije biti prikazan u ovom poglavlju.

Cjelovita instalacija s medija

Kod ovog načina instalacije na instalacijskom mediju nalaze se svi programski paketi za instalaciju *Debian*a. Taj je način instalacije pogodan za računala koja nisu mrežno povezana s Internetom ili imaju jako lošu vezu. Instalacijski mediji su veliki (tri DVD-a).

2.2.2. Priprema instalacije

Prije instalacije treba prikupiti podatke o mrežnim parametrima poslužitelja na koji će se instalirati operacijski sustav *Debian GNU/Linux*.

Ako je **konfiguracija mrežnih parametara dinamička** (DHCP), ti će se parametri podesiti automatski. Ako je **konfiguracija statička**, treba prikupiti IP-adresu, mrežnu masku, adresu mrežnog prolaza (*default gateway*) i adrese DNS-poslužitelja. I kod statičke i dinamičke konfiguracije potrebno je pripremiti **ime računala** i **njegovu domenu**.

Ime računala sastoji se od imena računala i poddomene odvojenih točkama. Svaki dio između točki može biti dugačak od 1 do 63 znaka, a sveukupno ime računala može biti do najviše 253 ASCII-znaka. Za ime računala mogu se rabiti slova od 'a' do 'z', brojevi od '0' do '9' te povlaka ('-'). Originalna speficikacija RFC 952 govori da svaka oznaka između točkica ne smije počinjati ili završavati povlakom. Primjer punog imena računala je kosjenka.srce.hr. Ime računala je kosjenka, domena je srce.hr.

Najčešće upotrebljavana arhitektura je **amd64** (64-bitna instalacija operacijskog sustava *Debian GNU/Linux*) pa prilikom odabira slike instalacijskog medija treba odabrati sliku za arhitekturu **amd64**.

Uz arhitekturu amd64 postoji još nekoliko arhitektura na kojima se može instalirati distribucija *Debian GNU/Linux*:

- **i386** za 32-bitna PC računala
- **ia64** za 64-bitna računala temeljena na procesoru *Intel Itanium*
- **armel** i **armhf** za računala temeljena na procesorima ARM
- **powerpc** za računala *Apple Macintosh PowerMac*
- **sparc** za procesore *Sun SPARC*, itd.

Više o arhitekturama možete pronaći na ovoj poveznici.

Važno je napomenuti da za svaku distribuciju treba pripremiti odgovarajući instalacijski medij.

Nakon što preuzmete sliku instalacijskog medija, potrebno ju je snimiti na instalacijske medije, kao što su CD ili USB, ili ju rabiti izravno kroz neku virtualizacijsku platformu (npr. *VMware Player* ili *VirtualBox*).

2.2.2.1. Odabir regionalnih postavki

Nakon pokretanja instalacije najprije treba namjestiti regionalne postavke (odabir kontinenta, države, jezika lokalizacije i rasporeda tipaka na tipkovnici).

Na prvom prikazu koji se pojavi nakon pokretanja instalacije odabere se jezik instalacije, materijali su izrađeni za engleski jezik, stoga odaberite **English**.

Zatim se odabire kontitent, odnosno država. S obzirom na to da Hrvatska (*Croatia*) nije dostupna na početnom popisu, treba odabrati mogućnost **other**.

U sljedećem koracima od ponuđenih se mogućnosti odabere **Europe** pa **Croatia**.

Nakon odabira države treba odrediti lokalizaciju operacijskog sustava. Tijekom ovog tečaja koristit će se engleska lokalizacija operacijskog sustava pa će se kod odabira lokalizacije odabrati mogućnost **United States**.

Međutim, raspored tipki na tipkovnici bit će postavljen na hrvatski jezik (*Croatian*) pa će se u tom koraku odabrati mogućnost **Croatian**.

2.2.2.2. Postavke mreže, određivanje administratorske lozinke i izrada prvog korisničkog računa

U sljedećem koracima instalacije određuju se postavke mreže. Prilikom određivanja postavki mreže treba upisati **ime računala** (*hostname*) i njegovu **domenu** (*domain name*).

Treba odrediti **lozinku administratorskog korisničkog računa** (*root password*). Iz sigurnosnih razloga u postupku instalacije potrebno je dva puta upisati odabranu administratorsku lozinku.

Nakon toga određuju se postavke korisničkog računa koji će se koristiti prilikom izvođenja radnji na operacijskom sustavu koje ne zahtijevaju administratorske ovlasti. Podaci potrebni za izradu prvog korisničkog računa su **puno ime** (*full name*) i **korisnička oznaka** (*username*).

Najčešći oblik korisničke oznake je prvo slovo imena i prezime (npr. iprezime).

Zatim treba upisati i dodatno potvrditi lozinku za novootvorenog korisnika. Ovim je postupkom izrađen prvi korisnički račun.

2.2.2.3. Particioniranje diskova

Prilikom instalacije treba izraditi **particije**. Postoje dvije metode izrade particija, **ručna** i **vođena**. Tijekom ovog poglavlja bit će prikazana **ručna metoda izrade particija** (odgovor *Manual* na pitanje *Partitioning method*). Nakon odabira ručne metode izrade particija treba potvrdno odgovoriti na pitanje vezano uz izradu particijske tablice na disku (*Create new empty partition table on this device?*).

U donjem je primjeru prikazana instalacija distribucije *Debian GNU/Linux* na disk od 32 GB. Pri tome će biti napravljena particija *boot* od 512 MB koju treba podesiti da se s nje može pokrenuti operacijski sustav (*bootable*) i montirati je na *mount point* /**boot**.

Zatim treba napraviti particiju **swap** od 4 GB *(How to use this partition).* Ta particija može biti primarna ili logička. U primjeru instalacije odabrana je primarna particija *(*odgovor *Primary* na pitanje *Type for the new partition*).

Ostatak će diska biti particija **root** montirana na točku montiranja (*mount point) /*. Ta particija će isto tako biti primarna particija (odgovor *Primary* na pitanje *Type for the new partition:*). Particija **root** mora biti primarna i konfigurirana tako da se s nje može pokrenuti operacijski sustav (*bootable*).

Detaljniji opis rasporeda particija i particioniranja diskova biti će opisan u sljedećim poglavljima ovog tečaja.

Nakon uspješnog particioniranja diska promjene treba zapisati na disk (potvrdno odgovoriti na pitanje *Write the changes to disks?*).

2.2.2.4. Odabir dodatnog softvera i prijava na sustav

Nakon završetka particioniranja moguće je **konfigurirati upravitelj paketa**. Prije početka preuzimanja programskih paketa pojavit će se pitanje *Scan another CD or DVD?*.

Ako radite cjelovitu instalaciju s medija, na ovo pitanje možete odgovoriti potvrdno. Računalo će tražiti ubacivanje jednog po jednog instalacijskog medija (CD-a ili DVD-a) i pokupit će popise programskih paketa koji se nalaze na tim medijima.

U primjeru instalacije koji će se prikazati u ovom tečaju na to je pitanje odgovoreno negativno (ne želimo instalirati dodatne pakete).

Kod instalacije dodatnog softvera potrebno je ostaviti odabrano „*Debian desktop environment*“ i „*Standard system utilities*“. Drugo možemo isključiti (npr. „*Print server*“ koji se automatski instalira).

Time smo odabrali **minimalnu grafičku instalaciju** – instalirat će se osnovni operacijski sustav i grafičko sučelje. Ako će se računalo na koje se instalira operacijski sustav *Debian* rabiti kao poslužitelj, možemo odmah uključiti druge potrebne elemente (odaberemo *Web server,* ako želimo da instalacija automatski instalira *web*-poslužitelj, *Print server,* ako želimo imati spojen printer, *SQL database,* ako želimo poslužitelj za upravljanje bazama podataka, *DNS server* za DNS poslužitelj itd.). Svi ti paketi mogu biti instalirani i naknadno.

Nakon instalacije potrebnog softvera na tvrdi će se disk instalirati **GRUB *boot loader***. Da bi se to dogodilo, potrebno je potvrdno odgovoriti na pitanje *Install the GRUB boot loader to the master boot record?*.

GNU GRUB (*Grand Unified Bootloader*) je pokretač operacijskog sustava koji je sposoban pokretati razne besplatne i komercijalne operacijske sustave. *GRUB* može raditi s *Linux*om, *Windows*ima i drugim operacijskim sustavima.

Time je instalacija operacijskog sustava *Debian GNU/Linux* završena i možemo se prijaviti u sustav pomoću prethodno izrađenog korisničkog računa.

2.2.3. Dodatni sadržaj

Debian GNU/Linux: https://www.debian.org/

Općenito o imenima računala: http://en.wikipedia.org/wiki/Hostname

2.3. Vježba 1: Instalacija distribucije operacijskog sustava Debian GNU/Linux

1. Preuzmite sliku instalacijskog medija i snimite ju na CD/DVD ili se njome koristite izravno kroz neku virtualizacijsku platformu (npr. VirtualBox).
2. Pokrenite računalo tako da se ono pokrene s instalacijskog medija (boot s CD-a/DVD-a/USB-a). Pritiskom na tipku [Enter] započinje instalacija.
3. Tijekom instalacije odaberite ove postavke iz tablice.

Select your location	other
Select your location	Europe
Select your location	Croatia
Configure locales	United States
Configure the keyboard	Croatian
Hostname	debian
Domain name	test.lan
Root password	
Re-enter password to verify	
Full name for the new user	Linux User
Username for your account	luser
Choose a password for the new user	
Re-enter password to verify	
Partitioning method	Manual
Partition disks	SCSI3 (0,0,0) (sda)
Create new empty partition table on this device?	Yes
Partition disks	pri/log X GB FREE SPACE
New partition size	512 MB
Mount point	/boot
Bootable flag	on
Partition disks	Done setting up the partition
Partition disks	pri/log X GB FREE SPACE
New partition size	4 GB

Partition disks	Primary (može biti i Logical)
Use as	swap area
Partition disks	Done setting up the partition
Partition disks	pri/log X GB FREE SPACE
Partition disks	Primary (može biti i Logical)
Partition disks	Finish partitioning and write changes to disk.
Write changes to disk?	Yes
Scan another CD or DVD?	No
Software selection	Debian desktop environment Standard system utilities
Install the GRUB boot loader to the master boot record?	Yes

4. Nakon prvog pokretanja operacijskog sustava prijavite se u sustav kao korisnik kojeg ste izradili prilikom instalacije.

Pitanja za ponavljanje

1. Koji načini instalacije Debiana postoje?

   _____

2. Koje sve znakove možemo koristiti u imenu računala (hostname)?

   _____

   _____

3. Naredbena linija

Trajanje poglavlja:
100 min

Po završetku ovoga poglavlja moći ćete:

- prepoznati elemente naredbi za pregled sadržaja dostupne dokumentacije (`man` i `whatis`)
- dati primjere uporabe navedenih naredbi
- objasniti način rada u interaktivnoj ljusci
- objasniti postupak deklaracije i brisanja varijabli u interaktivnoj ljusci
- prikazati popis definiranih varijabli pomoću naredbi `set` i `env`
- objasniti postupak preusmjeravanja standarnog ulaza i izlaza
- opisati postupak ulančavanja procesa i primjene naredbe `tee`
- dati primjere uporabe metaznakova
- opisati primjenu naredbe za ispis pokrenutih naredbi (`history`)
- dati primjer uporabe naredbe `alias` i koristiti se automatskim nadopunjavanjem (*tab completion*)
- opisati primjenu naredbe `exec`.

Ova cjelina obrađuje dostupnu dokumentaciju operacijskog sustava Linux. Upoznajemo se s radom u interaktivnoj ljusci, deklaracijom i upotrebom varijabli ljuske, preusmjeravanjima standardnog ulaza i izlaza te ulančavanjem procesa.

3.1. Dokumentacija

3.1.1. Stranice man

Sustavi *Linux* jako su dobro dokumentirani. Informacije o korištenju određene naredbe ili funkcije mogu se pronaći na tzv. *man*-stranicama. Službeni naziv *man*-stranica je ***Unix Programmers's Manual***, a pružaju informacije o naredbama, sistemskim pozivima, formatima datoteka i održavanju sustava. Standardni su dio svih sustava *Linux* i *Unix*.

Naredba za prikazivanje *man*-stranica je `man`. Argument naredbe `man` je *man*-stranica koja se želi prikazati. U pravilu je to ime naredbe o kojoj se želi saznati više.

U sljedećem primjeru prikazana je *man*-stranica naredbe `mkdir`. Za pristup terminalu potrebno je u grafičkom sučelju iz padajućeg izbornika *Activities* odabrati **Terminal**.

```
$ man mkdir
MKDIR(1) User Commands MKDIR(1)

NAME
 mkdir - make directories

SYNOPSIS
 mkdir [OPTION]... DIRECTORY...

DESCRIPTION
 Create the DIRECTORY(ies), if they do not already exist.

Mandatory arguments to long options are mandatory for short options too.
-m, --mode=MODE
 set file mode (as in chmod), not a=rwx - umask
-p, --parents
 no error if existing, make parent directories as needed
-v, --verbose
 print a message for each created directory
-Z set SELinux security context of each created directory to the default type
--context[=CTX]
 like -Z, or if CTX is specified then set the SELinux or SMACK security
context to CTX
--help display this help and exit
--version
 output version information and exit

AUTHOR
 Written by David MacKenzie.

REPORTING BUGS
 GNU coreutils online help: <http://www.gnu.org/software/coreutils/>;
 Report mkdir translation bugs to <http://translationproject.org/team/>;

COPYRIGHT
 Copyright © 2014 Free Software Foundation, Inc. License GPLv3+: GNU GPL
version 3 or later <http://gnu.org/licenses/gpl.html>;.
This is free software: you are free to change and redistribute it. There is NO
WARRANTY, to the extent permitted by law.

SEE ALSO
 mkdir(2)

Full documentation at: <http://www.gnu.org/software/coreutils/mkdir>;
 or available locally via: info '(coreutils) mkdir invocation'
```

Man-stranice podijeljene su u nekoliko dijelova:

- NAME - naziv naredbe i nazivi sličnih naredbi
- SYNOPSYS - prikazuje sintaksu naredbe i raspoložive opcije i argumente
- DESCRIPTION - pregled djelovanja naredbe
- OPTIONS - raspoložive opcije koje mijenjaju funkciju ili efekt naredbe

- OPERANDS - cilj naredbe na kojemu se naredba izvršava
- FILES - datoteke vezane za tu naredbu (konfiguracijske datoteke i sl.)
- SEE ALSO - upućuje na povezane naredbe i teme.

Nemaju sve *man*-stranice sve navedene dijelove, neke imaju i više, a neke i manje informacija. Međutim, sve *man*-stranice trebaju sadržavati minimalno ove dijelove: *NAME, SYNOPSYS* i *DESCRIPTION*.

Man-stranice su podijeljene u osam sekcija (*sections*). Jedna naredba se može nalaziti u više sekcija, uvijek se prikazuje sadržaj sekcije nižeg broja. Sekcije su ove:

- Sekcija 1 - Informacije o izvršnim datotekama
- Sekcija 2 - Sistemski pozivi
- Sekcija 3 - Pozivi biblioteka
- Sekcija 4 - Uređaji (datoteke u direktoriju **/dev**)
- Sekcija 5 - Konfiguracijske datoteke i njihov format
- Sekcija 6 - Igre
- Sekcija 7 - Makro paketi
- Sekcija 8 - Administratorske naredbe

Na primjer, `mkdir` se opisuje i u sekciji 1 (kao naredba za izradu direktorija) i u sekciji 2 (kao sistemski poziv za izradu direktorija). Ako se pokrene:

```
$ man mkdir
```

prikazat će se *man*-stranica iz sekcije 1. Ako se želi prikazati *man*-stranica o sistemskom pozivu, naredbi `man` se u argumentu navede broj sekcije:

```
$ man 2 mkdir
```

3.1.2. Naredba whatis

Naredba `whatis` služi za pretraživanje *man*-stranica po ključnoj riječi. Ako se određena naredba nalazi u više sekcija, ispisat će se sve sekcije.

Primjer pretraživanja *man*-stranica naredbe `mkdir`:

```
make
```

```
$ whatis mkdir
mkdir (2) - create a directory
mkdir (1) - make directories
```

Iz ispisa se vidi da se naredbai `mkdir` nalazi i u sekciji 2 (sistemski pozivi) i u sekciji 1 (informacije o izvršnim datotekama).

3.2. Naredbena linija

3.2.1. Interaktivna ljuska

Osnovni način interakcije s računalom je naredbena linija. Ljuska interpretira instrukcije utipkane na tipkovnici.

Kao posrednik između korisnika i operacijskog sustava služi program koji se zove **ljuska** (*shell*). Ljuska je zapravo programski jezik s varijablama, kontrolnim naredbama, potprogramima, prekidima i dr. Organizirana je kao tumač (*interpreter*) naredbi, što znači da pročita redak teksta, naredbu koju utipka korisnik, interpretira je i poduzme sve potrebne akcije za njezino izvođenje. Kada je naredba izvedena, ljuska daje informaciju korisniku (*prompt*) da je spremna prihvatiti sljedeću naredbu. *Prompt* ljuske završava znakom **$** za običnog korisnika ili znakom **#** za administratorskog korisnika.

Ljuska nije dio jezgre sustava, nego korisnički program. Svatko može napisati svoj program koji će imati ulogu ljuske, međutim poželjno je da to bude standardni program rasprostranjen na svim instalacijama *Linux*a (i *Unix*a), čime se postiže kompatibilnost rada na različitim računalima.

Ljuska je također programsko okruženje u kojem se mogu izvoditi automatizirani zadaci. Programi ljuske (*shell programs*) nazivaju se i **skripte**. Postoji nekoliko vrsta ljuski koje se rabe u okruženju sustava *Linux*. Najčešće su rabljene ljuske:

Najčešće ljuske	Putanja
The Bourne shell	/bin/sh
The Bourne again shell	/bin/bash
The Korn shell	/bin/ksh
The C shell	/bin/csh
Tom's C shell	/bin/tcsh
Z shell	/bin/zsh

Najčešće upotrebljavana ljuska na distribucijama *Linux* je **BASH** (*The Bourn again shell*) i bit će obrađena u ovom tečaju.

Sintaksa je naredbi ljuske:

```
naredba [opcije] {argumenti}
```

Naredba `echo` koristi se za prikaz teksta na zaslonu. U argument se stavlja ono što se želi ispisati:

```
$ echo "ovo je tekst"
ovo je tekst
```

Ljuska interpretira prvu riječ napisanu u naredbenoj liniji kao naredbu. Ta riječ može biti apsolutna ili relativna putanja do izvršne datoteke na disku. Ako ta datoteka postoji, ona će se izvršiti. Ako prva riječ ne započinje znakom „/“, tada će ljuska skenirati direktorije definirane u varijabli PATH i pokušati pokrenuti prvu datoteku na koju naiđe.

Na primjer, varijabla PATH sadrži samo direktorije **/bin** i **/usr/bin**, a naredba se `fdisk` nalazi u direktoriju **/sbin**. Kada korisnik pokuša izvršiti naredbu `fdisk`, ona neće biti pronađena u

direktorijima **/bin** i **/usr/bin** i neće moći biti pokrenuta. Korisnik ju tada može pokrenuti pozivanjem njezine apsolutne putanje:

```
# /sbin/fdisk
```

Korisnik isto tako može upisati relativnu putanju do datoteke:

```
# ../sbin/fdisk
```

Za razliku od operacijskog sustava DOS, u kojem je korisnik mogao pokrenuti naredbu upisivanjem samo imena naredbe (bez putanje) u tekućem direktoriju čija putanja nije definirana u varijabli PATH, u okruženju *Unix/Linux* to nije moguće. Za pokretanje izvršne datoteke koja se nalazi u tekućem direktoriju treba se koristiti relativnom ili apsolutnom putanjom. Slijedi primjer pozivanja naredbe `fdisk`, čija putanja nije definirana u varijabli PATH:

```
# cd /sbin
# ./fdisk
```

Svaki korisnik može definirati postavke svoje ljuske BASH u datoteci **.bashrc**. U njoj se mogu definirati varijable (npr. varijabla PATH), aliasi itd.

3.2.2. Varijable ljuske

Varijable ljuske slične su varijablama korištenim u drugim programskim jezicima. U imenu varijabli mogu se koristiti **samo alfanumerički znakovi**. Na primjer, BROJ=300 dodjeljuje vrijednost 300 varijabli nazvanoj BROJ.

Varijabla se inicijalizira naredbom (važno je naglasiti da nema razmaka ispred i iza znaka = te da naredba počinje imenom varijable, bez znaka $ koji u primjeru u nastavku označava *prompt*):

```
$ BROJ=300
```

Naredba `echo` služi za ispis teksta na zaslonu ili za ispis vrijednosti varijable.

Varijabla se poziva svojim imenom kojem prethodi znak $:

```
$ echo $BROJ
300
$ echo BROJ
BROJ
```

Iz primjera se vidi da je prva naredba ispisala vrijednost varijable, a druga ime varijable.

Ako se varijabla više ne koristi, ona se može obrisati naredbom `unset` pri čemu **znak $ ne prethodi imenu varijable**.

U sljedećem se primjeru vrijednost prethodno inicijalizarane varijable pokušala obrisati tako da je u imenu varijable korišten znak $. Ispis vrijednosti (naredba echo) pokazuje da je vrijednost varijable ostala nepromijenjena (300).

```
$ unset $BROJ
$ echo $BROJ
300
```

U sljedećem se slučaju naredba `unset` koristi na ispravan način, **bez korištenja znaka $**. Ispis vrijednosti (naredba `echo`) pokazuje da je vrijednost varijable obrisana.

```
$ unset BROJ
$ echo $BROJ
```

3.2.3. Vrste varijabli ljuske

Postoje dvije vrste varijabli:

- lokalne
- izvezene (*exported*).

Lokalne varijable dostupne su samo iz trenutačne ljuske. **Izvezene varijable** dostupne su iz trenutačne ljuske (roditelj) i svih ljuski (djece) koje su pokrenute iz te ljuske.

Naredbe `set` i `env` služe za ispis definiranih varijabli:

Naredba	Opis
`set`	Ispisuje **sve** varijable, i lokalne i izvezene.
`env`	Ispisuje **sve izvezene** varijable.

Izvezene varijable su globalne utoliko što ih djeca mogu referencirati.

Svaka lokalna varijabla može postati izvezena, koristeći naredbu `export`. U sljedećem primjeru vidimo da naredba `env` ne pronalazi varijablu BROJ, ali ju nalazi nakon izvršavanja naredbe `export`.

```
$ env | grep BROJ
$ export BROJ
$ env | grep BROJ
BROJ=300
```

Napomena

Naredba `grep` služi za pretraživanje izlaza neke naredbe. U sljedećim poglavljima bit će detaljno objašnjena naredba `grep` i ulančavanje procesa znakom `|`.

3.2.4. Osnovne predefinirane varijable

Kad se korisnik prijavi na sustav, pokrene se njegova ljuska u kojoj može izvršavati naredbe. Ta ljuska ima predefinirane varijable. Tablica prikazuje najčešće rabljene varijable.

Ime varijable	Značenje
DISPLAY	Rabi ju grafičko okruženje *X Windows System*, služi tome da grafičko sučelje zna gdje pokrenuti klijentsku aplikaciju.
HISTFILE	Putanja do korisnikove datoteke s povijesti naredbi.
HOME	Putanja do korisnikova direktorija.
LOGNAME	Ime korisnika pod kojim se pokreće trenutna ljuska.

PATH	Popis direktorija u kojima ljuska pretražuje izvršne programe, kada se naredba pokrene bez apsolutne ili relativne putanje do nje.
PWD	Trenutačni radni direktorij.
SHELL	Korisnikova ljuska (u većini slučajeva je to bash).

Naredbom `echo` mogu se prikazati vrijednosti svih tih varijabli:

```
$ echo $DISPLAY
:0
$ echo $HISTFILE
/home/tux/.bash_history
$ echo $HOME
/home/tux
$ echo $LOGNAME
tux
$ echo $PATH
:/usr/local/bin:/usr/bin:/bin
$ echo $PWD
/home/tux
$ echo $SHELL
/bin/bash
```

Postoji nekoliko specijalnih varijabli povezanih s upravljanjem procesima:

`$!`	Predstavlja ID procesa zadnjeg procesa djeteta.
`$$`	Predstavlja ID procesa trenutačne ljuske.
`$?`	Ima vrijednost 0 ako je zadnja naredba uredno izvršena. Ako nije, tada je vrijednost 1.

Ako se želi ispisati ID procesa trenutačne ljuske, dovoljno je pokrenuti naredbu:

```
$ echo $$
9823
```

Ako se želi ispisati izlazni kod prošle izvršene naredbe, dovoljno je pokrenuti naredbu:

```
$ echo $?
0
```

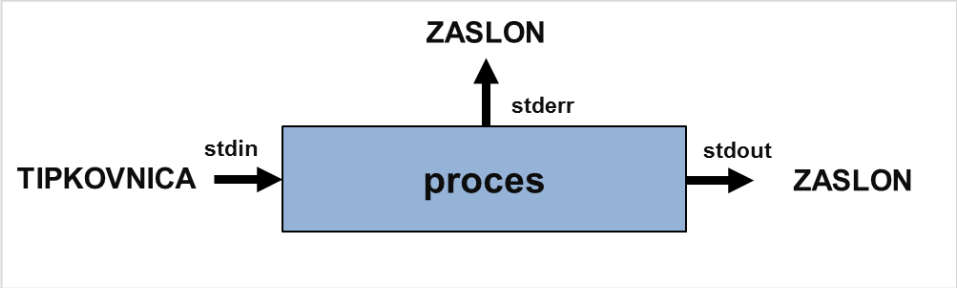
3.2.5. Preusmjeravanje standardnog ulaza i izlaza

Programima (procesima) aktiviranim iz ljuske automatski se pridjeljuju tri „otvorene“ datoteke ***stdin*** (standardni ulaz, eng. *standard input*), ***stdout*** (standardni izlaz, eng. *standard output*) i ***stderr*** (standardni izlaz za pogreške, eng. *standard error*) s brojevima 0, 1 i 2. Ti brojevi (*file descriptors*) opisuju (adresiraju) otvorene datoteke. Pojam „otvorena“ datoteka označava da određeni proces

ima vlasništvo nad dotičnom datotekom. Datoteka *stdin* (0) je otvorena za čitanje, rabi se kao standardni ulaz i obično je to tipkovnica. Datoteka *stdout*(1) je otvorena za pisanje i rabi se kao standardni izlaz. Po definiciji je to korisnikov terminal (zaslon). Datoteka *stderr*(2) je otvorena za pisanje i rabi se za ispisivanje pogrešaka. Po definiciji je to također zaslon terminala.

Početno stanje prilikom aktiviranja novog procesa (standardno zadavanje naredbe) prikazano je na sljedećoj slici:

```
$ naredba
```

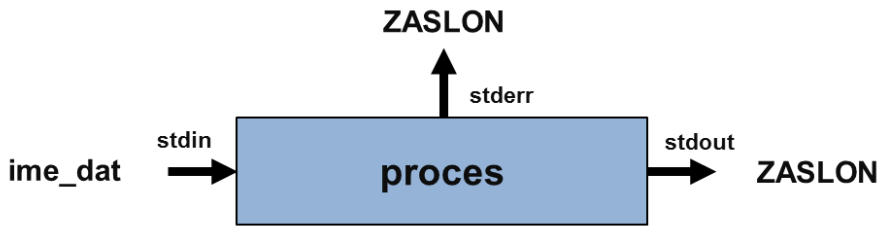


Ljuska može mijenjati dodijeljene ulazno-izlazne datoteke. To se postiže specijalnim znakovima <, > ili 2> u retku naredbe ispred imena datoteke za koju želimo da bude standardni ulaz, izlaz ili izlaz za pogreške.

U nastavku će biti opisani postupci preusmjerivanja (engl. *redirection*) za standardni ulaz i izlaz (<, >). Pritom izlaz za pogreške ostaje nepromijenjen (zaslon terminala). Time se izbjegava da poruke o pogreškama budu „sakrivene“ u nekoj datoteci. Znakove < i > tumači ljuska i ne prosljeđuje ih samoj naredbi. Zato nije potrebno posebno kodiranje.

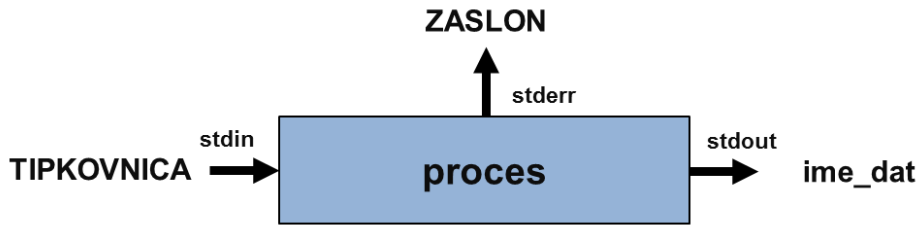
Sljedeća naredba i slika prikazuju preusmjeravanje datoteke **ime_dat** na standardni ulaz procesa, tj. naredbe.

```
$ naredba < ime_dat
```



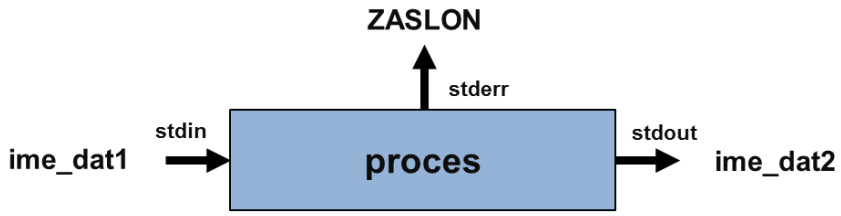
Sljedeća naredba i slika prikazuju preusmjeravanje standardnog izlaza procesa na datoteku **ime_dat**. Time će se presnimiti datoteka **ime_dat**.

```
$ naredba > ime_dat
```



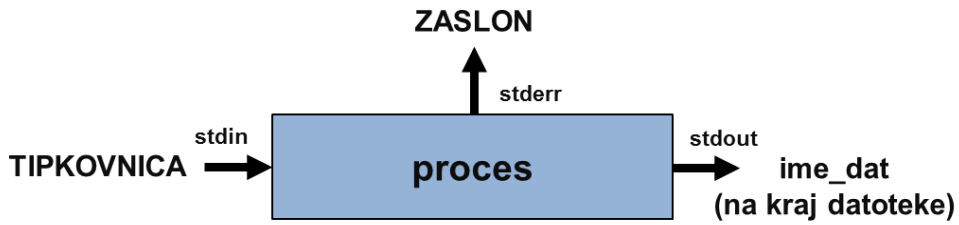
Sljedeća naredba i slika prikazuju preusmjeravanje datoteke **ime_dat1** na standardni ulaz procesa, te standardnog izlaza procesa na datoteku **ime_dat2**.

```
$ naredba < ime_dat1 > ime_dat2
```



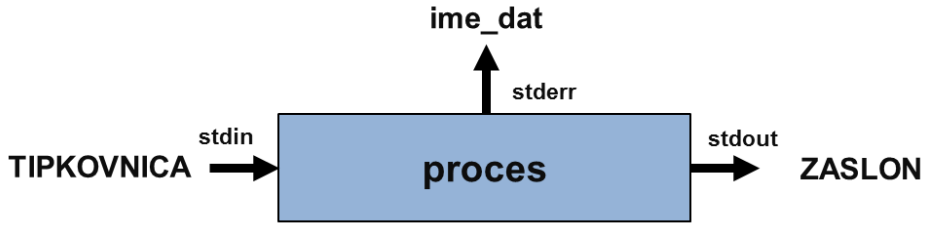
Sljedeća naredba i slika prikazuju preusmjeravanje standardnog izlaza procesa na datoteku **ime_dat**. Time se neće presnimiti datoteka **ime_dat**, tj. novi podaci će se zapisati na kraj datoteke.

```
$ naredba >> ime_dat
```



Ako se želi standardni izlaz za pogreške preusmjeriti u neku datoteku, to se postiže posebnim znakovima 2>. Slijedi primjer preusmjeravanja standardnog izlaza za pogreške u datoteku **ime_dat**.

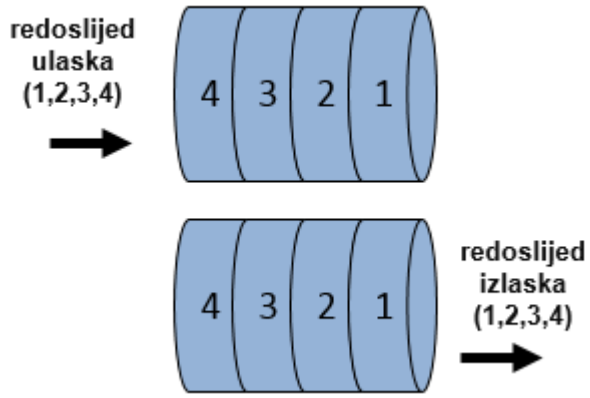
```
$ naredba 2> ime_dat
```



3.2.6. Ulančavanje procesa

Važna je značajka operacijskih sustava *Unix* i *Linux* u usporedbi s nekim drugim operacijskim sustavima mogućnost **ulančavanja procesa**, tj. stvaranja kanala (*pipes*) kojima se izlaz iz jednog procesa dovodi na ulaz drugog procesa. Po istom principu po kojem je u prethodnim slučajevima preusmjerivan ulaz – izlaz u neku datoteku, u okviru ljuske moguće je preusmjerivanje na drugi proces.

Tijekom takvog poziva naredbe nastaje sakrivena, privremena datoteka zvana *pipe* na principu *fifo repa* (prvi unutra, prvi van, eng. *fifo – first in first out*) koja omogućuje programima (procesima) da rade paralelno i uz sinkronizaciju sustava te da prenose podatke iz jednog procesa u drugi.

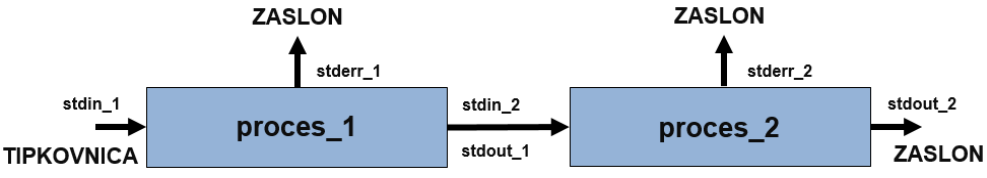


Notacija za povezivanje dvaju procesa kanalom vrlo je jednostavna. Između dviju naredbi treba utipkati znak |, što je dovoljno da ljuska pokrene mehanizam ulančavanja procesa. Znak | na hrvatskom rasporedu tipkovnice dobije se pritiskom na tipke **[AltGr] + [W]**.

```
$ naredba1 | naredba2
```

Jednostavna notacija imala je značajan utjecaj na programsku metodologiju korisnika operacijskih sustava *Unix/Linux* koji su, ohrabreni jednostavnošću, počeli kombinirati postojeće programe umjesto gradnje novih. Ideja je da se od niza malih komadića (programa) kombiniraju složeniji moduli s određenim ciljem. Tako je lakše definirati, dokumentirati i održavati manje cjeline, a povećava se pouzdanost modula izvedenih iz osnovnih programa. Jednostavni primjeri takvog kombiniranja naredba (programa) pokazani su u daljnjem tekstu.

Na sljedećoj je slici prikazano zadavanje naredbe za povezivanje dvaju procesa kanalima.



Ako želimo preusmjeriti standardni izlaz i u datoteku i na zaslon, to možemo napraviti naredbom `tee`.

Naredba `tee` čita ono što dobije na standardni ulaz, preusmjerava na svoj standardni izlaz i u datoteku koja je postavljana u argumentu naredbe `tee`.

```
$ naredba | tee ime_dat
```

Sljedećom naredbom ispisat ćemo sve datoteke koje počinju nizom **passwd** u direktorij **/etc**:

```
$ ls /etc/passwd*
/etc/passwd
/etc/passwd-
```

Ako se taj popis želi preusmjeriti u datoteku, dovoljno je u datoteku preusmjeriti standardni izlaz. Time se popis datoteka neće ispisati na zaslon (tj. standardni izlaz).

```
$ ls /etc/passwd* > /tmp/popis.txt
$ cat /tmp/popis.txt
/etc/passwd
/etc/passwd-
```

Ako se taj popis želi prikazati i na zaslonu (standardni izlaz) i preusmjeriti u datoteku, potrebno je rabiti naredbe `tee`:

```
$ ls /etc/passwd* | tee /tmp/popis.txt
/etc/passwd
/etc/passwd-
$ cat /tmp/popis.txt
/etc/passwd
/etc/passwd-
```

3.2.7. Metaznakovi

Metaznak	Opis
*	zamjenjuje bilo koju skupinu slova u riječi
?	zamjenjuje bilo koje slovo u riječi
[..]	zamjenjuje bilo koji od znakova u zagradama
~	označuje korisnikovo izvorno kazalo, tj. korisnikov kućni direktorij (*home directory)*
>	znači preusmjerivanje izlaza
<	znači preusmjerivanje ulaza
>>	znači dodavanje izlazu
|	znači povezivanje procesa u kanale
&	znači nalog za izvođenje procesa (naredbe) u pozadini
!	(u prvom stupcu naredbe) poziva jednu od prethodno zadanih naredbi

U nastavku je prikazano nekoliko primjera korištenja metaznakova u kombinaciji s naredbom `ls`:

```
ls /usr/bin/b*
```

Rezultat ove naredbe su sve datoteke koje počinju znakom **b** i nalaze se u direktoriju **/usr/bin**.

```
ls /usr/bin/?b*
```

Rezultat ove naredbe su sve datoteke koje imaju slovo **b** na drugom mjestu i nalaze se u direktoriju **/usr/bin**.

`ls a[0-9]`

Rezultat ove naredbe su datoteke koje počinju znakom **a** i na drugom mjestu imaju neki broj.

`ls [!Aa]*`

Rezultat ove naredbe su sve datoteke koje ne počinju slovima **a** ili **A**.

`ls ~`

Rezultat ove naredbe su sve datoteke u korisničkom direktoriju.

3.2.8. Navodnici

Posebno značenje metaznakova može biti poništeno znakom *escape*, koji su također metaznakovi.

Backslash (znak \) zove se znak *escape* i poništava značenje bilo kojeg metaznaka, prisiljavajući ljusku da ga doslovno interpretira.

Ako se želi ispisati metaznak *, ispred njega treba upisati znak *escape* \. Ako ne stavimo znak *escape* \, naredba `echo` će ispisati datoteke u tekućem direktoriju.

```
$ echo *
datoteka1.txt datoteka2.txt
$ echo \*
*
```

Jednostruki navodnici (' ') poništavaju značenje svih metaznakova osim znaka *backslash* (znak \).

```
$ echo $BROJ
300
$ echo '$BROJ'
$BROJ
```

Dvostruki navodnici (" ") su slabiji navodnici od jednostrukih i mogu poništiti značenje većine posebnih metaznakova, osim znaka za povezivanje procesa u kanale (|), backslasha (\) i varijabli ($var).

```
echo "$BROJ"
300
```

Navodnici *back tick* (``) izvršit će naredbu koja se nalazi u navodnicima.

U sljedećem će se primjeru izvršiti naredba `date` s argumentima, koja ispisuje samo današnji datum. Osim što je u *back tick* navodnicima, ubačena je i u varijablu $VRIJEME.

```
$ VRIJEME="Danas je `date +%d.%m.%Y.`"
$ echo $VRIJEME
Danas je 11.05.2015.
```

3.2.9. Povijest naredbi

Da bi se izlistao popis prije pokrenutih naredbi, ljuska ima ugrađenu naredbu `history`. Naredba `history` pokreće se bez argumenata i daje popis izvršenih naredbi:

```
$ history
1 VRIJEME="Danasnji datum je `date +%d.%m.%Y.`"
2 echo $VRIJEME
```

Ljuska popis pokrenutih naredbi snima u datoteku **~/.bash_history**. Korisnik može strelicama gore i dolje doći do neke od ranije izvršenih naredbi i izvršiti ju pristikom na tipku [Enter].

Ako korisnik kao prvi znak naredbe unese znak ! te iza njega neki drugi znak, ljuska će pokrenuti zadnju izvršenu naredbu koja je počinjala tim znakom. Ako odabere broj iz povijesti koju je dobio naredbom `history`, izvršit će se naredba pod tim rednim brojem.

Sljedeća naredba pokreće zadnju izvršenu naredbu koja je započinjala znakom x:

```
$ !x
```

Sljedeća naredba pokreće naredbu s rednim brojem 2 u povijesti naredbi (u našem slučaju je to naredba `echo $VRIJEME`).

```
$ !2
```

Sljedeća naredba pokreće zadnju izvršenu naredbu:

```
$ !!
```

Sljedeća naredba pokreće zadnju naredbu tako da zamijeni **string1** sa **string2** u toj naredbi:

```
$ ^string1^string2
```

U sljedećem primjeru zamijenit ćemo tekst "test" iz prethodne naredbe u tekst "test2".

```
$ echo "ovo je test"
ovo je test
$ ^test^test2
echo "ovo je test2"
ovo je test2
```

3.2.10. Aliasi i automatsko nadopunjavanje

Ako korisnik ima potrebu za češćim pokretanjem određene naredbe, može izraditi **alias**. Alias se izrađuje naredbom `alias`.

Sintaksa je sljedeća:

```
$ alias mojprogram='naredba [opcije] {argumenti}'
```

Slijedi primjer gdje se izrađuje alias naziva `trazi` koji pokreće naredbu `find /etc -name passwd`.

Alias `trazi` time pokreće naredbu `find` koja pretražuje direktorij **/etc** i traži sve datoteke koje se zovu **passwd**.

```
$ alias trazi='find /etc -name passwd'
$ trazi
/etc/pam.d/passwd
/etc/cron.daily/passwd
/etc/passwd
```

Ako se alias želi poništiti, to se može naredbom `unalias`. Dovoljno je u argumentu naredbe dodati alias koji se briše:

```
$ unalias trazi
```

Postoji i opcija **brzog nadopunjava naredbi**.

Kad korisnik započne pisati naredbu, npr. *alias*-naredbu `trazi`, može napisati nekoliko prvih slova (npr. `tra`) i pritisnuti tipku **[Tab]**. Ljuska će tada automatski završiti naredbu ili ispisati sve naredbe koje započinju nizom `tra`.

Na isti način može nadopunjavati imena datoteka na disku. Na primjer, za datoteku **/etc/passwd** dovoljno je napisati **/etc/pas** i pritisnuti tipku **[Tab]**.

3.2.11. Izvršavanje više naredbi

Korisnik može izvršavati više naredbi u nizu.

Više naredbi koje se izvršavaju jedna za drugom, bez obzira na uspješnost prethodno pokrenute naredbe:

```
naredba1 ; naredba2 ; naredba3
```

Naredbe se izvršavaju jedna za drugom samo u slučaju da prethodna naredba ima izlazni kod 0 (uspješno izvršena):

```
naredba1 && naredba2 && naredba3
```

Naredbe se izvršavaju jedna za drugom samo u slučaju da prethodna naredba ima izlazni kod različit od 0 (neuspješno izvršena):

```
naredba1 || naredba2 || naredba3
```

U nastavku slijede primjeri izvršavanja više naredbi u nizu.

Prva naredba prikazuje ispis pogreške jer datoteka **/etc/ne_postoji** stvarno ne postoji. Naredba `echo` prikazuje izlazni kod prve naredbe, koji je 2.

```
$ ls /etc/ne_postoji
ls: cannot access /etc/ne_postoji: No such file or directory
$ echo $?
2
```

Ova naredba prikazuje ispis datoteke koja postoji, a budući da je naredba uredno izvršena, njezin izlazni kod je 0.

```
$ ls /etc/passwd
/etc/passwd
$ echo $?
0
```

U sljedećem su se primjeru obje naredbe izvršile jer ne postoji uvjet izvršavanja druge naredbe:

```
$ ls /etc/ne_postoji ; ls /etc/passwd
ls: cannot access /etc/ne_postoji: No such file or directory
/etc/passwd
```

U sljedećem primjeru izvršila se samo prva naredba. Izlazni kod prve naredbe je 2, što znači da se prva naredba neuspješno izvršila i zbog toga se nije izvršila druga naredba:

```
$ ls /etc/ne_postoji && ls /etc/passwd
ls: cannot access /etc/ne_postoji: No such file or directory
```

U sljedećem primjeru izvršile su se obje naredbe. Izlazni kod prve naredbe je 2, što znači da se prva naredba neuspješno izvršila i zbog toga se izvršila druga naredba:

```
$ ls /etc/ne_postoji || ls /etc/passwd
ls: cannot access /etc/ne_postoji: No such file or directory
/etc/passwd
```

3.2.12. Naredba exec

Naredba `exec` rabi se kad želimo zamijeniti trenutačnu interaktivnu ljusku s nekim drugim programom:

```
exec program
```

Kad se korisnik prijavi u sustav kao administratorski korisnik *root*, automatski će se pokrenuti ljuska **bash**. Ako korisnik želi promijeniti trenutačnu ljusku u **zsh**, pokrenut će naredbu:

```
# exec zsh
```

Kod takve upotrebe naredbe `exec` ne stvara se novi proces, kao što bi se dogodilo da smo ovako pokrenuli novi proces:

```
# zsh
```

nego se postojeći zamjenjuje sa **zsh**.

U sljedećem je primjeru vidljivo da je pokretanjem ljuske **zsh** pomoću naredbe `exec` nova ljuska **zsh** dobila isti identifikacijski broj procesa kao i stara ljuska **bash** (9823). Znači, proces stare ljuske je nestao i umjesto njega je pod istim identifikacijskim brojem pokrenuta nova ljuska.

Naredba `ps` služi za ispisivanje popisa aktivnih procesa, a naredba `grep` za filtriranje linija koje sadrže određenu riječ.

```
# ps -ef | grep bash | grep -v grep
root 9823 16169 0 18:00 pts/5 00:00:00 bash
# exec zsh
# ps -ef | grep 9823 | grep -v grep
root 9823 16169 0 18:00 pts/5 00:00:00 zsh
```

3.3. Vježba 2: Naredbena linija

Vježba: Pokretanje terminala

1. Prijavite se na računalo svojim korisničkim imenom i lozinkom.
2. Kliknite mišem na *Applications* → *Accesories* → **Terminal** u desnom gornjem kutu.Time ćete pokrenuti terminal u kojem možete upisivati naredbe iz slijedećih vježbi.
3. Ukoliko želite podesiti font i njegovu veličinu, te boje kliknite mišem na *Edit* → *Profile Preferences*.

Vježba: Man-stranice

1. Koristeći se naredbama `man`, pročitajte man-stranicu naredbe `mkdir`.
2. Za koju se akciju koristi naredba `mkdir`?

   _____
3. Naredbom `whatis` pronađite sve man-stranice s ključnom riječi **mkdir**.
4. Pročitajte man-stranice u sekcijama 1 i 2.
5. U čemu je razlika između man-stranice u sekciji 1 i u sekciji 2 naredbe `mkdir`?

   _____

   _____

Vježba: *stdin-stdout-stderr*

1. Naredbom `find` potražite sve datoteke u direktoriju **/etc** koje počinju slovom **p** i preusmjerite rezultat u datoteku **/tmp/nadjeno.txt**.

   ```
   find /etc –name "p*" > /tmp/nadjeno.txt
   ```
2. Provjerite sadržaj novoizrađene datoteke naredbom `cat`. Nalazi li se u njoj popis svih datoteka i direktorija u direktoriju **/etc** koji počinju slovom **p**? _____

   ```
   cat /tmp/nadjeno.txt
   ```
3. Koristeći se identičnom naredbom nađite sve datoteteke u direktoriju **/etc** koje počinju slovom **s** i rezultat stavite na kraj datoteke **/tmp/nadjeno.txt**.

   ```
   find /etc –name "s*" >> /tmp/nadjeno.txt
   ```
4. Naredbom **`sort`** razvrstajte popis iz datoteke **/tmp/nadjeno.txt** i preusmjerite taj rezultat u novu datoteku **/tmp/nadjeno2.txt**.

   ```
   sort < /tmp/nadjeno.txt > /tmp/nadjeno2.txt
   ```
5. Provjerite sadržaj novoizrađene datoteke **/tmp/nadjeno2.txt** naredbom `cat`.

```
cat /tmp/nadjeno2.txt
```

Što je sadržaj datoteke **/tmp/nadjeno2.txt**?

_____

Vježba: Ulančavanje procesa

1. Naredbom **`find`** u direktoriju **/bin** pronađite sve datoteke koje završavaju slovom **s** i razvrstajte izlaz naredbom `sort`.

   ```
   find /bin -name "*s" | sort
   ```

2. Pokušajte pokrenuti naredbu `find` bez razvrstavanja izlaza.
   Proučite razliku.

3. Kako radi ulančavanje procesa u 1. zadatku?

   _____

   _____

   _____

4. Naredbom **`du`** ispišite veličinu svih direktorija u direktoriju **/usr** i poredajte po veličini (mogućnost `-n` naredbe `sort` služi za numeričko razvrstavanje).

   ```
   du -sk /usr/* | sort -n
   ```

5. Preusmjerite standardni izlaz ulančanih naredbi iz prošlog zadatka u datoteku **/tmp/izlaz.txt**.

   ```
   du -sk /usr/* | sort -n > /tmp/izlaz.txt
   ```

6. Koristeći se naredbom `tee` preusmjerite standardni izlaz ulančanih naredbi iz prošlog zadataka na standardni izlaz i u datoteku **/tmp/izlaz.txt**.

   ```
   du -sk /usr/* | sort -n | tee /tmp/izlaz.txt
   ```

Vježba: Varijable

1. Varijabli ALERT dodijelite vrijednost **virus**.

   ```
   ALERT=virus
   ```

2. Naredbom **`set`** provjerite je li ta varijabla definirana. Naredba `grep` ispisuje linije koje sadrže izraz ALERT.

   ```
   set | grep ALERT
   ```

3. Provjerite je li varijabla definirana naredbom `env`.

   ```
   env | grep ALERT
   ```

   Budući da varijabla nije izvezena, ne bi trebala biti definirana.

4. Zatim pokrenite još jednu ljusku i provjerite možete li prikazati vrijednost varijable.

   ```
   bash
   echo $ALERT
   ```

   Posljednja naredba ne bi smjela ispisati vrijednost varijable jer je pokrenuta nova ljuska, a varijabla nije izvezena. Pokrenite naredbu `exit` da biste izašli iz trenutačne novopokrenute ljuske i vratili se u staru ljusku.

5. Izvezite varijablu pomoću naredbe `export`.

   ```
   export ALERT
   ```

6. Naredbom **`env`** ponovno provjerite je li varijabla postala globalna (izvezena). Je li varijabla postala globalna? _____

7. Nakon toga pokrenite ponovno ljusku i provjerite možete li sada doći do vrijednosti te varijable.

   ```
   bash
   echo $ALERT
   ```

8. U novopokrenutoj ljusci promijenite vrijednost varijable.

   ```
   export ALERT=green
   ```

9. Naredbom `exit` izađite iz novootvorene ljuske i provjerite vrijednost varijable u roditeljskoj ljusci.

   Koja je vrijednost varijable u originalnoj roditeljskoj ljusci? _____

Pitanja za ponavljanje

1. U koliko su sekcija podijeljene man-stranice?

   _____

2. U čemu je razlika između lokalnih i izvezenih varijabli?

   _____

   _____

3. Koje se tri vrste datoteka automatski pridjeljuju procesima (programima) pokrenutim u ljusci?

   _____

   _____

4. Čemu služi ulančavanje procesa?

   _____

   _____

   _____

5. Kada se u ljusci djetetu promijeni vrijednost varijable, hoće li se promijeniti i u ljusci roditelju?

   _____

4. Upravljanje datotekama i direktorijima

Trajanje poglavlja:
125 min

Po završetku ovoga poglavlja moći ćete:

- pokazati kretanje po datotečnom sustavu
- primijeniti naredbe `pwd` i `cd`
- pronalaziti datoteke i direktorije
- rabiti naredbe `find`, `locate` i `which`
- izraditi, kopirati i premještati datoteke i direktorije
- primijeniti naredbe `mkdir`, `rmdir`, `rm`, `cp` i `mv`
- razlikovati permamentne i simboličke poveznice
- koristiti se naredbom `ln`
- izraditi datoteke
- primijeniti naredbe `touch` i `dd`.

Ova cjelina obrađuje osnovne naredbe za kretanje po datotečnom sustavu, pronalaženje datoteka i direktorija te upravljanje datotekama i direktorijima. Naučit ćemo izrađivati, kopirati i premještati datoteke i direktorije, te razliku između permanentnih i simboličkih poveznica.

4.1. Kretanje po datotečnom sustavu

4.1.1. Apsolutna i relativna putanja

Datoteci ili direktoriju može se pristupiti **punom putanjom**, koja započinje znakom / (tj. ishodišnim direktorijem - *root*), ili **relativnom putanjom**, koja započinje od trenutačnog direktorija u kojem se nalazi korisnik.

Apsolutna putanja je neovisna o trenutačnom direktoriju i započinje znakom **/**.

Primjeri apsolutne putanje:

```
/etc/passwd
/root/.bashrc
/usr/local/bin/command
```

Relativna putanja ovisi o tome u kojem se direktoriju nalazi korisnik i ne započinje znakom **/**. Može započeti znakovima .. (označava prethodni direktorij) ili znakom . (označava trenutačni direktorij).

Primjeri relativne putanje:

```
passwd
root/.bashrc
../local/bin/command
```

4.1.2. Naredbe pwd i cd

Kao i u bilo kojem drugom struktiriranom datotečnom sustavu, postoji nekoliko alata koji korisniku pomažu kretati se kroz datotečni sustav. Kod operacijskog sustava *Debian* to su najčešće naredbe `pwd` i `cd` koje su ugrađene u korisničku ljusku.

Naredba	Opis
`pwd`	Ova naredba prikazuje trenutačnu lokaciju korisnika. Lokacija se prikazuje u obliku apsolutne putanje do trenutačnog direktorija.
`cd`	Ova naredba služi za promjenu trenutačnog direktorija (cd - *change directory*).

U primjeru u nastavku, naredbom `pwd` prikaže se trenutačna lokacija (**/root**), a naredbom `cd` lokacija se promijeni u **/usr/local/bin/**. Da bi se provjerio rezultat primjene naredbe `cd`, provjerava se trenutačna lokacija naredbom `pwd`. Iz rezultata (**/usr/local/bin**) vidljivo je da je se naredbom `cd` promijenila trenutačna lokacija.

```
$ pwd
/root
$ cd /usr/local/bin/
$ pwd
/usr/local/bin
```

Oznaka ~ označava osobni direktorij korisnika. Ako se korisnik nalazi u nekom drugom direktoriju, u svoj se direktorij može vratiti naredbom `cd ~`.

```
$ pwd
/usr/local/bin
$ cd ~
$ pwd
/home/korisnik
```

Ako se korisnik želi vratiti u prethodni direktorij u kojem je bio, može se koristiti naredbom `cd -`.

```
$ pwd
/home/korisnik
$ cd -
$ pwd
/usr/local/bin
```

Ako korisnik želi otići u osobni direktorij nekog drugog korisnika, npr. **tux**, onda može koristiti oblik `cd ~tux`.

```
$ pwd
/home/korisnik
$ cd ~tux
$ pwd
/home/tux
```

4.1.3. Isprobajte naredbe

1. Prijavite se na sustav kao gost odabirom poveznice *login as guest*.
2. Upišite naredbu za provjeru trenutačne lokacije. U kojem se direktoriju nalazite?
3. Postavite se u direktorij **/usr/bin/** uporabom odgovarajuće naredbe.
4. Ponovno provjerite trenutačnu lokaciju. U kojem se sada direktoriju nalazite?

Odgovori na pitanja

1. Upišite naredbu za provjeru trenutačne lokacije. U kojem se direktoriju nalazite?

```
pwd
/root
```

2. Postavite se u direktorij **/usr/bin/** uporabom odgovarajuće naredbe.

```
cd /usr/bin/
```

3. Ponovno provjerite trenutačnu lokaciju. U kojem se sada direktoriju nalazite?

```
pwd
/usr/bin
```

4.2. Pronalaženje datoteka i direktorija

4.2.1. Naredba find

U *Linux*ovom okruženju ima više naredbi za pretraživanje datoteka i direktorija. Najčešće se rabe `find`, `locate` i `which`.

Naredba `find` služi za pretraživanje datotečnog sustava.

Njezina je sintaksa:

```
find <direktorij> <kriterij> [-exec {} \;]
```

Argument <direktorij> kaže naredbi `find` gdje da započne pretragu. Pretraga uključuje **taj direktorij i sve poddirektorije u njemu**. Argumentom <kriterij> definiramo prema kojem kriteriju

pretražujemo datoteke - ime datoteke, tip (direktorij ili datoteka), vlasnik, vrijeme pristupa, izrade ili modificiranja.

Sljedeća tablica prikazuje osnovne kriterije pretraživanja.

Osnovni kriteriji pretraživanja	
-type [f|d]	Tip pretrage može biti f za datoteku ili d za direktorij.
-name IME	Ime datoteke (mogu se koristiti regularni izrazi).
-user KORISNIK	Vlasnik tražene datoteke ili direktorija.
-atime BROJDANA	Vrijeme zadnjeg pristupa datoteci ili direktoriju izraženo u danima.
-ctime BROJDANA	Vrijeme izrade datoteke ili direktorija izraženo u danima.
-mtime BROJDANA	Vrijeme zadnje promjene datoteke ili direktorija izraženo u danima.
-amin BROJMINUTA	Vrijeme zadnjeg pristupa datoteci ili direktoriju izraženo u minutama.
-cmin BROJMINUTA	Vrijeme izrade datoteke ili direktorija izraženo u minutama.
-mmin BROJMINUTA	Vrijeme zadnje promjene datoteke ili direktorija izraženo u minutama.
-newer DATOTEKA	Datoteke stvorene prije datoteke DATOTEKA.

4.2.2. Nekoliko primjera korištenja naredbe find

Sljedeća naredba pretražuje datoteke u direktoriju **/home** koje se zovu **.zshrc**:

```
$ find /home –name .zshrc
/home/irako/.zshrc
/home/sabina/.zshrc
/home/tux/.zshrc
```

Naredba pretražuje datoteke u direktoriju **/etc** čije **ime** započinje znakom **x**:

```
$ find /etc -name "x*"
/etc/xinetd.d
/etc/xinetd.conf
/etc/xml
/etc/xml/xml-core.xml
/etc/xml/xml-core.xml.old
/etc/init.d/x11-common
/etc/init.d/xinetd
```

Naredba pretražuje sve datoteke na svim montiranim datotečnim sustavima koje su u vlasništvu korisnika **tux**.

```
$ find / -user tux
/home/tux/.bashrc
```

```
/home/tux/.bash_profile
/home/tux/.bash_history
```

Popis traženih datoteka naredba ispisuje na standardnom izlazu. Ako želimo obrisati navedene datoteke, ili promijeniti dozvole nad datotekama koje naredba nađe, to možemo pomoću opcije **–exec**.

Naredba pretražuje sve datoteke u vlasništvu korisnika **tux** i briše ih.

```
$ find / -type f -user tux –exec rm –f {} \;
```

Slično se može napraviti naredbom `xargs`. Naredba prihvaća popis datoteka sa standardnog ulaza i nad njima izvršava naredbu u argumentu.

```
$ find / -type f -user tux | xargs rm –f
```

Rezultat će biti isti kao i kod prethodne naredbe, obrisat će se sve datoteke koje pripadaju korisniku **tux**.

4.2.3. Naredba locate

Pretraživanje naredbom `find` može biti sporo. Pretraživanje svih montiranih datotečnih sustava može potrajati i desetke minuta.

Zbog toga postoji naredba `locate` koja pretražuje osjetno brže. Radi tako da naredba `updatedb` koja se pokreće iz ***crona*** (servisa koji u točno određeno vrijeme pokreće određene programe, obično noću) spremi popis datoteka i direktorija u lokalnu bazu podataka. Naredba `locate` pokreće upit u toj lokalnoj bazi podataka i puno brže dolazi do rezultata koji ispisuje na standardnom izlazu. Treba uzeti u obzir da pretražuje stanje montiranih datotečnih sustava u vrijeme zadnjeg izvršavanja naredbe `updatedb`.

Sintaksa je naredbe `locate`:

```
$ locate STRING
```

U argumentu se navodi dio imena datoteke ili direktorija koji se traži.

U sljedećem primjeru pretražit će se sve datoteke koje u sebi imaju **/etc/pass**. Rezultat će biti apsolutne putanje do tih datoteka.

```
$ locate /etc/pass
/etc/passwd
/etc/passwd-
```

4.2.4. Naredba which

Naredba `which` vraća punu putanju do naredbe koju pretražujemo unutar direktorija definiranih u korisnikovoj varijabli **PATH**.

Sintaksa je naredbe `which`:

```
$ which STRING
```

U argumentu se navodi dio imena datoteke ili direktorija koji se traži.

U sljedećem primjeru ispisat ćemo apsolutnu putanju do naredbe `ls`.

```
$ which ls
/bin/ls
```

Isprobajte navedenu naredbu u simuliranom okruženju. Prijavite se na sustav kao gost odabirom poveznice *login as guest*.

4.3. Upravljanje direktorijima

4.3.1. Izrada novog direktorija

Naredba za izradu novog direktorija je `mkdir`. Kao argument se koristi apsolutna ili relativna putanja do direktorija koji se želi izraditi.

Primjer je izrade direktorija **/tmp/novi**:

```
$ mkdir /tmp/novi
```

Korisna je opcija `-p`, koja automatski stvara sve poddirektorije koji su potrebni.

U sljedećem će primjeru biti napravljeni direktorij **/tmp/novi** i u njemu **/tmp/novi/dir**.

```
$ mkdir -p /tmp/novi/dir
```

4.3.2. Brisanje direktorija

Naredbe za brisanje direktorija su `rmdir` ili `rm -r`. Ako ste prijavljeni kao *root,* možete dodati opciju **-f** koja prisiljava na brisanje svih datoteka u direktoriju koji je zadan kao argument.

Napomena

Ako se rabi više opcija naredbe (npr. i -r i -f) tada se one mogu pisati zajedno kao `-rf`.

Naredba briše sve datoteke i poddirektorije unutar direktorija **/dir1,** tj. ostavlja direktorij **/dir1** praznim.

```
$ rm -rf /dir1/*
```

Naredba briše sve datoteke i poddirektorije uključujući i **/dir1**.

```
$ rm -rf /dir1/
```

4.3.3. Kopiranje datoteka i direktorija

Naredba `cp` služi za kopiranje datoteka i direktorija. Njezina je sintaksa:

cp [opcije] datoteka1 datoteka2

cp [opcije] datoteke direktorij

Važno je napomenuti da naredba `cp datoteka1 datoteka2` kopira **datoteka1** i ostavlja je nepromijenjenu.

Isto se tako može kopirati nekoliko datoteka u direktorij, pomoću liste direktorija ili zamjenskog znaka *.

U sljedećoj su tablici navedene najčešće korištene opcije naredbe `cp`.

Najčešće korištene opcije naredbe cp	
-d	Ne prati simboličke poveznice.
-f	Prisilno kopiranje.
-i	Interaktivni način rada.
-p	Čuva atribute datoteke.
-R ili -r	Rekurzivno kopiranje direktorija.

Primjeri korištenja

Naredba kopira sve datoteke i poddirektorije u direktoriju **/dir** bez samog direktorija **/dir**.

```
$ cp -r /dir/* /dir2/
```

Naredba kopira sve datoteke i poddirektorije u direktoriju **/dir** uključujući direktorij **/dir**.

```
$ cp -r /dir/ /dir2/
```

4.3.4. Premještanje i preimenovanje datoteka i direktorija

Naredba `mv` služi za premještanje i preimenovanje datoteka i direktorija.

Njena je sintaksa:

```
mv [opcije] staroime novoime
mv [opcije] izvor odredište
mv [opcije] izvor direktorij
```

Ako je **staroime** datoteka, a **novoime** direktorij, tada će premjestiti datoteku **staroime** u direktorij **novoime**.

Ako su izvor i odredište u istom datotečnom sustavu, tada se datoteka **neće kopirati** nego će se ažurirati *inode* (pokazivač na blok s podacima) s informacijom o novoj lokaciji.

Najčešće se rabe opcije **-f** (prisilno premještanje) i **-i** (interaktivni način rada), koje imaju isto značenje kao i kod naredbe `cp`.

Isprobajte naredbe

1. Prijavite se na sustav kao gost odabirom poveznice *login as guest*.
2. Iz direktorija **/usr/bin** kopirajte datoteku **mkfifo** u direktorij **/root**.
3. Uporabom naredbe **ls** prikažite sadržaj direktorija **/root**.
4. U direktoriju **/root** preimenujte datoteku **mkfifo** u **mkfifonew**. Provjerite rezultat naredbom **ls**.
5. Premjestite datoteku **mkfifonew** iz direktorija **/root** u direktorij **/var**. Naredbom **ls** provjerite je li direktorij **/root** prazan.
6. Izbrišite datoteku **mkfifonew** u direktoriju **/var**. Provjerite sadrži li direktorij **/var** datoteku **mkfifonew**.

Odgovori na pitanja

1. Iz direktorija **/usr/bin** kopirajte datoteku **mkfifo** u direktorij **/root**.

```
[root@localhost ~]# cp /usr/bin/mkfifo /root
```

2. Uporabom naredbe ls prikažite sadržaj direktorija **/root**.

```
[root@localhost ~]# ls
```

3. U direktoriju **/root** preimenujte datoteku **mkfifo** u **mkfifonew**. Provjerite rezultatat naredbom ls.

```
[root@localhost ~]# mv mkfifo mkfifonew
[root@localhost ~]# ls
```

4. Premjestite datoteku **mkfifonew** iz direktorija **/root** u direktorij **/var**. Naredbom ls provjerite je li direktorij **/root** prazan.

```
[root@localhost ~]# mv mkfifonew /var
[root@localhost ~]# ls
```

5. Izbrišite datoteku **mkfifonew** u direktoriju **/var**. Provjerite sadrži li direktorij **/var** datoteku **mkfifonew**.

```
[root@localhost ~]# cd /var
[root@localhost ~]# ls
[root@localhost ~]# rm mkfifonew
[root@localhost ~]# ls
```

Napomena

Molimo pričekajte nekoliko sekundi dok se ne pojavi pokazivač.

4.4. Permanentne i simboličke poveznice

4.4.1. Simbolička poveznica

Simbolička poveznica je *alias* ili prečac prema datoteci ili direktoriju. Izradom te poveznice kreirat će se novi *inode* (dio na disku koji sadrži pokazivač) koji pokazuje na isto mjesto s podacima. Naredba `ln -s` rabi se za izradu simboličkih poveznica.

Primjer korištenja naredbe `ln`. Za provjeru rezultata rabi se naredba `ls`. Opcija `-al` služi za detaljniji prikaz informacija o datotekama.

```
$ ln –s passwd passwd.sym

$ ls -al passwd passwd.sym
-rw-r--r-- 1 root root 2661 Mar  2 11:02 passwd
lrwxrwxrwx 1 root root    6 Mar  3 16:11 passwd.sym -> passwd
```

Iz ovog se prikaza vidi da je **passwd** datoteka, a da je **passwd.sym** simbolička poveznica koja pokazuje na datoteku **passwd**. Isto tako se vidi da je referentni broj **1** i za datoteku i za simboličku poveznicu.

Simboličke poveznice mogu se izraditi kroz različite datotečne sustave. To znači da se na jednom datotečnom sustavu može napraviti simbolička poveznica na drugi datotečni sustav.

Npr. ako je particija *root* odvojena od particije **/var/root**, moguće je napraviti simboličku poveznicu **/var/root/passwd.sym** koja pokazuje na **/etc/passwd**.

4.4.2. Permanentna poveznica

Permanentna poveznica je još jedno ime za isti *inode* i referentni broj za svaku datoteku se povećava izradom svake nove permanentne poveznice na tu datoteku. Naredba `ln` rabi se i za izradu **permanentnih poveznica**.

Primjer uporabe naredbe `ln`. Za provjeru rezultata rabi se naredba `ls`.

```
$ ln passwd passwd.link

$ ls -al passwd passwd.link
-rw-r--r-- 2 root root 2661 Mar  2 11:02 passwd
-rw-r--r-- 2 root root 2661 Mar  2 11:02 passwd.link
```

Iz ovog se prikaza vidi da je referentni broj povećan na **2** i da su te dvije datoteke jednakih veličina i vremena izrade.

Permanentne poveznice mogu biti izrađene samo unutar istog datotečnog sustava.

4.5. Izrada datoteka

4.5.1. Naredba touch

Datoteka se može izraditi na više načina. Najčešća naredba za izradu ili modificiranje datoteka je `touch`.

Njezina je sintaksa:

```
$ touch [opcije] datoteka
```

Ako datoteka ne postoji, naredba ju izrađuje. Isto tako je moguće mijenjati vrijeme pristupa datoteci koristeći opciju **–a**, vrijeme zadnje izmjene koristeći opciju **–m** ili pomoću opcije **–r** aplicirati vremenske atribute neke druge datoteke.

U sljedećoj tablici su navedene najčešće korištene opcije naredbe `touch`.

Najčešće opcije naredbe touch	
-a	Mijenja vrijeme pristupa datoteci.
-m	Mijenja vrijeme zadnje izmjene datoteke.
-r	Aplicira vremenske atribute neke druge datoteke.

Primjeri uporabe:

Naredba izrađuje datoteke **datoteka1.txt** i **datoteka2.txt** u tekućem direktoriju:

```
$ touch datoteka1.txt datoteka2.txt
```

Datoteka **datoteka** preuzima atribute datoteke **/etc/passwd**.

```
$ touch datoteka -r /etc/passwd
```

Napomena

Ako se želi izraditi datoteka koja započinje znakom - (npr. **-datoteka**), naredba `touch` rabi se u ovom obliku:

```
$ touch -- -datoteka
```

U protivnom bi **-datoteka** bila smatrana opcijom naredbe `touch` i ispisala bi se pogreška jer ta opcija nije ispravno zadana:

```
$ touch -datoteka
touch: invalid date format ‘atoteka’
```

4.5.2. Naredba dd

Druga često korištena naredba je `dd`. Tom se naredbom kopiraju datoteke s promjenjivim veličinama bloka.

Glavne opcije su if= (*input file*, ulazna datoteka) i of= (*output file*, izlazna datoteka).

U sljedećem primjeru naredba će iskopirati presliku diskete (/root/boot.img) na disketni uređaj (/dev/fd0):

```
$ dd if=/root/boot.img of=/dev/fd0
```

Budući da su danas disketni uređaji rijetki, a na CD ROM se ne može pisati, sliku se može iskopirati nekamo drugamo, npr. **/tmp/fd0**.

Za razliku od naredbe `cp`, naredba `dd` može kopirati cijeli uređaj i pritom sačuvati datotečni sustav koji leži na tom uređaju.

4.6.Vježba 3: Upravljanje datotekama i direktorijima

Napomena

Ovu vježbu potrebno je izvoditi s ovlastima korisnika *root*. U terminal je potrebno upisati:

`su -` pa lozinku korisnika *root* dodijeljenu prilikom instalacije.

Upravljanje datotekama i direktorijima

1. Pomoću naredbe `mkdir` izradite novi direktorij u direktoriju **/tmp** naziva **etc**.

   ```
   mkdir /tmp/etc
   ```

2. Koristeći se naredbom **`touch`** izradite datoteku **newfile** u direktoriju **/tmp/etc**.

   ```
   touch /tmp/etc/newfile
   ```

3. Postavite se na izvorišni direktorij (`cd /`).

4. Provjerite i zaokružite koje će od ovih naredbi ispisati sadržaj novoizrađene datoteke:

   ```
   cat etc/newfile
   cat /etc/newfile
   cat tmp/etc/newfile
   cat /tmp/etc/newfile
   ```

5. Promijenite ime datoteci **newfile** u **oldfile**. Neka datoteka ostane u istom direktoriju.

   ```
   mv /tmp/etc/newfile /tmp/etc/oldfile
   ```

6. Iskopirajte datoteku **/etc/passwd** u datoteku **newfile** u istom direktoriju.

   ```
   cp /etc/passwd /tmp/etc/newfile
   ```

7. Koristeći se naredbom `ls` provjerite koliko datoteka imate u direktoriju **/tmp/etc**.

   ```
   ls /tmp/etc
   ```

8. Obrišite direktorij **/tmp/etc** pomoću naredbe `rm`.

   ```
   rm -rf /tmp/etc
   ```

9. Ponovite vježbu od prvog koraka i obrišite direktorij **/tmp/etc** pomoću naredbe `rmdir`.

   ```
   rmdir /tmp/etc
   ```

   Dogodila se pogreška. Zašto?

   _____

Traženje datoteka u datotečnom sustavu

1. Pomoću naredbi `find` i `locate` nađite datoteku `shadow-`.

```
find / -name shadow-
locate shadow-
```

2. Usporedite brzine izvođenja naredbi `find` i `locate`.

   Zašto je naredba `locate` brža od naredbe `find`?

   _____

3. Koristeći se naredbom `touch` izradite datoteku **/etc/vjezba.txt**.
4. Pokušajte ju pronaći naredbama `find` i `locate`.

   Zašto je naredba `locate` ne nalazi?

   _____

5. Pokrenite naredbu `updatedb`.

   Pronalazi li je sada naredba `locate`?

   _____

6. Koristeći se naredbom `which` pronađite putanju do naredbe `adduser`.

   _____

Simboličke i permanentne poveznice

1. Izradite simboličku poveznicu **`/tmp/passwd.symlink`** koja pokazuje na datoteku **/etc/passwd**.

   `ln -s /etc/passwd /tmp/passwd.symlink`

2. Izradite permanentnu poveznicu **`/tmp/passwd.hardlink`** koja pokazuje na datoteku **/etc/passwd**.

3. `ln /etc/passwd /tmp/passwd.hardlink`

4. Naredbom `ls -al` ispišite te dvije poveznice.

   U čemu je razlika?

   _____

Pitanja za ponavljanje

1. U čemu je razlika između apsolutnih i relativnih putanja?

   _____

   _____

2. U čemu je razlika između naredbi `find` i `locate`?

   _____

   _____

3. Koje vrste poveznica mogu biti izrađene kroz različite datotečne sustave?

   _____

   _____

5. Obrada teksta

Trajanje poglavlja:
70 min

Po završetku ovoga poglavlja moći ćete:

- pregledavati datoteke
- koristiti se naredbama cat i tac
- koristiti se jednostavnim alatima za pregledavanje tekstualnih datoteka
- brojati količine redova, riječi i znakova u tekstnim datotekama
- spajati i razdvajati tekstne datoteke
- koristiti se naredbama `head`, `tail`, `wc`, `nl`, `od`, `hexdump`, `split`, `sort`, `uniq`
- upravljati tekstom
- koristiti se naredbama `cut`, `join`, `paste`, `fmt` i `tr`.

Ova cjelina obrađuje osnovne naredbe za pregledavanje tekstnih i binarnih datoteka. Obradit će se i naredbe za brojanje količine redova, riječi i znakova u tekstnim datotekama te spajanje i razdvajanje tekstnih datoteka.

5.1. Pregled datoteka

5.1.1. Naredba cat

Naredba `cat` služi za prikaz sadržaja neke datoteke. Njezina je sintaksa:

```
cat [opcije] datoteka1
```

U argument se može staviti i više datoteka:

```
cat [opcije] datoteka1 datoteka2 datoteka3
```

Najčešće se rabe ove opcije:

Opcija	Opis
-n	Uz svaku liniju ispisuje redni broj te linije.
-b	Isto kao opcija -n, ali se neće ispisati redni broj kod prazne linije.
-A	Ispisuje se i znak za novi red.

Sljedeća naredba ispisuje sadržaj datoteke **/etc/hosts**:

```
$ cat /etc/hosts
127.0.0.1 localhost

192.168.1.5 linux.srce.hr linux
```

Ova naredba ispisuje sadržaj datoteke **/etc/hosts** i redni broj linije (opcija `-n`).

```
$ cat -n /etc/hosts
1 127.0.0.1 localhost
2
3 192.168.1.5 linux.srce.hr linux
```

Sljedeća će naredba ispisati sadržaj datoteke **/etc/hosts** i redni broj linije, no preskočit će prazne linije (opcija `-b`).

```
$ cat -b /etc/hosts
1 127.0.0.1 localhost

2 192.168.1.5 linux.srce.hr linux
```

5.1.2. Naredba cat kao uređivač teksta

Naredba `cat` može se koristiti i kao osnovni uređivač teksta.

U sljedećem primjeru standardni izlaz naredbe preusmjerava se u datoteku (`cat > datoteka.txt`) i naredba za izlazak iz uređivača teksta je [Ctrl]+[D].

```
$ cat > datoteka.txt
neki tekst
koji ide u datoteku
[Ctrl]+[D]
```

Sljedećom će se naredbom ispisati sadržaj datoteke **datoteka.txt**.

```
$ cat datoteka.txt
neki tekst
koji ide u datoteku
```

5.1.3. Naredba tac

Naredba `cat` prikazuje datoteku od njezina početka do kraja. Ako se datoteka želi prikazati od kraja do početka, tome služi naredba `tac`. Sintaksa naredbe je identična naredbi `cat`.

U sljedećem će se primjeru ispisat datoteka /etc/hosts od kraja do početka:

```
$ tac /etc/hosts
192.168.1.5 linux.test.hr linux

127.0.0.1 localhost
```

5.2. Jednostavni alati

5.2.1. Naredbe head i tail

Naredbe `head` i `tail` najviše se koriste za analiziranje log-datoteka. Log-datoteke su tekstne datoteke u koje se pohranjuju sistemski zapisi rada sustava (*logs*). Te se naredbe također mogu rabiti i za sve druge tekstne datoteke kao npr. konfiguracijske datoteke, tekstne s podacima i sl. One prikazuju 10 linija teksta s početka ili kraja datoteke u slučaju kada nije određen broj linija koje će se prikazati.

U sljedećem će se primjeru primjenom naredbe `head` prikazati prvih 20 linija datoteke **/var/log/messages**:

```
head -n 20 /var/log/messages
```

Može se rabiti i ovaj oblik, rezultat je identičan, jer naredba smatra broj (-20) argumentom opcije (-n 20):

```
head -20 /var/log/messages
```

Naredba `tail` može prikazivati od nekog retka do kraja datoteke. Ako se u argument stavi -20, prikazat će zadnjih 20 linija datoteke. Ako se stavi +20, prikazat će se retci od dvadesetog do kraja datoteke.

Naredba je za prikaz zadnjih 20 linija datoteke **/etc/aliases**:

```
tail -20 /etc/aliases
```

Sljedeći primjer ispisuje datoteku **/var/log/messages** od njezina 25. retka do kraja datoteke:

```
tail -n +25 /var/log/messages
```

Log-datoteke stalno se povećevaju dodavanjem novih log zapisa na kraj datoteke. Ako se u realnom vremenu želi pregledati što se od svježih log-zapisa zapisuje u određenu log datoteku, može se rabiti naredba `tail -f`.

5.2.2. Naredbe wc i nl

Brojanje linija, riječi ili znakova

Naredba `wc` služi za brojanje broja znakova, riječi i linija u nekoj tekstnoj datoteci. Primjer je uporabe te naredbe:

```
$ wc /etc/passwd
224 437 12709 /etc/passwd
```

Znači, datoteka **/etc/passwd** sadrži 224 linije, 437 riječi i 12709 znakova. Ako se želi ispisati samo broj linija, dodaje se opcija -l:

```
$ wc -l /etc/passwd
224 /etc/passwd
```

Ako se želi ispisati samo broj riječi, tome služi opcija -w:

```
$ wc -w /etc/passwd
437 /etc/passwd
```

A kada se želi ispisati broj znakova, tada se rabi opcija -c:

```
$ wc -c /etc/passwd
12709 /etc/passwd
```

Brojanje linija

Naredba `nl` služi za ispis rednog broja linije kod prikazivanja datoteke.

Primjer kada se želi ispisati redni broj linije tekstne datoteke (izlaz je identičan kao naredba `cat -n`):

```
$ nl -ba /etc/hosts
1 127.0.0.1 localhost
2
3 192.168.1.5 linux.srce.hr linux
```

Primjer kada se ne žele brojati prazne linije (izlaz je identičan kao kod naredbe `cat -b`):

```
$ nl -bt /etc/hosts
1 127.0.0.1 localhost

2 192.168.1.5 linux.srce.hr linux
```

5.2.3. Naredbe od i hexdump

Sve dosad obrađene naredbe služile su za prikaz tekstnih datoteka. Postoji nekoliko alata za prikaz binarnih datoteka. Najčešće se rabe `od` (*octal dump*) i `hexdump`. Naredba `od` prikazat će svaki bajt binarne datoteke u oktalnoj, `hexdump` u heksadecimalnoj notaciji.

Primjer je uporabe naredbe `od`:

```
$ od /bin/ls
0000000 042577 043114 000402 000001 000000 000000 000000 000000
0000020 000002 000076 000001 000000 044200 000100 000000 000000
0000040 000100 000000 000000 000000 133160 000001 000000 000000
...
```

Primjer je uporabe naredbe `hexdump`:

```
$ hexdump /bin/ls
0000000 457f 464c 0102 0001 0000 0000 0000 0000
0000010 0002 003e 0001 0000 4880 0040 0000 0000
0000020 0040 0000 0000 0000 b670 0001 0000 0000
...
```

5.2.4. Naredba split

Ako se neka tekstna datoteka želi razdijeliti na više manjih datoteka, tome će poslužiti naredba `split`. Kriterij za smanjivanje je prema broju linija.

Primjer je uporabe naredbe:

```
$ split -l 5 /etc/passwd
$ ls
xaa xab xac xad xae xaf xag xah
```

Iz navedenog je primjera vidljivo da će se datoteka, preddefinirano, podijeliti na više manjih datoteka koje počinju znakom x.

Opcija **-l 5** u naredbi određuje da će se svaka podijeljena datoteka sastojati od 5 linija. U gornjem primjeru datoteka **/etc/passwd** se sastoji od najviše 40 linija te je njenom podjelom nastalo 8 datoteka.

Ako se umjesto x želi rabiti neki drugi znak ili niz znakova, to treba upisati u argument:

```
$ split -l 5 /etc/passwd passwd
$ ls
passwdaa passwdab passwdac passwdad passwdae passwdaf passwdag passwdah
```

Sljedeći primjeri prikazuju broj linija datoteke **/etc/passwd** prije razdvajanja na manje datoteke i broj linija svih novoizrađenih datoteka:

```
$ wc -l /etc/passwd
24
$ split -l 5 /etc/passwd test
$ wc -l testa*
5 testaa
5 testab
5 testac
5 testad
4 testae
24 total
```

Vidimo da se sve poklapa, da postoje 24 linije u oba slučaja. Ako se sve datoteke žele spojiti, koristi se naredba `cat`:

```
$ cat testa* > passwd2
$ wc -l passwd2
24 passwd2
```

Za razliku od prekidača **-l,** koji definira izlaznu datoteku po broju linija, postoji i prekidač **-c,** koji definira izlaznu datoteku u bajtovima.

5.2.5. Naredbe uniq i sort

Kod prikaza tekstnih datoteka često se pojavljuju uzastopne identične linije.

Naredba `uniq` ispisat će samo jednu uzastopnu liniju, makar je na svoj standardni ulaz dobila više istih linija.

Primjer je uporabe naredbe:

```
$ uniq > /tmp/UNIQUE
linija 1
linija 2
linija 2
linija 3
linija 3
linija 3
linija 1
```

Naredbom `cat` ispisat ćemo datoteku:

```
$ cat /tmp/UNIQUE
linija 1
linija 2
linija 3
linija 1
```

Ako se žele izbaciti sve iste linije, koje nisu uzastopne, može se koristiti kombinacija naredbi `sort` i `uniq`.

Naredba `sort` razvrstat će sve linije, tako da se istoznačne pojave jedna ispod druge, a izbacit će ih naredba `uniq`.

Primjer je uporabe te naredbe:

```
$ cat /tmp/UNIQUE | sort | uniq
linija 1
linija 2
linija 3
```

5.3. Upravljanje tekstom

5.3.1. Naredbe cut, paste i join

Ako se iz tekstne datoteke želi izbaciti dio teksta, rabi se naredba `cut`. Naredba `cut` će izbaciti dio znakova ili polja iz svake linije teksta. Opcija **-c** služi za rad sa znakovima.

Sljedećim primjerom prikazano je prvih 5 linija datoteke **/etc/passwd**.

```
$ head -5 /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
```

Sljedeća naredba prikazat će od petog do desetog znaka svake linije, te od petnaestog do kraja linije. Datoteka **/etc/passwd** je dugačka, zbog toga se rabi `| head -5` za prikaz prvih pet linija datoteke.

```
$ cut -c5-10,15- /etc/passwd | head -5
:x:0:0t:/root:/bin/bash
on:x:1aemon:/usr/sbin:/bin/sh
x:2:2:/bin:/bin/sh
x:3:3:/dev:/bin/sh
:x:4:6:sync:/bin:/bin/sync
```

Korisne opcije naredbe `cut` su **-d** i **-f**. Opcijom **-d** odrediti će se razdvojnik (*delimiter*), a opcijom **-f** polja koja se žele prikazati. Razdvojnik u datoteci **/etc/passwd** je znak **:** (dvotočka), a treba prikazati samo prvo i sedmo polje (korisnička oznaka i ljuska kojom se taj korisnik koristi). U prikazu nije potrebno prikazati sve linije, dovoljno je prvih pet (naredba `head`).

```
$ cut -d: -f1,7 /etc/passwd | head -5
root:/bin/bash
daemon:/bin/sh
bin:/bin/sh
sys:/bin/sh
sync:/bin/sync
```

Naredbom `cat` sadržaj se više tekstnih datoteka može spojiti u jednu tako da se sadržaji nižu jedan ispod drugog:

```
$ cat tekst1
1 jedan
2 dva
3 tri
$ cat tekst2
1 JEDAN
2 DVA
```

```
3 TRI
$ cat tekst1 tekst2
1 jedan
2 dva
3 tri
1 JEDAN
2 DVA
3 TRI
```

Naredba `paste` služi za spajanje sadržaja datoteka jednog pored drugog:

```
$ paste tekst1 tekst2
1 jedan 1 JEDAN
2 dva 2 DVA
3 tri 3 TRI
```

Naredbom `join` mogu se spojiti sadržaji datoteka prema određenom polju. Slijedi primjer:

```
$ join -j1 -j1 tekst1 tekst2
1 jedan JEDAN
2 dva DVA
3 tri TRI
```

5.3.2. Naredbe fmt i tr

Standardna veličina terminala je 80 x 25 znakova. Katkada tekst treba formatirati tako da stane 75 znakova po retku. Slijedi primjer neformatiranog teksta:

```
$ cat tekst.txt
Lorem ipsum dolor sit amet, consectetur adipiscing
elit. Suspendisse imperdiet felis convallis
lacus vulputate mollis. Mauris in erat eu nisl
lobortis pellentesque. Morbi vitae iaculis dolor. Curabitur eget diam
diam. Curabitur enim libero, fringilla in dapibus sit amet, scelerisque
quis sem. Morbi arcu odio, interdum et
sodales nec, gravida eget arcu.
```

Naredba `fmt` oblikovat će tekst na 75 znakova po retku.

```
$ fmt tekst.txt
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Suspendisse
imperdiet felis convallis lacus vulputate mollis. Mauris in erat eu nisl
lobortis pellentesque. Morbi vitae iaculis dolor. Curabitur eget diam
diam. Curabitur enim libero, fringilla in dapibus sit amet, scelerisque
quis sem. Morbi arcu odio, interdum et sodales nec, gravida eget arcu.
```

Naredba `tr` rabi se za translatiranje jednog skupa znakova u drugi. U sljedećem primjeru naredbom `tr` sva će se velika slova prebaciti u mala, a tekst će se formatirati naredbom `fmt`:

```
$ tr 'A-Z' 'a-z' < tekst.txt | fmt
lorem ipsum dolor sit amet, consectetur adipiscing elit. suspendisse
imperdiet felis convallis lacus vulputate mollis. mauris in erat eu nisl
lobortis pellentesque. morbi vitae iaculis dolor. curabitur eget diam
diam. curabitur enim libero, fringilla in dapibus sit amet, scelerisque
quis sem. morbi arcu odio, interdum et sodales nec, gravida eget arcu.
```

U već prikazanoj datoteci **/etc/passwd** naredba `tr` može zamijeniti sve dvotočke u razmake:

```
$ tr ':' ' ' < /etc/passwd | head -5
root x 0 0 root /root /bin/bash
daemon x 1 1 daemon /usr/sbin /bin/sh
bin x 2 2 bin /bin /bin/sh
sys x 3 3 sys /dev /bin/sh
sync x 4 65534 sync /bin /bin/sync
```

Vježba 4: Upravljanje tekstom

Napomena

Ovu vježbu potrebno je izvoditi s ovlastima korisnika *root*. U terminal je potrebno upisati:

`su -` pa lozinku korisnika *root* dodijeljenu prilikom instalacije.

Upravljanje tekstom

1. Koristeći se naredbom `cat` umetnite tekst u datoteku **poruka.txt**.

```
cat >> poruka.txt
linija 1
[Ctrl]+[D]
```

2. Napravite isto, ali se za izlaz iz naredbe `cat` umjesto naredbom [Ctrl]+[D] koristite ključnom riječi `STOP`.

```
cat >> poruka.txt << STOP
linija 2
STOP
```

3. Zatim naredbom `echo` dodajte tekst na kraj datoteke.

```
echo linija 3 >> poruka.txt
```

4. Naredbom `cat` provjerite sadržaj datoteke.

```
cat poruka.txt
```

5. Naredbom `tail` ispišite zadnje dvije linije datoteke **poruka.txt**.

```
tail -2 poruka.txt
```

6. Naredbom `head` ispišite prvu liniju datoteke **poruka.txt**.

```
head -1 poruka.txt
```

7. Naredbom `wc` izbrojite broj znakova, riječi i linija u datoteci **poruka.txt**.

```
wc poruka.txt
```

8. Naredbom `nl` ispišite datoteku **poruka.txt** te ispred svake linije dodajte redni broj.

```
nl –ba poruka.txt
```

9. Koristeći se naredbom `ifconfig`, `cut`, `grep` ispišite IP-adresu mrežnog sučelja **eth0**.

```
ifconfig eth0 | grep "inet addr" | cut -d: -f2 | cut -d" " -f1
```

10. Naredbom `tr` promijenite sve dvotočke u točka-zarez u datoteci **`/etc/passwd`**.

```
cat /etc/passwd | tr ':' ';'
```

11. Izradite datoteku **linije.txt** koja će izgledati ovako:

```
linija1
linija4
linija2
linija3
linija4
linija4
linija5
linija4
```

12. Koristeći se naredbom `split` razbijte datoteku **linije.txt** na više manjih datoteka od po jedne linije.

    ```
    split -l 1 linije.txt
    ```

13. U prošlom je zadatku datoteka **linije.txt** razbijena na 8 manjih datoteka, koje se zovu **xaa** do **xah**. Te datoteke treba ponovno spojiti u jednu koja se zove **linije2.txt**.

    ```
    cat xa* > linije2.txt
    ```

14. Koristeći se naredbom `sort` treba razvrstati datoteku **linije.txt** po abecedi.

    ```
    sort linije.txt
    ```

15. Primjetite da se pojavljuju uzastopne linije s brojem 4. Izbacite ih pomoću naredbe `uniq`.

    ```
    sort linije.txt | uniq
    ```

16. Koristeći se naredbom `fmt` formatirajte datoteku **linije.txt** tako da u jednom redu bude najviše 75 znakova po redu.

    ```
    fmt linije.txt
    ```

Prikaz binarnih datoteka

1. Koristeći se naredbom `od` prikažite binarnu datoteku **/bin/bash**.

   ```
   od /bin/bash
   ```

2. Koristeći se naredbom `hexdump` prikažite binarnu datoteku **/bin/bash**.

   ```
   hexdump /bin/bash
   ```

Pitanja za ponavljanje

1. U čemu je razlika između tekstnih i binarnih datoteka?

   _____

   _____

2. Koja naredba služi za formatiranje teksta na 75 znakova?

   _____

   _____

3. Kojom se naredbom tekstnu datoteka može razlomiti na više manjih?

   _____

   _____

6. Napredno upravljanje tekstom

Trajanje poglavlja:
55 min

Po završetku ovoga poglavlja moći ćete:

- rabiti tradicionalne i proširene regularne izraze
- koristiti se naredbama `grep`, `egrep` i `fgrep`
- rabiti *Stream Editor* – `sed`.

Ova cjelina obrađuje tradicionalne i proširene regularne izraze. Obradit će se i osnovne naredbe za pronalaženje i izmjenu sadržaja u tekstnim datotekama.

6.1. Regularni izrazi

6.1.1. Povijest

Regularni izrazi

U računarstvu i informatici, regularni je izraz (pravilni izraz, ispravni izraz, često i engleske skraćenice *regexp* ili *regex*, u množini *regexps*, *regexes* ili *regexen*) niz znakova koji opisuje druge nizove znakova (*string*) u skladu s određenim sintaksnim pravilima. Prvenstvena svrha regularnog izraza je opisivanje uzorka za pretraživanje nizova znakova.

Porijeklo regularnih izraza leži u teoriji automata i teoriji formalnih jezika, pri čemu su obje discipline teoretskog računarstva. Te discipline proučavaju modele računanja (automate) i načine opisa i klasifikacije formalnih jezika. Matematičar Stephen Kleene 1950-ih je opisao te modele koristeći se matematičkom notacijom zvanom ***regularni skupovi***. Ken Thompson je tu notaciju ugradio u uređivač QED, a zatim i u *Unix*ov uređivač *ed*, što je s vremenom dovelo do uporabe regularnih izraza u *grep*-u. Otad se regularni izrazi naširoko koriste u *Unix*u i pomoćnim programima temeljenim na *Unix*u kao što su *expr*, *awk*, *Emacs*, *vi*, *lex* i *Perl*.

Korištenje regularnih izraza u strukturiranim informacijskim standardima (za modeliranje dokumenata i baza podataka) pokazalo se vrlo važnim, počevši od 1960-ih te se proširujući 1980-ih konsolidacijom industrijskih standarda kao što je ISO SGML. Jezgra standarda jezika specifikacije strukture su regularni izrazi.

Regularnim se izrazima koriste mnogi uređivači teksta i pomoćni programi za pretragu i manipulaciju teksta ovisno o nekim uzorcima. Mnogi programski jezici podržavaju regularne izraze za manipulaciju nizom znakova (*strings*). Skup pomoćnih programa (uključujući uređivač *ed* i filter *grep*) koji se standardno distribuira s *Unix*ovim distribucijama znatno je doprinio promociji i popularizaciji koncepta regularnih izraza.

6.1.2. Osnovni koncepti

Regularni izraz, često zvan **uzorak** ili *pattern*, izraz je koji opisuje nizove znakova (*string*). Obično se rabe za davanje opisa nizova znakova, bez potrebe za nabrajanjem svih elemenata. Na primjer, niz znakova koji sadrži elemente *Handel*, *Händel* i *Haendel* može se opisati uzorkom **H(ä|ae?)ndel**. Kaže se da uzorak **sparuje** (*match*) svaki od navedena tri niza znakova.

Većina formalizama pruža ove operacije pri konstrukciji regularnih izraza:

Alternacija
Okomita crta razdvaja alternative. Na primjer, **gray|grey** se može skratiti u istovjetan izraz **gr(a|e)y** i pri tome spariti **gray** ili **grey**.

Grupiranje
Zagrade se rabe za definiranje područja djelovanja (*scope*) i prednosti operatora. Na primjer, **gray|grey** i **gr(a|e)y** su različiti uzorci, ali i jedan i drugi opisuju niz koji sadrži **gray** ili **grey**.

Kvantifikacija
Kvantifikator nakon znaka ili skupine njih određuje učestalost pojavljivanja izraza koji prethodi. Najčešće se rabe kvantifikatori **?**, *****, i **+**:

Znak	Opis	Primjer
?	Upitnik označava da se prethodni izraz pojavljuje **0 ili 1** puta.	**colou?r** sparuje i *color* i *colour*
*	Zvjezdica (*asterisk*) označava pojavljivanje **prethodnog izraza, 0,1 ili bilo koji veći broj** puta.	**go*gle** sparuje ggle, gogle, google, gooogle itd.
+	Znak plusa označava pojavljivanje prethodnog izraza **barem jednom**.	**go+gle** sparuje gogle, google, gooogle itd. (ali ne i ggle)

Ti se elementarni konstrukti mogu kombinirati u proizvoljno složene izraze, slično načinu na koji se mogu konstruirati aritmetički izrazi iz brojeva i operacija +, -, * i /.

Stoga su **H(ae?|ä)ndel** i **H(a|ae|ä)ndel** valjani uzorci, i štoviše, oba sparuju iste nizove znakova baš kao i primjer na početku lekcije. Uzorak **((great)*grand)?((fa|mo)ther)** sparuje bilo koji od nizova znakova koji u engleskom jeziku označavaju pretke *father*, *mother*, *grand father*, *grand mother*, *great grand father*, *great grand mother*, *great great grand father*, *great great grand mother*, *great great great grand father*, *great great great grand mother* i tako dalje.

6.1.3. Tradicionalni regularni izrazi na Unixu

„Osnovna“ sintaksa regularnih izraza na *Unix*u je prema POSIX-ovim definicijama danas zastarjela, iako se naširoko rabi radi unazadne kompatibilnosti. Većina pomoćnih programa na *Unix*u (npr. *grep* i *sed*) rabi tradicionalne regularne izraze, a prošireni se regularni izrazi koriste preko naredbenolinijskih argumenata.

Znak	Opis
.	Sparuje bilo koji znak samo jednom. Unutar [ ] ima svoje uobičajeno značenje (točka). Na primjer, "a.cd" sparuje "abcd", "a..d" sparuje "abcd".
[ ]	Sparuje jedan znak sadržan unutar uglatih zagrada. Na primjer, [abc] sparuje "a", "b", ili "c". [a-z] sparuje sva mala slova. Ta se dva stila mogu i miješati: [abcq-z] sparuje a, b, c, q, r, s, t, u, v, w, x, y, z, baš kao i [a-cq-z]. Znak '-' bi trebao biti shvaćen doslovno (kao literal) samo ako je prvi ili posljednji znak unutar zagrada: [abc-] ili [-abc]. Da bi se sparili znakovi '[' ili ']', najlakše je zatvarajuću uglatu zagradu postaviti prvu u obuhvaćajućim uglatim zagradama: [][ab] sparuje ']', '[', 'a' ili 'b'.
[^]	Sparuje jedan znak koji nije sadržan unutar uglatih zagrada. Na primjer, [^abc] sparuje bilo koji znak osim "a", "b", i "c". [^a-z] sparuje bilo koji znak

	koji nije malo slovo. Baš kao u prethodnim primjerima, ti se stilovi mogu miješati.
^	Sparuje početak linije (bilo koje linije, kad je primjenjen u višelinijskom načinu rada).
$	Sparuje kraj linije (bilo koje linije, kad je primjenjen u višelinijskom načinu rada).
()	Definira „označeni podizraz“. Što zagradama obuhvaćeni izraz sparuje, poslije može biti dohvaćeno za daljnju obradu, a način dohvata opisan je unosom za *n* (sljedeći redak). „Označeni podizraz“ je također „blok“. Ta osobina nije prisutna u nekim instancama regularnih izraza. U većini pomoćnih programa na *Unix*u (kao što su *sed* i *vi*), znak "\" (*backslash*) mora prethoditi otvorenim i zatvorenim zagradama.
\n	Pri čemu je *n* znamenka od 1 do 9 - sparuje *n*-ti spareni označeni podizraz. Taj konstrukt je teoretski **neregularan** i nije prihvaćen u proširenoj sintaksi regularnih izraza.
*	Izraz od jednog znaka nakon kojeg slijedi "*" sparuje nula ili više kopija sebe. Na primjer, "ab*c" sparuje "ac", "abc", "abbbc" itd. "[xyz]*" sparuje "", "x", "y", "zx", "zyx", i tako dalje.
+	Izraz od jednog znaka nakon kojeg slijedi "+" sparuje jednu ili više kopija izraza. Na primjer, "ab+c" sparuje "abc", "abbbc" itd. "[xyz]+" sparuje "x", "y", "zx", "zyx", i tako dalje.
{*x*,*y*}	Sparuje posljednji blok barem "x" i ne više od "y" puta. Na primjer, "a\{3,5}" sparuje "aaa", "aaaa" ili "aaaaa". Uočite da taj konstrukt nije prisutan u nekim instancama regularnih izraza.

Primjeri:

- ".at" sparuje bilo koji string od tri znaka poput *hat*, *cat* ili *bat.*
- "[hc]at" sparuje *hat* i *cat.*
- "[^b]at" sparuje sve sparene stringove iz regexa ".at" izuzev *bat.*
- "^[hc]at" sparuje *hat* i *cat* ali samo na početku linije.
- "[hc]at$" sparuje *hat* i *cat* ali samo na kraju linije.

6.1.4. Moderni (prošireni) regularni izrazi POSIX

Prošireni regularni izrazi POSIX slični su u sintaksi tradicionalnim regularnim izrazima na *Unix*u, osim nekih iznimki. Dodani su ovi metaznakovi:

Znak	Opis
+	Sparuje posljednji "blok" jedan ili više puta. Na primjer, "ba+" sparuje "ba", "baa", "baaa" i tako dalje.
?	Sparuje posljednji "blok" nula ili jedanput. Na primjer, "ba?" sparuje "b" ili "ba".
|	Operator izbora (ili unije skupova) sparuje ili izraz prije ili izraz poslije operatora. Na primjer, "abc|def" sparuje "abc" ili "def".

Znakovi *backslash* su odbačeni: \{...\} postaje {...} i ..... postaje (...).

Primjeri:

"[hc]+at" sparuje "hat", "cat", "hhat", "chat", "hcat", "ccchat" itd.
"[hc]?at" sparuje "hat", "cat" i "at".
"([cC]at)|([dD]og)" sparuje "cat", "Cat", "dog" i "Dog".

Ako se znakovi posebne namjene **(**, **)**, **[**, **]**, **.**, *****, **?**, **+**, **^** i **$** žele rabiti kao literal, ispred njih se stavlja znak \.

Primjeri:

"a\.(||)" sparuje string "a.)" ili "a.(".

6.1.5. Korisni linkovi

Više o regularnim izrazima:

http://en.wikipedia.org/wiki/Regular_expression

6.2. Pronalaženje sadržaja u datotekama

6.2.1. Naredba grep

Naredba `grep` služi za pretraživanje teksta prema zadanim obrascima. Ime naredbe nastalo je od prvih slova naredbi za uređivač teksta *ed*: ***g****lobal*, ***r****egular* ***e****xpression* i ***p****rint*.

Naredba `grep` pretražuje sadržaj datoteke ili standardni ulaz (STDIN) tražeći redove teksta koji odgovaraju zadanom obrascu koji može biti regularni izraz. Rezultat pretrage ispisuje se na standardni izlaz (STDOUT).

Sintaksa je naredbe `grep`:

```
grep [OPCIJE] UZORAK DATOTEKA
```

Uzorak koji se pretražuje može biti znak, riječ ili tradicionalni regularni izraz. Sljedeća naredba traži tekst *root* u datoteci /etc/passwd.

```
$ grep root /etc/passwd
root:x:0:0:root:/root:/bin/bash
```

Primjer je uporabe s te naredbe regularnim izrazom:

```
$ grep '^sy[ns]' /etc/passwd
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
```

Korisna opcija naredbe grep **-v**. Ona invertira izlaz, tj. prikazuje sve redove koji NE zadovoljavaju uzorak koji se pretražuje.
U sljedećem primjeru ispisat će se sve linije koje nisu prazne:

```
$ grep -v "^$" /etc/inittab
```

6.2.2. Naredbe egrep i fgrep

Naredbe `egrep` i `fgrep` slične su naredbi `grep`, uz male razlike.

Naredba `egrep` podržava proširene regularne izraze. Sve su opcije identične, samo se u uzorku mogu rabiti prošireni regularni izrazi.

Primjer je uporabe naredbe `egrep` s naprednim regularnim izrazom:

```
$ egrep '^sync|sys' /etc/passwd
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
```

Naredba `fgrep` uopće ne podržava regularne izraze pa se brže izvršava i služi za brzo pretraživanje riječi u datotekama. Zbog toga se naredba i zove `fgrep` što je kratica od *fast grep* (brzi *grep*).

6.3. Stream Editor – sed

6.3.1. Upotreba naredbe sed

Naredba `sed` (skraćeno od *stream editor*) je alat koji služi za raščlanjivanje i mijenjanje teksta pomoću regularnih izraza.

`sed` je linijski orijentiran alat za obradu teksta: učitava tekst, liniju po liniju s ulaza koji može biti tok (*stream*) ili datoteka, u unutrašnji međuspremnik. Učitavanjem linije započinje ciklus. U unutrašnjem međuspremniku `sed` primjenjuje jednu ili više operacija koje su definirane pomoću naredbi `sed` koje podržavaju regularne izraze. Svaka se linija nakon izvršavanja regularnog izraza ispisuje na standardni izlaz te započinje novi ciklus sljedećom linijom ulaza.

`sed` naredbe mogu se zadati iz naredbene linije (opcija **-e**) ili čitanjem iz datoteke (opcija **-f**).

Sintaksa je naredbe `sed` ovakva:

```
sed [opcije] 'naredbe' DATOTEKA
```

Najčešća je uporaba te naredbe zamjena teksta. Ako se na kraju `sed` naredbe stavi **g**, to znači da će se zamjena izvršiti na cijeloj liniji, a ne samo kod prvog pojavljivanja traženog izraza na koje `sed` naiđe u jednoj liniji. Ako se **g** izostavi, zamjena će se izvršiti samo kod prvog pojavljivanja izraza u jednoj liniji.

```
$ sed 's/regularniizraz/zamjena/g' ulaznadatoteka
```

Primjer je uporabe naredbe, pri čemu se početak linije koja započinje izrazom *root* mijenja u *tux*.

```
$ grep root /etc/passwd
root:x:0:0:root:/root:/bin/bash
$ grep root /etc/passwd | sed s/^root/tux/g
tux:x:0:0:root:/root:/bin/bash
```

U sljedećem primjeru biti će obrisane sve linije koje su zakomentirane (počinju znakom **#**):

```
$ sed '/^#/d' datoteka
```

6.3.2. Napredne mogućnosti naredbe sed

Naredba `sed` podržava da se više naredbi izvršava u jednom prolazu. Tome služi opcija **-e**.
Slijedi primjer datoteke koja će se mijenjati naredbom `sed`:

```
# ovo je pocetak datoteke

STARO
NOVO

# ovo je kraj datoteke
```

Slijedi primjer u kojem će se obrisati sve prazne linije i zamijenit će se sve riječi STARO s riječi NOVO:

```
$ sed -e '/^$/d' -e 's/STARO/NOVO/g' datoteka.txt
# ovo je pocetak datoteke
NOVO
NOVO
# ovo je kraj datoteke
```

Isto tako naredbe `sed` mogu biti zapisane u posebnu datoteku i pozivane opcijom **-f**:

```
$ cat sed.cmd
/^$/d
s/STARO/NOVO/g

$ sed -f sed.cmd datoteka.txt
# ovo je pocetak datoteke
NOVO
NOVO
# ovo je kraj datoteke
```

Vježba 5: Napredno upravljanje tekstom

grep, egrep, fgrep

1. Izradite datoteku **/tmp/datoteka.txt** koja sadrži linije:

```
danas je lijep,
i suncan dan.
sutrasnji dan
ce biti
kisovit.
# ove dvije linije
# su komentari
```

2. Koristeći se naredbom `grep` ispišite samo nezakomentirane linije. Nezakomentirane linije započinju znakom `#`.

```
grep -v ^# /tmp/datoteka.txt
```

3. Koristeći se naredbom `grep` ispišite samo linije koje završavaju zarezom (,).

```
grep -v ,$ /tmp/datoteka.txt
```

4. Nađite sve linije koje sadrže riječ **dan** (no ne riječ „danas“ – koristite opciju `-w` za traženje riječi)

```
grep -w dan /tmp/datoteka.txt
```

5. Nađite sve linije koje počinju slovom **s**.

```
grep ^s /tmp/datoteka.txt
```

6. Koristeći naredbu `egrep` nađite sve linije koje sadrže riječi **danas** i **biti**.

```
egrep 'danas|biti' /tmp/datoteka.txt
```

Regularni izrazi

1. U datoteku iz prošle vježbe dodajte linije:

```
dn
dan
daani
daaani
da+n
da*n
da?n
drani
darni
```

2. Istražite razlike rezultata koristeći `grep`, `egrep` i `fgrep`:

```
grep 'da+n' /tmp/datoteka.txt
grep 'da?n' /tmp/datoteka.txt
grep 'da.n' /tmp/datoteka.txt
grep 'daa*n' /tmp/datoteka.txt
grep 'da*r.' /tmp/datoteka.txt
```

Stream Editor

1. Koristeći se naredbom `sed` u prvoj liniji zamijenite „dan“ sa „sutra“.

   `sed s/dan/sutra/ /tmp/datoteka.txt`

2. Obrišite liniju u kojoj se pojavljuje riječ „suncan“.

   `sed /suncan/d /tmp/datoteka.txt`

3. U četvrtoj liniji zamijenite „ce biti“ s „nece biti“.

   `sed 's/ce biti/nece biti/' /tmp/datoteka.txt`

Pitanja za ponavljanje

1. Što sparuje $, a što ^ kod regularnih izraza?

   _____

   _____

2. U čemu je razlika između naredbi `egrep` i `fgrep`?

   _____

   _____

3. Čemu služi *Stream Editor* – sed?

   _____

   _____

7. Uređivač teksta vi

Trajanje poglavlja:
85 min

Po završetku ovoga poglavlja moći ćete:

- koristiti se uređivačem teksta vi
- prepoznati načine rada uređivača teksta vi
- kretati se po tekstu
- upravljati tekstom.

Ova cjelina obrađuje uređivač teksta vi. Obradit će se osnovno korištenje uređivača teksta vi, njegovi načini rada, kretanje po tekstu i upravljanje tekstom.

7.1. Uređivač teksta vi

7.1.1. Uređivači teksta

Za izradu novih datoteka i održavanje postojećih, koriste se različita programska pomagala među kojima uređivačima teksta (text editor) pripada najznačajnije mjesto. Uređivači se prvenstveno rabe za izradu i održavanje datoteka koje sadrže tekst (ASCII-znakove). U *Unix*ovoj i u *Linux*ovoj okolini postoji nekoliko uređivača teksta:

ed - standardni linijski uređivač koji je vrlo jednostavan i može se koristiti na bilo kojem terminalu

ex - poboljšana inačica uređivača teksta *ed*

vi (*visual*) - zaslonski uređivač teksta koji radi sa stranicama teksta (stranica je obično veličine zaslona terminala)

sed (*stream editor*) - omogućuje ispravke nad nizom podataka (redaka teksta) jedne datoteke.

Uređivač teksta *vi* ugodniji je i brži za rad od linijskih editora, ali zahtijeva složenije terminale (pozicioniranje pokazivača, brisanje zaslona i dr.). Budući da su takvi terminali danas opće prihvaćeni (VT100, VT200), a podržani su i u svim grafičkim okruženjima (X-terminali), u nastavku je detaljnije obrađen zaslonski **uređivač teksta *vi*** koji se sigurno može naći u svakoj *Linux*ovoj inačici, a dostupan je i za druge operacijske sustave.

Na samom početku treba napomenuti da ***vi*** može stvoriti odbojnost kod korisnika. Razmjerno je kompliciran za upotrebu, jer ima tri načina rada u kojima se funkcije znakova generiranih s tipkovnice drastično razlikuju. Obično ne rabi kontrolne tipke kao što su [PageUp] i [PageDown], kao ni funkcijske tipke, tako da se naredbe zadaju sa standardnih tipki i njihovom kombinacijom s tipkom **[Ctrl]**.

Ne posjeduje izbornike na koje su se korisnici navikli kod uređivača teksta koji su, uvjetno rečeno, *user friendly*. Međutim, treba imati u vidu da je osnova uređivača teksta ***vi*** definirana početkom sedamdesetih, istovremeno s početkom razvoja Unixa. Još tada je postavljen cilj da ***vi*** funkcionira na raznim tipovima terminala od kojih većina nije imala ni preveliki ni premoćan skup kontrolnih sekvenci, kao ni standardiziran izgled tipkovnice. Naravno, u tome se uspjelo, ali je cijena plaćena upravo činjenicama koje su pobrojane kao nedostaci ovog uređivača teksta.

Nakon boljeg upoznavanja s uređivačem teksta ***vi***, svakom će korisniku biti jasno da naredbe za globalnu zamjenu i pretraživanje te rad s međuspremnicima koje on nudi, predstavljaju glavni nedostatak spomenutih korisniku pristupačnijih uređivača teksta.

7.1.2. Načini rada uređivača teksta vi

Zaslonski uređivač teksta ***vi*** može se naći u jednom od tri načina rada:

zapovjedni način rada (*command mode*) - svi znakovi otkucani na tipkovnici ponašaju se kao naredbe

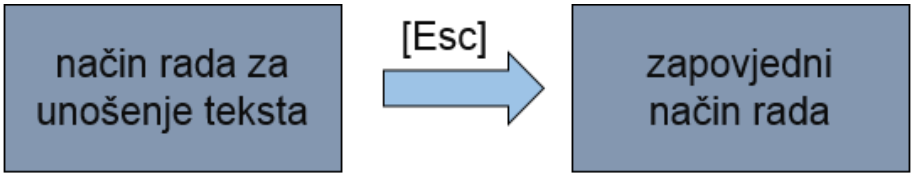
način rada za unošenje teksta (*insert mode*) - služi za unos teksta, tipke imaju normalno značenje

način rada zadnje linije (*last line mode)* - služi za unos dužih naredbi.

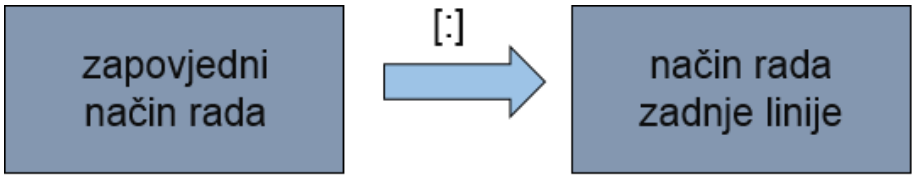
Nakon pokretanja, uređivač teksta ulazi u **zapovjedni način rada**.

Prelazak u način rada za unošenje teksta ili u način rada zadnje linije moguć je jedino iz zapovjednog načina.

Prelazak iz zapovjednog načina rada u način rada za unošenje teksta ostvaruje se većim brojem naredbi za dodavanje teksta (biti će pojašnjene u nastavku), ali se napuštanje načina rada za unošenje teksta i povratak u zapovjedni uvijek obavlja pritiskom na tipku **[Esc]**.



Prelazak u način rada zadnje linije moguć je jedino naredbom : (dvotočka).



Iz načina rada zadnje linije izlazi se unošenjem željene naredbe i njezinim izvršavanjem pritiskom na tipku **[Enter]** ili tipkom **[Esc]** kada se način rada zadnje linije odmah napušta.

7.1.3. Kretanje po tekstu

Za kretanje po tekstu u uređivaču teksta *vi* potrebno je najprije, pritiskom na tipku **[Esc],** prijeći u zapovjedni način rada, a zatim se koristi tipkama **[H]**, **[J]**, **[K]** i **[L]**.

Tipka **[H]** pomiče pokazivač jedan znak ulijevo, tipka **[J]** jedan znak dolje, tipka **[K]** jedan znak gore a tipka **[L]** jedan znak udesno.

U zapovjednom načinu rada naredbe su obično jedno slovo. Npr. naredbom **j** prelazi se na sljedeći red. Ako se želi izvršiti više istovjetnih naredbi, dovoljno je napisati broj ponavljanja i naredbu. Na primjer, 10j će pomaknuti pokazivač10 linija prema dolje.

Još su neke korisne naredbe za kretanje:

0 ili **^** - na početak reda

$ - na kraj reda

G - na kraj datoteke

***n*G** - u red broj *n*

w - na sljedeću riječ

b - na početak riječi

e - na kraj riječi

(- na početak rečenice

) - na kraj rečenice

{ - na početak odlomka

} - na kraj odlomka.

7.1.4. Naredbe za ulazak u način rada za unošenje teksta

Iz zapovjednog se načina rada u način rada za unošenje teksta može prijeći pritiskom na odgovarajuću tipku na tipkovnici:

i - unos teksta na mjestu pokazivača

a - unos teksta jedno mjesto iza pokazivača

I - unos teksta na početku reda

A - unos teksta na kraju reda

o - unos teksta jedan red ispod

O - unos teksta jedan red iznad.

Jednom kad se uđe u način rada za unošenje teksta, sve što se upisuje, unosit će se kao tekst u datoteku. Iz načina rada za unošenje teksta izlazi se pritiskom na tipku **[Esc]**.

7.1.5. Brisanje tekst

Ako se želi obrisati neki znak ili linija teksta, pritisne se tipka **[Esc]** za prijelaz u zapovjedni način rada i rabi se neka od ovih naredbi:

x - briše znak na mjestu pokazivača

X - briše znak na jednom mjestu ispred pokazivača

dd - briše cijelu liniju teksta

D - briše sve u liniji iza pokazivača

dw - briše od pokazivača do kraja riječi u kojoj je pokazivač

d$ - briše od pokazivača do kraja reda u kojoj je pokazivač

d) - briše od pokazivača do kraja rečenice

dG - briše od pokazivača do kraja teksta.

Ako se ne želi obrisati samo jedan znak ili samo jedna linija teksta, ispred ovih naredbi treba upisati broj (količinu) znakova ili linija teksta koje će se izbrisati. Da bi se, na primjer, obrisalo pet znakova počevši od mjesta na kojem se pokazivač trenutačno nalazi, treba prijeći u zapovjedni način rada (tipka **[Esc]**) i zatim utipkati **5x**. Ako želite obrisati liniju u kojoj se nalazite i liniju ispod nje (dakle dvije linije), treba prijeći u zapovjedni način rada i zatim utipkati **2dd**.

7.1.6. Pretraživanje teksta

Naredbe su za traženje određenog znaka u retku:

f<znak> - pomiče pokazivač do prvog (ako je zadan broj *n* ispred naredbe) do *n*-tog pojavljivanja znaka danog uz naredbu; pretraživanje je desno od pokazivača

F<znak> - isto kao i prethodna naredba, ali je pretraživanje lijevo od mjesta pokazivača

t<znak> - pomiče pokazivač udesno i zaustavlja se na znaku ispred zadanog znaka

T<znak> - pomiče pokazivač ulijevo i zaustavlja se na znaku iza zadanog znaka

; - ponavlja zadnju naredbu iz skupine t, F, t, T

, - isto kao i prethodna naredba, ali u obratnom smjeru od originalne naredbe.

Ako zadani znak nije pronađen u retku, pokazivač ostaje na mjestu prije početka pretraživanja, a iz terminala se čuje zvučni signal.

Ako se traži određeni niz znakova (*string),* tada se rabe ove naredbe:

/niz<ENTER> - pretražuje se tekst od mjesta pokazivača udesno dok se ne pronađe zadani niz znakova; tekst se pretražuje do kraja i zatim od početka do mjesta pokazivača prije zadavanja naredbe

?niz<ENTER> - radi isto što i prethodna naredba, ali u obratnom smjeru (od mjesta pokazivača ulijevo)

n - ponavlja zadnju / ili ? naredbu

N - kao prethodna naredba, ali uz obratni smjer pretraživanja.

7.1.7. Promjene dijelova teksta

Naredbe su za promjenu teksta:

s - zamjenjuje znak ispod pokazivača novim tekstom, akcija se završava pritiskom na tipku **[Esc]**

r - zamjenjuje samo znak ispod pokazivača

R - više znakova ispod pokazivača, akcija se završava pritiskom na tipku [Esc]

cw - zamjenjuje tekst od pokazivača do kraja riječi novim tekstom.

U načinu rada zadnje linije moguće je mijenjati tekst upotrebom regularnih izraza.

U način rada zadnje linije može se ući pritiskom na tipku **[:]** iz zapovjednog načina rada.

Ako se u cijelom tekstu želi promijeniti niz 'stari' u niz 'novi', dovoljno je utipkati:

```
:%s/stari/novi/g
```

Ako se želi svakoj liniji na početak dodati riječ 'test', dovoljno je utipkati:

```
:%s/^/test/g
```

A ako se želi svakoj liniji na kraj dodati riječ 'test', dovoljno je utipkati:

```
:%s/$/test/g
```

Prilikom promjene dijelova teksta mogu se koristiti svi dosad obrađeni regularni izrazi.

7.1.8. Poništavanje zadnje promjene u tekstu

Prednost je uređivača teksta *vi* mogućnost poništavanja zadnje promjene teksta (korisno ako je nešto promijenjeno greškom).
Naredbe su ove:

u (*undo*) - vraća sadržaj teksta kakav je bio prije zadnje promjene

U (*undo line*) - vraća sadržaj retka teksta kakav je bio prije svih promjena nad njim; djeluje samo na redak u kojem se nalazi pokazivač

Ctrl + r (*redo*) - obrnuto od *undo*

:e! - odbacuje sve promjene koje su bile rađene nad datotekom i ponovno je čita s diska no tom se naredbom treba koristiti s oprezom, jer se ne može poništiti.

7.1.9. Kopiranje teksta

Kopiranje teksta obavlja se u nekoliko koraka:

1. korak - kopiranje određenog dijela teksta u pomoćnu memoriju
2. korak - pomicanje pokazivača na mjesto u tekstu kamo želimo staviti kopiju
3. korak - kopiranje teksta iz pomoćne memorije na mjesto pokazivača.

1. korak

Naredbe su za kopiranje u pomoćnu memoriju:

y - kopiranje u pomoćnu memoriju (bez imena); način zadavanja naredbe je isti kao i kod naredbe za brisanje teksta. Razlika između naredba **d** i **y** je u tome što **d** briše tekst i sprema ga u pomoćnu memoriju, a **y** ga ne briše, ali ga sprema u pomoćnu memoriju.

"<slovo>y - isto isto kao i prethodna naredba, ali pomoćna memorija ima ime koje se sastoji od jednog slova abecede; tako se može kopirati više dijelova teksta u različite memorijske spremnike (maksimalno 26). Znak navodnika (") označava imenovanje spremnika, tj. pomoćne memorije.

yy - isto kao i prethodne naredbe ali se akcija odnosi na cijeli redak (isto radi i naredba *Y*).

Na primjer:

2yy kopira sadržaj retka u kojem se nalazi pokazivač i sadržaj sljedećeg retka u pomoćnu memoriju.
"aY ili **"ayy** kopiraju sadržaj retka u kojem se nalazi pokazivač u pomoćnu memoriju pod imenom *a*.

2. korak

Naredbe za pomicanje pokazivača na novo mjesto objašnjene su u prethodnom tekstu.

3. korak

Naredbe za vraćanje teksta iz pomoćne memorije na mjesto pokazivača:

p (*put*) - vraća sadržaj teksta iz pomoćne memorije bez imena na mjesto desno od trenutačnog mjesta pokazivača; sve drugo vrijedi kao i za naredbu *y*

"<slovo>p - vraćanje teksta iz pomoćne memorije s imenom

P - kao i prethodna naredba, ali lijevo od trenutačnog mjesta pokazivača

Pomicanje teksta slično je kopiranju teksta. Razlika je jedino u tome što se u prvom koraku kopiranja teksta koristi naredba y za kopiranje u pomoćnu memoriju, a kod pomicanja se teksta, umjesto naredbe y rabi naredba d za brisanje teksta. Sve drugo napisano za prethodne naredbe vrijedi i u ovom slučaju.

7.1.10. Spremanje promjena i izlazak

Ako se žele spremiti promjene, izaći ili izaći bez spremanja promjena, potrebno je ponovno prijeći u zapovjedni način rada pritiskom na tipku **[Esc]** te se zatim koristiti nekom od ovih naredbi:

:w - spremanje promjene

:q - izlazak iz uređivača teksta *vi,* ako nije bilo promjena od zadnjeg spremanja; ako je promjena bilo, program javlja grešku i ne izađe iz trenutačnog načina rada

:x - izlazak iz uređivača teksta *vi* i spremanje promjena, ako ih je bilo

:q! - izlazak iz uređivača teksta *vi* bez spremanja promjena

:wq - spremanje promjene i zatim izlazi iz *vi*-ja

:w ime_datoteke - spremanje promjene u datoteku s imenom *ime_datoteke*

:wq ime_datoteke - spremanje promjene u datoteku s imenom *ime_datoteke* i izlazak iz uređivača teksta *vi*

:15,24w ime_datoteke - spremanje od 15 do 24 linije u datoteku naziva *ime_datoteke*

ZZ - isto kao **:x**

:e - isto kao **:x**

:exit - isto kao **:x**

:quit - isto kao **:q**.

7.1.11. Dodatne naredbe

Kako je već pokazano, uređivač teksta *vi* ima velik broj naredbi.

U svakodnevnom radu korisne su i ove naredbe:

:e ime_datoteke - učitava datoteku s imenom *ime_datoteke,* ako takva postoji

:r ime_datoteke - učitava datoteku s imenom *ime_datoteke*, ako takva postoji, i ubacuje ju u trenutačno otvorenu, na mjestu gdje se nalazi pokazivač

:!<naredba> - pokreće naredbu iz naredbene linije i ispisuje njezin izlaz na zaslon

:r!<naredba> - pokreće naredbu iz naredbene linije i njezin izlaz stavlja u tekst na mjestu pokazivača.

7.1.12. Dodatni sadržaj

Interaktivne upute za rad u uređivaču teksta *vi*(m)
Za dodatno vježbanje rada u uređivaču teksta *vi* dostupne su i interaktivne upute.

VIM Adventures
Zabavna igra inspirirana naredbama uređivača teksta VIM.

Vježba 6: Uređivač teksta vi

1. U tekućem direktoriju izradite datoteku **vjezba.txt** rabeći ***vi*** te prepišite prvi odlomak teksta ovog poglavlja. U način rada za unašanje teksta može se ući pritiskom na tipku **[i]**.

   ```
   vi vjezba.txt
   ```

2. Isprobajte naredbe za kretanje po tekstu: h, j, k, l, ^, $, G, w, b, e, (,), { i }.

3. Isprobajte naredbe za ulazak u način rada za unošenje teksta: i, a, I, A, o i O. Svakog puta kad uđete u način rada za unošenje teksta, izađite iz njega tipkom [Esc] te uđite u novi način rada. U čemu je razlika između tih naredbi?

   i _____

   a _____

   I _____

   A _____

   o _____

   O _____

4. Smjestite pokazivač u sredinu teksta pa isprobajte naredbe za brisanje teksta: x, X, dd, D, dw, d$, d) i dG. U čemu su razlike?

   x _____

   X _____

   dd _____

   D _____

   dw _____

   d$ _____

   dG _____

5. Naredbom `u` vratite sadržaj teksta prije zadnjih dviju promjena. Pokušajte više puta pritisnuti naredbu `u` i vratite promjenu naredbom [Ctrl]+[r].

6. Koristeći se naredbama uređivača teksta iskopirajte taj odlomak teksta tri puta, jedan ispod drugog.

7. Rabeći *vi* promijenite svaku liniju tako da počinje `s BEGIN` i završava `s END`.

   ```
   :% s/^/BEGIN/g
   :% s/$/END/g
   ```

8. Istražite u čemu su razlike između ovih naredbi izlaza iz uređivača teksta vi: :x, ZZ, :quit, :wq, :q!. Koja od tih naredbi snima izmjene, a koja ne?

   :x _____

   ZZ _____

   :quit _____

   :wg _____

   :q! _____

Pitanja za ponavljanje

1. Koji su načini rada uređivača teksta *vi*?

   _____

   _____

2. Nakon pokretanja, u koji način rada ulazi uređivač teksta *vi*?

   _____

   _____

3. Koje tipke služe za kretanje po tekstu?

   _____

   _____

4. Što radi naredba 2yy uređivača teksta *vi*?

   _____

   _____

8. Upravljanje uređajima u direktoriju /dev

Trajanje poglavlja:
60 min

Po završetku ovoga poglavlja moći ćete:

- upravljati diskovima i particijama
- prepoznati diskove ili CD/DVD uređaje ovisno o nazivu u direktoriju /dev
- razlikovati primarne i proširene particije i način na koji mogu biti organizirane
- koristiti se alatima za particioniranje
- raditi s alatima za učitavanje operacijskog sustava.

Ova cjelina obrađuje strukturu diskova i particija. Naučiti ćemo koristiti alate za particioniranje diskova. Na kraju cjeline obradit će se najčešći programi za učitavanje operacijskog sustava (LILO i GRUB).

8.1. Diskovi i particije

8.1.1. Diskovi

Za razliku od operacijskog sustava *Microsoft Windows* koji sve uređaje za pohranu podataka imenuje velikim slovom i dvotočkom (npr. C:, D:, E: itd.) i svaki od njih ima svoje zasebno stablo direktorija, *Linux* drugačije pristupa radu s diskovima.

Kad se govori o diskovima, prvenstveno se misli na tvrde diskove, CD- i DVD-medije, USB-*stick*ove pa čak i zastarjele diskete koje se danas rijetko rabe. Osnovna je namjena tih uređaja fizičko spremanje podataka (informacija u obliku programa, vaših tekstova, slika i drugog) na površinu nekog medija:

tvrdi diskovi i diskete podatke spremaju na površinu magnetnog medija

CD/DVD - diskovi podatke spremaju na optički čitljivu površinu (pomoću lasera prepoznavaju se udubine u mediju, tako da se raspoznaje stanje 0 i 1)

USB *stick*ovi spremaju podatke u tzv. memorijske čipove *flash*.

Svim je diskovima zajedničko to što se podaci na njih spremaju u obliku datoteka i direktorija, preko nekog datotečnog sustava, i što se ti podaci ne brišu nakon isključivanja uređaja iz izvora napajanja.

U današnje vrijeme za svakodnevnu pohranu podataka u računalu rabe se isključivo tvrdi diskovi kapaciteta od nekoliko stotina GB do nekoliko TB. Za razmjenu podataka između računala koriste se CD- i DVD- diskovi te USB-*stick*ovi koji zbog toga pripadaju skupini izmjenjivih diskova, jer ih tijekom rada možemo izmjenjivati u CD/DVD-čitaču, odnosno stavljati/vaditi iz USB-sabirnice računala ovisno o tome koji nam podaci na njima trebaju.

Radi povećanja kapaciteta i pouzdanosti čuvanja podataka diskove možemo organizirati i u tzv. **RAID-polja** gdje cijelu seriju diskova proglašavamo jednim logičkim uređajem.

Tvrdi su diskovi na operacijskom sustavu *Linux* prikazani kao datoteke u direktoriju **/dev** pri čemu su **IDE-diskovi** prikazani kao datoteke koje počinju slovima **hd**, a **diskovi SCSI ili SATA** počinju slovima **sd**. Budući da se u jednom računalu može nalaziti više tvrdih diskova, operacijski sustav dodjeljuje još jedno slovo imenu direktorija tvrdog diska, počevši od a do z i ovisno o broju diskova.

Oznaka u direktoriju /dev	Fizički blok uređaj
/dev/hda	Primarni *master* IDE disk
/dev/hdb	Primarni *slave* IDE disk
/dev/hdc	Sekundarni *master* IDE disk
/dev/hdd	Sekundarni *slave* IDE disk
/dev/sda	Prvi SCSI ili SATA disk
/dev/sdb	Drugi SCSI ili SATA disk
/dev/sdc	Treći SCSI ili SATA disk

CD/DVD-uređaji počinju slovima **sr** i rednim brojem uređaja u računalu. Npr. ako postoje dva DVD-uređaja, prvi će biti **/dev/sr0**, a drugi **/dev/sr1**.

8.1.2. Particije

Particije su vezane uz tvrde diskove, a zapravo se mogu predočiti kao područja na nekom tvrdom disku (fizičkom disku), koja se opet ponašaju kao disk (logički disk). Tako se može postići privid da na jednom disku imamo više diskova, ali manjeg kapaciteta.

Particijama se koristimo:

ako želimo instalirati više od jednog operacijskog sustava; nemoguće je instalirati više od jednog operacijskog sustava po jednoj particiji.

ako operacijski sustav treba više od jedne particije za svoj uredan rad

ako se disk želi dodatno podijeliti za različite namjene

ako se na istom fizičkom disku želi rabiti više od jednog datotečnog sustava

Kod operacijskog sustava *Linux* postoje barem dvije particije: jedna **za operacijski sustav** i druga **za tzv. *swap***, odnosno privremenu radnu memoriju kada ponestane one u računalu (RAM-a).

Svaki tvrdi disk mora imati **barem jednu particiju**, što konkretno znači da se baš svaki tvrdi disk mora particionirati, jer je to uvjet da se na njega postavi neki datotečni sustav.

Kod particija treba razlikovati **primarne** (*primary*) i **proširene** (*extended*) particije:

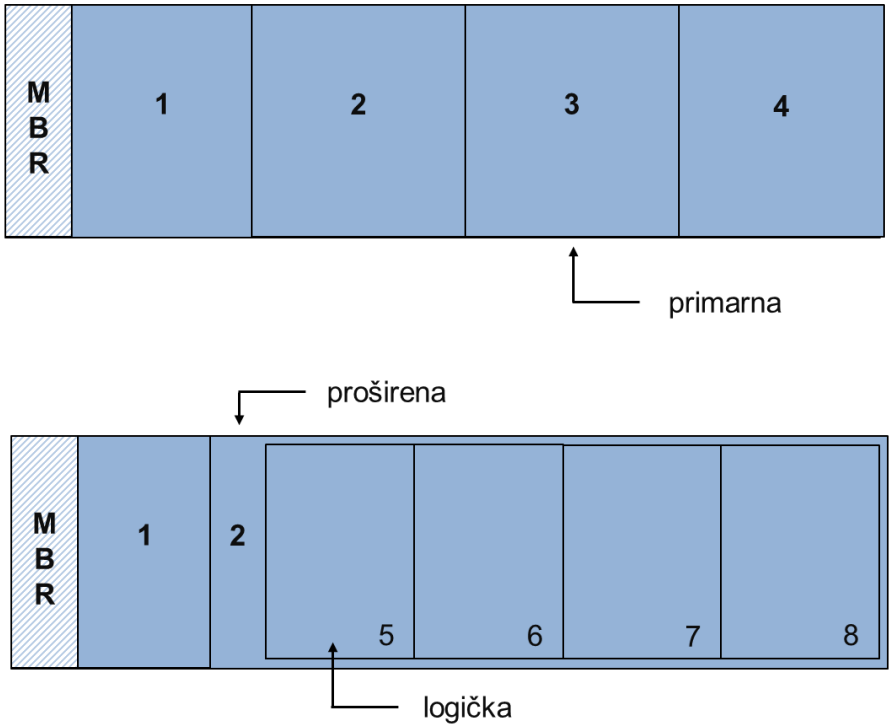
Primarna particija je nositelj datotečnog sustava. Zbog ograničenja u BIOS-u računala, na jedan fizički tvrdi disk mogu se postaviti **najviše četiri** primarne particije.

Proširena particija je nositelj (okvir) drugih primarnih particija. Na jedan se tvrdi disk može staviti najviše tri primarne particije i jedna proširena (*extended)*, koja u sebi može imati više **logičkih** particija.

Particije se također prikazuju u direktoriju **/dev**, tako da se na ime diska doda broj particije. Brojevi od 1 do 4 rezervirani su za primarne particije, a brojevi su od 5 na više za logičke:

Uređaj	Opis
/dev/hda1	Prva primarna particija na primarnom *master* IDE disku.
/dev/hda2	Druga primarna particija na primarnom *master* IDE disku.
/dev/sdc3	Treća primarna particija na SCSI ili SATA disku.
/dev/sdc6	Logička particija na SCSI ili SATA disku.

Na sljedećoj su slici prikazana dva diska. Na prvom su disku napravljene četiri primarne particije (1, 2, 3 i 4), a na drugom disku jedna primarna particija (1), jedna proširena (2) te unutar jedne proširene particije još četiri logičke particije (5, 6, 7 i 8).



MBR (*Master Boot Record*) je prostor na početku diska gdje se nalazi program za pokretanje operacijskog sustava.

8.2. Alati za particioniranje

8.2.1. Alati za particioniranje prije instalacije

Operacijski sustav *Linux* često se instalira na računalo s već instaliranim operacijskim sustavom *Microsoft Windows*. Ako postoji samo jedna particija preko cijelog diska **C:**, tada je treba smanjiti. Ako već postoji jedna particija (npr. **D:**, koja je prazna), ona se može obrisati i pripremiti za instalaciju *Linux*a.

Akcije brisanja i smanjivanja particije mogu se napraviti alatima kao što su:

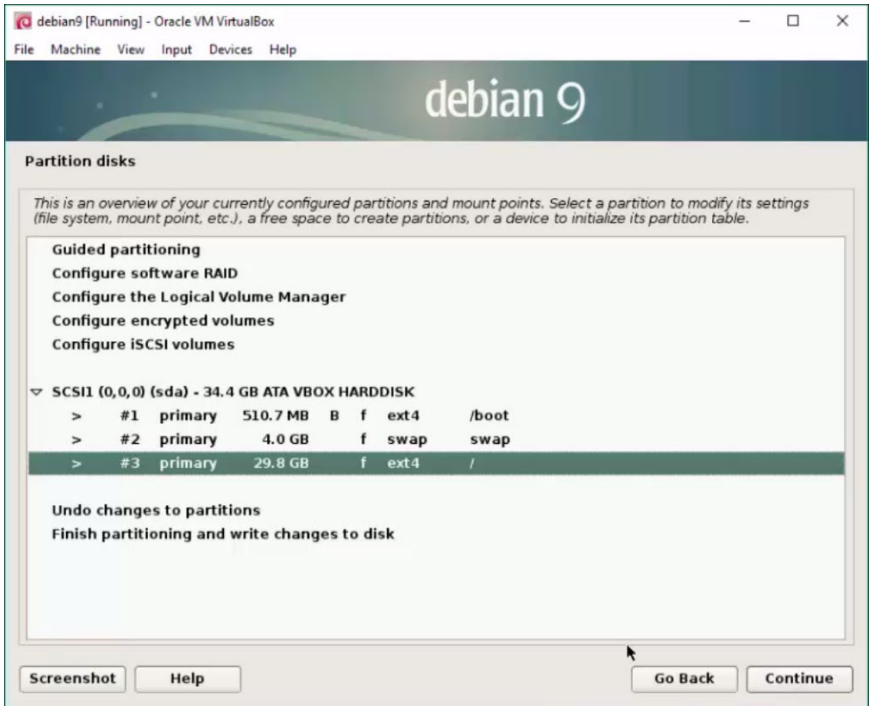
fips - jednostavan alat koji može smanjiti datotečne sustave FAT16 i FAT32

PartitionMagic - napredniji alat koji zna raditi sa svim drugim tipovima particija, kao što su NTFS, ext2, ext3, itd.

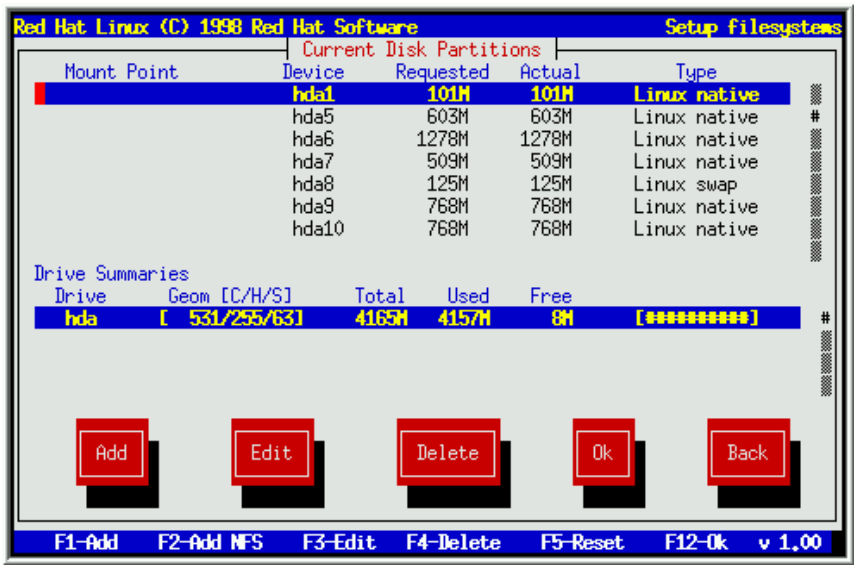
8.2.2. Alati za particioniranje tijekom instalacije

Diskovi se mogu reparticionirati i tijekom instalacije. Svaka distribucija ima svoju aplikaciju za particioniranje diskova tijekom instalacije. U poglavlju 2.2.2.3 Particioniranje diskova prikazano je particioniranje diskova *pomoću Debian Installera*, aplikacije koja korisnika vodi kroz instalaciju *Debiana*.

Izgled osnovnog ekrana alata za particioniranje diskova prikazan je na sljedećoj slici.



U drugim je distribucijama to moguće kroz ***DiskDruid*** (distribucije *CentOS, Red Hat*) ili ***diskdrake*** (*Mandrake, Mandriva*). *DiskDruid* je prikazan na sljedećoj slici.



8.2.3. Alati za particioniranje poslije instalacije

Jednom instalirana distrubucija *Linux*a dolazi s nekoliko alata za particioniranje diskova.

Najčešći su alati:

fdisk - najraširenij i najčešće korišten alat, podržava samo particijsku shemu MBR (*Master Boot Record*) koja dopušta particije do 2 TB

parted - nudi više mogućnosti od fdisk-a kao što je promjena veličine particije i podržava GPT (GUID *Partition Table),* koji dopušta particije do 9.4 ZB (ziliona bajtova, ili 10^{21}).

Te se naredbe moraju pokretati pod administratorskim ovlastima, tj. pod ovlastima korisnika *root*.

Obje naredbe imaju opciju **-l** koja prikazuje trenutačni raspored particija po diskovima.

Slijedi prikaz naredbom `fdisk`:

```
# fdisk -l
Disk /dev/sda: 21.5 GB, 21474836480 bytes
255 heads, 63 sectors/track, 2610 cylinders, total 41943040 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00020e94
   Device Boot Start End Blocks Id System
/dev/sda1 * 2048 40136703 20067328 83 Linux
/dev/sda2 40138750 41940991 901121 5 Extended
/dev/sda5 40138752 41940991 901120 82 Linux swap / Solaris
```

Zatim naredbom `parted`:

```
# parted -l
Model: VMware Virtual disk (scsi)
Disk /dev/sda: 21.5GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos

Number Start End Size Type File system Flags
 1 1049kB 20.5GB 20.5GB primary ext4 boot
 2 20.6GB 21.5GB 923MB extended
 5 20.6GB 21.5GB 923MB logical linux-swap(v1)
```

Obje naredbe daju isti rezultat: prikazuju tri particije od kojih je jedna primarna, a druga proširena unutar koje se nalazi jedna logička particija.

Stvaranje particije sastoji se od ovih koraka:

pokretanje naredbe `fdisk`

izrada nove particije (primarne ili proširene)

upisivanje početnog sektora

upisivanje završnog sektora

postavljanje vrste particije (Linux, swap, RAID...)

zapisivanje postavki.

8.3. Programi za učitavanje operacijskog sustava

8.3.1. GRUB

Punim nazivom *GRand Unified Bootloader, GRUB* je prvi program koji se pokreće s tvrdog diska nakon što mu BIOS prepusti kontrolu učitavanja operacijskog sustava. Izravno je zadužen za učitavanje jezgre operacijskog sustava, koja zatim učitava ostatak operacijskog sustava.

Taj je program trenutačno najrašireniji program za učitavanje operacijskog sustava u svijetu *Linuxa*, no nije i jedini. Naime, postoji i LILO- *bootloader* koji se i dalje koristi, ali manje.

Na zadnjoj verziji *Debian*a u upotrebi je verzija ***GRUB 2*** tog programa. Značajna su poboljšanja u odnosu na *GRUB*:

podrška za skripte

modularnost

mogućnost "spašavanja" (r*escue mod*)

teme

grafički izborni *boot* i poboljšani *splash*

pokretanje sustava sa slike LiveCD ISO koja se nalazi na čvrstom disku

nova struktura konfiguracijskih datoteka

podrška za ne-x86 platforme (npr. PowerPC)

univerzalna podrška za UUID-ove (*Universal Unique IDentifier* - jedinstveni identifikacijski kôd koji ima svaki uređaj za pohranu podataka i tako ne može doći do njihove zamjene).

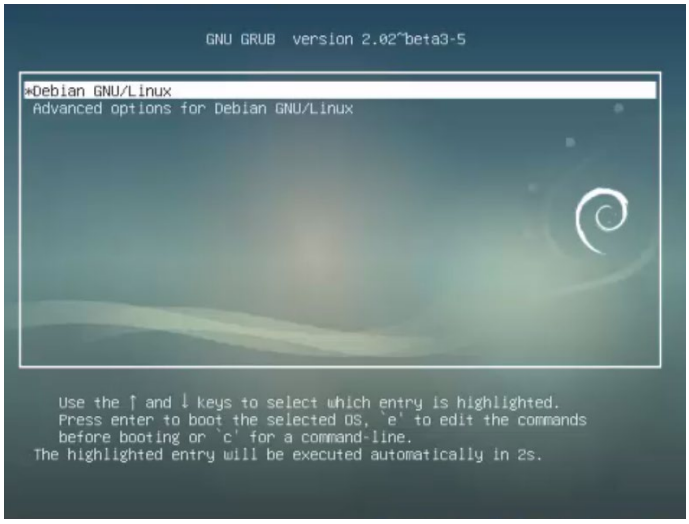
Najvažnija konfiguracijska datoteka je **/boot/grub/grub.cfg**, a u njoj se nalaze glavne postavke *GRUB*-a 2. Svaki odjeljak je označen s *"(### BEGIN)"* i poziva se na mapu **/etc/grub.d** iz koje su dobivene postavke. Datoteka se **grub.cfg** može osvježiti naredbom `update-grub` koju treba pokrenuti kao korisnik *root*.

Svaki puta kada se instalira nova jezgra, osvježit će se i datoteka **grub.cfg**. Međutim, ta datoteka nije predviđena za uređivanje pa ju je moguće samo čitati (*read only*).

Važna je i konfiguracijska datoteka **/etc/default/grub**. Sadržaj se može uređivati s *root* ovlastima. Kada se pokrene naredba `update-grub`, promjene će se odraziti u datoteci **grub.cfg**.

Jezgra operacijskog sustava i pripadajuće datoteke (kao **initrd**) nalaze se u direktoriju **/boot**. **initrd** (*initial ramdisk*) je pomoćna datoteka koja služi za učitavanje pomoćnog datotečnog sustava *root* prilikom pokretanja operacijskog sustava. U tom pomoćnom datotečnom sustavu nalaze upravljački programi za detektiranje hardvera kao što je tvrdi disk ili mrežna kartica.

Na slici je prikazan izgled programa *GRUB* prilikom pokretanja operacijskog sustava.



8.3.2. Podešavanje GRUB-a

Kad se *GRUB 2* želi podesiti i promijeniti, promjene se unose u datoteku **/etc/default/grub**. Datoteka **/boot/grub/grub.cfg** ne smije se uređivati nego se automatski podešava prema promjenama u datoteci **/etc/default/grub**.

Slijedi primjer datoteke **/etc/default/grub**:

```
# If you change this file, run 'update-grub' afterwards to update
# /boot/grub/grub.cfg.
# For full documentation of the options in this file, see:
# info -f grub -n 'Simple configuration'

GRUB_DEFAULT=0
GRUB_TIMEOUT=5
GRUB_DISTRIBUTOR=`lsb_release -i -s 2> /dev/null || echo Debian`
GRUB_CMDLINE_LINUX_DEFAULT="quiet"
GRUB_CMDLINE_LINUX=""

# Uncomment to enable BadRAM filtering, modify to suit your needs
# This works with Linux (no patch required) and with any kernel that obtains
# the memory map information from GRUB (GNU Mach, kernel of FreeBSD ...)
#GRUB_BADRAM="0x01234567,0xfefefefe,0x89abcdef,0xefefefef"

# Uncomment to disable graphical terminal (grub-pc only)
#GRUB_TERMINAL=console

# The resolution used on graphical terminal
# note that you can use only modes which your graphic card supports via VBE
# you can see them in real GRUB with the command `vbeinfo'
#GRUB_GFXMODE=640x480

# Uncomment if you don't want GRUB to pass "root=UUID=xxx" parameter to Linux
#GRUB_DISABLE_LINUX_UUID=true

# Uncomment to disable generation of recovery mode menu entries
#GRUB_DISABLE_RECOVERY="true"
```

```
# Uncomment to get a beep at grub start
#GRUB_INIT_TUNE="480 440 1"
```

U njoj se nalaze ove linije:

GRUB_DEFAULT

U ovoj je liniji upisan operacijski sustav koji će biti odabran kao zadan u izborniku *GRUB*. Možete upisati broj ili naziv.

GRUB_DEFAULT=0 - postavlja zadani izbor prema položaju na izborniku. Položaj se broji od 0 kao i na starom GRUB-u.(Prvi je 0, drugi je 1 itd.)

GRUB_DEFAULT=saved - ostavlja izbornik na posljednjem odabranom odabiru.

GRUB_DEFAULT="xxxx" - postavlja izbornik prema nazivu operacijskog sustava. Tako neće biti važan položaj u izborniku, nego samo ime.npr: **GRUB_DEFAULT="Debian GNU/Linux, with Linux 3.2.0-4-amd64"**

GRUB_HIDDEN_TIMEOUT=0

Tu se postavlja prikaz izbornika GRUB. Ako je potrebno prikazati izbornik, ta se linija zakomentira, tj. doda se znak # na početak te linije. (**# GRUB_HIDDEN_TIMEOUT=0**)

Za sve unose veće od 0, sustav će napraviti pauzu za toliki broj sekundi, ali se izbornik neće prikazati.

Ako se želi prikazati izbornik, do njega se može doći pritiskom na tipku [SHIFT].

Ako tipka [SHIFT] ne radi, do izbornika se može doći i pristiskom na tipku [ESC].

GRUB_HIDDEN_TIMEOUT_QUIET=true

true ne prikazuje odbrojavanje vremena. Izbornik se ne prikazuje.

false prikazuje odbrojavanje vremena. Izbornik se ne prikazuje.

GRUB_TIMEOUT=10

Postavlja vrijeme tijekom kojeg će biti prikazan izbornik *GRUB*. (Ako je aktivna opcija GRUB_HIDDEN_TIMEOUT, GRUB_TIMEOUT će se pokrenuti samo kad je izbornik prikazan.)

Ako se postavi vrijednost na -1, izbornik će se prikazivati dok god se ne odabere stavka iz izbornika.

GRUB_DISTRIBUTOR=`lsb_release -i -s 2> /dev/null || echo Debian`

Postavlja ime stavke u izborniku. (Debian, Ubuntu, Red Hat, Windows itd.)

GRUB_CMDLINE_LINUX

Ako postoji, ta linija uvozi sve iz naredbene linije "Linux".

GRUB_CMDLINE_LINUX_DEFAULT="quiet splash"

Ta linija uvozi sve s kraja linije "Linux". Ako se želi da se prilikom pokretanja sustava prikaže tekst procesa, treba ukloniti "quiet splash". Ako se želi vidjeti slika *splash* zajedno s tekstnim opisom procesa, koristi se "splash".

#GRUB_TERMINAL=console

Potrebno je odkomentirati ako se ne rabi grafički terminal.

#GRUB_DISABLE_LINUX_UUID=true

Potrebno odkomentirati ako se ne želi da GRUB jezgri operacijskog sustava preda parametar "root=UUID=xxx".

#GRUB_GFXMODE=640x480

Podešavanje rezolucije grafičkog izbornika. Postavke se odnose samo na izbornik pokretanja (*boot menu*), a ne na operacijski sustav koji se pokreće. Potrebno odkomentirati ako se želi koristiti ta opcija.

#GRUB_DISABLE_LINUX_RECOVERY=true

Potrebno odkomentirati tu liniju, ako se ne želi da se prikaže način rada "Recovery" jezgre u izborniku.

Nakon promjena u datoteci **/etc/default/grub** treba pokrenuti naredbu `update-grub` kao korisnik *root*:

```
# update-grub
Generating grub.cfg ...
Found linux image: /boot/vmlinuz-4.9.0-3-amd64
Found initrd image: /boot/initrd.img-4.9.0-3-amd64
 No volume groups found
done
```

8.3.3. LILO

Prije pojave *GRUB*-a, *LILO* je bio najpopularniji i najrašireniji program za učitavanje operacijskog sustava. Danas se sve značajnije distribucije koriste *GRUB*-om.

LILO je također prvi program koji se pokreće nakon što BIOS prepusti kontrolu učitavanja operacijskog sustava. Izravno je zadužen za učitavanje jezgre operacijskog sustava, koji dalje učitava ostatak operacijskog sustava.

LILO je skraćenica od *LInux LOader*, a sastoji se od dva dijela: od instalacijskog programa koji se pokreće iz operacijskog sustava i od dijela koji instalacijski program instalira na disk u MBR-u (*Master Boot Record*).

Konfiguracijska se datoteka nalazi u datoteci **/etc/lilo.conf**, a naredba za instalaciju u MBR-u je `lilo`. Dovoljno je pokrenuti naredbu bez parametara a ona će pročitati konfiguracijsku datoteku i smjestiti *LILO* u MBR.

Primjer je konfiguracijske datoteke:

```
boot = /dev/sda1
install = menu
default = "Linux"
prompt
timeout = 100

image = /boot/vmlinuz-3.2.0-4-amd64
 label = "Linux"
 root = /dev/sda1
 read-only
 initrd = /boot/initrd.img-3.2.0-4-amd64
```

Objašnjenje opcija prikazano je u sljedećoj tablici.

Opcija	Opis
boot	Mjesto gdje će *LILO* biti instaliran, tj. tvrdi disk s kojeg se pokreće operacijski sustav.
install	Odabir hoće li se rabiti tekstni prompt (*text*) ili izbornik (*menu*).
prompt	Daje korisniku na odabir, ako postoji više jezgri ili operacijskih sustava na računalu. Ako se ova opcija isključi, korisnik ne može izabrati operacijski sustav ili jezgru i pokreće se predodređeni.
default	Predodređeni operacijski sustav ili jezgra.
timeout	Broj desetinki sekundi koliko je korisniku dopušteno za odabir toga što želi pokrenuti.
image	Putanja do jezgre operacijskog sustava koja se pokreće.
label	Ime operacijskog sustava.
root	Lokacija particije *root*.
read-only	Montira particiju *root* samo za čitanje, da bi se ispravno izvršila provjera datotečnog sustava naredbom fsck.
initrd	Mjesto slike initrd.

Nakon izmjena u datoteci **/etc/lilo.conf** treba pokrenuti naredbu `lilo` kao korisnik *root*:

```
# lilo
```

Vježba 7: Upravljanje diskovima i particijama

Napomena

Ovu vježbu potrebno je izvoditi s ovlastima korisnika *root*. U terminal je potrebno upisati:

`su -` pa lozinku korisnika *root* dodijeljenu prilikom instalacije.

1. Priključite na računalo još jedan disk. S obziroma na to da je sistemski disk **/dev/sda**, taj će disk biti **/dev/sdb**.
2. Izradite novu primarnu particiju **/dev/sdb1** veličine 100 MB na disku **/dev/sdb** koristeći se naredbom `fdisk`.

   ```
   fdisk /dev/sdb
   ```

3. Napravite novi datotečni sustav na napravljenoj particiji, koristeći se naredbom `mkfs`.

   ```
   mkfs /dev/sdb1
   ```

4. Napravite novu logičku particiju **/dev/sdb5** veličine 100 MB na disku **/dev/sdb** koristeći se naredbom `cfdisk`.

   ```
   cfdisk /dev/sdb
   ```

5. Montirajte primarnu particiju na direktorij **/mnt**.

   ```
   mount /dev/sdb1 /mnt
   ```

6. U datoteci **/etc/default/grub** promijenite vrijednost varijable GRUB_TIMEOUT na vrijednost 7 sekundi. Ta varijabla postavlja vrijeme tijekom kojeg će biti prikazan GRUB-ov meni. Nakon isteka tog vremena, pokrenut će se operacijski sustav pod rednim brojem navedenim u varijabli GRUB_DEFAULT.
7. Osvježite konfiguracija GRUB-a naredbom `update-grub`.

   ```
   update-grub
   ```

8. Ponovo pokrenite računalo. Pogledajte čeka li sada GRUB-ov izbornik 7 sekundi.

Pitanja za ponavljanje

1. Koliko može biti primarnih, a koliko proširenih particija na jednom disku?

   _____

   _____

2. U čemu je razlika između alata za particioniranje **fdisk** i **parted**?

   _____

   _____

3. Koji se program za učitavanje operacijskog sustava danas najviše rabi?

   _____

   _____

9. Datotečni sustav

Trajanje poglavlja:
100 min

Po završetku ovoga poglavlja moći ćete:

- upoznati strukturu datotečnog sustava
- upoznati osnovne particije
- formatirati datotečni sustav
- provjeriti konzistentnost datotečnog sustava
- koristiti se kvotama
- nadzirati potrošnju diskovnog prostora
- koristiti se naredbama `blkid`, `mount`, `tune2fs`, `mkfs`, `fsck`, `debugfs` i `dumpe2fs`
- mijenjati dozvole i atribute nad datotekama
- koristiti se naredbama `chown`, `chgrp`, `chmod`, `umask`, `lsattr` i `chattr`.

Ova cjelina obrađuje strukturu datotečnog sustava. Naučit ćemo osnovne naredbe za upravljanje diskovima i particijama te kako promijeniti dozvole i atribute nad datotekama i direktorijima.

9.1. Struktura datotečnog sustava

9.1.1. Datotečni sustavi

Datotečni sustav je vrsta pohranjivanja i organiziranja računalnih datoteka na medij za pohranu podataka. Danas su funkcije datotečnih sustava dio jezgre operacijskih sustava.

Prilikom instalacije operacijskog sustava najčešće se može odrediti koji ćemo datotečni sustav rabiti kao osnovni na nekom računalu, no na više vanjskih medija dostupnih nekome računalu moguće je rabiti više datotečnih sustava.

Svaki sustav na svoj način vodi evidenciju o datotekama. Moguće je dodavanje podrške za dodatne sustave. Popis podržanih sustava nalazi se u datoteci **/proc/filesystems**.

Najčešći su datotečni sustavi:

FAT - rabio se u vrijeme DOS-a na PC-kompatibilnim računalima (utemeljenim na procesoru 8086), nasljednik mu je vfat ili FAT32

NTFS - datotečni sustav u uporabi na višezadaćnim inačicama operacijskog sustava *Microsoft Windows* (npr. NT4.0, 2000, XP)

ext2 - *Linux*ov datotečni sustav

ext3 - novija inačica, u odnosu na ext2 dodan je dnevnički sustav, tj. rabi se evidencija radnji koje treba izvršiti na vanjskom mediju prije samog izvođenja

ext4 - trenutačno najnovija inačica, podržava diskove veličine 1 egzabajta

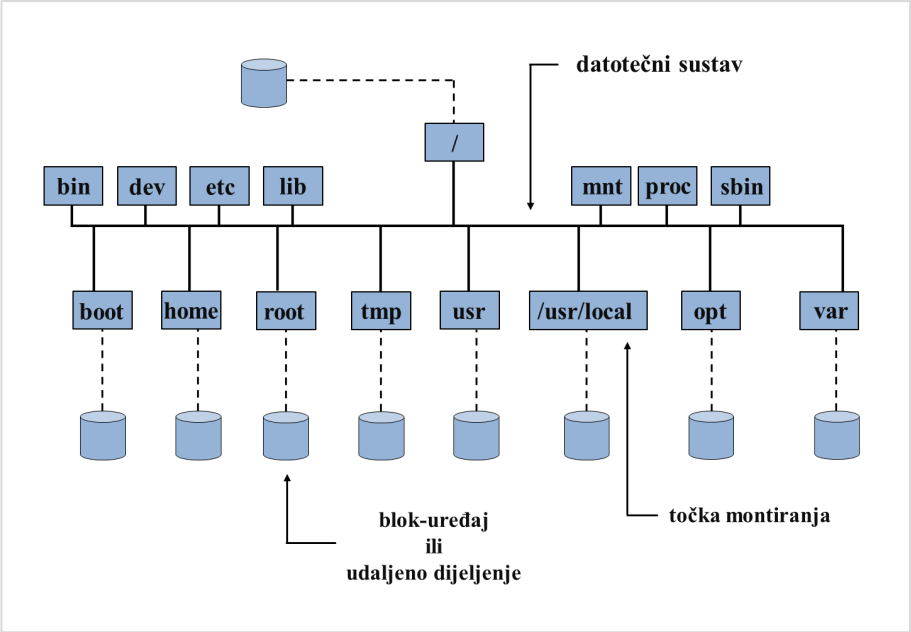
XFS - SGI razvija kao zamjenu za EFS, radi na većini distribucija *Linux*a

ReiserFS - prvi *Linux*ov datotečni sustav s dnevničkim sustavom.

Linux se koristi većinom dostupnih datotečnih sustava.

9.1.2. Struktura datotečnog sustava

Slika prikazuje strukturu datotečnog sustava na operacijskom sustavu *Linux*. Postoje mnogi resursi (ne moraju nužno biti samo lokalni tvrdi diskovi i particije, mogu biti i CD ili DVD-mediji, udaljeni dijeljeni disk, itd.) koji su spojeni na različite točke montiranja.



Za korisnika je datotečni sustav jednostavno stablo s direktorijima i poddirektorijima. Korijen tog stabla zove se korijenski direktorij ***root*** i prikazuje se znakom */*. On je prvi i ne nalazi se ni u jednom drugom direktoriju. Svi se drugi direktoriji i sadržaji nalaze unutar njega (grane tog drveta).

9.1.3. Standard hijerarhije datotečnog sustava

Linux je naslijedio hijerarhiju (strukturu) datotečnog sustava od *Unix*a, iako ne sasvim dosljedno (ovisi o distribuciji).

Hijerarhija datotečnog sustava prepoznaje:

datoteka (*file*) je neki podatak ili program, odnosno - nositelj sadržaja;

direktorij (*directory*) je „ladica“ koja objedinjuje datoteke, ali samostalno ne predstavlja nikakav sadržaj.

Razlikuju se dva logička pristupa rasporedu podataka:

samodostatna pakiranja, u kojima na jedno mjesto stavljamo jedan program i sve njegove popratne datoteke, biblioteke i pomoćne programe;

pakiranja datoteka prema svrsi i tipu, u kojima se jedan tip datoteka nalazi unutar jednog paketa makar se njima koriste različiti programi (npr. biblioteke svih programa se nalaze u direktoriju **biblioteke**).

Prednost samodostatnog pakiranja je u tome što je funkcionalno sve na jednom mjestu, no nedostatak je u tome što postoji puno duplikata. U računalu se taj nedostatak manifestira kao trošenje diskovnog prostora.

Prednost je pakiranja datoteka prema svrsi i tipu u tome što se tako prostor rabi učinkovitije (nema duplikata), ali je nedostatak teža pretraživost podataka. Međutim, računalo puno lakše pretražuje nego čovjek, tako da taj način pakiranja računalu ne predstavlja problem.

Platforma *Windows* više naginje prvom pristupu: većina se programa standardno nalazi u svojim direktorijima u direktoriju *Program Files*, a jedino se biblioteke stavljaju na zajedničkom mjesto (*dll* datoteke). Sustavi *Unix* imaju drugačiju filozofiju. *Unix* se sastoji od puno malih alata koji rade zajedno da bi napravili određeni zadatak i tako se programi međusobno rabe, a da bi se lakše pronašli svi se nalaze na jednom ili samo nekoliko mjesta. Biblioteke također imaju svoje zajedničko mjesto, pa ako neki program treba neku biblioteku, pretražuje samo biblioteke, a ne čitav sustav.

Stoga je organizacija *Linux Foundation* donijela **standard hijerarhije datotečnog sustava** (*Filesystem Hierarchy Standard - FHS*) koji definira strukturu datoteka i direktorija na operacijskim sustavima temeljenima na *Unix*u i na *Linux*u. Trenutačna inačica je 2.3, objavljena 29. siječnja 2004. godine.

9.1.4. Pregled osnovnih direktorija

Izgled i značenja pojedinih direktorija u stablu predočeni su u sljedećoj tablici.

Direktorij	Opis namjene
/	*Primarna hijerarhija,* direktorij *root* cijelokupne hijerarhije sustava, početak.
/bin	Izvršne datoteke važnih naredbi na razini tzv. *Single user moda*, naredbe za sve korisnike (npr. cat, ls, cp).
/boot	Datoteke potrebne za pokretanje sustava (npr. jezgra, datoteke *GRUB*), često i na zasebnoj particiji.
/dev	Datoteke koje predstavljaju fizičke ili virtualne uređaje (npr. diskovi, USB i drugi portovi).
/etc	Konfiguracijske datoteke sustava koje vrijede za cijeli sustav (ali ne i za korisničke programe i postavke koje su spremljene u korisničkom direktoriju */home/ime/*).
/home	Korisnički direktoriji *home* (i *Osobna mapa*) - sadrže korisničke privatne podatke i postavke. Često (i preporučeno) na posebnoj particiji, odvojenoj od sustava.
/lib	Važne biblioteke za programe iz direktorija */bin/* i */sbin/*.
/media	Mjesto/točka za montiranje za izmjenjivih medija, npr. CD-ROM ili USB memorija (od FHS-2.3).
/mnt	Privremeno montirani datotečni sustavi. Nisu nužni za funkcioniranje sustava.
/opt	Neobavezni, dodatni aplikacijski paketi i programi.
/proc	Virtualni datotečni sustav za prikaz rada jezgre i procesa u obliku tekstnih i sličnih datoteka.
/root	Direktorij *h*ome korisnika *root* . U pravilu se nalazi na istoj particiji gdje i cijeli sustav (sadržaj direktorija *root*).
/sbin	Važni sistemski programi (npr., init, route, ifconfig,...).
/srv	Specifični podaci posluženi od strane sustava...
/tmp	Privremeni podaci, koji se obično ne čuvaju nakon ponovnog pokretanja računala.
/usr	Sekundarna hijerarhija za korisničke podatke; sadrži glavninu više korisničkih alata i aplikacija.
/usr/bin	Manje važne izvršne datoteke programa i naredbe (nepotrebne u tzv. *Single user modu*); namijenjeno svim korisnicima.
/usr/include	Standardne datoteke *include*.

/usr/lib	Biblioteke programa u */usr/bin/* i */usr/sbin/*.
/usr/sbin	Manje važne sistemske datoteke (npr. *daemon*i za različite servise).
/usr/share	Datoteke koje su neovisne o arhitekturi (dijeljene datoteke), npr. slike/ikone ili dokumentacija.
/usr/src	Datoteke izvornog koda (npr. izvorni kod jezgre operacijskog sustava).
/usr/X11R6	X *Window System*, inačica 11, izdanje 6.
/usr/local	Tercijalna hijerarhija za lokalne podatke. Strogo prema standardu, /usr/local/ služi za podatke koji moraju biti pohranjeni na lokalnom računalu (suprotno od /usr/, koji mogu biti montirani preko mreže).
/var	Promjenjive datoteke kao što su logovi i sl.
/var/lib	Podaci koji se mijenjaju u pripadajućim programima (npr. baze podataka).
/var/lock	Zaključane datoteke. Datoteke koje drže tragove o programima koji se izvršavaju.
/var/log	Sistemske log-datoteke.
/var/mail	Korisnički sandučići e-pošte.
/var/run	Informacije o sustavu od zadnjeg pokretanja.
/var/spool	Spool za zadatke koji tek trebaju biti učinjeni.
/var/tmp	Privremene datoteke koje trebaju preživjeti ponovno pokretanje (*reboot*) računala.

9.2. Upravljanje diskovima i particijama

9.2.1. Linuxovi datotečni sustavi

Datotečni sustav način je pohranjivanja i organiziranja računalnih datoteka na medij za pohranu podataka. Danas su funkcije datotečnih sustava dio jezgre operacijskih sustava. Prilikom instalacije operacijskog sustava najčešće se može odrediti koji će se datotečni sustav rabiti kao osnovni na nekom računalu, no na više vanjskih medija dostupnih nekome računalu moguće je rabiti više datotečnih sustava.

Najzastupljeniji datotečni sustav na operacijskom sustavu *Linux* je **ext2**, a njegovi su nasljednici **ext3** i **ext4**.

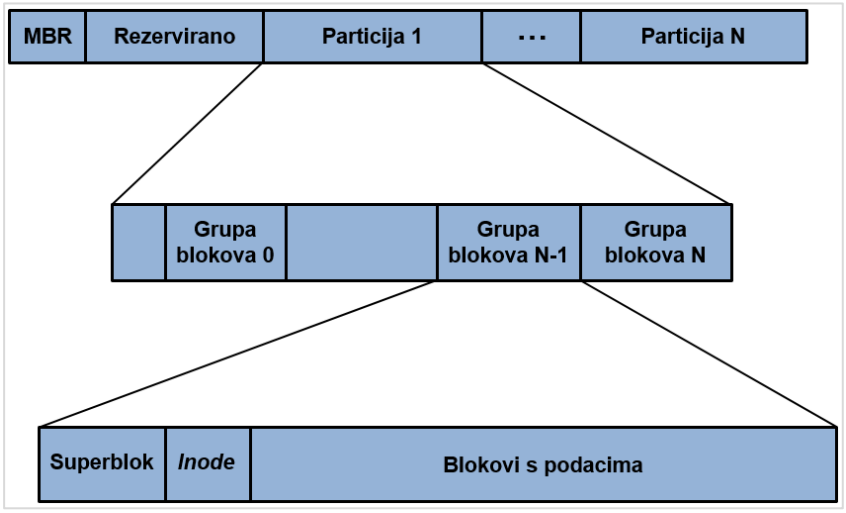
Datotečni sustav **ext2** sastoji se od blokova podrazumne veličine 1024 bajtova = 1 kB.

Postoje tri vrste blokova:

superblokovi (*superblocks*) – ponavlja se svakih 8193 bloka, sadrži informacije o veličini bloka, slobodnim inodovima, zadnjem vremenu montiranja itd.;

inodeovi (*inodes*) – sadrži pokazivač na blokove s podacima; svaki *inode* je veličine 256 bajtova i sadrži informacije o korisniku, skupini, dozvolama i vremenu stvaranja podatka na koji pokazuje;

blokovi s podacima (*data blocks*) - sadrže podatke.



9.2.2. U čemu je razlika između ext2, ext3 i ext4?

Datotečni su sustavi **ext2**, **ext3** i **ext4** obitelj datotečnih sustava, koji imaju snažnu unatrag i naprijed kompatibilnost. Oni se zapravo mogu smatrati jednim formatom datotečnog sustava s brojnim značajnim nastavcima. I **ext2**, **ext3** i **ext4** su samo imena implementacija koje se mogu naći u jezgri operacijskog sustava *Linux*.

Takav je način gledanja na stvari podržan činjenicom da datotečni sustavi dijele iste alate za korisnički prostor (**e2fsprogs** – **mke2fs, e2fsck, resize2fs, tune2fs...**). Na primjer, datotečni sustav koji je napravljen za uporabu s **ext3** može se montirati pomoću **ext2** ili **ext4**. Međutim, datotečni sustav sa specifičnim nastavcima **ext4** ne može se montirati pomoću **ext2** ili **ext3**. I datotečni sustav **ext3** kôd u jezgri zahtijeva prisutnost dnevnika (*journal*), koji općenito nije prisutan u particijama formatiranim za korištenje datotečnog sustava **ext2**.

Ukoliko se ext2 želi pretvoriti u ext3, dovoljno je uključiti prisutnost dnevnika naredbom `tune2fs`. Slijedi primjer za **/dev/sdb1**:

```
tune2fs -j /dev/sdb1
```

Datotečni sustav **ext4** kôd ima sposobnost montiranja i korištenja datotečnog sustava bez dnevnika.

Remy Card je u travnju 1992. godine napisao datotečni sustav **ext** da bi riješio dva ključna ograničenja datotečnog sustava ***Minix***, koji je prethodno bio samo datotečni sustav dostupan za *Linux*, a to su: naziv datoteke je mogao sadržavati samo 14 znakova, a veličina je datotečnog sustava koji podržava *Minix* bila najviše 64MB.

Datotečni sustav **ext** podržava blok uređaja do 2GB, a nazive datoteka do 255 znakova, ali (kao Minix) ima samo jedan *timestamp* za vrijeme posljednje promjene, posljednje vrijeme pristupa, i promjenu vremena *inode*.

Od siječnja 1993. kada je objavljen pa do danas, datotečni je sustav **ext2** dodatno povećao veličinu bloka na najviše 4 TB, dodao POSIX *timestamps* i omogućio podršku za različite veličine blokova.

9.2.3. Formatiranje datotečnog sustava

Datotečni sustavi se izrađuju, odnosno inicijaliziraju pomoću programa **mkfs.** Program **mkfs** je program vrste *front-end,* koji poziva posebne programe pomoću kojih se izrađuju različite vrste datotečnih sustava. Sintaksa programa **mkfs** je različita za različite distribucije, a najvažnije su opcije, zajedničke za sve inačice programa **mkfs,** objašnjene u nastavku.

Sintaksa je programa **mkfs**:

```
# mkfs [-t fstype] [-c | -l bblist] device
```

Objašnjenja su opcija:

Opcija	Opis
-t fstype	Opcija kojom se specificira vrsta datotečnog sustava koji treba izraditi. **fstype** može biti **ext2**, **ext3**, **ext4**, **reiserfs**, **msdos** ili bilo koja druga vrsta za koju u operacijskom sustavu postoji podrška.
-c	Zastavica kojom se programu **mkfs** nalaže da prije izrade datotečnog sustava ispita površinu medija na kojoj taj datotečni sustav radi i inicijalizira popis neispravnih blokova.
-l bblist	Opcija kojom se specificira datoteka s inicijalnim popisom neispravnih blokova. Ne treba se koristiti opcije -c i -l zajedno.
device	Posebna datoteka koja predstavlja particiju na kojoj se izrađuje datotečni sustav. Tom datotekom se kasnije predstavlja datotečni sustav (npr. **/dev/sda1**, **/dev/hdc3**).

Ovisno o izabranoj vrsti datotečnog sustava, naredba `mkfs` će pokretati naredbe koje se zovu `mkfs.<tip_datotečnog_sustava>`, tj. `mkfs.ext2`, `mkfs.ext3`, `mkfs.ext4`, itd.

Primjer je uporabe naredbe `mkfs`:

```
# mkfs -t ext2 -c /dev/sdb1
mke2fs 1.42.5 (29-Jul-2012)
Discarding device blocks: done
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
Stride=0 blocks, Stripe width=0 blocks
2512 inodes, 10000 blocks
500 blocks (5.00%) reserved for the super user
First data block=1
Maximum filesystem blocks=10485760
2 block groups
8192 blocks per group, 8192 fragments per group
1256 inodes per group
Superblock backups stored on blocks:
 8193
Checking for bad blocks (read-only test): done
Allocating group tables: done
Writing inode tables: done
Writing superblocks and filesystem accounting information: done
```

9.2.4. Provjera konzistentnosti datotečnog sustava

U slučaju da je datotečni sustav oštećen ili korumpiran, treba pokrenuti naredbu `fsck` na oštećenoj particiji. Minimimalni zahtjev je da datotečni sustav bude montiran samo za čitanje ili nemontiran.

Naredba `fsck` automatski će detektirati o kojem se datotečnom sustavu radi i pokrenuti odgovarajuću naredbu: fsck.ext2, fsck.ext3, fsck.ext4, itd.

Isto se tako zastavicom **-t** može izabrati vrsta datotečnog sustava koji se provjerava.

Sintaksa je ovakva:

```
fsck -t <tip_datotecnog_sustava> <uređaj>
```

Slijedi primjer uporabe naredbe:

```
# fsck /dev/sdb1
fsck from util-linux 2.25.2
e2fsck 1.42.12 (29-Aug-2014)
/dev/sdb1: clean, 126684/30531584 files, 58945211/122096128 blocks
```

9.2.5. Debugiranje datotečnog sustava

Naredbe `debugfs` i `dumpe2fs` služe za prikazivanje detaljnih informacija o radu datotečnog sustava.

Program **dumpe2fs** prikazuje informacije o datotečnom sustavu **ext2** na temelju podataka koje čita iz zaglavlja, odnosno superbloka. Program funkcionira na principu programa *Berkeley dumpfs* koji prikazuje informacije o datotečnom sustavu BSD FFS.

Naredba `debugfs` je interaktivni program koji korisniku omogućava izravan pristup strukturama datotečnog sustava, tako da se može koristiti za njihov oporavak, ako **fsck** to ne može napraviti automatski. Isto tako, `debugfs` omogućava korisniku da izvrši razne operacije nad objektima datotečnog sustava, poput markiranja zastavice čistoće, inicijalizacije, povezivanja struktura *inode* i *dir-info* te povratka obrisanih datoteka. Detaljan popis naredbi programa **debugfs** dobiva se unošenjem naredbe `help` u program.

Primjer je uporabe naredbe `dumpe2fs`:

```
# dumpe2fs /dev/sdb1 16:07
dumpe2fs 1.42.12 (29-Aug-2014)
Filesystem volume name: <none>
Last mounted on: /ext
Filesystem UUID: 9ce7f65c-a66a-4063-8573-44d4b91fd053
Filesystem magic number: 0xEF53
Filesystem revision #: 1 (dynamic)
Filesystem features: has_journal ext_attr resize_inode dir_index
filetype sparse_super large_file
Filesystem flags: signed_directory_hash
```

```
Default mount options: (none)
Filesystem state: clean
Errors behavior: Continue
Filesystem OS type: Linux
Inode count: 30531584
Block count: 122096128
Reserved block count: 6104806
Free blocks: 63150917
Free inodes: 30404900
First block: 0
Block size: 4096
Fragment size: 4096
Reserved GDT blocks: 994
Blocks per group: 32768
Fragments per group: 32768
Inodes per group: 8192
Inode blocks per group: 512
Filesystem created: Wed May 25 19:31:39 2011
Last mount time: Mon May 18 16:01:16 2015
Last write time: Mon May 18 16:07:43 2015
Mount count: 8
Maximum mount count: 35
Last checked: Mon Mar 2 11:08:16 2015
Check interval: 15552000 (6 months)
Next check after: Sat Aug 29 12:08:16 2015
Lifetime writes: 8075 MB
Reserved blocks uid: 0 (user root)
Reserved blocks gid: 0 (group root)
First inode: 11
Inode size: 256
Required extra isize: 28
Desired extra isize: 28
Journal inode: 8
Default directory hash: half_md4
Directory Hash Seed: 3c66e30d-2d47-408c-8ac4-aa421afd406d
Journal backup: inode blocks
Journal features: journal_incompat_revoke
Journal size: 128M
Journal length: 32768
Journal sequence: 0x0000e99e
Journal start: 0
```

9.2.6. Montiranje datotečnih sustava i datoteka /etc/fstab

Datotečni se sustavi montiraju automatski, pokretanjem operacijskog sustava. Da bi operacijski sustav znao kamo treba montirati koje datotečne sustave, za to postoji datoteka **/etc/fstab**. U nju su smještene sve informacije o montiranim datotečnim sustavima.

Slijedi primjer datoteke **/etc/fstab**:

```
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name
devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point> <type> <options> <dump> <pass>
proc /proc proc defaults 0 0
# / was on /dev/sda1 during installation
/dev/sda1 / ext3 errors=remount-ro 0 1
# /home was on /dev/sda3 during installation
/dev/sda3 /home ext3 defaults,user_xattr 0 2
# swap was on /dev/sda2 during installation
/dev/sda2 none swap sw 0 0
/dev/scd0 /media/cdrom0 udf,iso9660 user,noauto 0 0
# /ext was on /dev/sdb1 during installation
UUID=9ce7f65c-a66a-4063-8573-44d4b91fd053 /ext ext3 defaults 0 2
```

Svaki datotečni sustav opisan je jednom linijom datoteke. Svaka linija datoteke ima šest polja, odvojenih razmaknicama ili znakom Tab, na primjer `/dev/sda1 / ext3 errors=remount-ro 0 1`.

Značenja su ovih polja navedena redom.

Prvo polje (fs_spec) opisuje datotečne sustave koje treba aktivirati. Lokalni datotečni sustavi opisani su posebnim datotekama koje predstavljaju particije (/dev/sda2, /dev/cdrom) ili imenom sistema datoteka (LABEL=/home), a mrežni datotečni sustavi imenom poslužitelja i imenom dijeljenog direktorija (npr. posluzitelj:/direktorij).

Drugo polje (fs_file) opisuje točke montiranja. Za *swap* u ovo polje treba upisati *swap* ili *none*.

Treće polje (fs_vfstype) opisuje vrstu datotečnog sustava - npr. ext2, ext3, iso9660, swap.

U četvrtom polju (fs_mntops) navedene su zarezom ozdvojene opcije pomoću kojih će se aktivirati sustav datoteka.

Peto polje (fs_freq) određuje hoće li datotečni sustav biti uključen u popis sustava datoteka za sigurnosnu pohranu (*backup*), odnosno *dump* (vrijednost polja je 1) ili ne (vrednost polja je 0).

Šesto polje (fs_passno) opisuje redosljed kojim će program **fsck** provjeriti konzistentnost datotečnog sustava pri podizanju operacijskog sustava. Za datotečni sustav *root* fs_passno je 1, a za druge je 2.

Drugi je način za aktiviranje svih datotečnih sustava opisanih u **/etc/fstab** naredba `mount -a`. Tom će naredbom svi datotečni sustavi opisani u **/etc/fstab**, koji nemaju atribut *noauto*, biti montirani na točke montiranja navedene u drugom stupcu iste datoteke.

Ako se želi montirati određeni datotečni sustav koji nije naveden u datoteci **/etc/fstab**, to se može naredbom `mount`.

Sintaksa je sljedeća:

```
# mount -t <tip_datotečnog_sustava> -o <opcije> <uređaj> <mount-point>
```

Npr. ako se želi montirati particija **/dev/sdb1** na točki montiranja **/ext,** dovoljno je pokrenuti:

```
# mount /dev/sdb1 /ext
```

Jezgra operacijskog sustava provjerit će koji se datotečni sustav nalazi na toj particiji i montirati je na odgovarajuću točku.

Budući da poredak diskova ovisi o poretku kojim jezgra operacijskog sustava prepoznaje diskove, može se dogoditi da diskovi zamjene imena. Npr. da **/dev/sdb** bude **/dev/sda** i obrnuto. Time će se zamijeniti imena i particijama. Da bi se to izbjeglo, svakom disku je pridijeljen njegov jedinstveni broj, tj. *UUID* (*Universally Unique IDentifier*).

Naredba `blkid` služi za prikaz particija i pripadajućih UUID-ova:

```
# blkid
/dev/sda2: UUID="ff29dd8f-e771-4866-b009-b6e065876f02" TYPE="swap"
PARTUUID="07970a9a-02"
/dev/sdb1: UUID="9ce7f65c-a66a-4063-8573-44d4b91fd053" SEC_TYPE="ext2"
TYPE="ext3" PARTUUID="0f22e087-01"
/dev/sda1: UUID="e91b8b94-4784-49bc-bee9-7abd0f8d6564" TYPE="ext3"
PARTUUID="07970a9a-01"
/dev/sda3: UUID="4706f77c-246e-44df-92d2-404165509483" TYPE="ext3"
PARTUUID="07970a9a-03"
```

Taj se UUID može upisati u **/etc/fstab**, umjesto posebne datoteke koja predstavlja disk. U gornjem primjeru disk **/dev/sdb1** zapisan je u **/etc/fstab** preko UUID-a.

Naredba `tune2fs` služi za podešavanje parametara datotečnog sustava.

Kao što se particija može identicirati preko UUID-a, također se može identicirati i preko labele. Dovoljno je u datoteci **/etc/fstab** rabiti LABEL=ime_labele. Ako želite postaviti ime labele, to se može napraviti naredbom `tune2fs` i prekidačem -L.

```
tune2fs -L ime_labele particija
```

9.2.7. Kvote

Kvota je ograničenje koje sistemski administrator dodjeljuje korisnicima, a koje se tiče stupnja iskorištenja prostora na datotečnom sustavu.

Kvote se mogu postaviti na ove načine:

ograničavanjem broja indeksnih čvorova (odnosno broja datoteka) koje korisnici ili skupine mogu imati u okviru sustava datoteka za koji je kvota postavljena

ograničavanjem broja blokova na disku (vrijednost u kilobajtima) koji se mogu dodijeliti korisnicima ili skupinama.

Kvota ograničava korisnike, odnosno sprječava ih da rabe neograničenu količinu slobodnog prostora u datotečnom sustavu. Također, kvotama se mogu ograničiti veličine poštanskih pretinaca

korisnika. Na primjer, kvota od 10 MB može se dodijeliti svim korisnicima za particiju **/var**, nakon čega korisnici mogu u direktoriju **/var/spool/$LOGNAME** sačuvati najviše 10 MB.

Kvota se može postaviti za korisnike, skupine ili korisnike i skupine. Postavljanje kvota vrši se u nekoliko koraka. Najprije treba modificirati datoteku **/etc/fstab**. Kvota se mora postaviti posebno za svaki datotečni sustav, odnosno mora biti dopisana u svaku liniju datoteke **/etc/fstab** koja predstavlja datotečni sustav za koji se želi postaviti kvota. To se obavlja opcijama **usrquota** i **grpquota**.

Slijedi primjer definiranja kvote za korisnike i skupinu korisnika u datoteci **/etc/fstab**, za datotečni sustav **/dev/sda2** koji je montiran na **/home**.

/dev/sda2 /home ext2 defaults,nosuid,**usrquota**,**grpquota** 1 2

Nakon izmjena u datoteci **/etc/fstab** treba pokrenuti naredbu:

```
# mount -o remount /home/
```

Naredbom `quotacheck -acug` treba stvoriti datoteke na datotečnom sustavu u kojima se čuvaju informacije o potrošnji kvote.

Naredba `edquota` služi za postavljanje kvote određenom korisniku.

```
# edquota -u tux
Quotas for user tux:
 /dev/sda2: blocks in use: 0, limits (soft = 0, hard = 0)
 inodes in use: 0, limits (soft = 0, hard = 0)
```

Značenje je linija:

blocks in use: ukupan broj blokova (u kilobajtima) koje je korisnik upotrijebio na particiji

inodes in use: ukupan broj datoteka koje je korisnik smjestio na particiju.

Korisniku **tux** može se dodijeliti kvota od 5 MB na particiji **/dev/sda2** na ovaj način:

Quotas for user tux: /dev/sda2:
 blocks in use: 0, limits (soft = 5000, hard = 6000)
 inodes in use: 0, limits (soft = 0, hard = 0)

Značenje je parametara **soft** i **hard limit**:

Soft limit određuje najveću količinu prostora na datotečnom sustavu koju korisnik može iskoristiti za smještanje svojih datoteka. Budući da je u ovom primjeru soft=5000, korisnik **tux** će na particiju **/dev/sda2** moći smjestiti najviše 5 MB svojih datoteka.

Hard limit apsolutno ograničenje - korisnik ni na koji način ne može prijeći to ograničenje.

Alat `quotacheck` analizira potrošnju datoteka i direktorija na odgovarajućem datotečnom sustavu i na temelju toga izrađuje odgovarajuće datoteke **quota.user** i **quota.group**.

Sintaksa je naredbe:

```
# quotacheck [-u] [-g] [-a|filesystem]
```

9.2.8. Nadziranje potrošnje diskovnog prostora

Naredba `df` služi za nadziranje potrošnje datotečnih sustava. Pokretanjem naredbe `df` ispisat će se svi montirani datotečni sustavi i njihova trenutačna potrošnja. Opcija **-h** je korisna jer ispisuje veličine u megabajtima, gigabajtima ili terabajtima:

```
# df -h
Filesystem Size Used Avail Use% Mounted on
/dev/sda1   46G  14G   31G  31% /
/dev/sda3  411G 262G  129G  68% /home
tmpfs      797M  36K  797M   1% /run/user/2057
/dev/sdb1  459G 218G  218G  50% /ext
```

Naredba `du` služi za prikaz prostora koji zauzima određeni direktorij. Primjer je korištenja naredbe:

```
# du -sh /home
411G /home
```

9.3. Dozvole i atributi nad datotekama

9.3.1. Dozvole nad datotekama

Dozvole koje direktoriji i datoteke imaju u *Linux* datotečnom sustavu mogu izgledati kriptično, no zapravo se radi o vrlo jednostavnom sustavu koji je lako razumjeti i upotrebljavati. Budući da je, općenito gledano, u *Linux*u sve prikazano u obliku datoteke, na isti se način i pristupa i upravlja datotekama i uređajima te je jedna od važnijih stvari dobro razumijevanje sustava dozvola.

Čitanje, pisanje i izvršavanje tri su osnovne radnje koje možete napraviti s datotekom, a notacija slovima ih predstavlja kao:

r - čitanje (*read*)

w - pisanje (*write*)

x - izvršavanje (*execute*).

9.3.2. Dozvole nad direktorijima

Postupci su složeniji kod dozvola za direktorije – iako se još uvijek rabi notacija slovima r/w/x, dozvole imaju samo djelomične sličnosti s datotekama:

r - dozvola za pregled direktorija

w - dozvola za brisanje i izradu datoteka i poddirektorija

x - dozvola za ulaz u direktorij.

Dozvola za ulaz u direktorij možda izgleda čudno, no može se objasniti jednostavnim primjerom – ako korisnik treba pročitati neku datoteku i ima prava za to (barem što se tiče datoteke), moći će je pročitati samo ako može ući u direktorij u kojem se nalazi (dozvola *x*).

S druge strane, korisnik mora znati i ime datoteke, ako nema i dozvolu *r* na tom direktoriju, jer mu je sadržaj direktorija nevidljiv i neće moći vidjeti željenu datoteku.

9.3.3. Korisnici

Navedene dozvole izgledaju u redu za jednog korisnika, no *Linux* je sam po sebi postavljen kao višekorisnički sustav. Stoga se uvodi koncept vlasnika, pripadajuće skupine i svih drugih, označeno slovima:

o - vlasnik (*owner*)

g - skupina (*group*)

a - svi (*all*).

Svaka datoteka i direktorij imaju definiranog **vlasnika** i **vlasničku skupinu** (može, ali i ne mora biti povezano), zato se i dozvole primjenjuju odvojeno za vlasnika datoteke ili direktorija, vlasničku skupinu odnosno za sve druge.

Naredbom `ls` može se provjeriti stanje vlasništva i dozvola nad određenom datotekom ili direktorijem.

U sljedećem primjeru vidi se da je vlasnik direktorija **root**, vlasnička skupina je također **root**, vlasnik može čitati i pisati u tu datoteku, a vlasnička skupina i svi drugi mogu samo čitati.

```
$ ls -al /etc/passwd
-rw-r--r-- 2 root root 2416 Mar 9 11:55 /etc/passwd
```

Slično je i s direktorijima. U sljedećem primjeru vlasnik direktorija je *root*, vlasnička skupina je također *root*, vlasnik može pregledavati direktorij, izrađivati i brisati datoteke u njemu te ući u direktorij. Svi drugi mogu samo ući u direktorij i pregledavati datoteke.

```
$ ls -ald /etc/
drwxr-xr-x 124 root root 12288 May 18 12:39 /etc/
```

Napomena

Prva oznaka u izrazu `-rw-r--r--` ili `drwxr-xr-x` , dakle - ili *d*, označava objekt nad kojim se postavljaju dozvole. Znak - predstavlja datoteku, a znak *d* direktorij.

9.3.4. Naredbe chmod

Naredba `chmod` standardna je *Unix*ova naredba kojom određujemo prava pristupa određenoj datoteci ili određenom direktoriju. Poznavajući uporabu naredbe `chmod` možemo konfigurirati siguran sustav u kojem će se točno znati koji korisnici smiju čitati, koji pisati, a koji izvršavati određene datoteke i direktorije. Ako su pravila pristupa nepravilno postavljena vrlo je vjerojatno da aplikacije koje zahtijevaju određena prava pristupa neće dobro raditi, a i sam sustav može biti nesiguran. Zbog toga su osnovna pravila čitanja, pisanja i izvršavanja inicijalno postavljena u svakoj *Linux*ovoj distribuciji, a mogu se promijeniti po želji upravo sa naredbom `chmod`.

Sintaksa je naredbe sljedeća:

chmod [ugoa...][[+-=][dozvole...]...]

Slova **ugoa** znače:

u(ser) - korisnik vlasnik datoteke

g(roup) - drugi korisnici skupine vlasnika datoteke

o(thers) - drugi koji ne pripadaju skupini korisnika vlasnika datoteke

a(ll) - svi korisnici.

Znakovi +-= znače:

+ - dodavanje dozvole

- - uklanjanje dozvole

= - dodavanje dozvole datoteci i micanje svih dozvola koje nisu navedene.

U skup dozvola mogu se staviti slova **rwx**:

r - postavljanje dozvole čitanja datoteke ili direktorija

w - postavljanje dozvole pisanja u datoteku ili direktorij

x - omogućavanje izvršavanja datoteke (ili pretraživanje direktorija za direktorije).

U sljedećem će se primjeru datoteci **/tmp/test.txt** dodati prava da vlasnička skupina i svi drugi korisnici mogu u nju pisati, a naredbom `ls` provjerava se stanje dozvola.

```
# ls -al /tmp/test.txt
-rw-r--r-- 1 root root 0 May 18 13:09 /tmp/test.txt
# chmod go+w /tmp/test.txt
# ls -al /tmp/test.txt
-rw-rw-rw- 1 root root 0 May 18 13:09 /tmp/test.txt
```

U sljedećem će se primjeru skripti **/tmp/test.sh** dodati da svi drugi imaju pravo pisanja i izvršavanja.

```
# ls -al /tmp/test.sh
-rwxr-xr-- 1 root root 0 May 18 13:09 /tmp/test.sh
# chmod o+wx /tmp/test.sh
# ls -al /tmp/test.sh
-rwxr-xrwx 1 root root 0 May 18 13:09 /tmp/test.sh
```

9.3.5. Oktalna notacija i naredba chmod

U nekoliko su se prethodnih poglavlja za mijenjanje dozvola i vlasničkih odnosa nad elementima koristila slova, no često je jednostavnije i brže pregledati i postaviti dozvole u **oktalnoj notaciji** – jednoznamenkasti broj koji predstavlja određenu dozvolu, a mjesto znamenke označava na kojeg se korisnika što odnosi:

r => 4

w => 2

x => 1

Zbroj ovih vrijednosti odvojenih dozvola označava ukupnu dozvolu (npr. “rw” pravo je 4+2=6, “rx” je 4+1=5).

Ukupna se oznaka za dozvole sastoji od četiri znamenke – s desne strane na lijevo: svi, vlasnička skupina, vlasnik, posebna upotreba.

Ako samo vlasniku i vlasničkoj skupini želimo dati isključivo dozvolu čitanja neke datoteke, oznaka će izgledati ovako: 0440. Da bismo samo vlasniku omogućili pisanje i čitanje, a skupini i drugima samo čitanje, oznaku ćemo zapisati kao 0644.

Slijedi primjer uporabe naredbe `chmod` u slučaju oktalne notacije. Naredba `ls` služi za provjeru prethodno dodijeljenih dozvola.

```
# ls -al /tmp/test.txt
-rw-r--r-- 1 root root 0 May 18 13:09 /tmp/test.txt
# chmod 666 /tmp/test.txt
# ls -al /tmp/test.txt
-rw-rw-rw- 1 root root 0 May 18 13:09 /tmp/test.txt
```

9.3.6. Naredbe chown i chgrp

Naredba `chown` služi za promjenu vlasnika i vlasničke skupine određene datoteke ili direktorija.

U sljedećem će se primjeru datoteci **/tmp/test.sh** promijeniti vlasnik iz **root** u **tux**. Naredba `ls` služi za provjeru.

```
# ls -al /tmp/test.txt
-rw-r--r-- 1 root root 0 May 18 13:09 /tmp/test.txt
# chown tux /tmp/test.txt
# ls -al /tmp/test.txt
-rw-r--r-- 1 tux root 0 May 18 13:09 /tmp/test.txt
```

U sljedećem će se primjeru pomoću naredbe `chown` promijeniti i vlasnik i vlasnička skupina.

```
# ls -al /tmp/test.txt
-rw-r--r-- 1 root root 0 May 18 13:09 /tmp/test.txt
# chown tux:tux /tmp/test.txt
# ls -al /tmp/test.txt
-rw-r--r-- 1 tux tux 0 May 18 13:09 /tmp/test.txt
```

Ako se želi promijeniti samo skupina, to je moguće naredbom `chgrp`. Sintaksa je slična.

```
# ls -al /tmp/test.txt
-rw-r--r-- 1 root root 0 May 18 13:09 /tmp/test.txt
# chgrp tux /tmp/test.txt
# ls -al /tmp/test.txt
-rw-r--r-- 1 root tux 0 May 18 13:09 /tmp/test.txt
```

Ako se želi promijeniti vlasništvo nad više direktorija i datoteka u određenom direktoriju, potrebno je koristiti se opcijom **-R**. Slijede primjeri:

```
# chown -R root:root /tmp/direktorij/
# chown -R root /tmp/direktorij/
# chgrp -R root /tmp/direktorij/
```

9.3.7. Dodatne dozvole

Kod oktalne notacije četvrta znamenka, gledano s desne strane na lijevo, ima posebnu namjenu i kod datoteka se upotrebljava samo za izvršne datoteke. Vrijednost 4 služi da bi se pokrenutom programu iz određene datoteke dale ovlasti vlasnika te datoteke (*setuid*), bez obzira na to tko ga pokreće. Vrijednost 2 služi za definiranje vlasničke skupine (*setgid*).

Jedan od lakših primjera je naredba `ping`. Osnovni korisnik nema pravo pregleda mrežnih paketa, što je potrebno za naredbu `ping` da bi mogla pravilno raditi. Stoga se izvršnoj datoteci **ping** (čiji je vlasnik ***root***) dodaje zastavica *setuid* (npr. iz početne dozvole 0755 u 4755) da bi naredba `ping` imala pregled nad mrežnim paketima (tj. prava *root*) iako osnovni korisnik koji ju je pokrenuo to ne može.

Vrijednost 1 se zove i *sticky bit* i ima utjecaja samo na direktorije. Ako je postavljena ta dozvola nad direktorijem, pravo preimenovanja i brisanja datoteka u tom direktoriju imaju samo vlasnik datoteka (ili direktorija) i *root* (npr. 1777). Na primjer, iako će preko dozvola direktorija 1777 svaki korisnik moći postaviti datoteku u direktorij, samo će je vlasnik direktorija, vlasnik datoteke ili *root* moći obrisati. Čest je to slučaj u direktoriju **/tmp,** gdje svi mogu pisati, ali se ne preporuča da svaki korisnik može drugome brisati ili preimenovati datoteke.

Te dodatne dozvole postavljaju se naredbom `chmod`. Mogu se rabiti u opisanoj oktalnoj notaciji ili slovima:

X - omogućuje izvršavanje/pretraživanje samo ako se radi o direktoriju, ili ako je već postavljeno pravo izvršavanja za nekog korisnika

s - postavlja *bit* skupine ili korisnika

t - *sticky bit.*

9.3.8. Naredba umask

Naredba `umask` rabi se za postavljanje predefiniranih ovlasti prilikom izrade datoteka i direktorija.

Sintaksa:

`umask` - prikazuje trenutačnu masku

`umask umask_vrijednost` – postavljanje vrijednosti maske

Pretpostavljena (*default*) vrijednost je (0)022. Vrijednost *umask* u oktalnom se obliku odbija od pretpostavljenih vrijednosti potpunog pristupa za direktorije (777) i datoteke (666).

Primjer: Nova će datoteka imati ovlasti 666(8)-022(8)=644(8), a novi direktorij ovlasti 777(8)-022(8)=755(8).

Znači, ako vrijednost maske nije mijenjana, sve će novoizrađene datoteke imati dozvole 644, a svi će direktoriji imati ovlasti 755.

U sljedećem primjeru prvo će se provjeriti stanje maske, zatim će se izraditi direktorij i provjeriti trenutačno stanje dozvola. Nakon toga će se promijeniti maska i izradit će se novi direktorij. Vidljivo je da je stanje dozvola drugačije.

```
$ umask
022
$ mkdir /tmp/test
$ ls -ald /tmp/test
drwxr-xr-x 2 root root 4096 May 18 13:30 /tmp/test
$ umask 222
$ mkdir /tmp/test2
$ ls -ald /tmp/test2
dr-xr-xr-x 2 root root 4096 May 18 13:30 /tmp/test2
```

Vrijednost maske se poništi (*reset*) svakom prijavom, a za trajno zadržavanje vrijednosti treba ju postaviti u inicijalizacijsku datoteku.

9.3.9. Atributi

Postoje još neki posebni atributi koji se mogu postaviti nad datotekama i direktorijima.

Atribut	Opis
A	Zabranjuje promjenu vremena posljednjeg pristupa
a	Dozvoljava isključivo dodavanje novih podataka u datoteku, ali ne i izmjenu ili brisanje starih, ako je datoteka otvorena u režimu čitanja.
c	Datoteka s atributom c smješta se na disk u komprimiranom obliku, pri čemu kompresiju obavlja jezgra operacijskog sustava; prilikom čitanja, jezgra najprije obavlja dekompresiju datoteke.
d	Datoteka nije kandidat za sigurnosnu pohranu (*backup*) koja se vrši naredbom `dump`
i	Datoteka se ne može mijenjati ili brisati, ne može joj se promjeniti ime niti se može izraditi link koji pokazuje na tu datoteku.
j	Svi podaci se prvo ažuriraju u dnevniku, a zatim u datoteci, pod uvjetom da se rabi režim dnevnika *ordered* ili *writeback*.
s	Prilikom brisanja datoteke u sve blokove koji čine datoteku upisuju se nule.
S	Prilikom izmjene sadržaja datoteke promene se odmah upisuju na disk.
t	Neiskorišteni fragmenti poslednjeg bloka datoteke ne mogu se dodijeliti drugoj datoteci na korišenje.
u	Prilikom brisanja datoteke čuva se njezin sadržaj, čime je omogućen povratak obrisane datoteke.

Naredbe su za upravljanje atributima `lsattr` i `chattr`.

U sljedećem će se primjeru prvo izraditi datoteka naredbom `touch`, zatim će se provjeriti stanje atributa nad tom datotekom naredbom `lsattr`. Nakon toga će se dodati atribut i (koji govori da se datoteka ne može mijenjati ili obrisati) naredbom `chattr`. Zatim će se naredbom `ls` pokušati obrisati ta datoteka. Očito je da je naredba `rm` ne može obrisati dok se ne makne atribut **i**.

```
# touch testfile
# lsattr testfile
---------------- testfile
# chattr +i testfile
# rm -f testfile
rm: cannot remove `testfile': Operation not permitted
# chattr -i testfile
# rm -f testfile
# ls testfile
ls: cannot access testfile: No such file or directory
```

Vježba 8: Datotečni sustavi

Napomena

Ovu vježbu potrebno je izvoditi s ovlastima korisnika *root*. U terminal je potrebno upisati:

`su –` pa lozinku korisnika *root* dodijeljenu prilikom instalacije.

1. Izradite dvije nove particije na disku **/dev/sdb** koristeći se naredbom `fdisk`. Ako ste ih izradili u prošloj vježbi, možete se koristiti tim dvijema već napravljenim particijama.
2. Izradite na jednoj particiji datotečni sustav *ext2*, a na drugoj *ext3*. Sintaksa je `mkfs –t <tip> /dev/sdbX`.

```
mkfs -t ext2 /dev/sdb1
mkfs -t ext3 /dev/sdb5
```

3. Napravite dva direktorija na koje ćete montirati nove datotečne sustave. Direktoriji su **/mnt/ext2** i **/mnt/ext3**.

```
mkdir /mnt/ext2
mkdir /mnt/ext3
```

4. Montirajte datotečne sustave koristeći se naredbom `mount`. Sintaksa je `mount /dev/sdbX <točka montiranja>`.

```
mount /dev/sdb1 /mnt/ext2
mount /dev/sdb5 /mnt/ext3
```

5. Koristeći se naredbama `mount` i `df` provjerite jesu li datotečni sustavi ispravno montirani.
6. Koristeći se naredbom `blkid` ispišite sve raspoložive particije i pripadajuće UUID-ove.

   `blkid`

7. Odmontirajte (*unmount*) datotečne sustave i koristeći se naredbom `fsck` provjerite stanje datotečnog sustava na particiji **/dev/sdb1**.

```
umount /dev/sdb1
fsck /dev/sdb1
```

8. Koristeći se naredbom `dumpe2fs` ispišite sve informacije o datotečnom sustavu na particiji **/dev/sdb1**.

   `dumpe2fs /dev/sdb1`

9. Pretvorite datotečni sustav *ext2* (na particiji **/dev/sdb1**) u ***ext3*** koristeći se naredbom `tune2fs`.

   `tune2fs –j /dev/sdb1`

10. Montirajte datotečni sustav i provjerite je li sada ***ext3***.

Vježba: Dozvole nad datotekama i direktorijima

1. Prijavite se u sustav. Koristeći se naredbom `touch` napravite proizvoljnu datoteku i provjerite jesu li dozvole 644.
2. Promijenite **umask** na vrijednost 027. Izradite novu proizvoljnu datoteku i provjerite koje su sad dozvole.
3. Dodajte dva korisnika na sustav.

```
useradd -m korisnik1
useradd -m korisnik2
```

4. Promijenite lozinke korisnicima **korisnik1** i **korisnik2** koristeći se naredbom `passwd`.

```
passwd korisnik1
passwd korisnik2
```

5. Dodajte grupu ***prodaja*** koristeći se naredbom `groupadd`.

```
groupadd prodaja
```

6. Dodajte novootvorene korisnike u grupu ***prodaja***.

```
gpasswd –a korisnik1 prodaja
gpasswd –a korisnik2 prodaja
```

7. Izradite direktorij **/news**, postavite da su u vlasništvu grupe ***prodaja*** i postavite dozvole da grupa može čitati i pisati u tom direktoriju.

```
mkdir /news
chown .prodaja /news
chmod 770 /news
```

8. Prijavite se u sustav kao korisnik **korisnik1** i provjerite možete li izraditi datoteke u direktoriju **/news**.
9. Prijavite se u sustav kao korisnik **korisnik2** i provjerite možete li izraditi datoteke u direktoriju **/news**.

Vježba: Atributi nad datotekama i direktorijima

1. U trenutačnom direktoriju izradite datoteku **testna_datoteka.txt**.

   ```
   touch testna_datoteka.txt
   ```

2. Koristeći se naredbom `lsattr` provjerite koji su atributi podešeni nad datotekom **testna_datoteka.txt**.

   ```
   lsattr testna_datoteka.txt
   ```

3. Koristeći se naredbom `chattr` dodajte atribut kojim datoteka ne može biti mijenjana ili obrisana (atribut **i**).

   ```
   chattr +i testna_datoteka.txt
   ```

4. Koristeći se naredbom `lsattr` provjerite je li taj atribut dodan.

   ```
   lsattr testna_datoteka.txt
   ```

5. Koristeći se naredbom `rm` pokušajte obrisati datoteku.

   ```
   rm -f testna_datoteka.txt
   ```

6. Možete li je obrisati? _____

   Uklonite atribut i probajte je ponovno obrisati.

Kvote

1. Naredbom apt-get potrebno je instalirati paket quota koji sadrži alate koji služe za podešavanje kvota:

   ```
   apt-get install quota
   ```

2. U datoteku **/etc/fstab** za **particiju /dev/sdb1** podesite kvotu. Potrebno je dodati ovu liniju na kraj te datoteke:

   ```
   /dev/sdb1 /mnt ext2 defaults,nosuid,usrquota,grpquota 1 2
   ```

7. Nakon izmjene u **/etc/fstab** treba ponovo montirati datotečni sustav koristeći se naredbom `mount`.

   ```
   mount –a
   ```

8. Koristeći se naredbom `edquota` korisniku **korisnik1** iz prošle vježbe podesite kvotu.

   ```
   edquota –u korisnik1
   ```

Pitanja za ponavljanje

1. Kako je organiziran *Linux*ov datotečni sustav?

   _____

   _____

2. Čemu služi direktorij **/home**?

   _____

   _____

3. Kojom se naredbom formatira datotečni sustav?

   _____

   _____

4. Koja naredba služi za postavljanje predefiniranih ovlasti prilikom izrade datoteka i direktorija?

   _____

   _____

10. Upravljanje procesima

Trajanje poglavlja:
90 min

Po završetku ovoga poglavlja moći ćete:

- pregledavati pokrenute procese koristeći se naredbama `ps` i `top`
- upravljati procesima koristeći se naredbama `kill`, `killall`, `nice` i `renice`
- upravljati poslovima koristeći se naredbama `bg`, `fg` i `jobs`.

Ova cjelina obrađuje što je to proces i kako upravljati procesima u operacijskom sustavu Linux. Na kraju cjeline biti će obrađeno i upravljanje poslovima u operacijskom sustavu Linux.

10.1. Upravljanje procesima

11.1.1. Proces

Linux upravlja poslovima koristeći se procesima. Svakom se procesu pri pokretanju dodjeljuje **jedinstveni identifikacijski broj** (PID – *Process Identification Number*). Proces može kreirati podprocese i tako stvarati hijerarhijsku strukturu s odnosom roditelj – dijete. Neke jednostavne naredbe koje su ugrađene u ljusci ne kreiraju odvojeni proces. Primjer je naredba `cd`.

Pri pokretanju operacijskog sustava prvi se pokreće proces **systemd** s PID-om **1** koji inicializira ostale procese. Na starijim distribucijama Linuxa (npr. do Debiana 8), taj proces se zvao **init**.

Procesi se dijele prema nekoliko kriterija:

daemon - proces koji postoji zbog specifične uloge (npr. *Apache daemon* za servis http), pokreće se u pozadini i neaktivan je dok ih se ne pozove

parent - proces koji kreira druge procese; svaki proces osim procesa **init** ima roditeljski proces

child - pokreće ga drugi, roditeljski proces s oznakom PPID (*parent* PID)

orhpan - aktivni proces čiji je roditeljski proces prekinut; takav proces preuzima proces **init** koji mu postaje roditeljski

zombie (defunct) - *child*-proces koji se sa svojim izlaznim podacima ne vraća roditeljskom procesu i ostaje „izgubljen“ u sustavu; može se izbrisati iz tablice procesa jedino ponovnim pokretanjem (*restart*) operacijskog sustava.

10.1.2. Stablo procesa

Naredba `pstree` služi za ispis stabla procesa. U ispisu se vidi da je proces **systemd** glavni proces bez svojeg roditelja. Roditeljski proces **apache2** ima petero djece, proces **pstree** koji prikazuje to stablo je dijete procesa **zsh**, a proces **zsh** je dijete procesa sshd, koji ima još jednog roditelja **sshd**. Svim je tim procesima roditelj **systemd**.

```
$ pstree
systemd─┬─acpid
        ├─atd
        ├─atop
        ├─5*[getty]
        ├─in.tftpd
        ├─inetd
        ├─master─┬─pickup
        │        ├─qmgr
        │        └─tlsmgr
        ├─apache2───5*[apache2]
        ├─cron
        ├─ntpd
        ├─rsyslogd───4*[{rsyslogd}]
        ├─sshd
        ├─sshd───sshd───zsh───pstree
        ├─udevd───2*[udevd]
        └─vsftpd
```

10.1.3. Naredba ps

Naredba `ps` prikazuje popis aktivnih procesa.

Sintaksa je:

```
$ ps [opcije]
```

Najčešće se rabe opcije prikazane u tablici:

Opcija	Značenje
ps	Prikazuje informacije o svim procesima trenutačnog korisnika u trenutačnoj ljusci.
ps -e	Prikazuje informacije o svim procesima svih korisnika.
ps -f	Prikazuje sve raspoložive informacije o procesima trenutačnog korisnika.
ps -u userid	Prikazuje informacije o procesima određenog korisnika.
ps -ef	Prikazuje sve raspoložive informacije o svim procesima svih korisnika.

Primjer je uporabe naredbe `ps` u kojem se prikazuju svi procesi svih korisnika:

```
# ps -ef
UID PID PPID C STIME TTY     TIME CMD
root  1    0 0  2014 ?   00:08:24 init [2]
root  2    0 0  2014 ?   00:00:00 [kthreadd]
root  3    2 0  2014 ?   00:24:50 [ksoftirqd/0]
root  6    2 0  2014 ?   00:00:00 [migration/0]
root  7    2 0  2014 ?   00:03:42 [watchdog/0]
root  8    2 0  2014 ?   00:00:00 [cpuset]
root  9    2 0  2014 ?   00:00:00 [khelper]
root  10   2 0  2014 ?   00:00:00 [kdevtmpfs]
...
```

Objašnjenja stupaca opisana su sljedećom tablicom.

Vrijednost	Značenje
UID	Jedinstveni identifikacijski broj vlasnika procesa (*User Identification Number*).
PID	Jedinstveni broj procesa (*Process Identification Number*).
PPID	Jedinstveni broj procesa roditelja (*Parent Process Identification Number*).
C	Prioritet procesa.
STIME	Vrijeme pokretanja procesa.
TTY	Oznaka terminala gdje je proces pokrenut.
TIME	Ukupna količina procesorskog vremena koje je proces zauzeo.
CMD	Ime programa koji je pokrenuo proces.

10.1.4. Naredba top

Procesi se u realnom vremenu mogu pratiti naredbom `top`. Naredba ispisuje podatke koliko je dugo računalo uključeno, koliko je opterećenje računala (*load average*), podatke o broju procesa i raspoloživim resursima poput procesora i memorije. Zatim slijedi detaljan popis procesa sličan rezultatu naredbe `ps`.

U nastavku je dan prikaz izvršavanja naredbe `top`:

```
top - 11:36:27 up 19 days, 16:32,  7 users,  load average: 0.38, 0.39, 0.35
Tasks: 262 total,   1 running, 261 sleeping,   0 stopped,   0 zombie
Cpu(s): 27.4%us,  1.5%sy,  0.0%ni, 71.1%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:   8143932k total,  8051184k used,    92748k free,   517532k buffers
Swap:  5787636k total,   185156k used,  5602480k free,  1774952k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
32516 www-data  20   0  536m  80m  21m S   42  1.0   0:22.41 apache2
14857 mysql     20   0  756m 273m 5364 S   14  3.4 184:31.95 mysqld
25909 root      20   0 19204 1524 1028 R    2  0.0   0:00.44 top
    1 root      20   0 10428  684  648 S    0  0.0   0:09.91 init
    2 root      20   0     0    0    0 S    0  0.0   0:00.00 kthreadd
    3 root      RT   0     0    0    0 S    0  0.0   0:02.25 migration/0
    4 root      20   0     0    0    0 S    0  0.0   2:53.85 ksoftirqd/0
    5 root      RT   0     0    0    0 S    0  0.0   0:00.00 watchdog/0
    6 root      RT   0     0    0    0 S    0  0.0   0:02.46 migration/1
    7 root      20   0     0    0    0 S    0  0.0   0:36.84 ksoftirqd/1
    8 root      RT   0     0    0    0 S    0  0.0   0:00.00 watchdog/1
    9 root      20   0     0    0    0 S    0  0.0   0:14.65 events/0
   10 root      20   0     0    0    0 S    0  0.0   0:21.49 events/1
   11 root      20   0     0    0    0 S    0  0.0   0:00.00 cpuset
   12 root      20   0     0    0    0 S    0  0.0   0:00.00 khelper
   13 root      20   0     0    0    0 S    0  0.0   0:00.00 netns
```

10.1.5. Signali procesa

Procesi se mogu zaustaviti slanjem **signala** procesima. Postoje 63 različita signala. Signal se rabi za obavještavanje procesa ili procesne niti o nekom događaju. Svaki signal ima svoj jedinstveni naziv tj. kraticu koja počinje sa SIG (npr. SIGINT) i odgovarajući broj te po primitku signala proces reagira na određeni način.

Naredba `kill` služi za slanje određenog signala procesu.

Sintaksa je naredbe `kill`:

```
$ kill SIGNAL PID_procesa
```

Predodređeni je signal koji se šalje pokretanjem naredbe `kill` je SIGTERM s vrijednosti 15. „Ubijanjem“ roditeljskog procesa „ubijaju se“ i procesi koje je taj proces pokrenuo. Najčešće su korišteni signali prikazani u tablici

Signal	Kod	Značenje
SIGHUP	1	Kod primanja ovog signala proces se obično pokrene iznova s istim PID-om, ponovno učitavajući svoje konfiguracijske datoteke.
SIGINT	2	Šalje se signal procesu da prekine svoje izvođenje. Istovjetno je pritiskanju kombinacije tipaka [Ctrl]+[C].
SIGKILL	9	Šalje se signal procesu da se odmah prekine. Proces taj signal ne može ignorirati.
SIGTERM	15	Šalje se signal procesu i proces sam sebe prekida. Proces taj signal može i ignorirati.
SIGSTOP	17	Šalje signal procesu i proces se privremeno zaustavlja. Proces taj signal ne može ignorirati.

U sljedećem se primjeru naredbom `ps` provjerava postoji li proces **vsftpd**, zatim se šalje signal SIGKILL (9) (prekidanje procesa) te se na kraju provjerava je li proces zaustavljen, tj. postoji li još uvijek.

```
# ps -ef | grep vsftpd
root 2181 31984 0 17:56 pts/0 00:00:00 grep vsftpd
root 27529 1 0 2014 ? 00:00:00 /usr/sbin/vsftpd
# kill -9 27529
# ps -ef | grep vsftpd
root 2183 31984 0 17:56 pts/0 00:00:00 grep vsftpd
#
```

Postoji i naredba `killall` koja zaustavalja procese prema imenu, bez poznavanja PID-a procesa.

Sintaksa je ovakva:

```
$ killall SIGNAL ime_procesa
```

U sljedećem se primjeru ostvarila ista funkcionalnost kao i u prošlom, uz tu razliku da se umjesto naredbe `kill` rabila naredba `killall`.

```
# ps -ef | grep vsftpd
root 2199 1 0 17:58 ? 00:00:00 /usr/sbin/vsftpd
root 2201 31984 0 17:58 pts/0 00:00:00 grep vsftpd
# killall vsftpd
# ps -ef | grep vsftpd
root 2204 31984 0 17:58 pts/0 00:00:00 grep vsftpd
#
```

10.1.6. Niceness i prioritet izvođenja procesa

Niceness određuje koliko će procesi često doći na red za izvođenje. Vrijednost se kreće od -20 (češće dolazi na red) do 19 (rjeđe dolazi na red). ***Niceness* nije isto što i prioritet** - sustav dodjeljuje prioritet na temelju *nicenessa* kojeg zadaje korisnik i to najčešće tako da pribraja *niceness* na zadani prioritet procesa, ali *ne mora biti tako.*

Većina korisničkih programa ima isti ***niceness***, 0 (nula). Procesi prioriteta *realtime* imaju prednost nad ostalima bez obzira na *niceness*.

Korisnici, osim korisnika ***root***, mogu postaviti vrijednosti **od 0 do 19** (ta je postavka predodređena, regulira se u konfiguracijskoj datoteci **/etc/security/limits.conf**).

Postoje dvije naredbe za podešavanje prioriteta procesa:

naredba `renice` mijenja *niceness* u odnosu na trenutačni, radi na već pokrenutim procesima

naredba `nice` mijenja *niceness* u odnosu na zadani, koristi se kod pokretanja procesa.

Slijedi sintaksa naredbe `renice`. **NI** je *niceness* procesa, a **PID** je njegov jedinstveni identifikacijski broj.

renice <+/-NI> -p <PID>

Slijedi sintaksa naredbe `nice`. Izvršavanjem naredbe pokrenut će se proces s određenim prioritetom.

nice -<NI> <proces>

U sljedećem primjeru najprije će se naći PID procesa **vsftpd** naredbom `ps`, a zatim će se promijeniti prioritet tog procesa naredbom `renice`:

```
# ps -ef | grep vsftpd
root 30861 1 0 13:12 ? 00:00:00 /usr/sbin/vsftpd
root 30869 31984 0 13:12 pts/0 00:00:00 grep vsftpd
# renice -5 30861
30861 (process ID) old priority 0, new priority -5
```

U sljedećem primjeru pokrenut će se proces **vsftpd** s prioritetom -5, zatim će se prioritet promijeniti na +10.

```
# nice --5 vsftpd &
[1] 31083
# renice +10 -p 31083
31083 (process ID) old priority -5, new priority 10
```

10.1.7. Procesi i ljuska

Pokretanjem procesa ljuska stvara posao (*job*). Posao procesu pridjeljuje atribute kao što su terminal kojem proces (posao) pripada te ulazne i izlazne uređaje (*stdin, stdout, stderr*). Posao može objediniti više procesa koji su medusobno ovisni.

Na primjer, procesi vezani *pipeom* (|) objedinjeni su u jedan posao:

$ cat /etc/passwd | grep korisnik

Posao ima vlastiti identifikator - *Job ID* (JID) kojeg dodjeljuje ljuska. U svakoj pokrenutoj ljusci brojanje JID-ova počinje od 1 (u dvije ljuske može postojati više istih JID-ova). Zatvaranjem ljuske završavaju se svi poslovi u njoj.

Posao pokrenut u ljusci može biti u dva načina rada:

foreground - prednji plan (fokus) u ljuski

background - rad u pozadini.

Kod pokretanja procesa iz terminala ljuska postavlja posao u *foreground*. Za pokretanje procesa u pozadini rabi se operator **&**.

Slijedi primjer pokretanja naredbe `find` u pozadini. Ljuska ispisuje JID i PID procesa.

```
# find / -name passwd &
[1] 2447
#
```

Treba naglasiti da proces u pozadini oslobađa *prompt* i mogu se upisivati nove naredbe.

Popis poslova koji se izvode u trenutačnoj ljusci dobije se naredbom `jobs`.

```
# jobs
[1] + running find / -name passwd
#
```

Proces koji je trenutačno u prednjem planu može se suspendirati slanjem signala SIGTSTP (kombinacijom tipki **[Ctrl] + [Z]**).

```
# find / -name passwd
/etc/passwd
^Z
bash: suspended find / -name passwd
#
```

Ljuska ispisuje JID suspendiranog posla. Posao se može nastaviti naredbama:

`fg` - u prednjem planu

`bg` - u pozadini

Posao se može staviti u prednji plan pozivanjem naredbe `fg`:

```
# fg
[1] + running find / -name passwd
```

Prompt nije oslobođen, jer se posao „vrti“ u prednjem planu. Ako se posao želi prebaciti u pozadinu i ostaviti da se izvršava, treba pritisnuti kombinaciju tipki **[Ctrl] + [Z]** i pokrenuti naredbu `bg`:

```
# fg
[1] + running find / -name passwd
^Z
bash: suspended find / -name passwd
# bg
[1] + continued find / -name passwd
#
```

Prompt je opet oslobođen, posao se „vrti“ u pozadini. Ako se se proces želi prekinuti, to se može naredbom `kill` tako da se u argument stavi JID ispred kojeg se nalazi znak %:

```
# kill %1
[1] + terminated find / -name passwd
```

Ako se želi da neki program nastavi raditi i nakon što se korisnik odjavi sa sustava, tada taj program treba pokrenuti pomoću naredbe `nohup`. Kao argument se stavlja naredba koja se želi pokrenuti.

U sljedećem će primjeru naredba `nohup` pokrenuti program `obrada` koji će spremiti rezultate u datoteku rezultati.

```
$ nohup obrada > rezultati &
```

Vježba 9: Upravljanje procesima

1. Koristeći se naredbom `pstree` ispišite stablo svih aktivnih procesa. Možete rabiti mogućnost `-p` da se ispišu i identifikatori procesa.
   Koji je proces glavni, roditeljski svim drugim procesima?

   _____

2. Koristeći se naredbom `ps` ispišite sve aktivne procese. Nađite proces **init** i provjerite mu identifikator.

   ```
   ps –ef | grep init
   ```

3. Pokrenite naredbu `top` i pogledajte aktivne procese.
   Iz programa top izlazi se pritiskom na tipku **[q]**.

4. Pokrenite **vi**. Privremeno zaustavite rad tog posla tipkama [Ctrl]+[z]. Tako taj posao ostaje u pozadini.

5. Naredbom `jobs` provjerite koji su aktivni poslovi.

6. Pokrenite **xeyes**. Također ga stavite u pozadinu. Ponovno se koristeći naredbom `jobs` provjerite koji su aktivni poslovi.

7. Koristeći se naredbom `bg` nastavite rad procesa **xeyes**.

8. Koristeći se naredbom `fg` vratite proces **vi** u prednji plan.

9. Koristeći se naredbom `kill` ubijte procese **vi** i **xeyes**.

10. Koristeći se naredbom `nice` ponovno pokrenite **xeyes**, ali ovog puta prilikom pokretanja postavite ***niceness*** na vrijednost 5.

    ```
    nice -5 xeyes
    ```

11. Koristeći se naredbom `renice` promijenite ***niceness*** procesu **xeyes** na 10.

    ```
    renice -n 10 <PID>
    ```

Ukoliko želite izaći iz programa **xeyes** dovoljno je pritisnuti tipke [Ctrl]+[c] u terminalu gdje je pokrenut program. Ta kombinacija tipki šalje procesu signal SIGINT čime proces prekida svoje izvođenje.

Pitanja za ponavljanje

1. Koji je ekvivalentni signal kombinaciji tipka [Ctrl]+[c]?

   _____

   _____

2. Koji signal služi da bi se neki proces ponovno pokrenuo i učitao svoje konfiguracijske datoteke?

   _____

   _____

3. Koji je podrazumijevani signal koji se šalje procesima, koristeći se naredbom `kill`?

   _____

   _____

   Koje signale proces ne može ignorirati?

   _____

11. Instalacija softvera

Trajanje poglavlja:
80 min

Po završetku ovoga poglavlja moći ćete:

- instalirati softver iz izvornog kôda
- razlikovati statične i dijeljene (dinamičkih) knjižnice
- koristiti se naredbama `ldd` i `file`
- koristiti se Debianovim paketnim sustavom
- koristiti se Red Hatovim paketnim sustavom.

Ova cjelina obrađuje osnove instalacije softvera iz izvornog kôda te razlike između statičnih i dijeljenih biblioteka. U drugom dijelu cjeline obradit će se dva najčešća sustava za upravljanje paketima: Debianov paketni sustav dpkg i Red Hatov paketni sustav rpm.

11.1. Instalacija iz izvornog koda

11.1.1. Uvod

Kad se izrađuje neki softver (jezgra operacijskog sustava, program, igra, alat...), on se izrađuje u nekom od programskih jezika (najčešće C, ali ima i drugih). Tako pisani softver prepoznatljiv je samo ljudima i to onima koji razumiju taj programski jezik, no takav kôd je potpuno nepoznat računalu i on ga ne zna izvršavati. Da bi se programi mogli izvršavati, moraju se iz izvornog kôda prevesti u izvršni kôd. Taj se proces zove **kompajliranje**.

Slijedi primjer jednostavnog programa koji na ekran ispisuje "Hello World!". Sastoji se od dviju datoteka - **main.c** i **Hello.c**:

Glavni dio programa u datoteci **main.c**:

```
#include <stdlib.h>
int main() {
  Hello();
}
```

Funkcija koja ispisuje tekst na ekran u datoteci **Hello.c**:

```
#include <stdio.h>
void Hello() {
  printf("Hello World!\n");
}
```

Prevođenje se provodi naredbom `gcc` (GNU C *Compiler*). Potrebno je prevesti obje datoteke i spojiti ih u jednu izvršnu. Nakon prevođenja program je pokrenut:

```
$ gcc -c main.c
$ gcc -c Hello.c
$ gcc -static -o app main.o Hello.o
$ ./app
Hello World!
```

Velika većina programa za *Linux* distribuira se u obliku izvornog kôda pisanog u programskom jeziku C, a u arhivi se nalazi i datoteka **Makefile** koja služi za automatizirano prevođenje programa.

Gornji se postupak može automatizirati pomoću naredbe `make`, ali prije toga treba pripremiti datoteku **Makefile**:

```
SHELL=/bin/sh
CC = /usr/bin/gcc
app: main.o Hello.o
   $(CC) -static -o app main.o Hello.o
main.o: main.c
   $(CC) -c main.c
Hello.o: Hello.c
   $(CC) -c Hello.c
```

Zatim je dovoljno pokrenuti naredbu `make`:

```
$ make
/usr/bin/gcc -c main.c
/usr/bin/gcc -c Hello.c
/usr/bin/gcc -static -o app main.o Hello.o
$ ./app
Hello World!
```

11.1.2. Statične i dijeljene knjižnice

Obično se funkcije kojima se program koristi stavljaju u posebne datoteke, **knjižnice**. Prilikom prevođenja te knjižnice mogu biti spojene na glavni program:

statično: cijela funkcija je ugrađena u glavni program

dijeljeno (dinamički): funkcija se nalazi u posebnoj datoteci (knjižnici) i program ju učitava prema potrebi.

Naredbama `file` i `ldd` može se vidjeti je li neki program statički ili dinamički preveden. Iz izlaza naredbi (u nastavku) vidljivo je da je program preveden statički, tj. sve se nalazi u jednoj izvršnoj datoteci.

```
$ file app
app: ELF 64-bit LSB executable, x86-64, version 1 (GNU/Linux),
statically linked, for GNU/Linux 2.6.26,
```

```
BuildID[sha1]=0x02126ad69752e9768e102cf5f449e798d18a2038, not stripped
$ ldd app
 not a dynamic executable
```

Ako se funkcija **Hello()** želi staviti u posebnu knjižnicu, potrebno je pokrenuti naredbe `gcc` s ovim opcijama:

```
$ gcc -c -fPIC Hello.c
$ gcc -shared -o libfoo.so.1.0 Hello.o
```

Tako je napravljena knjižnica **libfoo.so.1.0**:

```
$ ls -al libfoo.so.1.0
-rwxr-xr-x 1 irako staff 6419 May 14 15:48 libfoo.so.1.0
```

Nakon toga treba prevesti glavni program i povezati ga s knjižnicom:

```
$ gcc -o app-shared main.c libfoo.so.1.0
$ ./app-shared
Hello World!
```

Naredbama `file` i `ldd` može se provjeriti kako je program povezan s knjižnicom. Iz primjera se vidi da je aplikacija **app-shared** povezana s knjižnicom **libfoo.so.1.0** i s nekoliko sistemskih knjižnica.

```
$ file app-shared
app-shared: ELF 64-bit LSB executable, x86-64, version 1 (SYSV),
dynamically linked (uses shared libs), for GNU/Linux 2.6.26,
BuildID[sha1]=0x44c0422139de934aeaaf7ad08538e70b97734d14, not stripped
$ ldd app-shared
linux-vdso.so.1 => (0x00007fff153ff000)
libfoo.so.1.0 (0x00007f617b977000)
libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f617b5e0000)
/lib64/ld-linux-x86-64.so.2 (0x00007f617bb7a000
```

Tako prevedena datoteka morala bi zauzimati na disku daleko manje mjesta jer u izvršnu datoteku nisu ubačene sve potrebne knjižnice. Naredbom `ls` provjerit će se njihove veličine:

```
$ ls -al app app-shared
-rwxr-xr-x 1 irako staff 788807 May 14 15:55 app
-rwxr-xr-x 1 irako staff   7122 May 14 15:48 app-shared
```

Zbog toga je velika većina programa na *Linux*u prevedena dinamičkim povezivanjem. Te dinamičke knjižnice nalaze se u direktorijima **/lib** i **/usr/lib**.

11.1.3. Arhiva s izvornim kodom

Projekti otvorenog kôda često se distribuiraju kao ***tarball***, kompresirana arhiva ***tar***. U tim se arhivama nalaze dokumentacija, izvorni kôd i sve potrebne skripte za prevođenje programa u izvršni kôd.

Nekompresirana arhiva ima nastavak **.tar**. Na primjer, ako je projekt razvijan u direktoriju **moj-projekt-1.0** on se obično zapakira u arhivu naredbom `tar`:

```
$ tar c moj-projekt-1.0 > moj-projekt-1.0.tar
```

Istovjetna je naredba (s opcijama cf (c -*create*, f -*file*).):

```
$ tar cf moj-projekt-1.0.tar moj-projekt-1.0
```

Kako su neki projekti prilično veliki, njihovo preuzimanje s mreže može trajati dugo i zbog toga su te arhive komprimirane.

Najčešći su programi za arhiviranje:

compress

gzip

bzip2

xz.

Alat za kompresiju	Alat za dekompresiju	Dekompresija s prikazom na ekran	Nastavak datoteke
compress	uncompress	zcat	.Z
gzip	gunzip	zcat	.gz
bzip2	bunzip2	bzcat	.bz2
xz	unxz	xzcat	.xz

Ti alati mogu komprimirati samo jednu datoteku, zato se cijeli projekt zapakira u jednu arhivu *tar*, i tada se kompresija radi jednim od ta četiri alata.

Znači izrada se arhive sastoji od dva koraka: od izrade arhive i njezina kompresiranja:

```
$ tar cf moj-projekt-1.0.tar moj-projekt-1.0
$ bzip2 moj-projekt-1.0.tar
```

To se može napraviti i samo naredbom `tar`, dodavanjem zastavice (**Z** za compress, **z** za gzip, **j** za bzip2 i **J** za xz):

```
$ tar cjf moj-projekt-1.0.tar.bz2 moj-projekt-1.0
```

U gornjem primjeru je korištena zastavica **j** za bzip2.

Otpakiravanje arhive obavlja se zastavicom **x**:

```
$ tar xjf moj-projekt-1.0.tar.bz2
```

11.1.4. Instalacija iz izvornog koda

Jednom kad se projekt otpakira, treba ga prevesti. Većina se projekata otvorenog kôda prevodi u tri koraka:

`configure` - skripta koja pregledava koja se arhitektura rabi i nalazi li se sve potrebno u sustavu te izrađuje datoteku **Makefile**

`make` - naredba koja čita **Makefile** i prevodi izvorni kôd u izvršni

`make install` - time se izvršni kod instalira na računalo, u direktorije u **/lib**, **/usr/lib**, itd.

Slijedi primjer instalacije programa **mboxgrep**. Prvo se pokreće `configure` koji provjeri nalazi li se sve potrebno u sustavu i izgenerira datoteku **Makefile**..

```
$ ./configure
checking for gcc... gcc
checking for C compiler default output... a.out
checking whether the C compiler works... yes
...
configure: creating ./config.status
config.status: creating Makefile
config.status: creating src/config.h
```

Zatim slijedi prevođenje izvornog kôda u izvršni kod naredbom `make`:

```
$ make
cd src; make
make[1]: Entering directory '~/mboxgrep-0.7.9/src'
gcc -g -O2 -I. -I. -c info.c
gcc -g -O2 -I. -I. -c main.c
gcc -g -O2 -I. -I. -c mh.c
gcc -g -O2 -I. -I. -c scan.c
gcc -g -O2 -I. -I. -c maildir.c
gcc -g -O2 -I. -I. -c mbox.c
gcc -g -O2 -I. -I. -c misc.c
gcc -g -O2 -I. -I. -c wrap.c
gcc -g -O2 -I. -I. -c getopt.c
gcc -g -O2 -I. -I. -c getopt1.c
gcc -g -O2 -I. -I. -c md5.c
gcc -g -O2 -o mboxgrep info.o main.o mh.o scan.o maildir.o mbox.o misc.o
wrap.o getopt.o getopt1.o md5.o -lz
```

Nakon uspješnog prevođenja, program treba instalirati pomoću naredbe `make install`.
Potrebno je naglasiti da taj korak treba pokrenuti korisnik *root* jer običan korisnik ne može pisati po direktorijima gdje se nalaze programi.

```
# make install
cd src; make install
make[1]: Entering directory ~/mboxgrep-0.7.9/src'
/usr/bin/install -c -d /usr/local/bin
```

```
/usr/bin/install -c -s mboxgrep /usr/local/bin
make[1]: Leaving directory '~/mboxgrep-0.7.9/src'
cd doc; make install
make[1]: Entering directory '~/mboxgrep-0.7.9/doc'
/usr/bin/install -c -d /usr/local/man/man1
/usr/bin/install -c -m 0644 mboxgrep.1 /usr/local/man/man1
/usr/bin/install -c -d /usr/local/info
/usr/bin/install -c -m 0644 mboxgrep.info /usr/local/info
make[1]: Leaving directory '~/mboxgrep-0.7.9/doc'
```

Nakon toga se može provjeriti je li program instaliran. Program se može pokrenuti:

```
$ ls -al /usr/local/bin/mboxgrep
-rwxr-xr-x 1 root root 32632 May 14 16:19 /usr/local/bin/mboxgrep
$ /usr/local/bin/mboxgrep 16:25
Usage: mboxgrep [OPTION] PATTERN MAILBOX ...

Try `mboxgrep --help' for more information.
```

11.2. Upravljanje paketima

11.2.1. Programski paketi

Programski paket je skup izvršnih, konfiguracijskih, bibliotečnih i dokumentacijskih datoteka, podešenih tako da instalacijom omogućuju osnovnu funkcionalnost programa koji je zapakiran.

U odnosu na distribuiranje izvornog kôda, paketi imaju više prednosti:

podešavanje programa za vlastite potrebe

čistoća sustava

lakše održavanje sustava

jako olakšana nadogradnja (više) sustava

promjena konfiguracija na više računala jednokratnim procesom.

Znači, umjesto da krajnji korisnik mora prevoditi sve programe na svojem operacijskom sustavu, to za njega rade održavatelji paketa. Oni za njega prilagode i prevedu program, upakiraju ga u programski paket.

11.2.2. Debianov paketni sustav

Naredba `dpkg` je sustav održavanja paketa za *Debian GNU/Linux*. Ime **dpkg** dolazi od **D**ebian **P**ac**k**a**g**e. Nastavak za *Debian*ove programske pakete je **.deb**.

Paket se obično imenuje ovako:

<ime_paketa>_<verzija>_<arhitektura>.deb

Primjeri su imenovanja nekih *Debian*ovih paketa:

openssl_0.9.7_amd64.deb

freeradius_2.0-1_i386.deb

aosi-aai_3.2.1_all.deb

*Debian*ov paket je **ar arhiva** od dvije arhive:

data.tar.gz - nalaze se podaci koji dolaze s paketom (libovi, binovi, dokumentacija...)

control.tar.gz - nalazi se sve o paketu i instalaciji paketa (razne skripte (preinst, postinst, prerm, postrm), control, conffiles...). Datoteke iz control.tar.gz se poslije instalacije paketa nalaze u **/var/lib/dpkg/info/<paket>.<datoteka>**.

S paketom dolaze i instalacijske skripte koje se pokreću prije ili poslije instalacije ili brisanja paketa. Te su skripte:

preinst – prije instalacije paketa

postinst – poslije instalacije paketa

prerm – prije brisanja paketa

postrm – poslije brisanja paketa.

Tijek je instalacije paketa:

otpakiraju se kontrolne datoteke (iz **control.tar.gz**) u **/var/lib/dpkg/info**

ako postoji starija inačica istog paketa, pokreće se skripta **prerm** starog paketa (ako postoji)

pokreće se skripta **preinst** novog paketa (ako postoji)

otpakiraju se podaci (iz **data.tar.gz**)

ako postoji starija inačica istog paketa, pokreće se skripta **postrm** starog paketa

pokreće se skripta **postinst** novog paketa.

Tijek je brisanja paketa:

pokreće se skripta **prerm**

obrišu se datoteke koje dolaze s paketom

pokreće se skripta **postrm**.

11.2.3. Naredba dpkg

Naredbom `dpkg` programski se paketi mogu instalirati, brisati ili se mogu dobiti informacije o već instaliranim paketima.

Najčešće su opcije naredbe `dpkg` prikazane u tablici.

Opcija	Duga opcija	Značenje
-l	--list	Prikazuje popis instaliranih paketa.
-s	--status	Prikazuje informacije o određenom paketu.
-I	--info	Prikazuje informacije o .deb datoteci.
-L	--listfiles	Prikazuje sadržaj instaliranog paketa.
-i	--install	Instalira ili nadograđuje (ako je paket već instaliran) paket iz datoteke .deb
-r	--remove	Briše paket, ali ostavlja konfiguracijske datoteke.
-P	--purge	Briše paket, skupa s konfiguracijskim datotekama.

Postoji razlika između opcija **--remove** i **--purge**. Opcija **--remove** briše paket, ali ostavlja konfiguracijske datoteke, a opcija **--purge** briše paket i konfiguracijske datoteke.

Slijedi primjer prikaza popisa instaliranih paketa. U prvom se stupcu nalazi stanje paketa (i znači da je uredno instaliran), u drugom stupcu nalazi se ime paketa, u trećem instalirana inačica, u četvrtom arhitektura, a u petom kratak opis paketa.

```
# dpkg -l
Desired=Unknown/Install/Remove/Purge/Hold
| Status=Not/Inst/Conf-files/Unpacked/halF-conf/Half-inst/trig-aWait/Trig-pend
|/ Err?=(none)/Reinst-required (Status,Err: uppercase=bad)
||/ Name Version Architecture Description
+++-==================-===========-======-===============================================
ii aacplusenc           0.17.5-dmo2 amd64 High-Efficency AAC (AAC+) Encoder.
ii accountsservice      0.6.37-3+b1 amd64 query and manipulate user account information
ii acl                  2.2.52-2    amd64 Access control list utilities
ii acpi                 1.7-1       amd64 displays information on ACPI devices
ii acpi-support-base    0.142-6     all   scripts for handling base ACPI events such as
the power button
ii acpid                1:2.0.23-2  amd64 Advanced Configuration and Power Interface
event daemon
ii adduser              3.113+nmu3  all   add and remove users and groups
...
```

Slijedi prikaz detaljnih inforamacija o paketu **ocsinventory-agent-srce**:

```
# dpkg -s ocsinventory-agent-srce
Package: ocsinventory-agent-srce
Status: install ok installed
Priority: optional
Section: net
Installed-Size: 52
Maintainer: Ivan Rako <irako@srce.hr>
Architecture: all
Version: 1:2.0.5~srce2
Depends: carnet-tools-cn (>= 2.7), ocsinventory-agent (>= 2:2.0.5), libcrypt-
ssleay-perl, libsys-hostname-long-perl, lsb-release
Conflicts: ocsinventory-agent-cn
Description: Hardware and software inventory tool (client)
 Open Computer and Software Inventory Next Generation is an
 application designed to help a network or system administrator to
 keep track of the hardware and software configurations of computers
```

```
 that are installed on the network. It also allows deploying
 software, scripts and files on client computers.
```

Opcijom **-L** prikazuje se popis svih datoteka koje paket donosi.

```
# dpkg -L ocsinventory-agent-srce
/.
/usr
/usr/share
/usr/share/doc
/usr/share/doc/ocsinventory-agent-srce
/usr/share/doc/ocsinventory-agent-srce/README.Srce
/usr/share/doc/ocsinventory-agent-srce/changelog.gz
/usr/share/doc/ocsinventory-agent-srce/copyright
/usr/share/perl5
/usr/share/perl5/Ocsinventory
/usr/share/perl5/Ocsinventory/Agent
/usr/share/perl5/Ocsinventory/Agent/Backend
/usr/share/perl5/Ocsinventory/Agent/Backend/OS
/usr/share/perl5/Ocsinventory/Agent/Backend/OS/Generic
/usr/share/perl5/Ocsinventory/Agent/Backend/OS/Generic/Hostname.pm
package diverts others to:
/usr/share/perl5/Ocsinventory/Agent/Backend/OS/Generic/Hostname.pm.divert
```

Opcijom -r paket se briše iz sustava:

```
# dpkg -r ocsinventory-agent-srce
(Reading database ... 70958 files and directories currently installed.)
Removing ocsinventory-agent-srce ...
A opcijom se -i instalira:
# dpkg -i ocsinventory-agent-srce_1%3a2.0.5~srce2_all.deb
(Reading database ... 70959 files and directories currently installed.)
Preparing to replace ocsinventory-agent-srce 1:2.0.5~srce2 (using ocsinventory-
agent-srce_1%3a2.0.5~srce2_all.deb) ...
Unpacking replacement ocsinventory-agent-srce ...
Setting up ocsinventory-agent-srce (1:2.0.5~srce2) ...
OCS Inventory server: https://mon.srce.hr/ocsinventory
```

11.2.4. Advanced Packaging Tool

Naredba `dpkg` dobra je za individualno instaliranje paketa bez međuovisnosti, ali kod paketa često jedan paket ovisi o drugom. Tada je za ispravan rad potrebno instalirati oba paketa. Isto tako, naredba `dpkg` ne zna raditi s repozitorijima paketa, za razliku od alata **APT**. APT može preuzeti zadnju inačicu paketa iz repozitorija, vidjeti koji su sve paketi potrebni za instalaciju, preuzeti sve pakete o kojima taj paket ovisi i instalirati ih.

Ime **APT** je skraćenica izraza *Advanced Packaging Tool*. S APT-om dolazi više alata (apt-cache, aptextracttemplates, apt-setup, apt-cdrom, aptftparchive, apt-show-source, apt-config, aptget...), a najpotrebnija su dva:

apt-get – alat za manipulaciju paketa.

apt-cache – alat za manipulaciju popisa paketa.

Najvažnije su konfiguracijske datoteke za **apt**:

/etc/apt/apt.conf - glavna konfiguracijska datoteka za APT.

/etc/apt/sources.list - popis repozitorija s kojih APT preuzima pakete.

11.2.5. Naredba apt-cache

Naredba za manipulaciju popisa paketa. Najčešće korištene opcije prikazane su u tablici:

Akcija	Opis
search *string*	Pretražuje lokalni popis dostupnih paketa i ispisuje rezultat.
show *paket*	Prikazuje sve informacije o dostupnom paketu.
depends *paket*	Prikazuje popis ovisnosti, tj. popis paketa o kojima je ovisan paket u argumentu.

Slijedi prikaz pretrage za paketom **ocsinventory-agent**.

```
# apt-cache search ocsinventory-agent
ocsinventory-agent-cn - Hardware and software inventory tool (client)
ocsinventory-agent - Hardware and software inventory tool (client)
ocsinventory-agent-srce - Hardware and software inventory tool (client)
```

Slijedi prikaz detaljnih informacija o traženom paketu.

```
# apt-cache show ocsinventory-agent-srce
Package: ocsinventory-agent-srce
Version: 1:2.0.5~srce2
Architecture: all
Maintainer: Ivan Rako <irako@srce.hr>
Installed-Size: 52
Depends: carnet-tools-cn (>= 2.7), ocsinventory-agent (>= 2:2.0.5), libcrypt-
ssleay-perl, libsys-hostname-long-perl, lsb-release
Conflicts: ocsinventory-agent-cn
Priority: optional
Section: net
Filename: pool/mon/o/ocsinventory-agent-srce/ocsinventory-agent-
srce_2.0.5~srce2_all.deb
Size: 2980
SHA256: 314bae63ee05eb1958941a444a0d518865551cbab2f4c34318a752d0f2811975
SHA1: eea9b6aedad179e04bc870b52a212a53b53c51bc
MD5sum: c34220493fd2b5f458e062b0bca90e2e
Description: Hardware and software inventory tool (client)
 Open Computer and Software Inventory Next Generation is an
 application designed to help a network or system administrator to
 keep track of the hardware and software configurations of computers
 that are installed on the network. It also allows deploying
 software, scripts and files on client computers.
```

11.2.6. Naredba apt-get

Naredba `apt-get` služi za manipulaciju paketima. Najčešće su opcije prikazane u tablici.

Akcija	Opis
update	Osvježavanje popisa paketa s repozitorija.
install *paket*	Instalacija određenog paketa.
upgrade	Nadogradnja svih paketa na novu inačicu.
dist-upgrade	Nadogradnja svih paketa na novu inačicu, rabi se za nadogradnju između distribucija.
remove *paket*	Brisanje određenog paketa.
clean	Brisanje arhive preuzetih paketa.

Najčešće korištene akcije su za nadogradnju postojećeg operacijskog sustava. Najprije treba osvježiti popis paketa s repozitorija naredbom `apt-get update`.

```
# apt-get update
Hit http://ftp.srce.hr srce-stretch Release.gpg
Hit http://ftp.srce.hr srce-stretch Release
Hit http://ftp.srce.hr srce-stretch/main Sources
Hit http://ftp.srce.hr srce-stretch/mon Sources
Hit http://ftp.srce.hr srce-stretch/main amd64 Packages
...
Reading package lists... Done
```

Zatim se nadogradi operacijski sustav pokretanjem naredbe `apt-get upgrade`:

```
# apt-get upgrade
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages will be upgraded:
 ocsinventory-agent-srce
1 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Need to get 0 B/2,980 B of archives.
After this operation, 0 B of additional disk space will be used.
Do you want to continue [Y/n]?
(Reading database ... 70959 files and directories currently installed.)
Preparing to replace ocsinventory-agent-srce 1:2.0.5~srce1 (using
.../ocsinventory-agent-srce_1%3a2.0.5~srce2_all.deb) ...
Unpacking replacement ocsinventory-agent-srce ...
Setting up ocsinventory-agent-srce (1:2.0.5~srce2) ...
OCS Inventory server: https://mon.srce.hr/ocsinventory
```

Brisanje se paketa vrši naredbom `apt-get remove`.

```
# apt-get remove ocsinventory-agent-srce
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages will be REMOVED:
 ocsinventory-agent-srce
0 upgraded, 0 newly installed, 1 to remove and 0 not upgraded.
After this operation, 53.2 kB disk space will be freed.
Do you want to continue [Y/n]?
(Reading database ... 70958 files and directories currently installed.)
Removing ocsinventory-agent-srce ...
```

Taj se paket može ponovno instalirati naredbom `apt-get install`:

```
# apt-get install ocsinventory-agent-srce 17:47
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
 ocsinventory-agent-srce
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 2,980 B of archives.
After this operation, 53.2 kB of additional disk space will be used.
Get:1 http://ftp.srce.hr/srce-debian/ srce-stretch/mon ocsinventory-
agent-srce all 1:2.0.5~srce2 [2,980 B]
Fetched 2,980 B in 0s (0 B/s)
Selecting previously unselected package ocsinventory-agent-srce.
(Reading database ... 70954 files and directories currently installed.)
Unpacking ocsinventory-agent-srce (from .../ocsinventory-agent-
srce_1%3a2.0.5~srce2_all.deb) ...
Setting up ocsinventory-agent-srce (1:2.0.5~srce2) ...
OCS Inventory server: https://mon.srce.hr/ocsinventory
```

11.2.7. RPM Package Manager

*Red Hat*ov paketni sustav ujedno je i najzastupljeniji paketni sustav za *Linux*. Prvobitna skraćenica za RPM je *Red Hat Package Manager*, a danas je rekurzivna skraćenica za *RPM Package Manager*. Ime RPM odnosi se na .rpm format datoteke, datoteke u tom formatu i na upravljanje paketima.

Imenovanje paketa slično je kao i na *Debian*u:

<ime_paketa>_<verzija>.<arhitektura>.rpm

Iako su neke kratke opcije slične, njihove različite akcije ovise o njihovoj poziciji u naredbenoj liniji. Prva opcija koja se daje naredbi `rpm` je glavna (*major*), druge su pomoćne (*minor*). Npr. u sljedećoj naredbi opcija **i** je glavna, a opcija **v** pomoćna:

```
rpm -iv paket.rpm
```

Tablica prikazuje popis glavnih načina rada (opcija) naredbe `rpm`:

Kratka opcija	Duga opcija	Značenje
-i	--install	Instalira paket.
-U	--update	Nadograđuje ili instalira paket.
-F	--freshen	Samo nadograđuje paket.
-V	--verify	Prikazuje podatke kao što su veličina paketa, dozvole, itd.
-q	--query	Ispituje instalirani ili neinstalirani paket.
-e	--erase	Deinstalira paket.

Sljedeća tablica prikazuje pomoćne opcije naredbe `rpm`.

Kratka opcija	Značenje
a	Odnosi se na sve instalirane pakete.
c	Zajedno s -q prikazuje popis konfiguracijskih datoteka.
d	Zajedno s -q prikazuje popis dokumentacije.
f	Zajedno s -q prikazuje kojem paketu pripada koja datoteka.
i	Zajedno s -q prikazuje informacije o određenom paketu.
l	Zajedno s -q prikazuje popis svih datoteka i direktorija u paketu.
p	Zajedno s -q prikazuje podatke o neinstaliranom paketu.
v	Opširniji prikaz.

Postoje tri načina pretrage. Može se ispitivati datoteka neinstaliranog paketa, instalirani paket ili samo datoteka koju je paket instalirao:

Način pretrage	Opcije
Neinstalirani paket	-qp
Instalirani paket	-q
Datoteka	-qf

U sljedećim ćemo primjerima razmatrat paket **zsh**. Prvo će se ispitati popis svih datoteka koje donosi paket koji još nije instaliran.

```
# rpm -qpl zsh-4.3.10-7.el6.x86_64.rpm
/bin/zsh
/etc/skel/.zshrc
/etc/zlogin
/etc/zlogout
/etc/zprofile
/etc/zshenv
/etc/zshrc
/usr/lib64/zsh
/usr/lib64/zsh/4.3.10
/usr/lib64/zsh/4.3.10/zsh
```

```
/usr/lib64/zsh/4.3.10/zsh/attr.so
...
```

Nakon tog paket ćemo instalirati.

```
# rpm -i zsh-4.3.10-7.el6.x86_64.rpm
```

Nakon toga će se provjeriti koje je sve datoteke paket instalirao:

```
# rpm -qpl zsh-4.3.10-7.el6.x86_64.rpm
/bin/zsh
/etc/skel/.zshrc
/etc/zlogin
/etc/zlogout
/etc/zprofile
/etc/zshenv
/etc/zshrc
/usr/lib64/zsh
/usr/lib64/zsh/4.3.10
/usr/lib64/zsh/4.3.10/zsh
/usr/lib64/zsh/4.3.10/zsh/attr.so
...
```

Zatim se sljedećom naredbom provjerava kojem paketu pripada datoteka **/bin/zsh**:

```
# rpm -qf /bin/zsh
zsh-4.3.10-7.el6.x86_64
```

Ako se paket želi obrisati, rabi se ova naredba:

```
# rpm -e zsh
```

Kad se želi dobiti popis svih instaliranih paketa, tada se rabi naredba:

```
# rpm -qa
sos-2.2-38.el6.centos.2.noarch
hal-info-20090716-3.1.el6.noarch
basesystem-10.0-4.el6.noarch
wireless-tools-29-5.1.1.el6.x86_64
libcurl-7.19.7-37.el6_4.x86_64
zsh-4.3.10-7.el6.x86_64
...
```

11.2.8. Yellowdog Updater, Modified

Kao i kod *Debian*a, i za *RPM* postoji program koji olakšava manipulaciju paketima. To je **yum** (*Yellowdog Updater, Modified*), uslužni program otvorenog kôda za upravljanje paketima naredbenom linijom za operacijske sustava *Linux* rabeći **rpm**.
Kao i kod APT-a, **yum** zna manupilirati s repozitorijima paketa kojima se može pristupati lokalno ili preko mreže. Najčešće opcije naredbe `yum` prikazane su u tablici.

Opcija	Značenje
install *paket*	Instalacija paketa.
update	Nadogradnja svih paketa na zadnju inačicu.
erase *paket*	Brisanje paketa.
remove *paket*	Brisanje paketa, isto kao opcija *erase*.
list	Prikaz popisa instaliranih paketa.
reinstall *paket*	Ponovno instaliranje paketa.
deplist *paket*	Prikaz popisa paketa o kojima je paket ovisan.

Slijedi primjer instalacije paketa **zsh** pomoću naredbe `yum`.

```
# yum install zsh
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package zsh.x86_64 0:4.3.10-9.el6 will be installed
--> Finished Dependency Resolution
...
Downloading Packages:
zsh-4.3.10-9.el6.x86_64.rpm | 2.1 MB 00:00
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
Installing : zsh-4.3.10-9.el6.x86_64 1/1
Verifying : zsh-4.3.10-9.el6.x86_64 1/1
Installed:
zsh.x86_64 0:4.3.10-9.el6
Complete!
```

Slijedi brisanje istog paketa.

```
# yum remove zsh
Resolving Dependencies
--> Running transaction check
---> Package zsh.x86_64 0:4.3.10-9.el6 will be erased
--> Finished Dependency Resolution
...
Running rpm_check_debug
```

```
Running Transaction Test
Transaction Test Succeeded
Running Transaction
 Erasing : zsh-4.3.10-9.el6.x86_64 1/1
 Verifying : zsh-4.3.10-9.el6.x86_64 1/1
Removed:
 zsh.x86_64 0:4.3.10-9.el6
Complete!
```

Vježba 10: Instalacija softvera

Vježba: Instalacija softvera iz izvornog kôda

1. Preuzmite arhivu s izvornim kodom programa **grep**.

   `cd /tmp; wget http://ftp.gnu.org/gnu/grep/grep-2.21.tar.xz`

2. Otpakirajte arhivu koristeći se naredom `tar`.

   `tar xfvJ grep-2.21.tar.xz`

3. Uđite u direktorij **grep-2.21** i pokrenite skriptu **configure** koja pregledava koja se arhitektura koristi, je li sve potrebno u sustavu i izrađuje datoteku **Makefile**. Dodajte mogućnost `--prefix` tako da odredište instalacije bude u **/tmp**, tako da se ne dira sistemski grep.

   ```
   cd grep-2.21
   ./configure --prefix=/tmp/grep
   ```

4. Pokrenite prevođenje izvornog koda u izvršni, koristeći se naredbom `make`.

   `make`

5. Instalirajte prevedeni program.

   `make install`

6. Provjerite što se nalazi u direktoriju **/tmp/grep/bin**. Koliko ima izvršnih datoteka?

   _____

7. Koristeći se naredbama `file` i `ldd` provjerite je li datoteka **/tmp/grep/bin/grep** dinamički ili statički prevedena u binarni kôd.

   _____

*Debian*ov paketni sustav

1. Koristeći se naredbom `dpkg` provjerite koji su paketi instalirani na računalu.

```
dpkg --list
dpkg -l
```

2. Koristeći se naredbom `apt-get` instalirajte programski paket **joe**. Programski paket **joe** je tekstni uređivač teksta.

```
apt-get install joe
```

3. Koristeći se naredbom `dpkg` provjerite je li paket uredno instaliran.

```
dpkg --list joe
dpkg -l joe
dpkg --status joe
dpkg -s joe
```

4. Koristeći se naredbom `apt-get` ili `dpkg` obrišite paket **joe**, ali da pri tom ostanu konfiguracijske datoteke.

```
dpkg --remove joe
dpkg –r joe
apt-get remove joe
```

5. Koristeći se naredbom `dpkg` provjerite je li je paket i dalje dostupan.
Primijetite da je status **rc** (paket obrisan, ali su ostale konfiguracijske datoteke). Provjerite koje su to konfiguracijske datoteke.

```
dpkg -s joe
dpkg --status joe
```

6. Naredbom `ls` provjerite postoji li stvarno konfiguracijska datoteka /**etc/joe/joerc**.

```
ls /etc/joe/joerc
```

7. Koristeći se naredbom `dpkg` ili `apt-get` obrišite i konfiguracijske datoteke.

```
dpkg --purge joe
dpkg -P joe
apt-get --purge remove joe
```

8. Koristeći se naredbom `ls` provjerite je li obrisana konfiguracijska datoteka **/etc/joe/joerc**.

```
ls /etc/joe/joerc
```

9. Korištenjem naredbe `apt-get` instalirajte paket **srce-keyring**.
Možete li ga instalirati? _____

10. Dodajte repozitorij Srca u konfiguracijsku datoteku **/etc/apt/sources.list**.

```
deb http://ftp.srce.hr/srce-debian/ srce-wheezy main
```

11. Osvježite lokalni popis paketa pomoću naredbe `apt-get update`.
12. Koristeći se naredbom `apt-cache` provjerite je li sada paket dostupan.

    ```
    apt-cache search srce-keyring
    ```

13. Instalirajte paket **srce-keyring**.

    ```
    apt-get install srce-keyring
    ```

Pitanja za ponavljanje

1. Koji je standardni postupak instalacije iz izvornog kôda?

   _____

   _____

1. U čemu je razlika između mogućnosti `--purge` i `--remove` programa `dpkg`?

   _____

   _____

2. Kako se zove *front-end* za *Debian*ov paketni sustav?

   _____

   _____

3. Koja su dva glavna alata *front-enda* za *Debian*ov paketni sustav?

   _____

   _____

4. U koju se datoteku upisuju repozitoriji *Debian*ovih paketa?

   _____

   _____