

Šesta beogradska gimnazija  
Milana Rakića 33  
Beograd

Maturski rad iz informatike  
Operativni sistem Linux

Mentor:  
Olivera Mihailović  
Profesor informatike

Učenik:  
Vojislav Lazić IV<sub>9</sub>

Beograd, jun 2019.

# Sadržaj

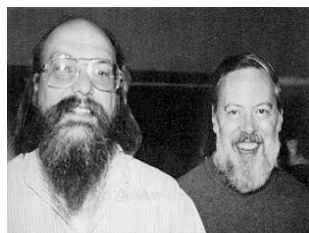
<b>1</b>	<b>Istorija Linuxa</b>	<b>3</b>
<b>2</b>	<b>Komponente Linux sistema</b>	<b>5</b>
2.1	Bootloader . . . . .	5
2.2	Kernel . . . . .	6
2.3	Daemoni . . . . .	7
2.4	Shell . . . . .	7
2.5	X window sistem . . . . .	7
2.6	Desktop okruženje . . . . .	8
<b>3</b>	<b>Osnovne komande u Linux-u</b>	<b>9</b>
<b>4</b>	<b>Hierarhija sistema datoteka</b>	<b>11</b>
<b>5</b>	<b>Redirekcija i "piping"</b>	<b>12</b>
<b>6</b>	<b>Dozvole i vlasništvo</b>	<b>13</b>
	<b>Literatura</b>	<b>15</b>

# 1 Istorija Linuxa

Za stvaranje Linux-a bilo je potrebno nekoliko komponenti, najvažnija od kojih je Unix. Unix su stvorili Ken Thompson i Denis Riči. Njih dvojica, zajedno sa timom inženjera u Belovim laboratorijama, su radili na Multics sistemu (Multiplexed Information and Computing Service), pravljen sa idejom da bude sistem koji može da radi više poslova u isto vreme. Thompson i Riči su u tom periodu počeli da rade na svom sopstvenom sistemu, zasnovan na Multics-u, po imenu Unix, prvi put objavljen 1970. godine. Kasnije, kad je C programski jezik, koji je Riči napisao zajedno sa Brijanom Kernigenom, postao dovoljno razvijen, Unix je potpuno prepisan u C-u, što je pomoglo njegovom rasprostranjenju u razne akademske institucije i poslove. Zbog promenljive i prilagodljive prirode Unix-a, razni univerziteti su počeli da prave svoje verzije Unix-a, jedan od najpopularnijih je bio BSD (Berkeley Software Distribution), koji je još u upotrebi danas.

1983. godine, Ričard Metju Stalman je započeo GNU projekat, namenjen da bude slobodna alternativa za Unix. Do ranih 90-ih, napisano je dovoljno softvera da se napravi citav operativni sistem. Jedino što je nedostajalo je "kernel" ili "jezgro" operativnog sistema, deo koji bi trebao sve ostale komponente da spoji. GNU je imao, i još ima, u pravljenju svoje kernel, GNU Hurd, ali nikad nije završen. Postojao je i kernel zasnovan na BSD-u, ali bez dovoljno funkcionalnosti.

Nedostatak besplatnog i korisnog kernel-a, je nerviralo Linusa Torvaldsa, pa je stoga odlučio da napiše svoji sopstveni. Torvalds je bio upoznat već sa Minix-om i sa GNU alatkama i dok je bio student informatike na Univerzitetu u Finskoj je počeo da radi na projektu koji bi kasnije postao Linux kernel. 25.-og avgusta 1991. godine, Torvalds je postavio na "Usenet newsgroup-i" o svom projektu. Nastavio je da bude projekat na kome je samo on radio, ali s vremenom je steklo sve više pažnje od drugih programera. Danas je preko 15000 programera doprinelo preko 17 miliona linija koda.



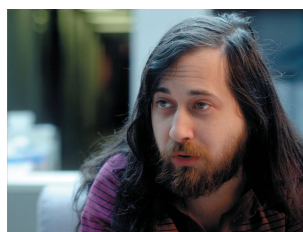
(a) Ken Thompson i Denis Riči, tvorci UNIX-a



(b) Linus Torvalds, tvorac Linux-a



(a) Flopi diskovi sa verzijom 0.12 Linux-a



(b) Ričard Metju Stalman, tvorac GNU-a

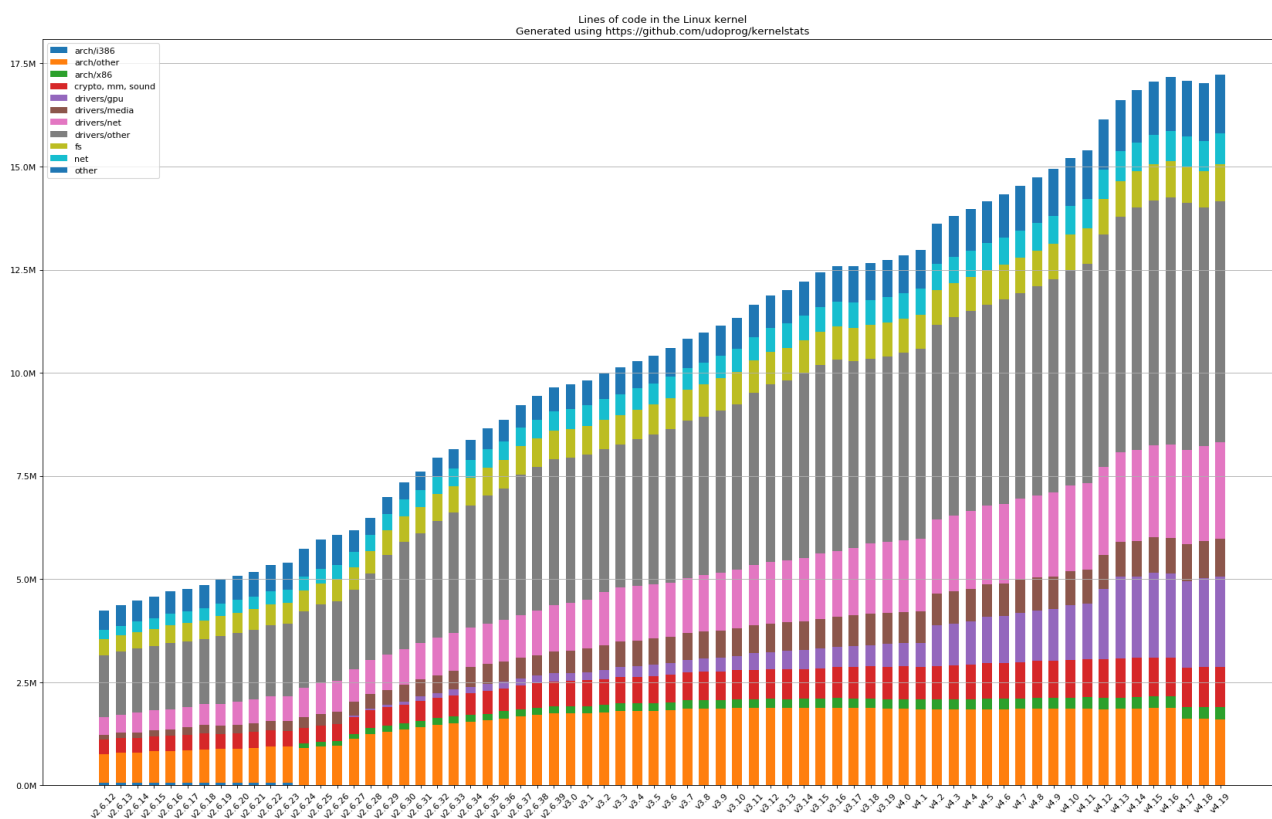


Figure 1: Broj linija koda u Linux kernel-u od verzije 2.6.12 do danas

## 2 Komponente Linux sistema

Linux se instalira putem neke od mnogobrojnih "Linux distribucija" što su potpuni operativni sistemi sa već instaliranim aplikacijama i drajverima da zadovolje razne potrebe. Postoje distribucije namenjene da prosečnom korisniku za svakodnevne poslove (Ubuntu, Debian, Linux Mint), distribucije orijentisane ka bezbednosti (Kali Linux), naučnom istraživanju (CAELinux), edukaciji (Edubuntu), itd.

Jedino što sve distribucije imaju zajedničko je Linux kernel, dok većina rasprostranjenijih distribucija ima i iste osnovne komponente.

### 2.1 Bootloader

Boot loader je program koji se pokreće pre bilo kog operativnog sistema. Njegov posao je da nadje operativni sistem (ili više njih) i da ga pokrene.

Na Linuxu postoji nekoliko bootloader-a:

- GRUB (**GR**and **U**nified **B**ootloader) - najpopularniji, deo GNU programa napravljenih za GNU Hurd kernel.
- LILO (**L**inux **L**oader) - razvoj je prekinut jer nije podržavao sisteme sa više od jednog operativnog sistema.
- SYSLINUX - skup neintenzivnih bootloader-a, najčešće se koriste za podizanje sistema sa drajvova malih kapaciteta, kao fleš drajv, DVD...



Figure 2: Primer GRUB prozora tokom paljenja kompjutera

## 2.2 Kernel

Kernel je najvažniji i najosnovniji deo svakog operativnog sistema. Kernel je zadužen da pokrene svaku komponentu potrebu za korišćenje sistema, da služi kao posrednik u komunikaciji između softvera i hardvera, i delova softvera međusobno. Kernel je potreban tokom celog korišćenja računara, pa je stoga neophodno da on bude što manji i što efikasniji. Osnovi delove kernela su obično:

- rasporednik - određuje kako će razni procesi koristiti snagu procesora
- supervizor - odobrava kontrolu kompjutera procesu koji je na redu
- rukovodilac zahteva - rukuje svim zahtevima upućenim kernelu
- menadžer memorije - dodeljuje lokacije na memoriji procesima kernela

Postoji 4 glavne kategorije kernel-a:

- monolitski - obično se nadju kod Unix-sličnih operativnih sistema, kao kod Linux-a i FreeBSD-a. Oni sadrže sve osnovne funkcije OS-a i drajvere potrebne za korišćenje hardvera kao hard diskova, grafičkih kartica, printera. Moderni monolitski kernel-i imaju opciju da odrede koji moduli kernel-a će se koristiti, time smanjujući količinu koda kernel-a.
- microkernel-i - imaju samo minimalan broj usluga kao menadžer memorije, sistem za komunikaciju između procesa i menadžer procesa. Sve ostale funkcije su implementirane nezavisno od kernel-a. Primeri mikrokernel-a su GNU Herd, MINIX i Mac OS X.
- hibridni - kompromis između monolitskih i mikrokernel-a. Osimišljeni se pre nego što je otkriveno da su mikrokernel-i daleko efikasniji od hibridnih. Eksperimentiše se sa exokernel-ima. Glavna razlika između njih i ostalih vrsta kernel-a je što se jedino bave zaštitom harvera umesto menadžmentom hardvera. Ovim pristupom exokernel-i omogućuje programerima da bolje odrede kako najefikasnije da koriste raspoloživ hardver.

## 2.3 Daemoni

Daemon je program na Linux sistemima koji radi u pozadini, bez direktne kontrole korisnika. Oni obično služe da odgovaraju na zahteve drugih kompjutera na mreži, ali također, da reaguju na softverske i hardverske promene na samom kompjuteru.

Na primer, na daemon-e mogu da utiču određeno vreme ili datum, stvaranje fajla u specifičnom folderu, zahtev napravljen preko interneta, itd. Daemon-i se vode u sistemu kao potprocesi "init" procesa, što je prvi proces koji se pokreće sa kompjuterom.

Na većini novih Linux sistema, daemon-i se pale samo po potrebi i na zahtev jednog glavnog daemon-a - "xinetd".

## 2.4 Shell

Shell služi da obezbedi isključivo tekstualni "interfejs" za korisnika. Njegova primarna svrha je čita komande iz konzole i da ih pokrene.

"Shell" ili "ljuska" se odnosi na to da je to spoljašnji sloj opertivnog sistema tj. shell je posrednik izmedju korisnika i unutrašnjih delova sistema. Osim za sa samo pokretanje programa, shell-ovi imaju sposobnost da usmeravaju output? jedne komande da bude korišćen kao input? druge komande - "piping" (prvo uvedeno još u UNIX-u) i također da služe kao programski jezik - sintaksa komandi može da se koristi za pisanje "shell skripti".

Postoje razni shell-ovi, od kojih je najpopularniji "bash" (Bourne-again shell), koji je nadogradnja na "sh" (Bourne shell) - originalni UNIX shell.

## 2.5 X window sistem

X window sistem ili samo "X" je sistem za menadžment grafičkih interfejsa (GUIs) na Linux-u. "Window sistem" je kolekcija softvera koja omogućava korisniku lakšu kontrolu nad stvaranjem prozora i drugih grafičkih elemenata na kompjuteru.

Važna karakteristika X-a je što je odvojen od opertivnog sistema, za razliku kod Windows-a i starijih verzija Mac sistema gde je window sistem bio sastavni deo operativog sistema. Na Linux-u, i bez window sistema, sistem može da se kontroliše kroz komandni interfejs (CLI).

Još jedna važna karakteristika X-a je što je on samo mehanizam za korisnički interfejs, bez da određuje kakav će on biti. Ovo daje korisniku slobodu da sam odredi kakav interfejs želi, za raliu od drugih opertivnih sistema gde je unapred određeno kako će interfejs izgledati. Korisnik nije u kontaktu sa X-om. Aplikacije su same zadužene za svoje ponašanje i izgled, X služi samo da ih prikaže. Ovo je dovelo do toga da svaki program izgleda drugačije, jer je svaki dolazio sa svojim sopstevnim podešavanjima. Ovaj problem je rešen sa desktop okruženjima.

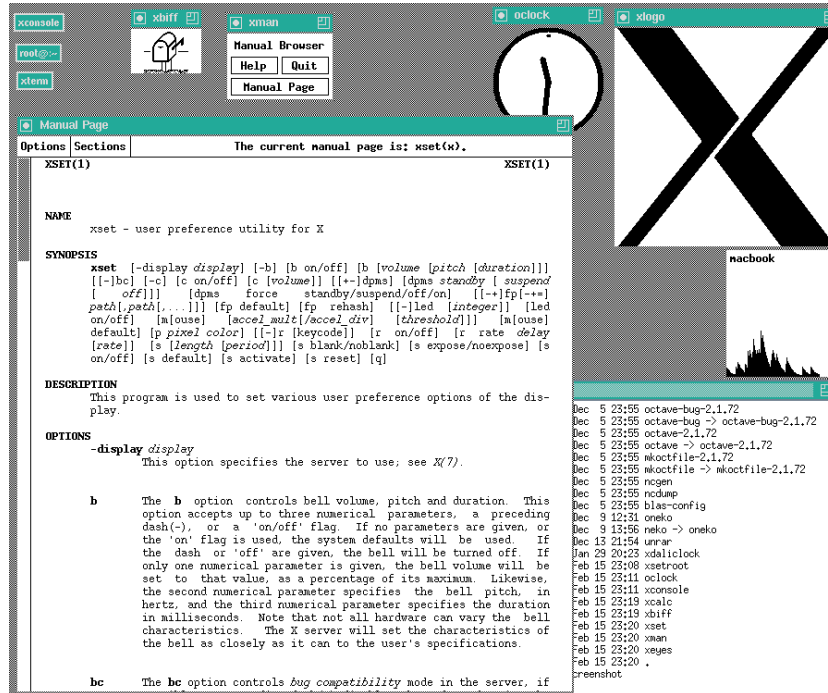
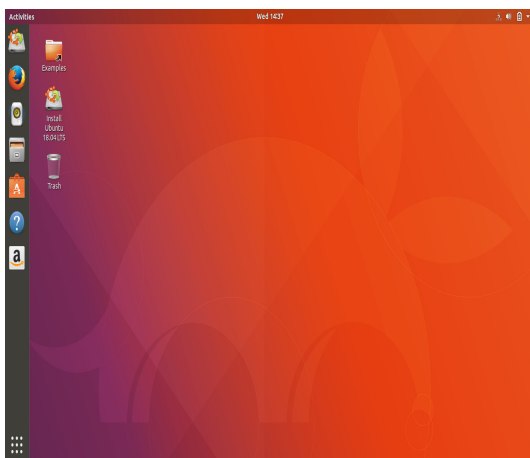


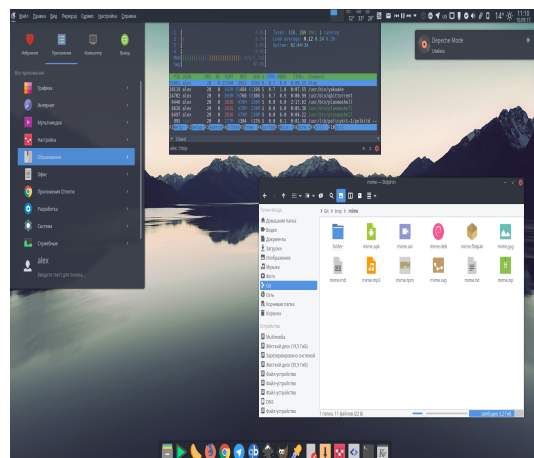
Figure 3: X u svojim ranim fazama, kasne 80-te i rane 90-te

## 2.6 Desktop okruženje

Desktop okruženje je skup softvera i drajvera koji rade na već postojećem sistemu sa gorenavedenim komponentama i služe da korisniku pruže što jednostavnije korišćenje sistema. Najpopularnija desktop okruženja su GNOME, KDE, Xfce, LXDE, itd.

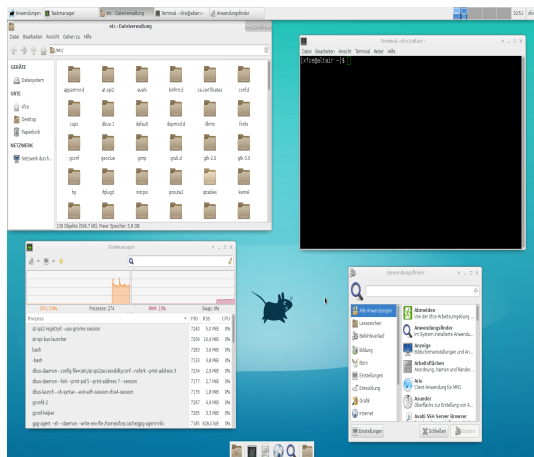


(a) GNOME desktop okruženje

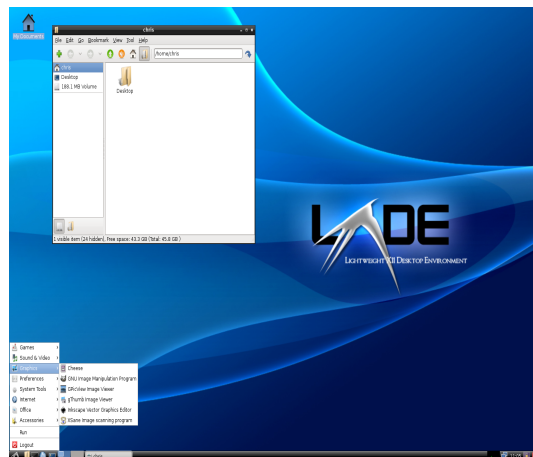


(b) KDE desktop okruženje





(a) XFCE desktop okruženje



(b) LXDE desktop okruženje

### 3 Osnovne komande u Linux-u

Prednost GNU/Linux operativnih sistema nad drugim je ekstensivnost Linux-ovog "terminal-a". Terminal se koristi za svaku radnju na sistemu, kao manevrisanje kroz sistem, manipulacija fajlovima i folderima, pokretanje i gašenje programa, kontrola trenutnih procesa, itd.

Dalje su objašnjene neke od osnovnih Linux komandi.

- **pwd** - Svaki put kad se otvori shell, njegov rad je koncentrisan u jedan folder (ovaj folder je pri otvaranja novog shell-a "home" folder). **pwd** ispisuje folder u kome se trenutno radi. Skraćeno od "print working directory".
- **ls** - Ispisuje sve fajlove i podfoldere u zadatom folderu ili u trenutnom folderu ako ništa nije zadato. Na komande u Linux-u mogu da se dodaju opcije koje mogu dodatno da definišu tačno kakav će rezultat komande biti. Na primer, na **ls** komandu može da se doda **-a** opcija da bi se prikazali i skriveni fajlovi i folderi čija imena počinju sa ".". Skraćeno od "list".
- **cd** - Menja folder u kome se trenutno radi na zadati folder, npr. **cd Documents**. Za vraćanje u prethodni folder koristi se **cd ..** (".." uvek označava folder relativno iznad trenutnog, dok "." označava trenutni folder). Skraćeno od "change directory".
- **mkdir** i **rmdir** - Pravi novi folder odnosno briše (pod uslovom da je prazan) zadati folder. Skraćeno od "make directory" i "remove directory".
- **rm** - Briše zadati fajl. Može da se koristi sa opcijom **-r** (rekurzivno) da bi izbrisalo sadržaj zadanog foldera. Skraćeno od "remove".
- **cp** - Kopira zadati fajl ili folder na datu lokaciju. Ova komanda uzima dva argumenta,

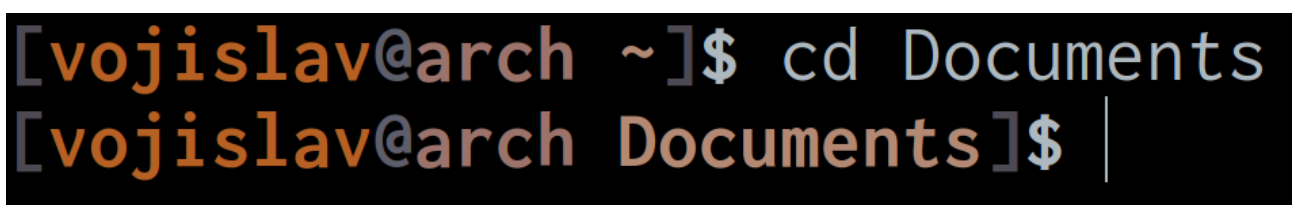
odvojena razmakom. Na primer, `cp subfolder folder` ("subfolder" će se kopirati na lokaciju `/folder/subfolder`). Skraćeno od "copy".

- `mv` - Premešta fajl ili folder na datu lokaciju. Takodje uzima dva argumenta. Skrećeno od "move".
- `cat` - Spaja i ispisuje sadržaj fajlova (ako je dat samo jedan fajl, ispisuje njegov sadržaj). Skraćeno od "concatenate".
- `sudo` - Stavlja se kao prefiks bilo kojoj komandi da bi se pokrenula sa administratorskim ovlašćenjima. Za pokretanje je potrebna lozinka "root" korisnika (administratora). Skraćeno od "SuperUser do".
- `man` - Ispisuje upustva za korišćenje i sve opcije za bilo koju komandu. Skraćeno od "manual".

A terminal window with a black background and orange and white text. The prompt is [vojislav@arch ~]\$ and the command pwd is entered. The output is /home/vojislav. The prompt is then [vojislav@arch ~]\$ followed by a vertical bar cursor.

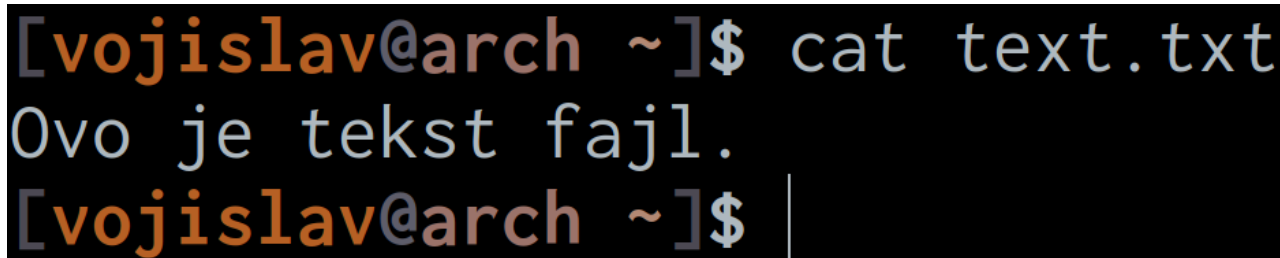
```
[vojislav@arch ~]$ pwd
/home/vojislav
[vojislav@arch ~]$ |
```

Figure 4: Primer `pwd` komande

A terminal window with a black background and orange and white text. The prompt is [vojislav@arch ~]\$ and the command cd Documents is entered. The output is [vojislav@arch Documents]\$ followed by a vertical bar cursor.

```
[vojislav@arch ~]$ cd Documents
[vojislav@arch Documents]$ |
```

Figure 5: Primer `cd` komande



```
[vojislav@arch ~]$ cat text.txt
Ovo je tekst fajl.
[vojislav@arch ~]$ |
```

Figure 6: Primer `cat` komande

## 4 Hierarhija sistema datoteka

Hierarhija sistema datoteka ili Filesystem Hierarchy Standard (FHS) definiše strukturu foldera u Linux sistemima. U FSH-u, svi fajlovi i folderi se nalaze u `root` foleru (označen kao `/`), raspoređeni po specifičnim subfolderima na osnovu njihove funkcije.

- `/bin` - sadrži neophodne programe na sistemu, kao `cat`, `ls`, `cp`.
- `/boot` - fajlovi za bootloader
- `/dev` - fajlovi neophodni za koriscenje uredjaja.
- `/etc` - fajlovi za konfiguraciju programa i procesa.
- `/home` - svi fajlovi i folderi korsnika
- `/lib` - "biblioteke" neophodne za pokretanje programa iz `/bin`.
- `/media` - namenjen za pristup sadržaju priključenih uredjaja, kao fleš drajvova, CD-ROM-ova, itd.
- `/mnt` - namenjeno za pristup privremenim uredjaja
- `/proc` - sadrži inforamacije o sistemskim procesima i kernel-u u obliku fajlova.
- `/usr` - sadrži sve instalirane programe koji se ne nalaze u `/bin`.
- `/var` - "promenljivi" fajlovi - fajlovi koji beleže razne promene i podatke tokom rada.
- `/tmp` - privremeni fajlovi i folderi koji se brišu pri gašenju sistema.

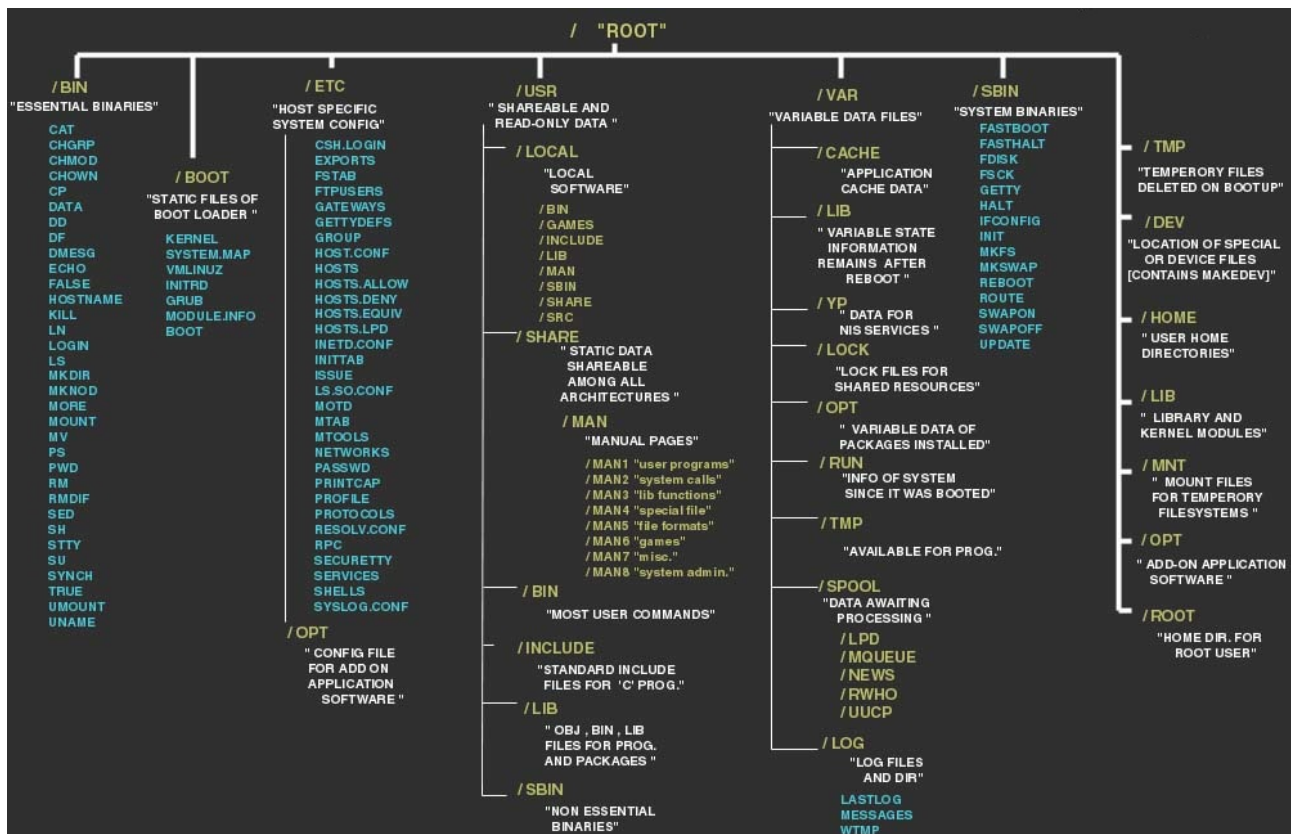


Figure 7: Vizuelno predstavljena hierarhija sistema datoteka

## 5 Redirekcija i "piping"

Gore je već spomenuta sposobnost Linux shell-a da menja tipičan tok i destinacija output-a komandi.

Redirekcija je uzimanje tipičnog output-a neke komande i slati ga negde drugačije od njegove uobičajene destinacije. Na Linux sistemima, svaka komanda i samim tim, svaki proces se sadrži od tri *toka podataka*: jednog ulaznog toka, standard input (ono što korisnik daje) i dva izlazna toka - standard output (predviđeni rezultat komande) i standard error (moguće greške tokom izvršavanja komande).

Redirekcija se sastoji od dva elementa: argument za komandu i redirekcionih operatora. Argument je fajl ili neki drugi izvor podataka koji se daje komandi da bude korišćena kao input. Na primer, u komandi `head`, koja ispisuje prvih deset linije datog teksta, argument je ili fajl sa tekstom ili neki drugi izvor teksta. `head` takodje može da se koristi bez argumenta, gde će čitati prvih deset linija koje je korisnik uneo, ali važno je napomenuti da ne može svaka komanda da se koristi bez argumenata.

Redirekциони operatori se koriste za menjanje destinacija toka podataka. Čine ih većinom tri glavna operatora: `<`, `>` i `|`.

">" uzima standard output i standard error i preusmerava njihovo ispisivanje u neki fajl, za razliku od uobičajenog ispisivanja na ekranu. Na primer, komanda `cat file1 > file2`, uzima output komande `cat file1`, što bi bilo samo ispisivanje njenog sadržaja i ispisuje ga u `file2`. Ako se već neki tekst nalazio u `file2`, on će biti potpuno zamenjen sadržajem `file1`. Varijacija na > operator je >>, koji umesto da potpuno zameni sadržaj jednog fajla drugim, doda sadržaj jednog fajla na kraj drugog. Na primer, komanda `cat file1 >> file2` dodaje sadržaj od `file1` na kraj `file2`.

"<" uzima kao input komande neki fajl, umesto da čita standard input. U suštini je isto kao korišćenje fajla kao argument komande, ali je ipak vredno znati.

"|" se koristi u "piping"-u i drugačiji je od ostalih redirekcionih operatora. Razlikuju se po tome što koriste output neke komande kao input druge. Na primer, output `ls` komande je lista svih fajlova i foldera u trenutnom folderu, postaje input `wc` komande koja, ako je pokrenuta bez argumenata, broji količinu redova, linija i karaktera onoga što joj je dato: `ls | wc`. Mogućnost spajanja komandi u jednu koristeći "piping", može znatno da skрати neki shell skript sa dosta komandi.

## 6 Dozvole i vlasništvo

Važna odlike Linux i njemu sličnim sistemima je što svaki objekat(fajlovi, folderi, itd.) ima svoje dozvole za pristup. Ovaj sistem je veoma važan za održavanje visokog nivoa bezbednosti i stabilnosti Linux sistema.

Svaki objekat ima tri tipa dozvola: za čitanje(read), pisanje(write) i pokretanje(execute). Ova tri tipa se odnose na dozvole: *vlasnika objekta*(korsnik koji ga je napravio), *grupe*(skup korsnika sa istim pravima na pristup) i na sve ostale korisnike.

Na svakom Linux sistemu postoji automatski napravljen administratorski korisnik ili *root*, čija se prava razlikuju od ostalih korisnika, u tome što on ima puna prava na svaki objekat u sistemu. Samim tim, ima sposobnost da menja dozvole drugih korisnika nad tim objektom.

Dozvole nekog objekta mogu da se predstavljaju na nekoliko načina, jedan od kojih je tekstualno, preko stringa od 10 karaktera. Prvi karakter predstavlja tip objekta(- za običan fajl, d za folder). Ostalih 9 predstavlja kakve dozvole imaju vlasnik, grupa i ostali korisnici nad tim objektom(3 tipa dozvole za 3 tipa korisnika).

Na primer, komandom `ls -l` vidimo sve fajlove i foldere u trenutnom folderu, ali sa `-l` opcijom vidimo i karakteristike tog objekta, kao i ko ima kakve dozvole nad njim.

Jedan tekstualni prikaz dozvola izgleda ovako: `-rwxrw-r--`. Po prvom karakteru možemo da zaključimo da je objekat fajl. Od sledeća 3 karaktera vidimo dozvole vlasnika fajla tj. da ima čitanje(*r*), pisanje(*w*) i pokretanje(*x*) prava. Od druga 3 vidimo prava grupe i od poslednja 3

prava ostalih korisnika.

Vlasništvo takodje može da se prikaže brojučano tj. preko tri broja, od 0 do 8, gde svaki broj označava tip vlasništva svakog tipa korisnika. Brojevi imaju sledeća značenja:

- **0** - nikakva vrsta dozvola
- **1** - samo pravo na pokretanje
- **2** - samo pravo na pisanje
- **3** - pravo na pokretanje i pisanje
- **4** - samo pravo na čitanje
- **5** - pravo na čitanje i pokretanje
- **6** - pravo na čitanje i pisanje
- **7** - sva prava

Prava nekog objekta mogu da se menjaju sa `chmod` komandom koristeći broječanu metodu(prava nekog objekta jedino mogu da menjaju vlasnik fajla i *root* korisnik).

## Literatura

- [1] The linux information project. <https://www.linfo.org/>.
- [2] The linux documentation project. <http://www.tldp.org/>.