**UNIVERZITET SINGIDUNUM**

**ADMINiSTRACIJA I DEPLOYMENT LINUX SERVERA**

**-diplomski rad –**

**Mentor: Kandidat:**

Prof.dr Aleksandar Jevremović Aleksandar Milutinović

**Novi Sad, 2022.**

# Sadržaj

[Sadržaj 1](#_Toc116755512)

[Uvod 4](#_Toc116755513)

[1. Unix kao preteča Linux 5](#_Toc116755514)

[2. Linux distribucije 8](#_Toc116755515)

[3. Struktura Linux sistema i permisije 10](#_Toc116755516)

[3.1 Direktorijumi 10](#_Toc116755517)

[3.2 Permisije u Linux sistemima 13](#_Toc116755518)

[3.2.1 Grupe permisija 13](#_Toc116755519)

[3.2.2 Promena permisija i komande 13](#_Toc116755520)

[3.2.3 Funkcija setuid 14](#_Toc116755521)

[3.2.4 Funkcija setgid 14](#_Toc116755522)

[3.2.5 Funckcija promene grupe i vlasnika 15](#_Toc116755523)

[3.2.6 Lepljivi bit 15](#_Toc116755524)

[3.3 Objašnjenje Sudo komande 16](#_Toc116755525)

[4. instalacija CentOS operativnog sistema 18](#_Toc116755526)

[4.1 Opšta konfiguracija 19](#_Toc116755527)

[5. Podešavanje udaljene konekcije SSH klijentom 25](#_Toc116755528)

[5.1 Opšta konfiguracija 25](#_Toc116755529)

[6.Automatizacija pravljenja virtualnog okruzenja 30](#_Toc116755530)

[6.1 Uvod 30](#_Toc116755531)

[6.2 Konfiguracija Virtualnog okruženja 31](#_Toc116755532)

[7. Daljinsko administriranje pomoću SSH 32](#_Toc116755533)

[7.1 Generisanje i upotreba ssh-keygen 33](#_Toc116755534)

[8. Networking/DNS 36](#_Toc116755535)

[8.1 A record 38](#_Toc116755536)

[8.2 AAAA rekord 39](#_Toc116755537)

[8.3 CNAME rekord 40](#_Toc116755538)

[8.4 MX rekord 41](#_Toc116755539)

[8.5 PTR rekord 41](#_Toc116755540)

[8.6 NS rekord 42](#_Toc116755541)

[8.7 SOA rekord 43](#_Toc116755542)

[8.8 SRV rekordi 44](#_Toc116755543)

[8.9 TXT rekordi 44](#_Toc116755544)

[9. Veb serveri, baze podataka i server za email. 45](#_Toc116755545)

[9.1 instalacija httpd(apache) na CentOS 46](#_Toc116755546)

[10.1.1 Osnove Apache konfiguracije 47](#_Toc116755547)

[10.1.2 Kako ovo funkcioniše 48](#_Toc116755548)

[9.2 MySQL, Maria DB baze podataka 49](#_Toc116755549)

[9.2.1 Podešavanje secure instalacije 49](#_Toc116755550)

[9.2.2 Listanje kreiranje i selektovanje podataka i tabela 50](#_Toc116755551)

[9.3 Upotreba MTA (Mail transfer agent) 52](#_Toc116755552)

[10. Pravljenje osnovne konfiguracije neke domene upotrebom wordpress 54](#_Toc116755553)

[10.1 Uvod 54](#_Toc116755554)

[10.2 instalacija Wordpress aplikacije 54](#_Toc116755555)

[11. Shell skripte 59](#_Toc116755556)

[11.1 Uvod 59](#_Toc116755557)

[11.2 Početak pisanja skripti 59](#_Toc116755558)

[11.3 Varijable 60](#_Toc116755559)

[11.4 Uzimanje unosa iz terminala 61](#_Toc116755560)

[11.5 Regularni izrazi 63](#_Toc116755561)

[11.6 Sed 64](#_Toc116755562)

[11.7 AWK 66](#_Toc116755563)

[12. Firewall i nadgledanje sistema 67](#_Toc116755564)

[12.1 Uvod 67](#_Toc116755565)

[12.2 iptables 67](#_Toc116755566)

[12.2.1 iptables dodavanje i uklanjanje pravila 68](#_Toc116755567)

[12.3 Nadgledanje sistema 70](#_Toc116755568)

[Zaključak 72](#_Toc116755569)

[Literatura 73](#_Toc116755570)

# Uvod

Glavni problem koji razmatram u ovom radu, jeste upotreba Linux servera kao i njegova administracija koristeći različite metode za rešavanje problema, koji se javljaju u ovom polju.

Glavni cilj mog istraživanja jeste da predstavim, a i da olakšam upotrebu i administraciju Linux sistema budućim Linux administratorima koji bi čitali ovaj rad.

Takođe još jedan cilj jeste da se približi razumevanje ovih operativnih sistema iz prostog razloga što je izuzetno moćan i omogućava korisniku potpunu slobodu, što nemamo u Windows operativnim sistemima.

Tehnologije koje ću koristiti su Virtual Box okruženje za virtuaizaciju, Vagrant koji omogućava jednostavnu automatizaciju kreacije servera, shell skripte pisane za specifične zadatke, operativni sistem CentOS 7

U svom radu, koristiću funkcionalni pristup gde ću kroz niz primera i upotreba različitih alata na Linux operativnom sistemu, ću prikazati različite probleme sa kojima se neki Linux sistemski administrator može susresti na serveru.

# 1. Unix kao preteča Linux

Kao preteča za Linux operativne sisteme bio je UNIX, koji je još uvek aktuelan operativni sistem. Možemo ga naći u ekosistemu Apple kompanije, kao neki od značajnih konkurenata Windows kompanije, gde je macOS (Apple) operativni sistem koji koristi UNIX.

UNIX je nastao 1969 godine i njegov nastanak se pridaje Denis Ričiju(Dennis Ritchie) i Ken Tomsonu (Ken Thompson). Operativni sistem je nastao u Belovim labaratorijama(Bell Labs) i kao osnova korišćen je Multicsa, prethodno neuspeli projekat višekorisničkog operativnog sistema.

U početku ovaj operativni sistem nije imao podršku većih kompanija, a i sam operativni sistem nije bio na zavidnom nivou kao što je danas. UNIX je inicijalno bio sistem koji je mogao da radi samo na jednom zadatku, suprotno Multicsa koji je mogao da vrši više zadataka, za šta je bio i inicijalno razvijan. Porastom kompleksnosti i upotrebe ovog operativnog sistema, pošto je zamenio tada korišćeni operativni sistem DEC, moralo je da se nađe rešenje kako bi se omogućila skalabilnost ovog operativnog sistema.

Godine 1973 operativni sistem je napisan u jeziku C, što je omogućilo prenosivost koda na buduće verzije. Prva verzija koja je bila u upotrebi van Bellovih labaratorija je bila pod nazivom V6 što je bilo šesto izdanje već postojećeg operativnog sistema.

Nakon inicijalnih primena, druge značajnije kompanije kao što su AT&T su stvorile svoje varijacije na postojeći dizajn. AT&T verzija koja je razvijena bila je System V Release 4 (SVR4) godine 1982.

Sam operativni sistem se razvijao zavidnom brzinom iz prostih razloga dostupnosti izvornog koda i jednostavnosti dizajna. Poznatije varijante su nastale na Univerzitetu u Berkliju pod nazivom Berkley Software Distributions (BSD).

U ovim verzijama UNIX operativnih sistema se javlja virtulana memorija , straničenje na zahtev i TCP-IP protokoli. Neke od varijanata ovog operativnog sistema su Darwin, Dragonfly BSD, FreeBSD, NetBSD i OpenBSD sistemi.

Neke kompanije su razvijale i svoje verzije UNIX operativnog sistema sa višestrukim radnim stranicama i serverima kao što su Digitalov Tru64, IBM AIX, Sunov Solaris….

Dobre osobine UNIX sistema pokazale su se u sistemskim pozivima(nekoliko stotina), pregledan dizajn, sistem datoteka.

Sve ove osobine omogućile su lakše upravljanje podacima i uređajima svodeći ih na na skup prostih sistemskih poziva. Open(), read(),write(),ioctl(), close(), uz sve to da je pisan u jeziku C koji se smatra jednim od robusnijih programskih jezika, koji se koristi u operativnim sistemima, a i šire (varijacije već postojećeg recimo C++). Omogućuje prenosivost kroz upotrebu C programskog jezika što je jedna od značajnih prednosti kada se piše OS.

UNIX veoma brzo kreira procese i ima jedinstven sistemski poziv fork(). Konačno UNIX obezbeđuje prostu i robusnu komunikaciju između procesa. UNIX danas je napredan operativni sistem koji omgućava široki spektar funkcija. Primena UNIX se svodi od uređaja koji koriste na stotine procesora do malih ugrađenih (embedded) uređaja.

Linux je nastao na Univerzitetu u Helsinkiju 1991 godine, i njegovo začeće se pridaje Linus Torvalds, koji je razvio ovaj sistem kao sredstvo za učenje koji koriste intelove mikroprocesore i386 ili 80386.

Razlog iz kojeg je ovaj operativni sistem nastao, je problemi koje je Linus imao sa licencama tada već dobro utemeljenog operativnog sistema UNIX.

Prvobitno je razvijen kao emulator koji se povezivao sa UNIX sistemima na fakultetu, a vremenom se razvijao i unapređivao.

Prva zvanična verzija ovog operativnog sistema je puštena na internet 1991, od ove tačke Linux postaje kolaborativni projekat.

Linux radi na sledećim procesorima:

MD x86-64, ARM, Compaq Alpha, CRiS, DEC VAX, H8/300, Hitachi Su-

perH, HP PA-RiSC, iBM S/390, intel iA-64, MiPS, Motorola 68000, PowerPC, SPARC, UltraSPARC i v850

Kako se razvijao Linux tako su na scenu stupile mnoštvo kompanija koje su imale svoju verziju Linuxa, a i generalno su se specijalizovale za ovaj operativni sistem.

Neke kompanije su Red Hat , iBM ,Novell….. oni nude svoja rešenja za desktop sisteme, servere embedded sisteme.

Linux je uzeo veliki deo koda od UNIX operativnih sistema , iz tog razloga bi se moglo reći da je klon. Tokom razvijanja operativnog sistema korišćene su ideje na kojima se zasniva UNIX kao portabilnost , jednostavnost ... no kada se sve uzme u obzir Linux ne izgleda kao potpuna kopija zato što je krenuo drugom putanjom od UNIX operativnog sistema na kojem je zasnovan.

Jedna od vodećih ideja Linuxa jeste da je sve open source tj. izvorni kod je dostupan svima da rade sa njim šta hoće, što je obuhvaćeno licencom otvorenog koda GPL (General Public License version 2.0), jedina obaveza koju bi imao neko ko je preuzeo ovaj kod jeste da daju ista prava koja sami koriste, uključujući i raspolaganje izvornim kodom.

Osnovini delovi operativnog sistema Linux su jezgro (kernel), C biblioteka(library) , prevodilac(code interpreter), kao i niz alata, osnovne sistemske funkcije(kao što su proces prijavljivanja i interpreter komandi).

Linux raspolaže i mnoštvom desktop okruženja, upravljač prozorima (window manger, slično desktop okruženju samo što se fokusira na minimalizmu i slobodi modifikovanja), takođe postoji i mnoštvo aplikacija za Linux komercijalnih i slobodnih.

# 2. Linux distribucije

Jedna od najvećih prepreka za nekog ko počinje bilo šta sa Linux operativnim sistemima je koju distribuciju izabrati.

Pregledom na sledeću stranicu

[*https://upload.wikimedia.org/wikipedia/commons/1/1b/Linux\_Distribution\_Timeline.svg*](https://upload.wikimedia.org/wikipedia/commons/1/1b/Linux_Distribution_Timeline.svg) može se videti koliko je to velika prepreka prilikom odabira, postoje stotine varijacija na već postojeći Linux operativni sistem.

Kada bi se razvrstalo na nešto jednostavniju podelu, možda kao početna grana bi se uzela Debian pošto je to i možda napopularnija grana Linuxa, ili RedHat grana koja je takođe popularna.

Jedna od popularnijih firimi u Debian grani je Ubuntu na koju se nakdane popularnije verzije nadovezuju, moglo bi se reći da je sve nakon Ubuntu samo drugačiji izgled, ali nije uvek tako sa svim granama koje su nastale odavde, Debian sistemi koriste APT upravljač paketima za instalaciju programa i softvera.

Imamo i SUSE kao jedna od popularnih grana, SUSE sam po sebi je grana Slackware Linux, koji je uzgred jedna od najstarijih inicijalnih grana Linux operativnog sistema .

SUSE je nastao samo godinu dana nakon SlackWare, i veću popularnost je stekao negde oko 2010 izlaskom OpenSuse operativnog sistema, upravljač paketima koji ova distribucija koristi je YAST. SUSE nudi i plaćenu verziju svog operativnog sistema odatle imamo i granu openSUSE koji je dostupan svima.

Sledeće u grupi većih ili značajnijih operativnih sistema je grupa RedHat Linux. Popularne distribucije koje spadaju u ovu grupu su Fedora, CentOS, kao i sam RedHat operativni sistem koji je razvijan za komercijalna rešenja.

Upravljač paketima koji se koristi je RPM(red hat package manager) yum (koji se nalazi na CentOS i Fedora).

Uglavnom sve verzije od RedHat su visoko kvalitetne verzije, jedne od većih pokretača web servera na internetu su RedHat i CentOS, gde je CentOS besplatna verzija pristupačnija pa je klijenti često koriste za svoj server u nekim kompanijma koje se bave ovim delatnostima.

Postoji i DNF package manager kod novijih verzija kao što su AlmaLinux koji bi trebao da zameni CentOS u nekoj bližoj budućnosti, ali nije ništa sigurno povodom ove konstatacije.

Takođe manje pomenuta, ali istaknuta distribucija Linuxa je Arch Linux, prednost ove distribucije u odnosu na sve ostale je da je specifično pravljen da korisnik ima potpunu kontrolu od samog početka počevši od instalacije.

Arch Linux operativni sistemi imaju pristup velikoj količini repozitorijuma koje su postavljene od samog osnivača pa do običnog korisnika.

Open Source AUR(Arch user repositories) koje su u suštini pokretane od grupe korisnika koji postavljaju ove aplikacije na repozitorijum, a zvanični upravljač paketima je pacman.

Tipovi operativnih sistema koji bi trebalo da se navedu su sledeći:

Rolling release: Ovo su bleeding edge ili najaktuelniji operativni sistemi sa svim aplikacijama na najnovijoj verziji, za serverske potrebe uglavnom se izbegavaju rolling release verzije operativnih sistema iz prostog razloga što ove verzije mogu da budu nestabilne, a za pravilan rad servera podrazumeva se operativni sistem koji nema takve probleme.

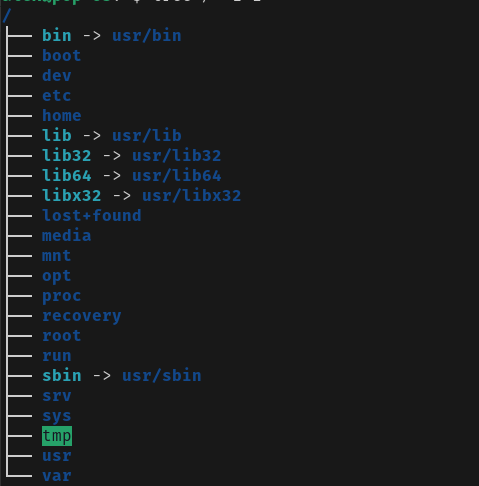
Stable Release: Ovo su verzije na kojima su aplikacije testirane i puštene u rad u ekosistemu operativnog sistema. Stabilne verzije operativnih sistema su zastupljene kod korisnika, a i na serverima iz prostog razloga što oni imaju manju količinu problema koje dolaze sa ne stabilnim softverom.

Navažnije je izdvojiti da upotreba distribucije zavisi od korisnika i šta ta distribucija nudi što odgovara korisniku u datom trenutku.

# 3. Struktura Linux sistema i permisije

Linux kao i svi trenutno prisutni operativni sistemi ima svoju strukturu fajlova i kako se vrše interakcije nad njima. Kako bi se bolje razumelo bolje je gledati njegovu strukturu kroz terminal da bi dobili neki stepen vizualizacije kako se sve to organizuje.

*Slika 1: Struktura fajlsistema*



Linux distribucije uglavnom pokazuju sličnu stukturu direktorijuma, ali ponekad postoje varijacije koje su uglavnom sitne i specifične za operativni sistem.

## 3.1 Direktorijumi

/bin

Ovo je direktorijum koji sadrži binary fajlove, tj neke aplikacije i programe koji mogu da se pokrenu na OS. Ako prelistamo ovaj direktorijum možemo naći komande koje često koristimo u terminalu npr. ls komandu za listanje fajlova, kao i druge osnovne alatke za pravljenje i brisanje fajlova i direktorijuma.

/boot

Boot direktojum sadrži sve informacije za pokretanje operativnosg sistema i skladištenje.

Ovaj direktorijum može biti prilično komplikovan za popravku ukoliko dođe do njegove korupcije, tako da uglavnom bilo kakva konfiguracija se izbegava, sem ako korisnik zna apsolutno šta želi da uradi ovde.

/dev

dev direktorijum sadrzi fajlove uređaja. Mnogi od ovih fajlova se generišu prilikom pokretanja operativnog sistema ili tokom rada računara, recimo ako ubacimo webkameru ili USB, novi uređaj će se pojaviti u ovom direktorijumu.

/etc

Sadrži konfiguracione fajlove i početne skripte. Ovaj direktorijum dobija ime od najranijih Unix konvencija koji je stojao za “et cetera”, iz razloga zato što je ovo bilo mesto za stavljanje fajlova za koje sistemski administrator nije bio siguran gde da stavi.

/home

Ovaj direktorijum sadrži početni direktorijum svih korisnika, osim administratora, tj sadrži podatke svakog specifičnog korisnika.

/lib

Fajlovi biblioteke su uskladišteni u ovaj direktorijum. Može da sadrži isečke koda aplikcije koji iscrtavaju prozor na desktopu, ili šajlu fajlove u hard disk.

/media

Eksterni mediji, kao što su USB, priključeni su na ovaj direktorijum, kao i drugi uređaji USB hard diskovi , SD kartice eksterni SSD … dok je računar uključen.

/mnt

Ovaj direktorijum je mesto gde bi ručno postavili uređaje za skladištenje ili particije. Ne koristi se često u sadašnja vremena.

/opt

Uglavnom ima više objašenja za ovaj direktorijum, ali po pravilu bi trebalo da bude da ovaj direktorijum sadrži opcione fajlove koji nisu neophodni za pokretanje operativnog sistema.

/proc

Ovaj direktorijum se često naziva virtuelni direktorijum, i sadrži mnoštvo informacija počevši od specifikacije sistema i procesa koji se pokreću u datom trenutku. Neki pod direktorijumi ovog direktorijuma su /proc/meminfo, /proc/mounts i /proc/partitions. Bitno je izdvojiti, ako se bilo koji od ovih pod direktorijuma izlistali sa cat, more ili less komandom, dobili bi neke informacije na terminalu koje bi nam bile korisne za informisanje o uređajima na operativnom sistemu.

/root

Ovaj direktorijum je sličan korisničkom /home direktorijumu, jedina razlika jeste da je ovaj direktorijum rezervisan za root korisnika i drugi korisnici ne mogu da mu pristupe bez upotrebe sudo komande koja bi im dala povećane privilegije pristupa.

/run

Sistemski procesi koriste ovaj direktorijum da čuvaju privremene podatke

/sbin

isto što i /bin stim što je razlika da sadrži aplikacije koje samo superuser koristi ili su mu potrebni u radu.

/usr

Sadrži sekundarne programe,biblioteke i dokumentacije o programima povezanim sa korisnikom.

/var

Direktorijum log fajlova, sadrži informacione fajlove koji pomažu prilikom otkrivanja nekih većih kvarova barem na serverskim sistemima. Direktorijum var stoji za promenljivi tip podataka (varijable) iz tog i skraćenica, takođe ovaj direktorijum sadrži i druge fajlove koji su promenljivog tipa.

/sys

Virtualni direktorijum kao i /proc koji sadrži informacije o uređajima koji su povezani na operativni sistem.

## 3.2 Permisije u Linux sistemima

U Linux sistemima postoje tri vrste dozvola:

* Dozvola za čitanje (r), ili 4 u brojčanom zapisu, korisnik može da čita ili da proveri fajl
* Dozvola za pisanje (w), ili 2 u brojčanom zapisu, korisnik može da modifikuje fajl
* Dozvola za izvršavanje (x), ili 1 u brojčanom zapisu, korisnik može da izvršava fajl

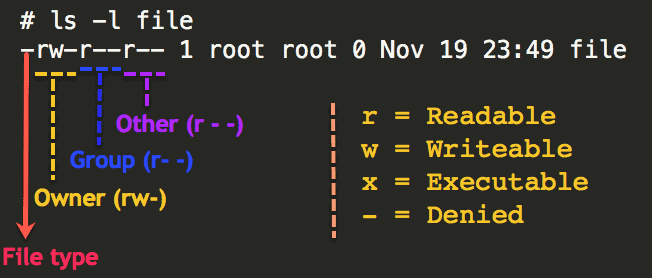
### 3.2.1 Grupe permisija

Owner – permisije vlasnika se odnose samo na vlasnika fajla ili sadržaja, neće uticati na akcije korisnika.

Group –grupne permisije se odnose na grupu koja je dodeljena fajlu ili direktorijumu, neće uticati na akcije drugih korisnika.

All-users – permisije svih korisnika se odnose na sve korisnike koju pokušavaju da pristupe fajlu, ovo se najviše gleda tokom rada.

*Slika 2 : Permisije fajla*



*izvor: thegeekdiary.com/understanding-basic-file-permissions-and-ownership-in-linux*

### 3.2.2 Promena permisija i komande

Tehnike promene permisije zavise od preferencije korisnika, ali generalno barem kod sistemskih administratora oni koriste numerički metod gde broj 4,2 ili 1 predstavlja neku vrstu permisije koja je ranije navedena.

**$ chmod 777 fajl.txt**

Ovo će dati sve permisije jednom fajlu na sistemu , ili u simboličkom smislu odgovara RWX RWX RWX , read write and execute za vlasnika, grupu i sve ostale korisnike.

### 3.2.3 Funkcija setuid

Funkcija setuid (set user identity) dozvoljava korisniku da pokreće fajlove samo sa dozvolama vlasnika (owner) grupe. Uglavnom se koristi kada je neophodno da se korisnicima obezbede veće privilegije, bez upotrebe sudo komande tj. da se korisnici dodaju u wheel grupu.

Podešvanje ovog načina izvršavanja fajla se vrši sa već ugrađenom funkcijom:

**$ chmod u+s fajl.txt**

ili

**$ chmod 4777 fajl.txt**

*Slika 3 : Permisije fajla nakon setuid*



Dozvole nakon bilo koje od dve prethodne komande, biće kao sa slike (slika 3), indikacija da je podešen ovaj način izvršavanja se primećuje po ‘s’ karakteru u polju za vlasnika (owner), a i sam sistem ga osvetli drugačijom bojom da se skrene pažnja na fajl.

### 3.2.4 Funkcija setgid

setgid (set group identity) funckija je slična prethodnoj funkciji setuid, stim da umesto da menja vlasničke (owner) dozvole, ova funkcija menja grupne (group) dozvole.

Podešavanje ove funckionalnosti se vrši na sličan način stim da se menja argument (flag) na 2 umesto 4 ili simbolički se dodeljuje grupi (g) set argument (s).

**$ chmod g+s fajl.txt**

ili

**$ chmod 2777 fajl.txt**

*Slika 4 : Permisije fajla nakon setgid*



Dozvole nakon bilo koje od dve prethodne komande biće kao sa slike (slika 4), može da se primetiti da je podešeno po ‘s’ karakteru u dozvolama grupe (group), takođe kao i sa vlasničkim (owner) dozvolama, fajl je osvetljen drugačijom bojom, tako da skrene pažnju da je funkcija aktivna.

### 3.2.5 Funckcija promene grupe i vlasnika

Komanda ili funkcija promene vlasnika, grupe omogućava manipulaciju fajla u zavisnosti od potreba sistemskog admnistratora ili sudo korisnika operativnog sistema. Najčešća upotreba ove komande na više korisničkom serveru, jeste da dodelimo fajl korisniku nakon čega korisnik može da izvršava željene promene.

Sintaksa komande:

**$chown user:group fajl**

*Slika 5 : Promena vlasnika i grupe*



*Slika 6: Rezultat komande chown*

### 3.2.6 Lepljivi bit

Dodatna komanda za manipulaciju specijalnih privilegija pored setuid i setgid bi bila upotreba lepljivog bita (sticky bit). Sistemski administrator bi koristio ovu komandu kako bi obezbedio specijalnu privilegiju koja omogućava samo vlasniku fajla da briše i prepravja fajl, dok drugi korisnici mogu samo da čitaju sadržaj. Vrlo često možemo videti uoptrebu lepljivog bita (sticky bit) u /tmp direktorijumu gde na više korisničkom serveru svaki korisnik kreira svoj proces koji nešto radi, a procesi generišu fajlove (pod imenom korisnika koji je pokrenu aplikaciju) u tmp direktorijumu koji su im nephodni za pravilan rad. Lepljivi bit će onemogućiti drugim korisnicima da manipulišu sa privremenim fajlovima koji su kreirani od strane drugog korisnika.

**$ chmod +t fajl.txt**

ili

**$chmod 1777 fajl.txt**

*Slika 7: Dozvole /tmp direktorijuma sa lepljivim bitom*

*Slika 8: Dozvole fajl.txt sa lepljivim bitom*

Sa slika (slika 7 i slika 8) može da se primeti upotreba lepljivog bita po karakteru ’t’ koji nam prikazuje da je fajl podešen sa specijalnom dozvolom.

## 3.3 Objašnjenje Sudo komande

Česta komanda koja se vidi prilikom rada na bilo kom Linux serveru, ako korisnik nije inicijalno pristupio serveru kao root korisnik je sudo.

Sudo (super users do) komanda se koristi, kada želimo da izvršimo komandu koja zahteva povećane privilegije koje dolaze sa root korisnikom.

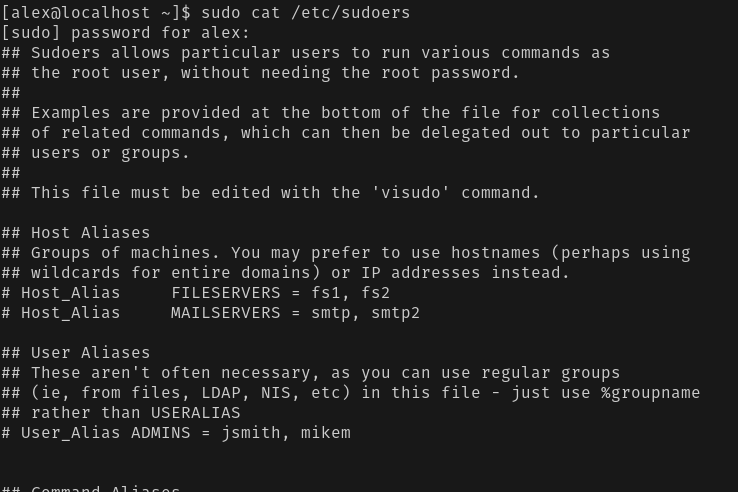
Ako se pokreće komanda za koju nemam privilegije kao korisnik dobijam sledeće:

*Slika 9 :Sudoers fajl*



izvršavanjem ove iste komande sa sudo dobijam:

*Slika 10 :Sudoers fajl sadržaj*



može da se vidi sa slike (slika 10) korišćenjem povećanih privilegija dozvoljen mi je pristup sudoers fajlu kojem ima pristup root korisnik.

Prolaskom kroz sudoers fajl, pri kraju ću naići na %wheel iznad ovog odeljka piše objašnjenje čemu služi.

*Slika 11 :Sudoers fajl wheel grupa*



Sledeće pitanje koje se nalaže jeste da li je korisnik zapravo u grupi wheel ?   
Ovo lako može da se proveri pregledom sledećeg fajla /etc/group

*Slika 12: Sudoers fajl - pripadnost korisnika*



Može se primetiti sa (slika 12) da je grupa wheel prvi segment, drugi segment je x što znači da su upotrebljene skrivene lozinke, 10 je iD grupe, i jedini član wheel grupe je alex. Na novijim sistemima ovo se često zove i admin pa postoje tako i varijacije u sudoers fajlu.

Mogućnost pokretanja superkorisnika pomoću komande sudo, nije pravo svakog korisnika na sistemu, ovo zavisi od kompanije i prava korisnika koji taj učesnik ima u timu administratora.

# 4. instalacija CentOS operativnog sistema

Pre same instalacije nephodno je izabrati odgovarajću arhitekturu za operativni sistem, ako ne izaberem pravilno dobiću upozorenje što nije strašno samo po sebi, ali može da uspori instalaciju. (Treba ponovo da se skida image sa pravilnom arhitekturom)

*Slika 13: Greška arhitekture sistema*



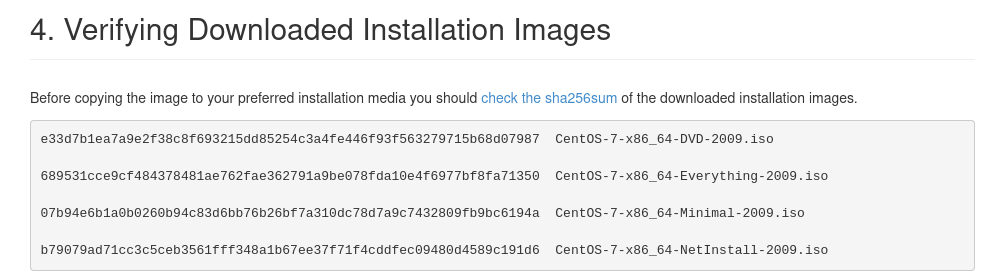
Takođe nije loše proveriti da li je ISO (fajl koji je preuzet sa interneta koji radi isto kao i CD, prilikom boot up sekvence) operativnog sistema ispravan proverom sha256sum

*Slika 14: Shasum CentOS iso fajla*



Kako bi potvrdio da li ovo odgovara zvaničnom sha256sum moram posetiti sledeću stranicu <https://wiki.centos.org/action/show/Manuals/ReleaseNotes/>. Ovde moramo da pronađem verziju operativnog sistema u mom slučaju je CentOS, tako da je dokumentacija CentOS koja ovo obuhvata, postoji i za Debian i druge distribucije Linux.

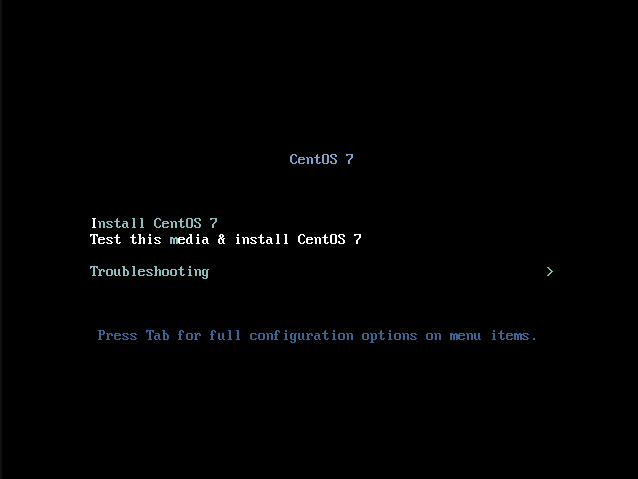
*Slika 15: Prikaz shasum256 operativnog sistema na dokumentaciji*



## 4.1 Opšta konfiguracija

Pre svega treba obezbediti boot order (princip pokretanja medijuma na kojem se nalazi instalacije operativnog sistema) kako bi rekli sistemu da pokrene ISO fajl koji se nalazi na USB ili nekom drugom medijumu. Ovo podešavanje može da se odradi kroz BIOS koji nam nudi odabir medijuma koji će biti upotrebljen prilikom pokretanja računara.

Nakon odabranog medijuma, instalacija počinje nakon što se izabere opcija install CentOS 7, samo po sebi sve je razumljivo, zato što se modernije distribucije trude da ovaj proces učine što lakšim za konfiguraciju.

*Slika 16: Početni prozor instalacije*  


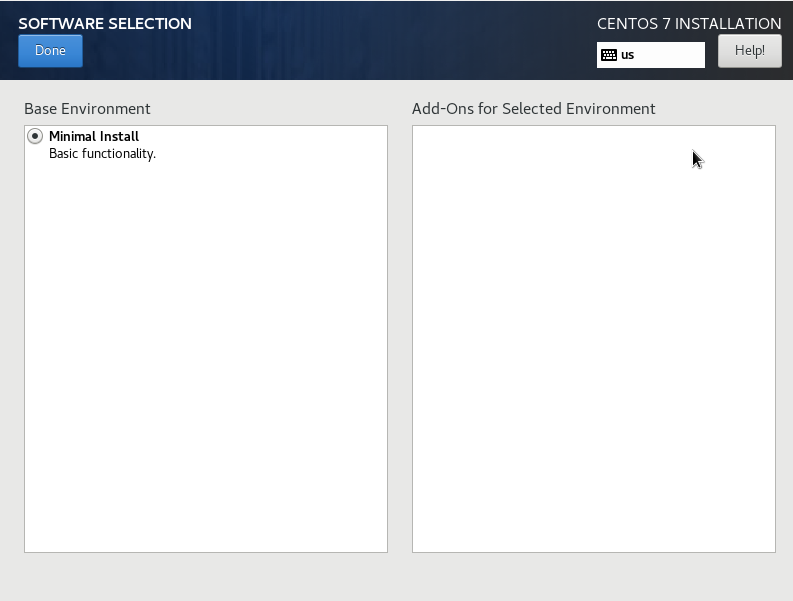
*Slika 17: Odabir jezika instalacije*



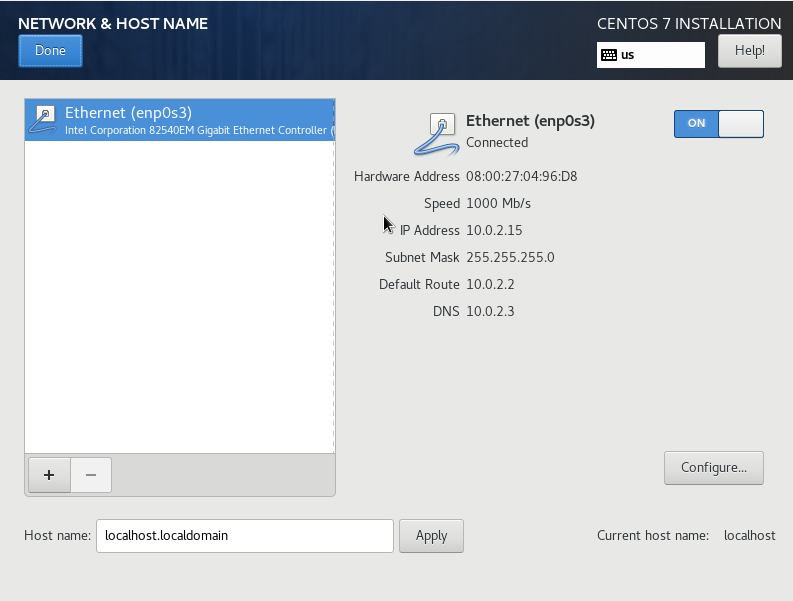
*Slika 18: Glavni instalacioni prozor*



Pošto je ideja da ovo bude server, odabirom opcije Software Selection pokušaću da napravim server što manjim moguće, tj. da se sve radi kroz terminal. Bitno je ovo odraditi pošto aplikacije na serveru same po sebi zauzimaju dosta memorije i dodavanje GUI (Graphical user interface) zauzima još prostora, stim se uglavnom serveri sa GUI izbegavaju.

*Slika 19: Odabir paketa instalacije*  


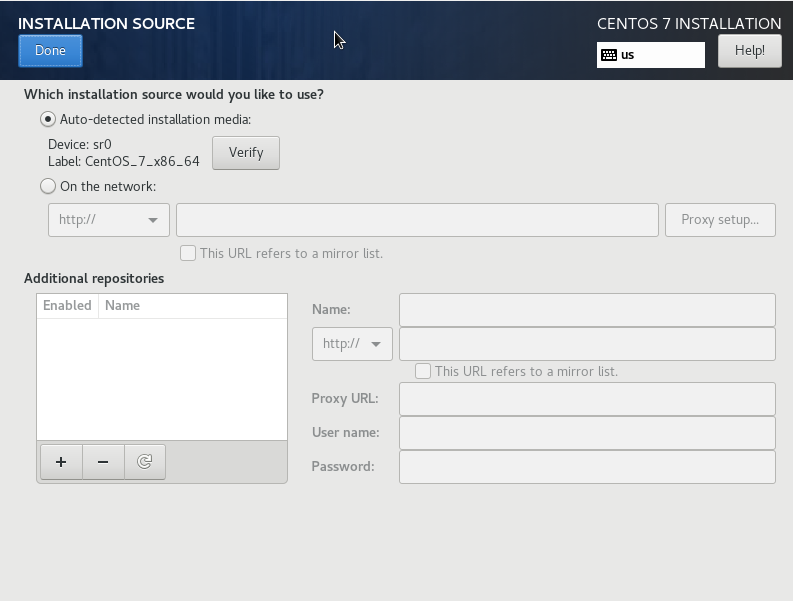
*Slika 20: Uključivanje network uređaja*



Sledeće što je bitno jeste, aktivirati mrežni uređaj odabirom opcije na dugmetu, nakon što se klikne (slika 20) vide se početna podešavanje za mrežu.

Pošto je ovo virtualno okruženje, VirtualBox kreira NAT mrežu sa podrazumevanim podešavnjem, što znači da se VM nalazi na istoj mreži kao i host računar (Virtualna mašina je sama na mreži, ali sa putanjom ka spoljašnjem svetu preko host mašine).

*Slika 21: Odabir instalaciong izvora*

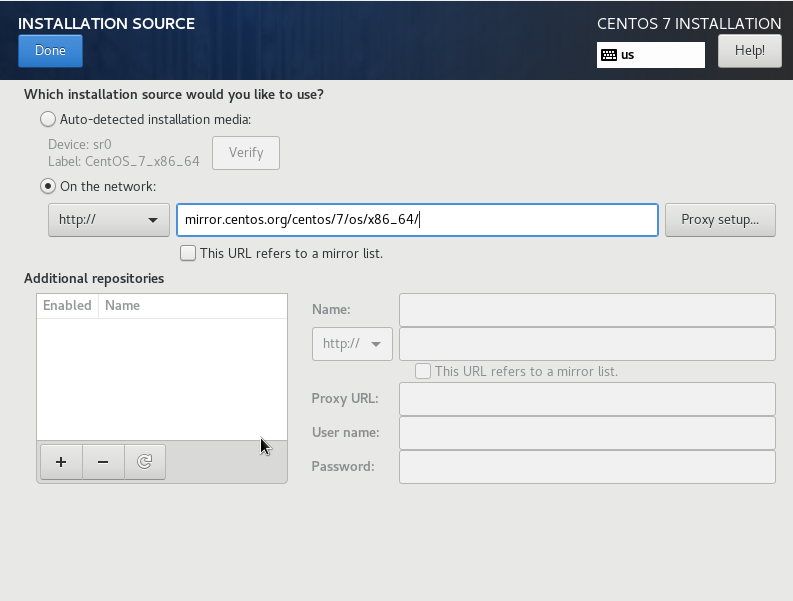


Takođe veoma bitno tokom ove instalacije je podešavanje instalacionog izvora, unutar ovog ekrana je automatski izabran medium, ISO diska (sr0 je oznaka za drive diska).

Promeniću ovo dugme na On the Network (ovo je zbog prethodnog podešavanja za konekcije ka internetu).

Nakon podešavanja dobijam:

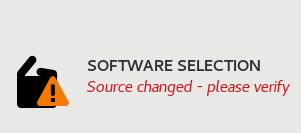
*Slika 22: Unos instalacionog izvora*



Može se videti (slika 22) da sam uneo samo mirror (alternativni server) koji će izvor (source) koristiti prilikom instalacije

Ovom promenom kada se vratim na početni prozor sistem će izbaciti upozorenja da je izvor promenjen i da se mora podesiti ponovo (slika 23)

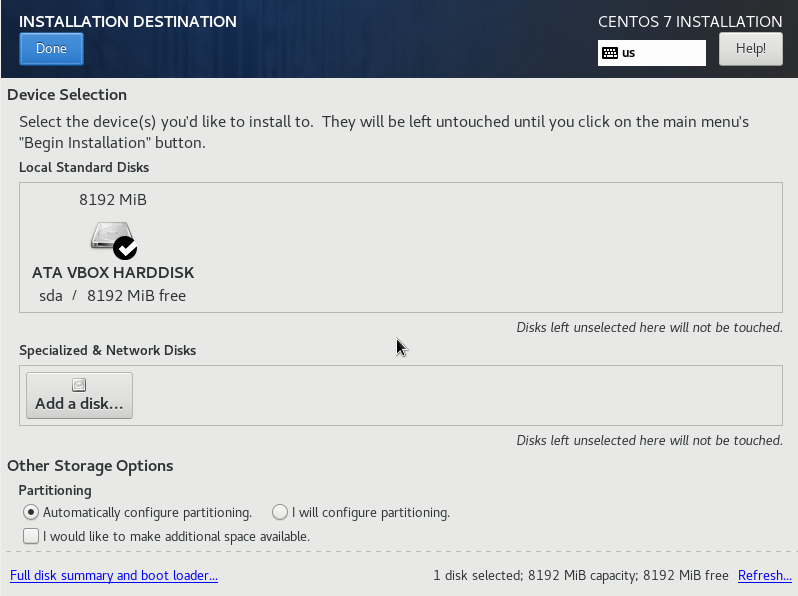
*Slika 23: Potvrda selekcije*



Za sada ću ostaviti isto kao i prethodno na minimal installation, samo treba da se potvrdi u interfejsu.

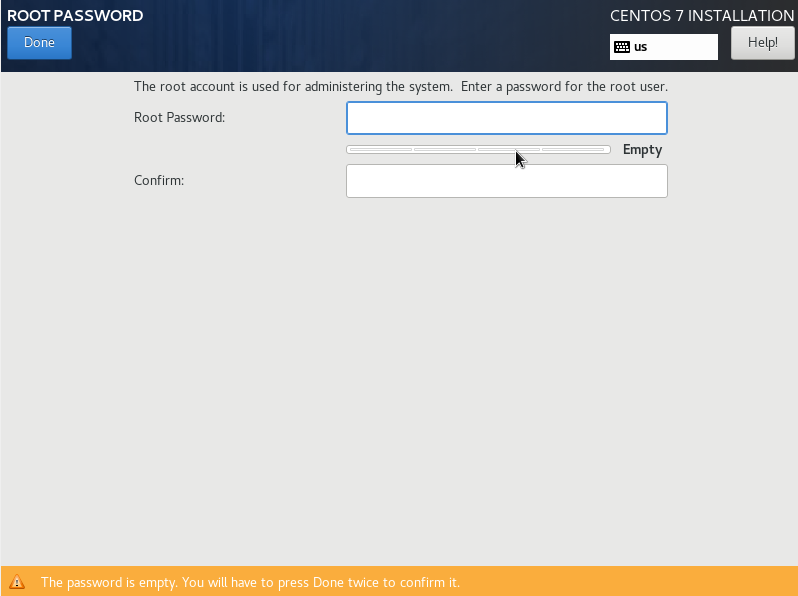
Poslednja stavka je destinacija instalacije (installation destination), u ovom slučaju dozvoliću instalaciji da kreira particije kako želi.

*Slika 24: Odabir diska instalacije*

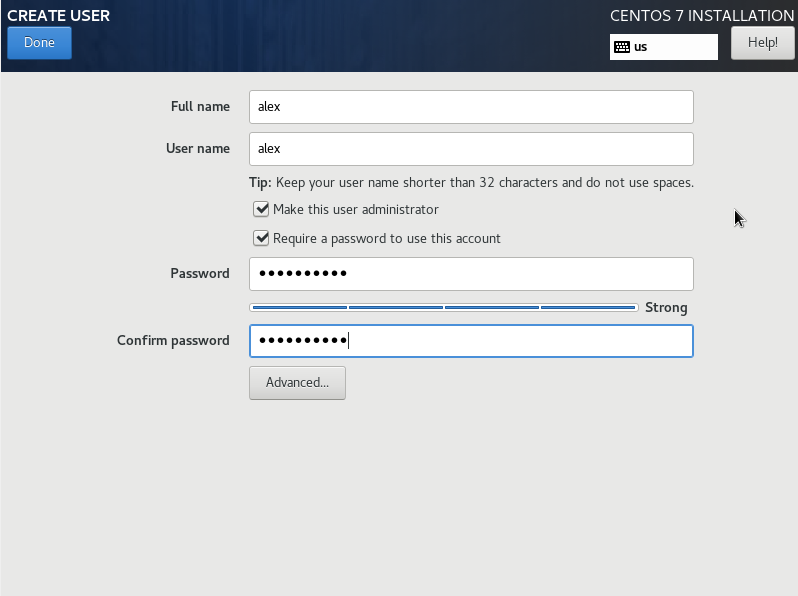


Tokom instalacije pojavljuje se prozor koji mi nudi podešavanje root koriniska. Kada počnemo instalaciju, uglavnom na CentOS 8 i 9 ovo se nalazi na inicijalnom prozoru.

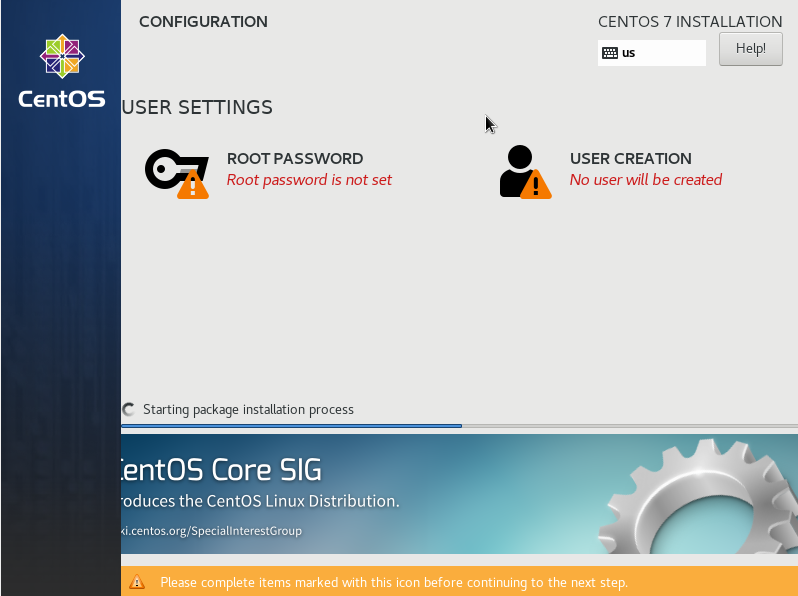
*Slika 25: Kreacija root korisnika*



*Slika 26: Kreacija korisnika*

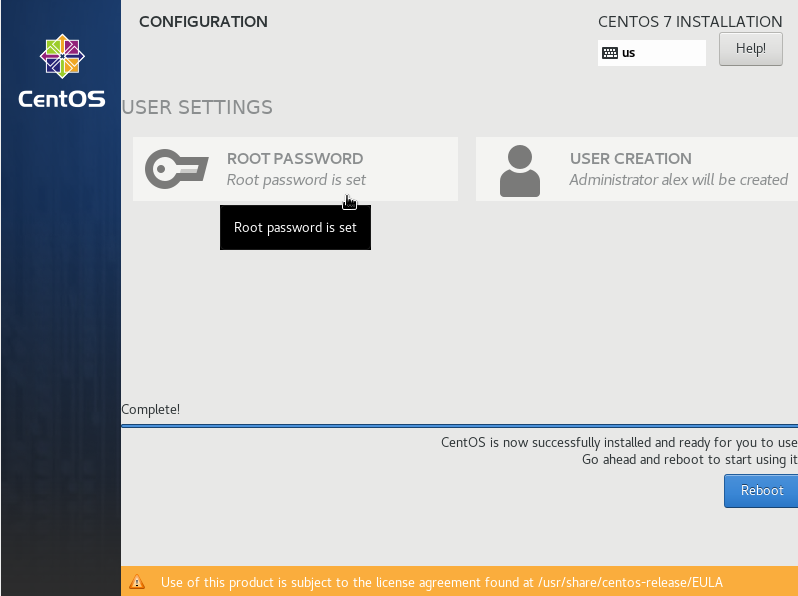


*Slika 27: instalacija pre kreiranje korisnika*



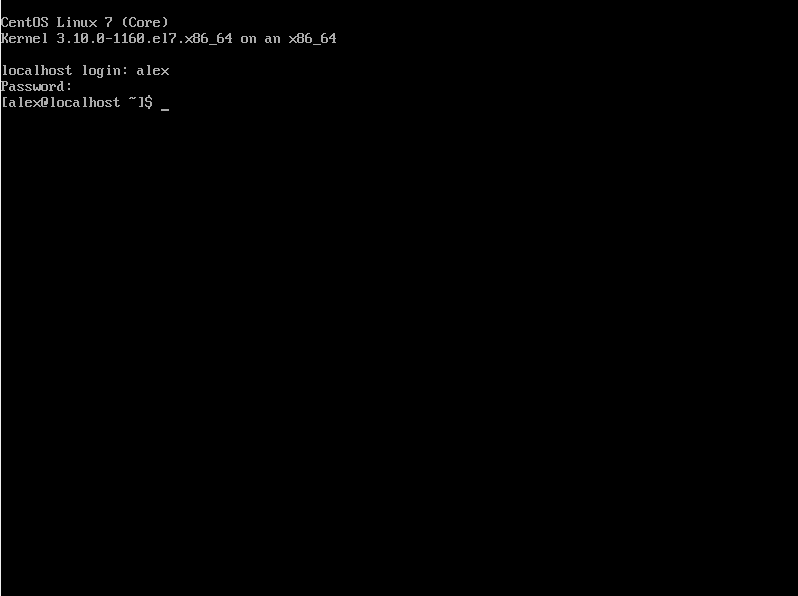
Izabrana je opcija da se korisnik načini administratorom ovo će ga dodeliti sudo korisnicima tj. korisnicima sa povećanim privilegijama na sistemu.

*Slika 28: instalacije nakon kreiranja korisnika*



Prvi login na sistem izgleda ovako:

*Slika 29: Server Login*



# 5. Podešavanje udaljene konekcije SSH klijentom

Generalno kada se radi nešto na serverima podešava se SSH klijent koji omogućava udaljenu konekciju do server ili nekog računara u zavisnosti od njegove funkcije.

Samim tim što se primeri ovde svode na korištenje virtualne mašine pogodno je podesiti ovaj tip konekcije zato što nam olakšava obavljanje zadataka, pošto je virtualna mašina malo sporija.

## 5.1 Opšta konfiguracija

Prvo što ću da uradim jeste da testiram postojanje SSH klijenta, a i da li je port otvoren: *Slika 30: Status sshd servisa*



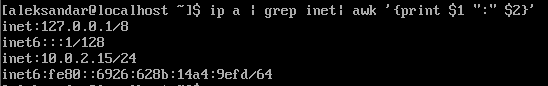
Sa slike (slika 30) servis je aktivan i sluša na portu 22, ali nešto nije uredu kada se konektujem sa host mašine, konekcija se ne ostvaruje. Pretpostavka je da će eventualno vreme čekanja isteći i dobiću ili timeout error, ili refused to connect.

*Slika 31: Login na udaljeni server*



Da bi dobio ovu početnu IP adresu mogu koristiti komandu ‘IP a’ ovo će mi dati neophonde informacije o adapterima koji su priključeni na server

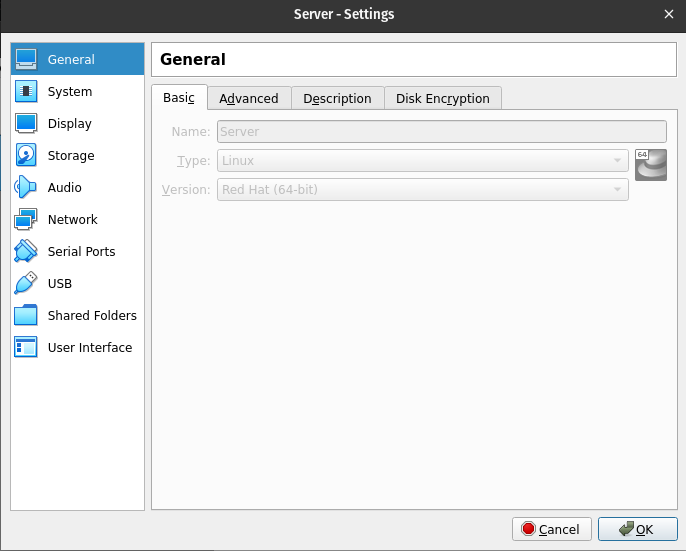
*Slika 32: Upotreba IP komande*



imam loop back IP 127.0.0.1 i IP servera koji je 10.0.2.15, ako bi dobio refused to connect implicira se na to da je IP blokiran od strane servera.

Kako bi konfigurisao udaljenu (remote) konekciju (sa hosta) moraću nameštam port forwarding u Virtualnoj mašini (pošto simuliramo postojanje servera preko ove mašine).

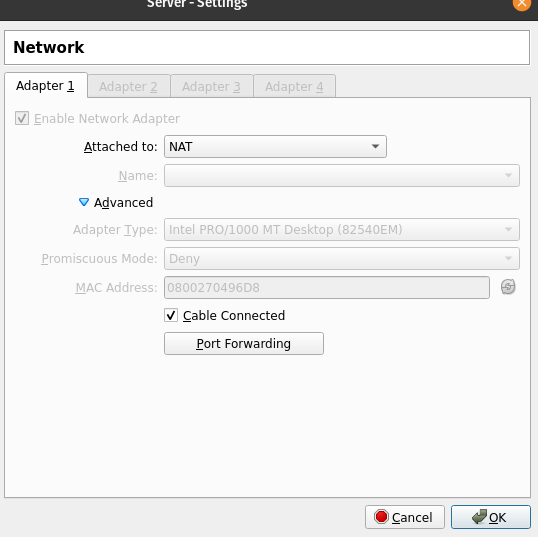
*Slika 33: Prozor podešavanja VM*



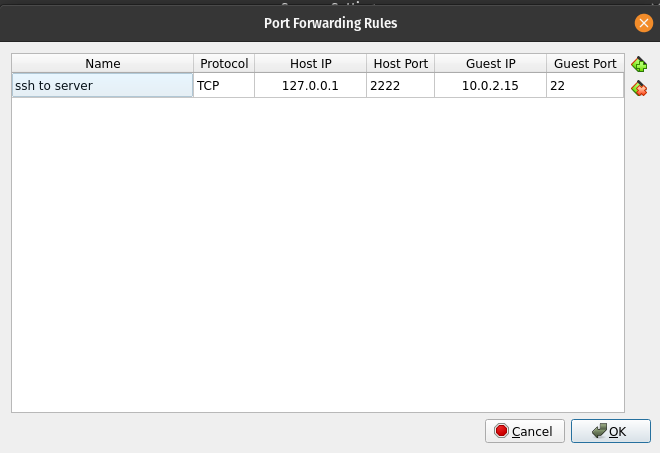
Da bi konfigurisao prethodno navedeno, idem na network u virtual box aplikaciju i pritiskom na advanced dugme dobijamo opciju odabira podešavanje port forwarding

(Port forwarding je metod ručnog podešavanjue, kako saobraćaj treba da prođe kroz NAT mrežu)

*Slika 34: Prozor podešavanja NAT konekcije VM*

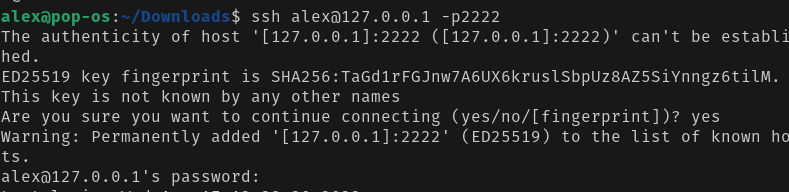


*Slika 35: Podešavanje pravila za port forwarding*



Podešavanje koje sam ovde postavio jeste kada želim SSH konekciju upišem sledeće ssh alex@127.0.0.1 -p2222 server će da prepozna ovaj tip konekcija ka njemu i prebaciće ga na port 22, koji koristi SSH klijent na virtualnoj mašini pri čemu sam ostvario vid konekcije koji je uglavnom dostupan prilikom administriranja servera.

*Slika 36: Login na server koji je kreiran u virtualnom okruženju*

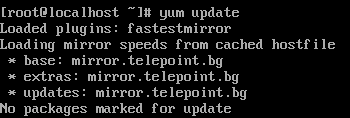


(Guest IP je IP servera koji možemo dobiti pozivom komande ‘ip a’)

Sledeće što ću uraditi jeste update (metod unapređivanja već postojećeg softvera) koji će povući sve poslednje verzije softvera koji je pokrenut na serveru, ukoliko za to postoji potreba. Kako bi obavio ovaj zadatak koristću komandu:

**$ yum update**

***S****lika 37: Upotreba yum update komande*



Prilikom rada često je nephodno proveriti verziju kernela (jezgra), a to se može odraditi na upotrebom sledeće komande:

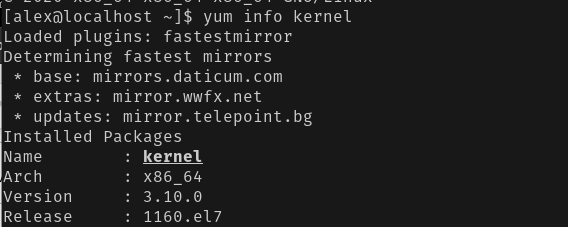
**$ uname -a**

*Slika 38: Upotreba komande uname*



Zatim pokretanjem sledeće komande:

*Slika 39: Prikaz poslednje verzije kernela u yum*



Može se videti da je kernel verzija ista kao i verzija pozvana uname komandom što implicira da je kernel poslednje verzije.

*Slika 40: Verzija kernela*



# 6.Automatizacija pravljenja virtualnog okruzenja

## 6.1 Uvod

Pošto sama instalacija može da bude prilično monotona, i često bi morali ponovo da postavljamo virtuelno okruženje radi testiranja nekih specifičnih problema, iz tog razloga razvijen je sistem koji automatizuje ovaj proces.

Prednosti ove metode kreiranja virtuelnog okruženja su:

• Eliminiše vreme potrebno za ručno kucanje odgovora u VM prozor

• Omogućava automatizaciju pokretanja tekstova za softver u razvojnom okruženju

• Omogućava deljenje tekstualnih fajlova koji se ponašaju kao ‘recepti’ za način izgradnje VM-A, umesto prebacivanja velikih VM imidža. (Ovde se misli na /vagrant direktorijum)

Ovo je oblik infrastructure as code (IAC).

Bitno je napomenuti jedan od metoda automatizacije raspoređivanja okvira, a to je kickstart fajl, koji se često koriste u velikim raspoređivanjima za automatsko odgovaranje na pitanja koje program postavlja korisniku. Ovi fajlovi se podešavaju, ili su napisani “od nule”, a zatim hostovani na veb serveru, na instalacionoj mreži, spremni da ih nekonfigurisana mašina preuzme.

Na lokalnim mašinama, kickstart fajlovi nisu praktični i ne ubrzavaju posao. Kada se testira ova postavka na lokalnim mašinama potrebno je nešto brzo i jednostavno, ali takođe nešto veoma moćno.

Korisna alatka za ovo je Vagrant, razvijena od strane kompanije “Hashicorp” kao softver otvorenog koda, a može da se upotrebi za automatsko obezbeđivanje VM, pa čak i celih razvojnih okruženja.

Okruženja su obezbeđena u obliku Vagrant fajlova koji se nalaze na klaudu, ovi fajlovi imaju već postojeće konfiguracije što nam skraćuje posao postavljanja novih virtualnih mašina. Ukratko podešavanje Vagrant okruženja u VM…. (Ova aplikacija je dostupna i na Windows mašini, ali prikazaću instalaciju kroz Linux instalaciju)

Pre svega mora da se instalira aplikacija, na Linux mašinama možemo to izvesti kroz upravljač paketima u slučaju Debian sistema to bi bio apt.

## 6.2 Konfiguracija Virtualnog okruženja

1) **$ sudo apt install vagrant** (pošto u bazi podataka postoji ova aplikacija biće automatski instalirana i podešena)

2) Kreiram negde direktorijum na Linux mašini, upotrebom **mkdir** komande koja nam omogućava ovu akciju. (Uglavnom je preporučljivo da ovaj direktorijum bude unutar home direktorijuma radi lakšeg upravljanja)

3) inicijalizujem direktorijum kao mesto za Vagrant konfiguraciju koristeći

**$ vagrant init**

4) Naredna komada je malo kompleksnija o njoj neću pričati sada, ali ona samo menja tekst na određenoj lokaciji u ovom slučaju za vagrant fajl koji je kreiran nakon inicijalizacije.

**$ sed -i ‘s#config.vm.box = “base”#config.vm.box = “centos/7”#g’ Vagrant file**

Šta se dešava ovde jeste da govorim vagrantu da želim da koristim CentOS7 image za ovu virtualnu mašinu (Unutar kreiranog vagrant fajla postoji base koji će biti zamenjen sa centos/7)

5) Startujem virtualnu mašinu koristeći **vagrant up**

6) Testiram login sa **vagrant ssh** (Ovo je unapred već kreirano tako da je od samog početka dostupno)

imam i **vagrant destroy** koji briše istancu potpuno iz vm box.

Bilo mi je nephodno da objasnim prvo ručno podešavanje, a zatim automatsko na neki način želim da prikažem mali proces automatizacije u ovom okruženju.

# 7. Daljinsko administriranje pomoću SSH

U realnom svetu sistemski administrator obavlja poslove udaljeno, ali u isto vreme može da bude na licu mesta kraj servera i da pravi neke promene.

Pretpostavka je da neće svaki put ići do data centra (mesto gde se nalaze serveri), ako se neke stvari mogu rešiti samo što se poveže na udaljeni server.

Ovo je razlog zašto će sistemski administrator koristiti SSH (secure shell) konekcije koje omogućavaju administratoru Linux sistema da pristupi udaljenom serveru i napravi neke promene koje su neophodne.

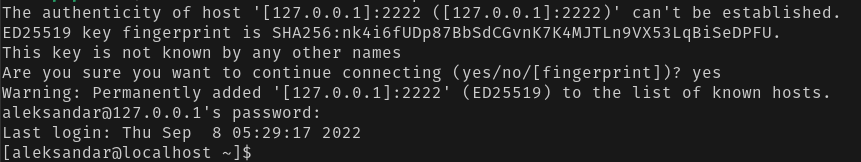
Secure Shell protokol je jedan od korišćenih primera ima mnogo varijacija na ovo, ali pošto je jedan od poznatijih, a i bezbednijih odlučio sam da prestavim ovaj. Dolazi kao fabrički servis na skoro svim Linux sistemima.

Komanda koja se koristi:

**$ ssh root@192.168.56.101 -p522 komanda ssh**

Prvi deo komande koristi se kao korisnik koji je root u ovom slučaju drugi deo komande nakon @ simbola se odnosi na server IP, ovo čak može i da se zameni sa hostname (ime servera) i treći deo -p odnosi se na port koji će biti korišćen u ovom slučaju to je 522 nakon ukucane komande dobiću ključ, koji bi trebao da potvrtdim i nakon toga bi uneo lozinku.

*Slika 41: Potvrda prilikom udaljene SSH konekcije ka serveru*



Uglavnom je ovo uredu metoda konekcija (Slika 41), ali može i na alternativni način da se uspostavi konekcija ka serverom, to može da se izvrši generisanjem ključa.

## 7.1 Generisanje i upotreba ssh-keygen

Većina ljudi koristi jednostavne lozinke koje se mogu lako pogoditi koristeći različite algoritme, klasična metoda je brute force (metoda sile) gde se pokušavaju sve moguće lozinke , ili imamo i dictionary attack (metoda rečnika) koji koristi zadate reči da pronađe lozinku.

Jedna metoda koja je razvijena za sprečavanje ovih napada, a i za identifikaciju samo autorizovanih korisnika je korišćenje kriptografskih ključeva koji mogu biti privatni i javni.

To funkcioniše na sledeći način, kada je javni deo ključa na serveru, može da se izvrši SSH konekcija sa mašine koja poseduje privatni ključ.

Ovaj tip konekcije može donekle da olakša povezivanje, gde se izbegavaju neki ekstra koraci kao što je upisivanje lozinke, tj upisuje se samo prvi put dok svaki naredni ta fraza (passphrase) ostaje zapamćena na mašini i koristi se iznova ukoliko želim da se povežem na isti server.

Kako bi to izgledalo između dva servera? Prvo što bi uradio spojio bi se na server pošto ja u mojim primerima koristim VM (virtualna mašina),

*Slika 42: Login na virtualnu mašinu*



iz prethodnog odeljka (Automatizacija pravljenja virtualnog okruzenja) može da se vidi šta je vagrant pošto je ovo generisano okruženje daje nam olakšicu konekcije ka novoj mašini koja se zove centos1.

Drugo što bi uradio na nekoj mašini jeste da generišem neki par ključeva koji će mi biti koristan prilikom konekcije ka drugom serveru.

*Slika 43: Generisanje ključa*



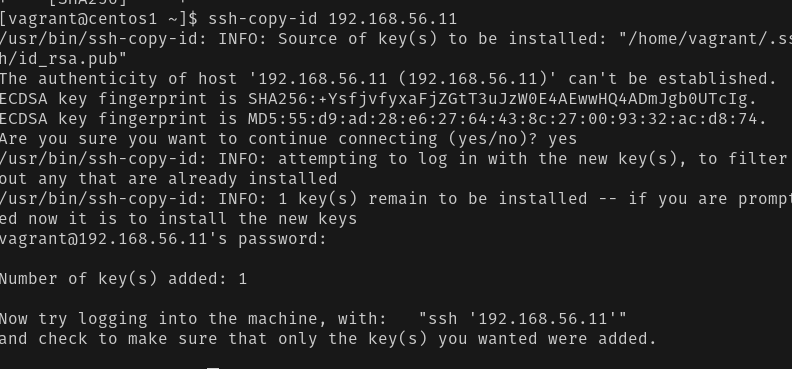
Ključ je generisan čak i komanda pita, ako želim negde drugde da sačuvam.

Unosim passphrase koji želim, i ono biva sačuvano na serveru kao id\_rsa i id\_rsa.pub.

Naravno ovo samo po sebi ništa ne znači, da bi mogao ovo koristiti neophodno je da kopiram javni ključ (public key) na odredišni server.

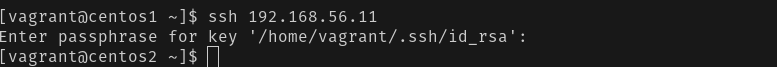
Ovo mogu uraditi ručno ili koristeći komandu **ssh-copy** kadestinacionom serveu koji prihvata ključ kada mu se unese lozinka koju će zatražiti prilokom razmene.

*Slika 44: Kopiranje ključa na udaljeni server*



Može da se primeti da mi komanda traži frazu (passphrase) koji sam zadao prilikom generisanja ovog ključa, nakon unosa omogućena mi je konekcija.

*Slika 45: Login na udaljeni server preko ključa*



Naravno ovaj tip ključa koji je korišćen nije jedini postoje još i ED25519, dsa, ecdsa, rsa.

Pošto je SSH prilično opširan potrudio sam se ovde da izdvojim neke osnove, korisne stranice su zvanična dokumentacija i manual (man) stranice iz terminala vezane za komandu ssh.

https://www.openssh.com/manual.html

# 8. Networking/DNS

Mi kao ljudi ne pamtimo dobro brojevne adrese u pretrazi nekih željenih sadržaja, recimo da nas neko pita koja je IP adresa google.com?

Naravno neko na ovo pitanje će znati odgovor, ali uglavnom se svodi da neće imati odgovor, ako ga prvobitno ne pretraži.

Odavde nastupa DNS koji pretvara URL u brojeve ili IP adresu koju računar razume, pošto se u mašinskom jeziku ne koriste imena.

Recimo da tražimo neku stranicu, mi možemo upisati željeno ime amazon.com, dešava se da IP adresa nije ista na dva računara.

Računar ili ti server u ovom slučaju ima zadatak, ako je velika potražnja za jednom IP adresom on će izvesti takozvano balansiranje ili ti load balancing kako bi smanjio opterećenje na server, i samim tim što bi korisnik pamtio IP adresu ako može da bude različita za dve zemlje recimo Amerika i Ujedinjeno Kraljestvo.

IP adrese mogu da budu iz dve grupe iPv4 ili iPv6 gde je iPv6 mnogo novija verzija i trebala bi da zameni iPv4 u bližoj budućnost zbog nedostatka IP adresa.

DNS je glavni gradivni element modernog interneta, i može da se koristi analogija da je kao ‘Telefonski imenik’ uzima ime i pretvara ga u telefonski broj u našem slučaju pretvara ime domene u IP adresu. Kada bi pretraživali neki broj na našem telefonu vrlo često ćemo prelistati naš imenik, a zatim izabrati neki broj ovo se zove lokalno skladište, imamo isti slučaj i na računaru, gde on ima svoj lični lokalni fajl koji se zove hosts fajl koji razrešava konekciju u zavisnosti kako je podešen.

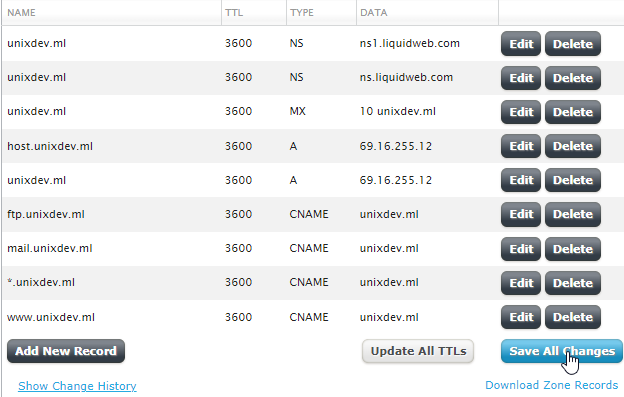
Vrlo čest primer upotrebe hosts fajla je da organizacije podese ovaj fajl da se samo razrešava na imena koja se nalaze unutar organizacije i tako se kreira mala DNS mreža razrešavanja.

Uprkos ovome na internetu stvari funkcionišu malo drugačije, imamo distribuirane DNS sisteme koji razrešavaju imena kao što su google.com, Facebook itd…

Na distribuirnim DNS sistemima DNS rekordi se razrešavaju u zavisnosti gde je DNS zona koja je zadužena za domenu ili registrar u zavisnosti od konfiguracije.

Kada se nalazimo u serveru ovo se razrešava preko BIND servisa, ali na višem nivou prvo se postavlja na mestu gde nam je sajt.

*Slika 46: DNS zona*



*izvor: liquidweb.com*

Iz primera podešavanja rekorda (Slika 45), može da se primeti kako trebaju da se razrešavaju na serveru.

*Slika 47: DNS zona podešavanje*



*izvor: liquidweb.com*

Svaki hosting ima svoj specifičan UI, ali uglavnom bi trebali da budu isti, stim da postoje neke razlike prilikom postavljanja CNAME, npr korišćenje @ simbola da ukazuje na domenu, no to nije bitno za sada.

Treba prvo početi od mogućih rekorda koji mogu da budu na serveru a to su:

* A
* AAAA
* CNAME
* MX
* PTR
* NS
* SOA
* SRV
* TXT

## 8.1 A record

A record ili adresni rekord, dodeljejue IP adresu domeni ili subdomeni.

Kada je DNS bio razvijen prvobitno bilo je preporučljivo da dva A rekorda se referiraju na istu IP adresu.

Na primer imamo nekidomen.tld i želimo da dodelimo 10.10.0.1 IP adresu našem web serveru, onda bi trebali da kreiramo A rekord sa “www.nekidomen.tld” kao Fully Qualified Domain Name i u polju vrednosti bi postavili IP adresu servera 10.10.0.1.

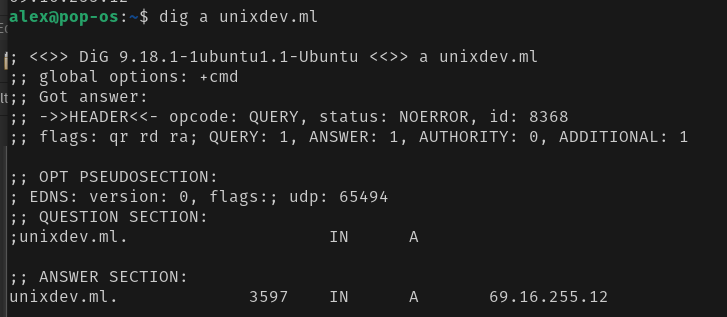
pretpostavka je da će svi zahtevi za www.nekidomen.tld biti poslani na server sa IP adresom koju smo specificirali u polju vrednosti DNS rekorda.

Korisno je upotrebiti A rekord kada imamo subdomene koje se nalaze na različitim sistemima.

Korisni savet, koji bi možda bio preporučljiv je da koristimo wild card “\*.nekadomena.tld” A rekord koji nam omogućava da bilo šta stoji ispred kao subdomena “bilošta.nekidomen.tld” i zatim se razrešava u zavisnosti od IP koji mu je dodeljen u polju vrednosti (value field). Ali uglavnom je preporučljivo koristiti CNAME rekorde za ovo, nego wildcard A rekord.

Korisna komanda koja pomaže Sistemskom administratoru da proveri rekorde neke domene je **dig** i to može da se uradi ovako:

*Slika 48: Komanda prikaza A rekorda domene*



Komanda mi daje način na koji pita domenu šta joj je adresni rekord nakon što specificram šta tražim i za koju domenu imam flag a koji kaže komandi da traži samo A rekorde, naravano ovo može biti bilo šta drugo recimo CNAME, MX, AAAA …

Svako polje znači nešto ako pogledam Answer sekciju videću TTL (time to live) koji je 3597 sekundi onda imam iN koje znači internet, A što se odnosi na rekord i IP adresu ili ti value field koji je popunjen.

## 8.2 AAAA rekord

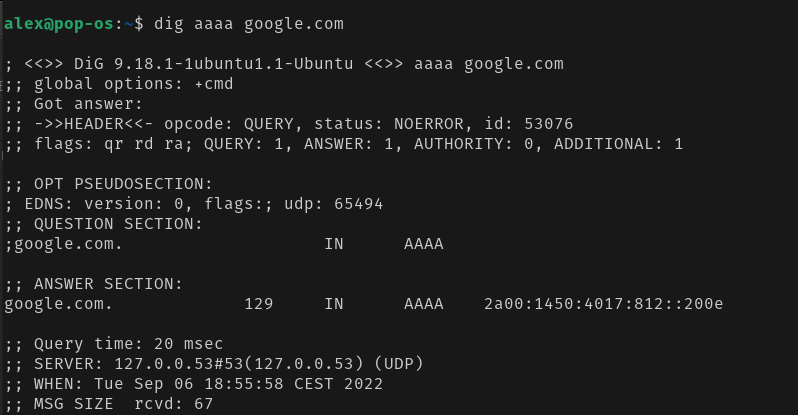
AAAA rekord ili iPv6 adresni rekord mapira hostname na 128-bitnu iPv6 adresu.

Regularni DNS adresni rekord ili ti A je definisan na 32-bita kao iPv4 adresa, zbog nedostatka ove verzije razvijen je ovaj rekord koji će se poklapati sa iPv6 adresama, ovo nam omogućava da domene asociramo i ovako ako to želim.

Četiri A slova su mnemonika (veština pamćenja) da nas asocira da je iPv6 četiri puta veći od iPv4.

AAAA rekord je struktuiran na sličan način kao i A rekord oba su binarni i master fajl format, samo što je iPv6 veći…

*Slika 49: Komanda prikaza AAAA rekorda domene*



AAAA rekordi su to da pomognu prilikom tranzicije i koegzistencije između iPv4 i iPv6 mreža.

iPv4 nameserver može da vrati iPv6 adresu recimo.

## 8.3 CNAME rekord

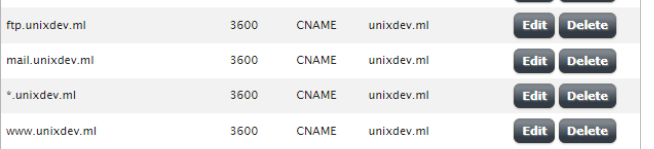
CNAME rekord ili canonical name rekord čini da domensko ime može da se predstavi kao alias druge (alias domen dobija sve subdomene i DNS rekorde orginalnog).

CNAME bi trebao da se koristi kako želimo da asociramo novi subdomen na već postojeći A rekord možemo uradit recimo “www.nekadomena.tld” na “nekadomena.tld”, koji bi trebao da ima već postojeću dodeljenu IP adresa kao A rekord.

Ovaj metod podešavanja mi omogućava da imam koliko želim subdomena, bez da specificiram IP na svaki rekord. Generalno pravilo je koristi CNAME, ako imaš više servisa koji pokazuju na istu IP adresu. Na ovaj način, ako prepravimo A rekord sve ostale domene servisi koji pokazuju na glavni A rekord će prepraviti svoju IP adresu.

Primer CNAME rekorda na nekom serveru: sa primera (Slika 43) može da se vidi par servisa koji koriste CNAME da bi pokazivali na isti IP kao A rekord glavne domene.

*Slika 50: Prikaz CNAME na hostingu*

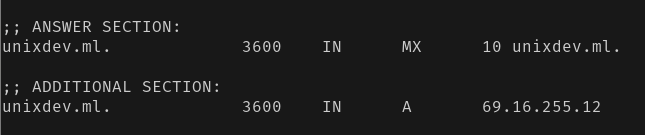
 *izvor: liquidweb.com*

Recimo ftp.unixdev.com vidm da je subdomena FTP servis koji je namenjen za istoimeni protokol koji će server koristiti nakon toga imamo TTL 3600 (u sekundama) i na kraju imam value koji je unixdev.ml koji se odnosi na ime domene ili ti glavni A rekord.

## 8.4 MX rekord

MX rekord, ili mail exchange rekord ovaj rekord mapira ime domena na listu mail exchange servera za tu domenu, tj ovaj rekord govori mail serveru gde treba da salje mailove u sustini.

*Slika 51: Struktura MX rekorda*



Kao što se može videte (Slika 51) MX rekord pokazuje da svi mail idu na @ ili unixdev.ml i trebaju biti rutirani na istoimeni domen, tj njegov lokalni inbox.

Takođe DNS rekord pokazuje da domena unixdev.ml je lociran na IP 69.16.255.12 i da se rutira na taj mailserver. Ovde se završava posao MX rekorda i onda preuzima mail server na toj IP adresi, on skuplja mail i distribuira ih na nekog korsinika na serveru.

Bitna stvar da ima tačku na kraju ( .) nakon imena domene u MX, ako ne bi imao ovu tačku nakraju domena bi izgledala nešto ovako “unixdev.ml.unixdev.ml”. Broj 10, indicira preferencioni broj ili broj prioriteta, gde je niži broj veći prioritet. Odatle stoji da je mail uvek rutiran na server koji ima najveci prioritet ili najmanji preference broj. Ako imam samo jedan Mail server, sigruno je koristiti 0 kao najveći prioritet.

## 8.5 PTR rekord

PTR rekord ili pointer rekord mapira iPv4 adresa na canonical name za hosta.

Postavljanje PTR rekorda za hostname u in-addr.arpa domen koji odgovara IP adresi koja implementira reverse DNS lookup za tu adresu

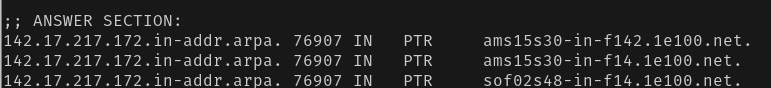
Na primer www.nekadomena.net ima IP adresu 122.0.3.16, ali PTR rekord to mapira kao

16.3.0.122.in-addr.arpa

*Slika 52:* PTR rekord na osnovu A rekorda sa prethonde slike



*Slika 53: A rekord prikaz*



IP adresa je ispisana sa desna na levo, i dodat je in-addr.arpa kao što se vidi (Slika 53) ovo je sada ne levoj strani gde je domen, a na desnoj strani kao vrednost su serveri namenjeni za google ili google.com.

Ovaj rekord se uglavnom koristi za bezbednost i anti-spam mere, gde većina web servera i email servera rade reverse DNS lookup da provere da li host zapravo dolazi od iste lokacije odakle prikazuje da dolazi. Preporučljivo je da imamo postavljen PTR, takođe je veoma preporučljivo da imamo ovo postavljeno ukoliko pokrećemo SMTP-mail server.

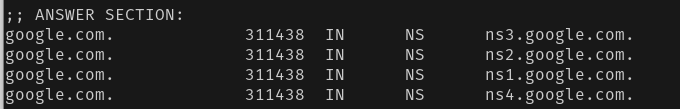
## 8.6 NS rekord

NS rekordi ili name server rekord koji mapira ime domena na listu DNS servera koji su autorativni za taj domen. Delegacija zavisi od NS rekorda.

NS rekord ili Autorativni name server pokazuje gde su Autorativni name serveri za specifičnu domenu. NS rekord Autorativnih Name servera za bilo koji domen bice pretstavljen na Parent serveru. Ovi rekordi se zovu delegacioni rekordi i kao rekordi na Parent serveru pokazuju delegaciju domene na autorativne rekorde.

NS rekordi koji mogu da se pronađu na parent serveru trebali bi da odgovaraju NS rekordima na autorativnom serveru, ali postoje NS rekordi koji se vide na Autorativnom serveru i nisu isti kao na parent serveru. Ova postavka je obično korišćena da se konfiguriše skriveni name server.

*Slika 54: Primer NS rekorda*



## 8.7 SOA rekord

SOA ili State of Authority rekord specificira DNS server koji daje autorativne informacije o domeni na interntenu, email domen administrator, domenski unikatni serijski broj, nekoliko tajmera koji odgovaraju refresh-osvezavanju zone.

SOA (State of Authority) rekord je navažniji deo zone fajla.

SOA rekord je način da domenski administrator da osnovne informacije o domeni, kao što su koliko se puta se update, kada je poslednji update rađen, kada da se proveri za više informacija, koji je email domenskog administratora itd… Zone fajl može da sadriži samo jedan SOA rekord.

Dobro podešeni SOA rekord može da samnji upotrebu mreže (bandwidth) između nameservera, poveća brzinu kojom može da se pristupu stranici, omogući stranici da bude aktivna čak i kada je glavni DNS server van funkcije.

*Slika 55: SOA rekord*



Prvo polje je ime ili google.com koji sam pretražio, drugo polje je time to live koje je 43 sekundi, iN stoji za internet što je implikacija da je domena na internetu.

SOA je tip rekorda nakon toga imam nameserver gde se domen nalazi i nakon toga imam kome odgovara email za ovo što je cloudflare.

imamo serijski broj koji je 472681414 koji daje kada je poslednja revizija izvršena trebalo bi da je format yyyymmddnn (nn je broj revizije) nakon toga imam refresh rate u sekundama koji je 900 update retry koji je 900, expiry koji je 1800 i 60 koji stoji kao minimum ili osnovno vreme koliko bi trebali slave(podčinjeni) serveri da čuvaju zone fajlove.

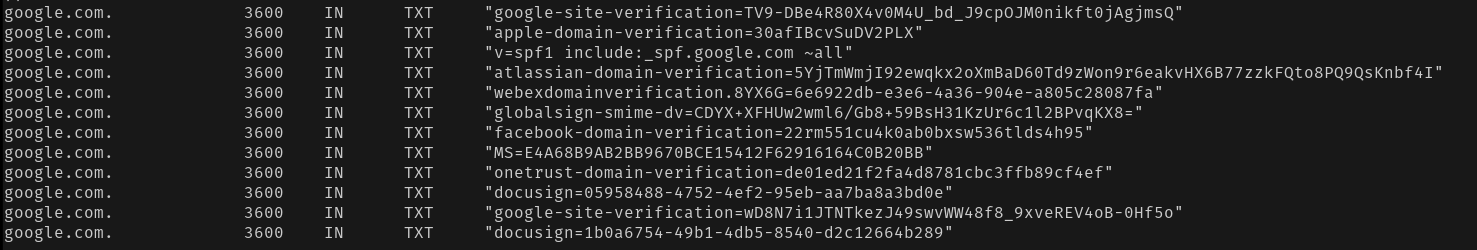
## 8.8 SRV rekordi

Teorija iza SRV rekorda jeste da ako imam ime domene (domain name) npr primer.com, neki servis recimo http, koji se pokreću na tcp u ovom slučaju, DNS upit će biti izvršen da se pronađe hostname koji daje ove informacije o domeni koje mogu a nemoraju da budu unutar domene.

## 8.9 TXT rekordi

TXT rekord dozvoljava administratoru da ubaci proizvoljan tekst u DNS rekord. Na primer, ovaj rekord se koristi da se implenetiraju SPF rekordi i njihova specifikacija.

*Slika 56: Primer sintakse TXT rekorda*



Prikazan je i prethodno pomenut spf rekord ovde

v=spf1 include:\_spf.google.com -all ovo se koristi za verifikaciju email servera.

# 9. Veb serveri, baze podataka i server za email.

Veći deo Veba se pokreće na Linux serverima (sa segmentiranim i mračnijim uglovima Veba na Windows).

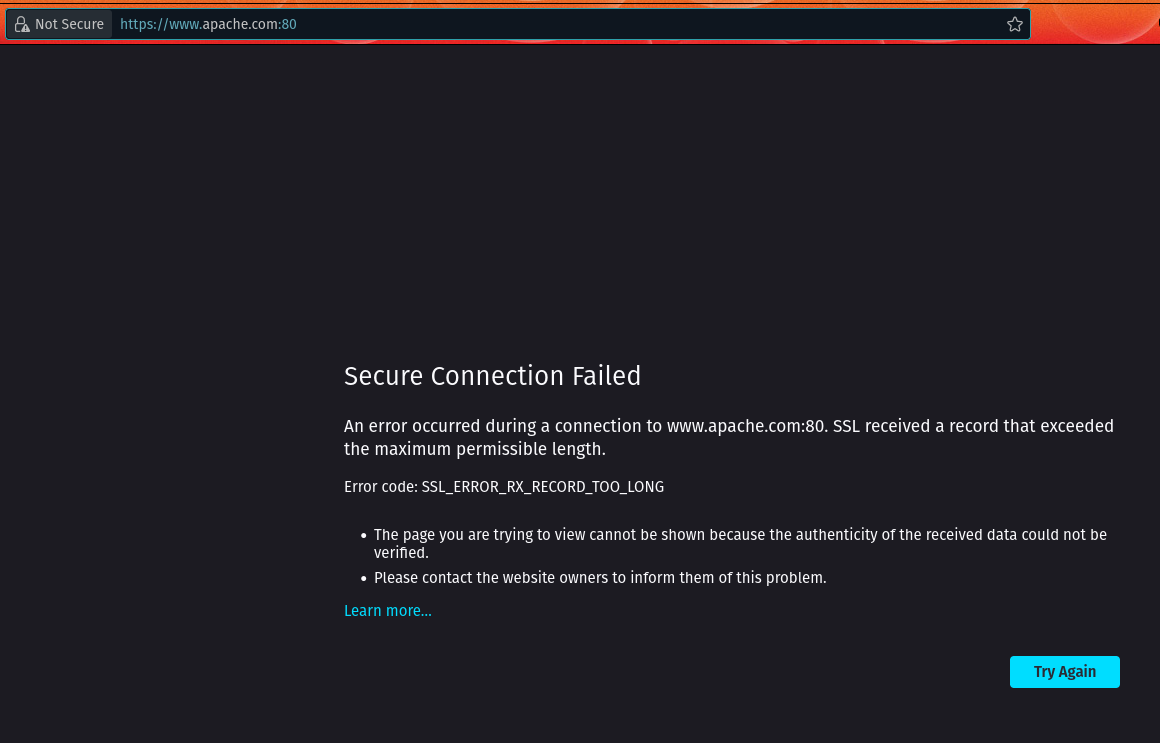
Postoje dve vrste veb servera, oni koji se pokreću sa Apache i oni koji se pokreću Nginx, gde je Apache stariji i još uvek vodi u trci najdominantnijeg web servera, ali Nginx nije daleko iza zbog rastuće popularnosti.

Veb server je komponenta sa kojom je velik zahtev kada se otvara veb stranica.

Tradicionalno port na kojem se osluškuje je 80 za Hypertext Transfer Protocol (HTTP), ili port 443 za Hypertext Transfer Protocol Secure (HTTPS).

Kada se ukuca URL u pretraživač, ovi portovi su generalno skriveni, osim ako su eksplicitno definisani, na primer ako unesemo https://google.com u Chrome ili Firefox, biće učitan veb sajt, ali neće prikazati koji je port. Isto tako ako se stranica pretraži sa portom na primer 80 dobićemo isti rezultat.

*Slika 57: Secure connection problem*



Dobili smo nesigurnu konekciju sa portom 80 preko sigurnog protokola HTTPS, nakon pokušane pretrage.

## 9.1 instalacija httpd(apache) na CentOS

Na CentOS sistemima Apache je httpd.

instalacija softvera se vrši preko yum package manager na sledeći način:

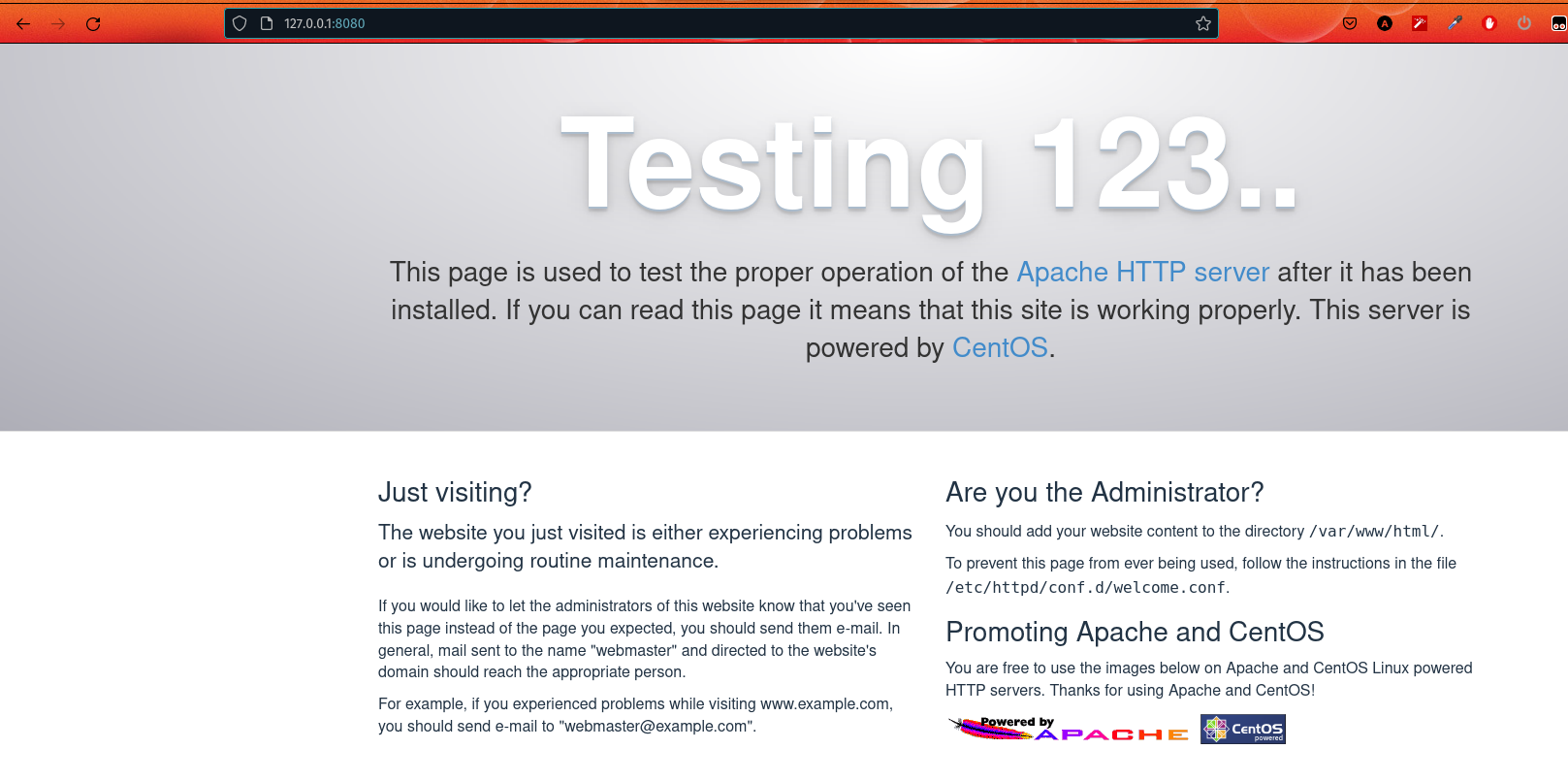
$ yum install httpd -y (flag y je da ne moramo da potvrđujemo paket nakon što se počne instalirati odnosno da izbegnemo fiziči input yes potvrde)

Nakon toga treba da omogućim servis, tj da podesimo da servis but stalno aktivan prilikom pokretanja sistema recimo ovo možemo porediti sa aplikacija koje se pale kada pokrenemo Windows na primer anti virus….

$ systemctl enable –now httpd (now stoji da odmah ponovo pokrene i aktivira servis)

Kada sam podesio servis/daemon trebalo bi da izgleda ovako:

*Slika 58: Apache početna stranica*

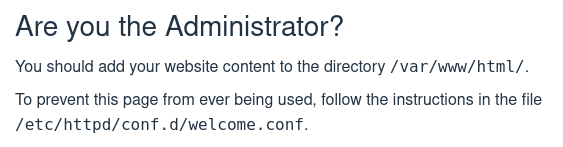


Ovaj splash ili ti index stranica mi govori da je Apache podešen po podrazumevanim podešavanjima.

### 10.1.1 Osnove Apache konfiguracije

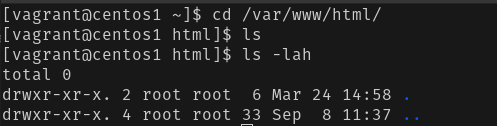
Pogledom na stranicu uočava se sledeća poruka:

*Slika 59: Poruka gde se nalazi konfiguracija od apache stranice*



primećujem da sam Apache nudi gde treba sadržaj stranice da se nalazi.

*Slika 60: Sadržaj www/html direktorijuma*



Kao što se vidi, sam direktorijum je prazan, ako nešto upišemo ovde recimo

$ cat << HERE | tee -a /var/www/html/index.html

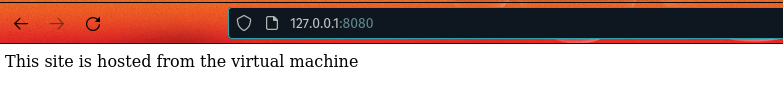
This site is hosted from the virtual machine

HERE

Recimo da kreiram fajl index.html koji će sadržati informaciju da je ovaj sajt zapravo hostovan na virtualnoj mašini.

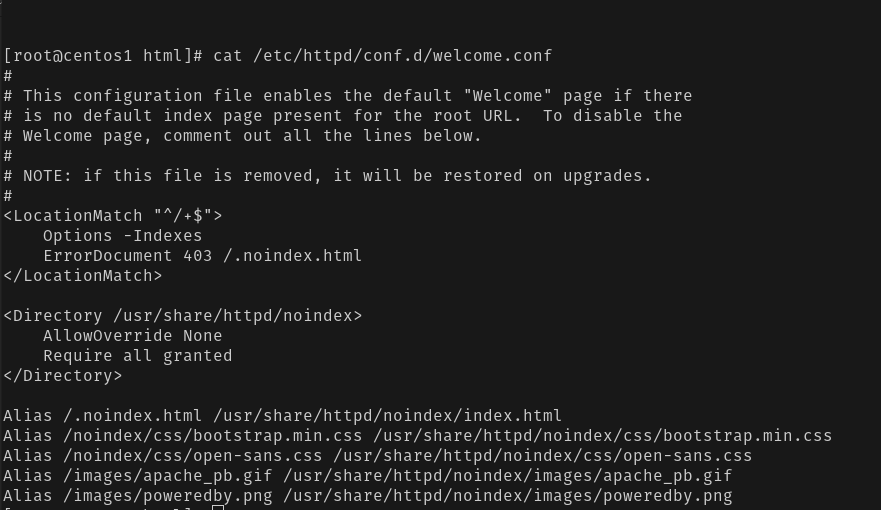
Samom kreacijom ovog fajla apache server će ovo očitati i promeniti sadržaj na stranici:

*Slika 61: Prikaz index.html stranice*



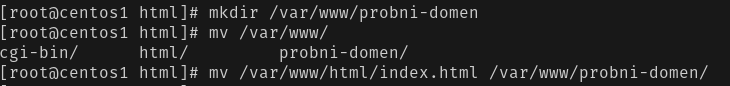
upotrebljen je direktorijum da zameni stranicu, ali za sada ne znam odakle zapravo vuče konfiguracije, ako pogledam sledeći fajl: $ cat /etc/httpd/conf.d/welcome.conf

*Slika 62: Welcome konfiguracioni fajl*



Nešto važno za ovaj fajl jeste da, konfiguracioni fajl uključuje podrazumevanu stranicu Welcome, ako ne postoji index.html unutar zadatog direktorijuma odakle isčitava fajlove. Pošto postoji velika količina vebsajtova (virtualnih hostova) na jednom hostu nije idelano da se samo zada index.html. iz ovog razloga mogu da napravim direktorijum za vebsajt odakle će čitati informacije. Prvo što treba da se odradi jeste da kreiramo direktorijum za naš domen i pomerim index.html fajl u njega

*Slika 63: Kreiranje docroot domene*



Zatim moram kreirati konfiguracioni fajl za ovaj domen u direkotrijumu kao /etc/httpd/conf.d/probni-domen.conf koji će govoriti odakle će čitati ovu domenu.

*Slika 64: Kreiranje fajla probni-domen.conf i njegov sadržaj*



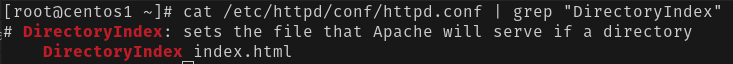
### 10.1.2 Kako ovo funkcioniše

*Slika 65: Prikaz DocRoot u httpd.conf*



Razlog zbog kojeg može da se prebaci fajl u direktorijum /var/www/html i da ga prikažem u pretraživaču je podešavanje DocumentRoot unutar apache konfiguracije. DocumentRoot se isčitava iz /var/www/html, a razlog zašto apache isčitava index.html je sledeći:

*Slika 66: Prikaz indeksiranja fajlova koji se prvi čita*



Ova linija obrađuje koji fajl će biti učitan kada bude zatražen direktorijum. Bez obzira što imam definisane varijable u /etc/httpd/conf/httpd.conf, mogu da dodam konfiguraciju za veb sajtove pod direktorijumom, što se može videti u primeru sa virtual hostom.

## 9.2 MySQL, Maria DB baze podataka

Tradicionalno baze podataka su odlično mesto za skladištenje uređenih podataka specifičnog tipa i veličine, implikacija je da možemo imati bazu podataka za bilo šta. Pošto su uglavnom poznatije relacione baze koristićemo MySQL baze, no ne baš MySQL pošto je to orginalno ime koje sada pripada Oracle firmi već MariaDB što je grana ove baze i dostupna je svim koji žele da je koriste zbog Open-Source licence ove baze su generalno poznate i široj javnosti iz razloga zato što jedan od većih interfejsa kao što je WordPress koristi ovu bazu. Dobra praksa sa softverom je da se instalira pre nego što se počne upotrebljivati, instalacije se vrši na sledeći način:

*Slika 67: instalacija mariadb servera preko yum*



Nakon toga moram aktivirati servis, slično kao i sa apache:

### 9.2.1 Podešavanje secure instalacije

Pošto instalacija mysql sama po sebi ne podešava nikakav password ili root korisnika sama po sebi, uglavnom je preporučljivo da se pokrene skripta koja vrši osnovno podešavanje

mysql\_secure\_installation

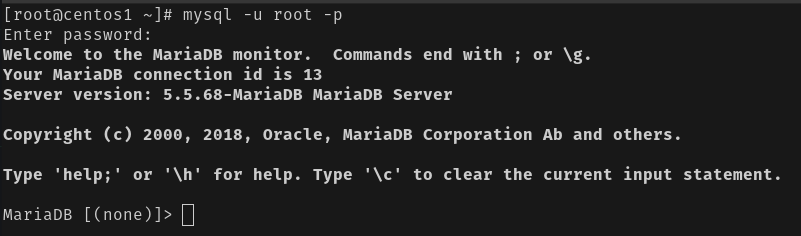
Ova skripta će pitati niz pitanja koja su:

* Enter current password for root: ovo možemo ostaviti prazno pošto nema password
* Set root password: Y
* New password: nekipass
* Remove Anonymous users: Y
* Dissallow root login remotely: Y
* Remove test database and access to it: Y
* Reload privilege tables now: Y

(Uglavnom su svi zahtevi razumljivi šta radi kada damo yes opciju)

Nakon osnovnog podešavanje trebalo bi mi biti omogućeno da pristupim bazi:

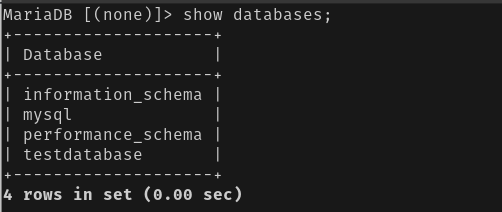
*Slika 68: Login na mysql servis*



### 9.2.2 Listanje kreiranje i selektovanje podataka i tabela

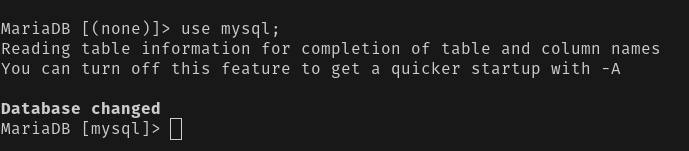
Bazom može da se upravlja iz terminala, koji se otvara pokretanjem iz serverskog shella. Osnovna komanda bi bila da proverim koje baze imam aktivne na serveru.

*Slika 69: MySQL command shell*

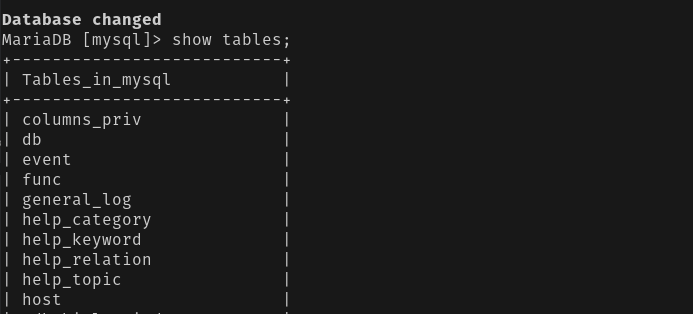


Recimo da želim da selektujemo neku od ponuđenih baza mogu koristiti sledeće;

*Slika 70: Prikaz use komande u mysql*



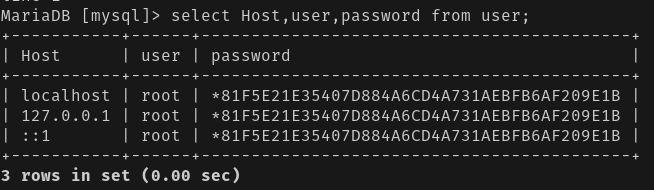
*Slika 71: Promena baze podataka*



Može da se primeti sve ovo može da se radi iz terminala bez ikakvog pristupa nekom eksternom servisu kao što je recimo phpmyadmin koji može da se podesi na server.

Neke česte komanda koje se koriste u mysql bazama su takođe insert, select, create database drop.

*Slika 72: Selekcija vrednosti iz tabele user*



Ovde sam selektovao Host, user, password iz tabele user da mi prikaže informacije sadržane ovde. Kreiranje baze podataka:

*Slika 73: Upotreba create database komande*



Brisanje baze podataka

*Slika 74: Upotreba drop database komande*



iz komandi iznad uočava se njihova upotreba koju bi sistemski administrator koristio prilikom kreacije baze (create) ili brisanja baze (drop) isto tako postoje i komande za kreiranje i unošenje u table sa kojima često radi sistemski administrator:

*Slika 75: Upotreba create komande za tabele*



*Slika 76: Upotreba insert komande za unos vrednosti u tabelu*

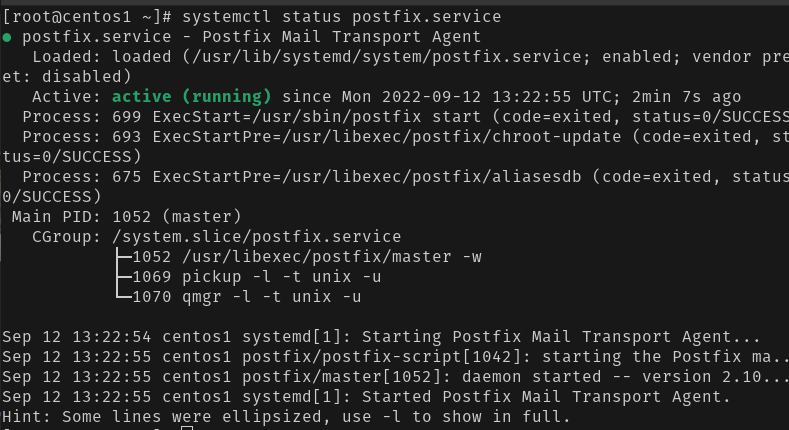


Takođe postoje i dozvole za baze podataka, koje su takođe važne koliko i regularne dozvole fajl sistema. Na primer ne želim da dve wordpress instalacije na istom hostu mogu da čitaju bazu podataka međusobno, pa se iz tog razloga keriraju posebni korisničkni nalozi za svaku od njih, i dodeljuju im se posebne baze podataka unutar MariaDB. Na zvaničnoj dokumentaciji imaju sve moguće implementacije ovih dozvola za korisnika: https://dev.mysql.com/doc/refman/8.0/en/privileges-provided.html

## 9.3 Upotreba MTA (Mail transfer agent)

E pošta je možda nešto najučestalije u poslu sistemskog administratora, iako sam email sistem je već jako zastarela tehnologije još uvek se koristi u velikoj meri. Česta ili ti tradicionalna upotreba email je da sistem-os vrši neke provere dokumentuje ih i šalje dokumentaciju administratoru ukoliko nešto krene po zlu ili neke osnovne informacije o zdravlju servera. Neki poznati email agenti ili ti MTA su EXiM i PostFiX gde je postfix dosta stariji pa na nekim serverima se pribegava ovome, dok je EXiM noviji i ima veću mogućnost konfiguracije, upotreba ovog MTA je učestala kod Cpanel hostovanih servera. Jedna zanimljiva stvar dovoljno nam je samo da imamo aktivan MTA kako bi slali mail nekome, rekordi uopšte nemoraju biti postavljen sem osnovnog što stoji za IP servera ili ti A rekord. Servis koji je zadužen za slanje mail:

*Slika 77: Provera status postfix servisa*



Default mta na sistemu:

*Slika 78: Provera fabričkog agenta za mail na sistemu*

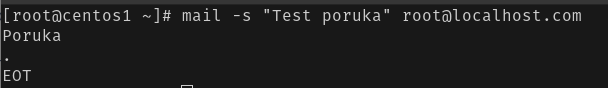


instalirao sam malu aplikaciju na serveru koja će mi omogućiti slanje sa servera na neku e poštu

*Slika 79: instalacija mailx aplikacije na server*



*Slika 80: Slanje test poruke*



Primer pokazuje slanje poruke na neki mail, ali ovo neće proći iz razloga zato što nije konfigurisano. No bitna stavka svakog administratora je, da imamo logove koje možemo proveriti zašto nešto nije prošlo recimo. Servis koji je zadužen za slanje mail:

*Slika 81: Greška prilikom slanja*



ne postoji konekcija ka hostu to je problem.

Konfiguracioni fajl za ovaj MTA se nalazi u etc/postfix/main.cf dok aliasi ili ti mapa naloga koja će se koristiti za mapiranje korisnika je u etc/aliasis. Kratka predstava kako ovaj agent radi na realnom sistemu, no u normalnim hosting okruženjima ovo je već sve podešeno za upotrebu. Za bilo šta uvek je dobro osvrnuti se na zvaničnu dokumentaciju. <https://www.postfix.org/documentation.html>

# 10. Pravljenje osnovne konfiguracije neke domene upotrebom wordpress

## 10.1 Uvod

Mali uvod oko WordPress, možda i najpopularniji CMS (Content Management System) za domene, vrlo često viđeno i na zvaničnim hosting komapanijama kao osnovna instalacija za server.

izabrao sam ovo pošto cela instalacija se može obaviti veoma brzo uz prethodno instalirane servise kao što je mysql (Maria DB) i httpd koje sam prelazio u prethodnim poglavljima.

Pošto je WordPress baziran na PHP neophodno je obezbediti odgovarajuće biblioteke a i sam PHP koji bi trebao da bude oko verzije 7.4 koja je relativno nova ali ima još par iznad ove. Na zvaničnom repozitorijumu centos u yum, 5.6 php verzija je dostupno iz tog razloga moraćemo da dobavimo određeni repozitorijum koji sadrži novije verzije PHP instalacija.

## 10.2 instalacija Wordpress aplikacije

Prvo moram instalirati pakete za remi repozitorijume

*Slika 82: nstalacije epel-release paketa*



Dodajem repozitorijum u listu za yum kako bi mogao da ih koristim.

*Slika 83: Dodavanje repozitorijuma za instalaciju*



instaliram yum alatke koje će mi omogućiti da podešavam verzije PHP

*Slika 84: instalacija dodatnih yum alatki*



Stavljamo php 7.4 kao fabričku verziju koristeći novo instalirane alatke iz paketa yum-utils

*Slika 85: Uključivanje remi-php74 iz repozitorijuma*



U suštini rekao sam yum da skida 7.4 verziju sa repozitorijuma koji smo specificirali prethodno, tu nam je došla korisna alatka yum utils.

instlairamo dodatne biblioteke za php koje će mi biti korisne za WordPress kasnije

*Slika 86: instalacija PHP paketa i biblioteka*



Nakon instaliranih paketa i biblioteka server bi trebao da pokreće PHP verziju koja mu je bila specificirana.

*Slika 87: Provera verzije PHP*



(Takođe nije loše ponovo poktenuti httpd nakon ovih izmena)

Pošto je wordpress instalacija dostupna svima sa zvaničnog sajta počeću odatle.

Pre svega treba dobaviti paket, ali u CLi (command line interface) nemamo ni browser, a ni pristup internetu sem ako ne napravimo to nekako, no postoji mnoštvo komandi koje će mi omogućiti isto.

Naravno moram imati link odakle ću preuzeti neophodne sadržaje

https://wordpress.org/latest.zip

Komanda wget je jedna od poznatjih, ali imamo i curl koji dolazi kao default i možemo njega da koristimo.

Skinućemo paket wordpress u direktorijum /var/www/html/ pošto je to mesto odakle apache isčitava podatke.

*Slika 88: Preuzimanje wordpress fajla sa interneta upotrebom wget*



Trebao bi da dobijem ovakav neki podatak:

*Slika 89: Prikaz preuzetog fajla*



Upotrebom bilo tar komande ili unzip mogu izvući podatke iz ovog fajla.

Kada to obavim dobijamo direktorijum wordpress:

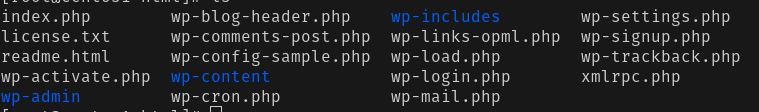
*Slika 90: Prikaz otpakovanog fajla*



Pošto Apache isčitava podatke u zavisnosti kako mu je podešen docroot, ili odakle će čitati konfiguracije, nephodno je fajlove iz wordpress kopirati na root nivo. (Virtual host odeljak pokazuje kako bi napravio drugačiju konfiguraciju)

Kada izvučem sve fajlove trebao bi dobiti ovako nešto:

*Slika 91: Prikaz otpakovanog fajla*



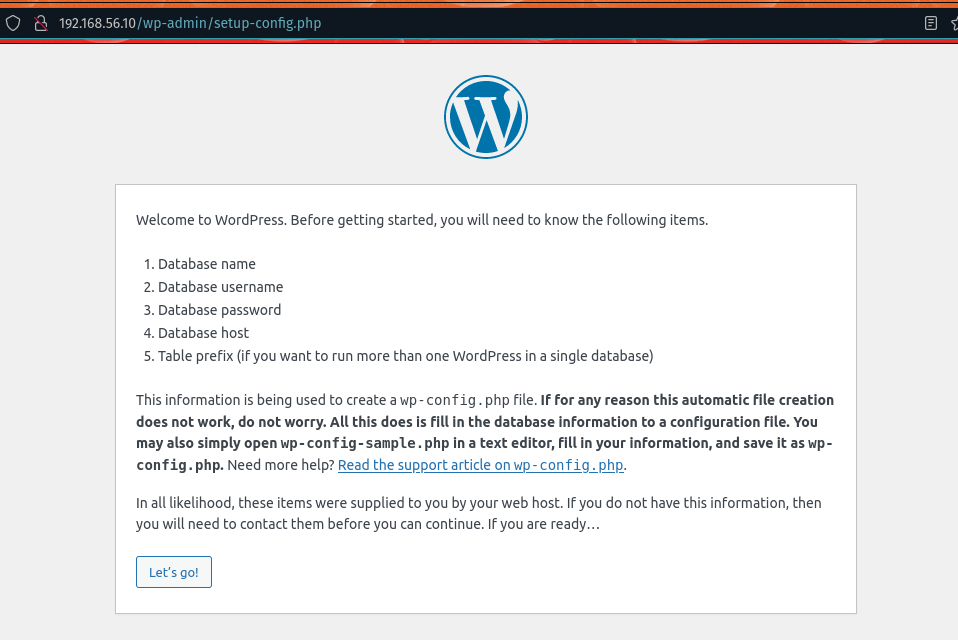
Naravno ovo izgleda ovako sada, ali koristio sam Linux komande kako bi pomerao fajlove recimo:

*Slika 92: Brisanje wordpress direktorijuma*



Nakon ovih podešavanja pozivom određenog Wordpress fajla trebali bi dobiti sledeću stranicu:

*Slika 93: Prozor instalacije WordPress*

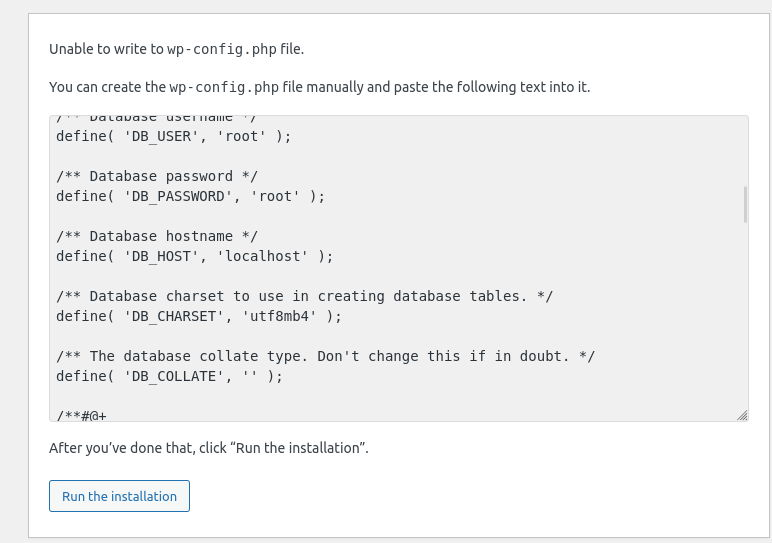


instalacija je prilično jasna i jednostavna tako da ću preskočiti detaljisanje o svakom koraku. Morao sam pristupiti fajlu setup-config.php koji se nalazi u direktorijumu wp-admin kako bi vršio ovu instalaciju. Moraću i da napravim konekciju ka bazi da bi ovo uopšte funkcionisalo u wordpress postoji wp-config.php koji će mi pomoći u ovome, ali i pre toga moraću kreirati bazu na serveru. Kreirao sam bazu wordpress koju ćemo koristiti za instalaciju:

*Slika 94: Prikaz kreirane baze za wordpress*



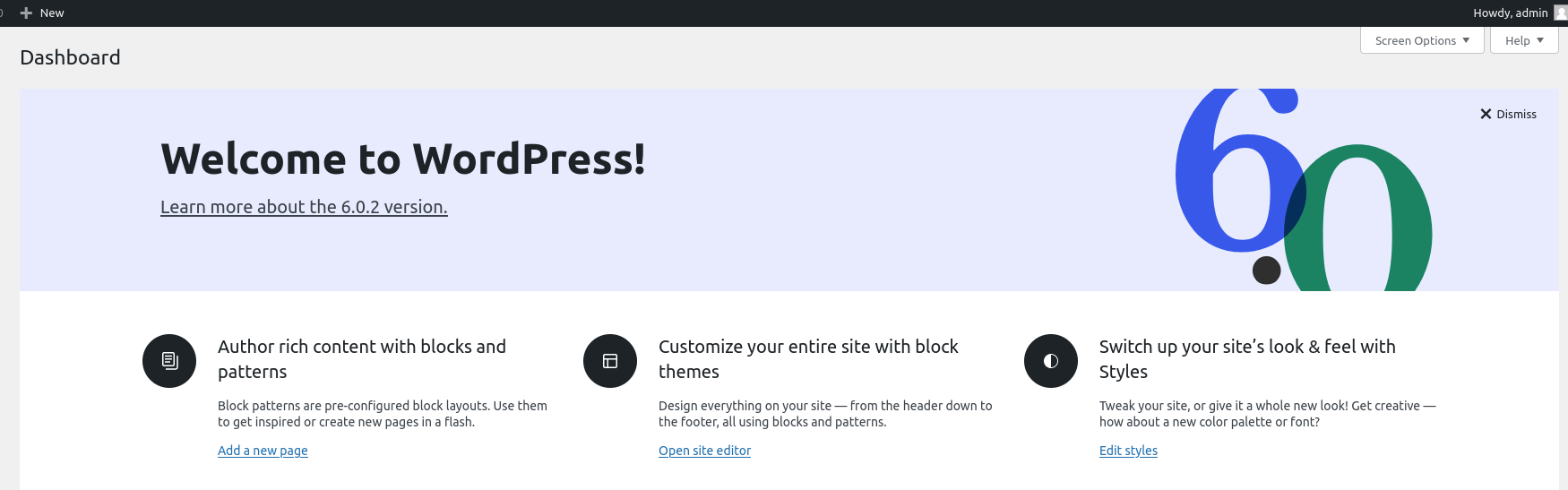
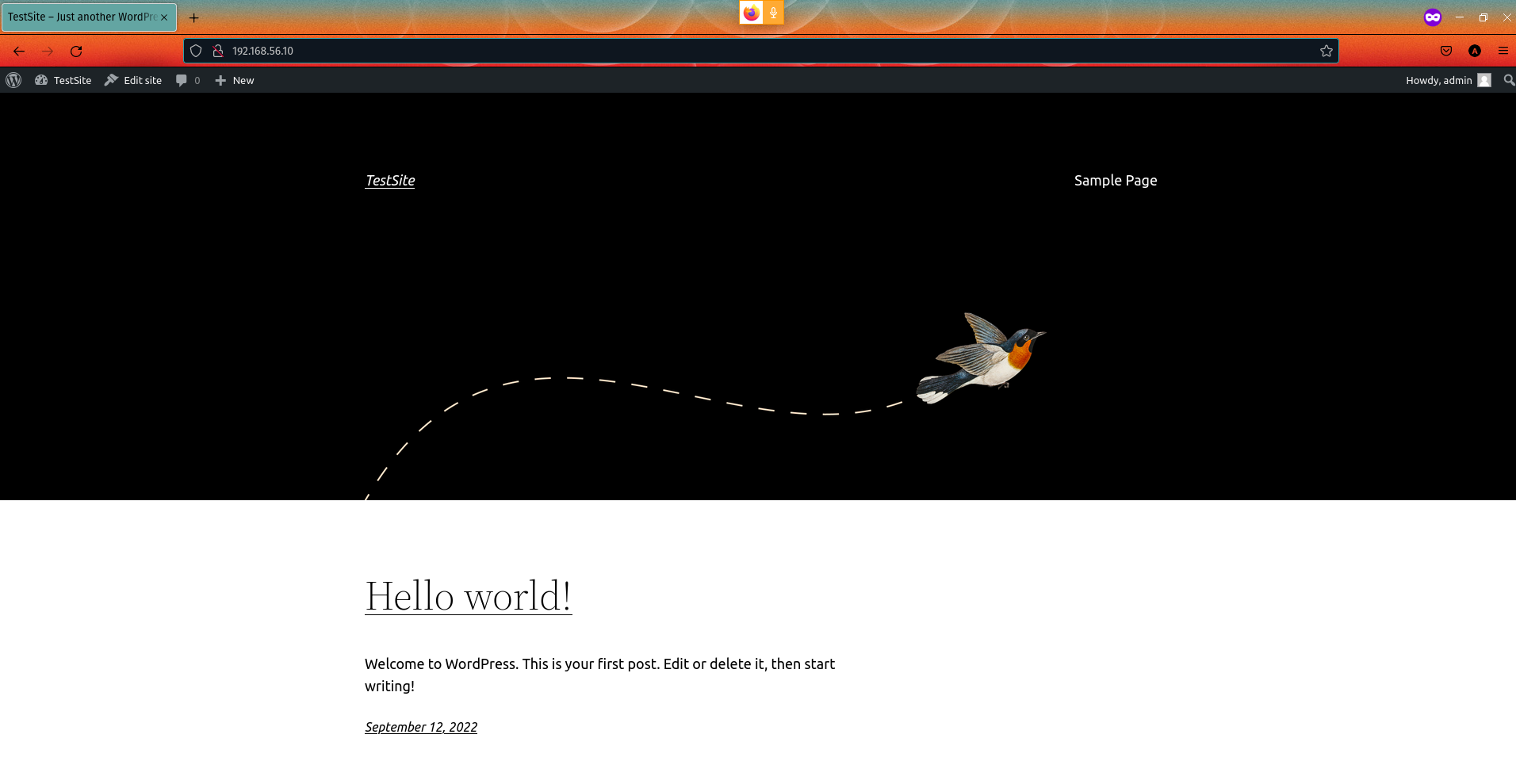
*Slika 95: WordPress konfiguracioni fajl*



To je kada prođem kroz upitnik koji mi daje wordpress prilikom instalacije.

Nakon popunjavanja wp-config.php i pokretanje instalacije dobijam još jedan upitnik sa kojim finaliziramo instalaciju. (preskočiću ga ovde pošto je samo forma koju korisnik popunjava) Nakon uspešne instalacije i kada unesemo zadate informacije koje nam WordPress traži dolazim do sledeće stranice sa kojom je završena instalacija:

*Slika 96: Admin stranica i Početna stranica WordPress nakon isntalacije*

Wordpress nam nudi mnoštvo mogućnosti, ali ima i svoje mane naravno, uglavnom u programerskom svetu ukoliko je klijent u mogućnosti pribegava se custom sajtovima što hakerima otežava da kompromituju domenu, Wordpress sam po sebi je siguran ali može da se poremeti malicioznim injection code napadima češće nego neki custom kod napisan za istu svrhu.

# 11. Shell skripte

## 11.1 Uvod

Shell skripte su niz komandi koje se izvršavaju u cilju da se automatizuju neki zadaci na serveru kako ne bi administrator morao da piše ručno.

Same po sebi skripte su jednostavne kao i način na koji se pišu, dosta se pozajmljuje iz C jezika na kojem je i samo Linux jezgro(kernel) napisano.

Bazično razmišlanje iza ovih skripti jeste da, ako znamo da manipulišemo stvarima, možemo pisati skripte.

## 11.2 Početak pisanja skripti

Svaka skripta počinje sa “shebang” i putanjom do shella koji želimo da skripta koristi, a to izgleda ovako:

#!/bin/bash

# Početak neke skripte

Ova kombinacija simbola #! govori našem interpreteru kako da izvršava skriptu ovo može biti i recimo #!/bin/python, ako želimo da se skripta izvršava kao Python kod. Ako se pogleda na drugu liniju možemo videti sličan početni simbol # stim što se ovo ne čita kao početak za interpreter već kao komentar u samom fajlu.

Recimo da želim da napravim skriptu za pretragu fajlova većih od 1M home/alex/Videos direktojuma find komandu da ne bi morao da stalno pišemo punu komandu, to bi izgledalo ovako:

**#!/bin/bash**

**# Find skripta**

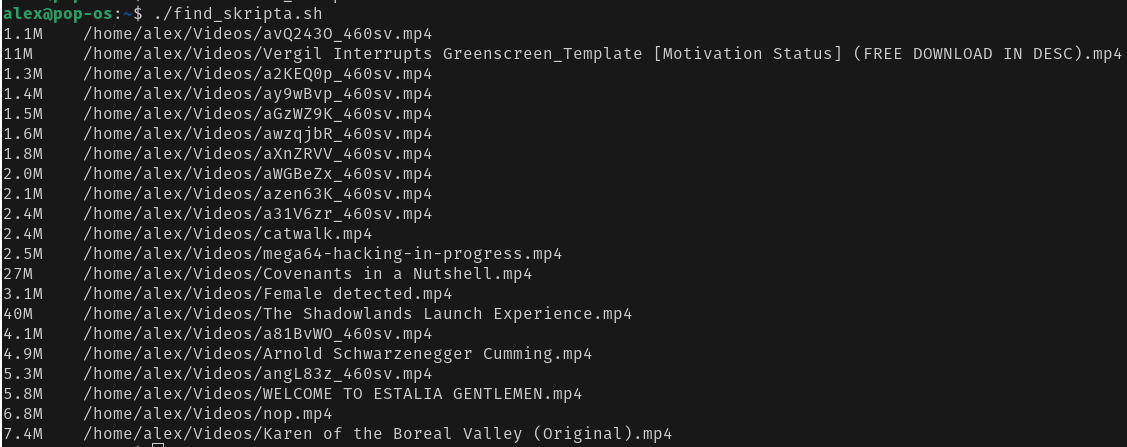
**find /home/alex/Videos -maxdepth 1 -type f -size +1M -exec du -h {} + | sort**

Nakon napisane skripte nephodno je dati određene permisije koje omogućavaju korisniku da pokreće skriptu, treba dati executable permisiju.

To se može izvršiti na sledeći način **chmod + x find\_skripta.sh**, ovo će omogućiti svim korisnicima da pokreću ovu skriptu koji su trenutno registrovani na operativnoj mašini.

Kada izvršim skriptu barem na mojoj mašini dobijam sledeći rezultat

Slika 97: Rezultat izvršavanja find\_skripta.sh



Skripte se mogu pokretati bilo navigiranjem do direktorijuma gde se nalazi i pisanjem sledećeg unosa

./find\_skripta.sh

ili

Mogu dati pun path (full path) do skripte

/home/alex/find\_skripta.sh

Rezultat je isti u oba slučaja

## 11.3 Varijable

U realnom slučaju je skripta je korisna, ali neću da sve bude čvrsto postavljeno, recimo da hoću da menjam vrednosti pretrage odakle počinje.

U ovim slučajevima korisne su mi varijable koje će mi omogućiti ovu slobodu, pisanje varijabli u shell skriptu imaju svoju konvenciju, a to je da imena moraju biti velikim slovima IMEVARIJABLE poželjno je da to radimo, ako želimo da delimo našu skriput sa nekim, u suprotnom nije potreba.

Forma pisanje varijable je sledeća

MESTO\_PRETRAGE=”unosimo nešto šta želimo”

Prikazaću ovo i na primeru koristiću prethodno napravljenu skriptu da demonstriram upotrebu varijabli:

#!/bin/bash

MESTO\_PRETRAGE="/home/alex/Videos"

find "${MESTO\_PRETRAGE}" -maxdepth 1 -type f -size +1M -exec du -h {} + | sort

Varijabla sadrži putanju koju find treba da koristi, a unutar same find komande pozvao sam ovu varijablu kako bi se primenila na mestu gde se zahteva putanja pretrage sa ${iME\_VARiJABLE}, vitičaste zagrade su dobra konvencija ako želim da razdvojimo dve varijable recimo da nam se putanja sastoji iz dva dela

“${MESTO\_PRETRAGE}${MESTO\_PRETRAGE\_NiZE}”

dupli navodnici impliciraju da treba da se čitaju varijable ako ih ima, tako da ne bi trebali da ih mešamo sa jednostrukim navodnicima gde govorimo skripti da isčitava sve kako piše, dobiću ${MESTO\_PRETRAGE} kao izlaz naše komande, ako je ištampamo u bash, ovo je korisno ako samo želim da napišem neki string.

## 11.4 Uzimanje unosa iz terminala

Skripte koje nemaju interakciju sa korisnikom su korisne, ali šta ako mi trebaju neke informacije svaki put kada se skripta pokreće? Ovde su mi korisne da uzimam argumente sa terminala prilikom pokretanja skripte ili da koristim read komandu.

želim da napišem skriptu koja će mi samo oštampati moje ime

#!/bin/bash

echo “Pozdrav ${1} !”

Naravno moramo dati potrebne permisije ovom fajlu i to bi trebalo da izgleda ovako:

*Slika 98: Prikaz upotrebe argumenta*



Skripta je uzela prvi argument $1 koji joj je prosleđenm, i oštampala ga je kako sam napisali komandu ispisa.

Kao što je prethodno pomenuto mogu naterati skriptu samim pokretanjem da traži moj unos kako bi obavila neki zadatak, a to može da se uradi na sledeći način

Koristiću prethodno napisanu skriptu sa find komandom kako bi predstavio upotrebu

#!/bin/bash

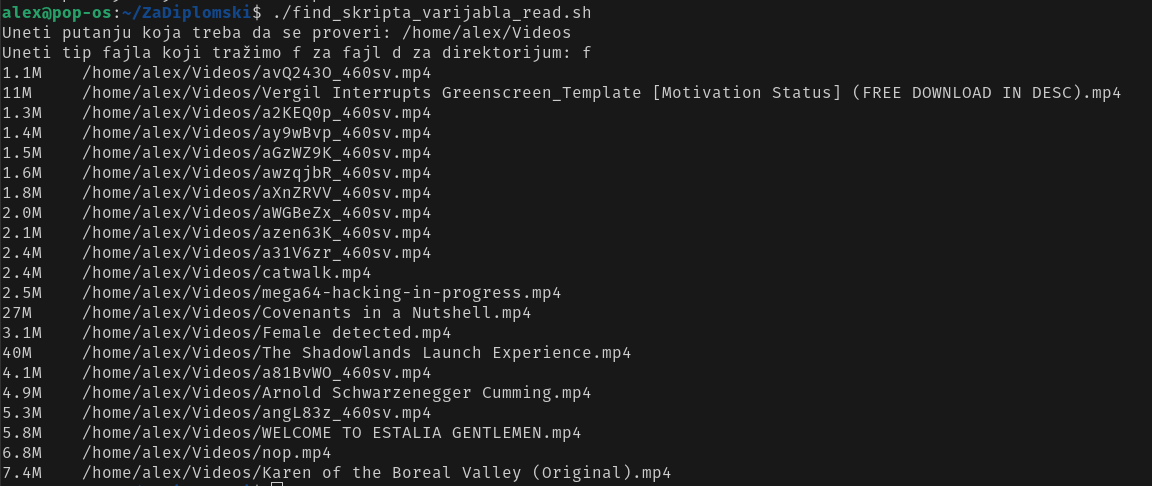
read -p "Uneti putanju koja treba da se proveri: " MESTO\_PRETRAGE

read -p "Uneti tip fajla koji tražimo f za fajl d za direktorijum: " TiP

find "${MESTO\_PRETRAGE}" -maxdepth 1 -type "${TiP}" -size +1M -exec du -h {} + | sort

Rezultat će biti sledeći:

*Slika 99: Rezulat upotrebe find skripte sa varijablom*



Dobija isti rezultat, ali sa uptrebom prompta koji od mene traži da unesem nešto kako bi se izvršila skripta

Ovde sam pretstavio samo neke osnove, shell skripte idu mnogo dublje od ovoga, mogu da se koriste funkcije, if odluke, for petlje, while petlje … itd.

Korisno je naći neku knjigu ili se okrenuti nekim generalnim sajtovima koji se bave obučavanjem pisanja programa.

Recimo dobra stranica koju sam često posećivao prilikom pisanja ovog dela je:

https://www.tutorialspoint.com/unix/shell\_scripting.htm

## 11.5 Regularni izrazi

Često u radu bilo sa programskim jezicima ili Linux sistemima naići ćemo na pojam regularni izrazi. Formalna definicija za njih bi bila sekvenca karaktera koji se intpretiraju kao neka pretraga ili ti obrada koja treba da se izvrši na fajlu.

Ovi izrazi imaju značajnu ulogu i prilikom pisanja skripti koje vrše neku automatizaciju pošto ne može baš sve da se izvrši upotrebom jedne komande već ponekad moramo da i podešavamo izlaz kako bi se pravlino izvršilo na komandi.

U narednoj tablici ću predstaviti neke meta karaktere i kako oni utiču na izlaz grep komande:

*Tabela 1: Specijalni karakteri regex*

|  |  |  |
| --- | --- | --- |
| BR | Komanda upotrebe | Rezultat |
| 1. | **grep user fajl.txt** | Grep će pretražiti pojam user u fajlu fajl.txt i daće nam sve pronađene linije koje sadrže ovaj pojam |
| 2. | **grep user \*.txt** | Grep će tražiti pojam user u fajlovima koji se završavaju sa .txt često se ovaj meta karakter naziva wildcard |
| 3. | **grep '^A' fajl.txt** | Grep će tražiti sve linije koje počinju sa slovom A |
| 4. | **grep '2$' fajl.txt** | Grep će prikazati sve linje koje se završavaju sa brojem 2 |
| 5. | **grep '1..' fajl.txt** | Grep će prikazati linije koje sadrže 1 i bilo koja 2 karaktera iza njega uračunavajući space koji se tretira kao karakter |
| 6. | **grep '.2' fajl.txt** | Grep će prikazati linje koje koje sadrže 2 i bilo koji karakter ispred njega uračunavajući space |
| 7. | **grep '^[Ai]' fajl.txt** | Grep će prikazati sve linije koje počinju sa A ili i (velikim slovima samo) |
| 8. | **grep '[^0-9]' fajl.txt** | Grep traži linije koje sadrže najmanje jedan alfanumerički karakter |
| 9. | **grep '[A-Z][A-Z] [A-Z]' fajl.txt** | Grep traži reč koja sadrži veliko slovo, pa opet veliko slovo, razmak i ponovo veliko slovo. |
| 10. | **grep '[a-z]{8}' fajl.txt** | Grep traži reč koja sadrži malo slovo ali kao 8 uzastopnih malih slova |

Ovde sam naveo samo neke meta karaktere ima još mnoštvo načina kako može da se pretražuje tekst sa grep ili ti da se obrađuje pretraga sa meta karakterima.

Dobra stranica za podešavanje meta karaktera za neku pretragu je:

<https://regex101.com>

## 11.6 Sed

Sed ili neiteraktivni stream editor, popularna je alatka kod sistemskih administratora, pa sam hteo da izdvojim neke osnove u ovom odeljku.

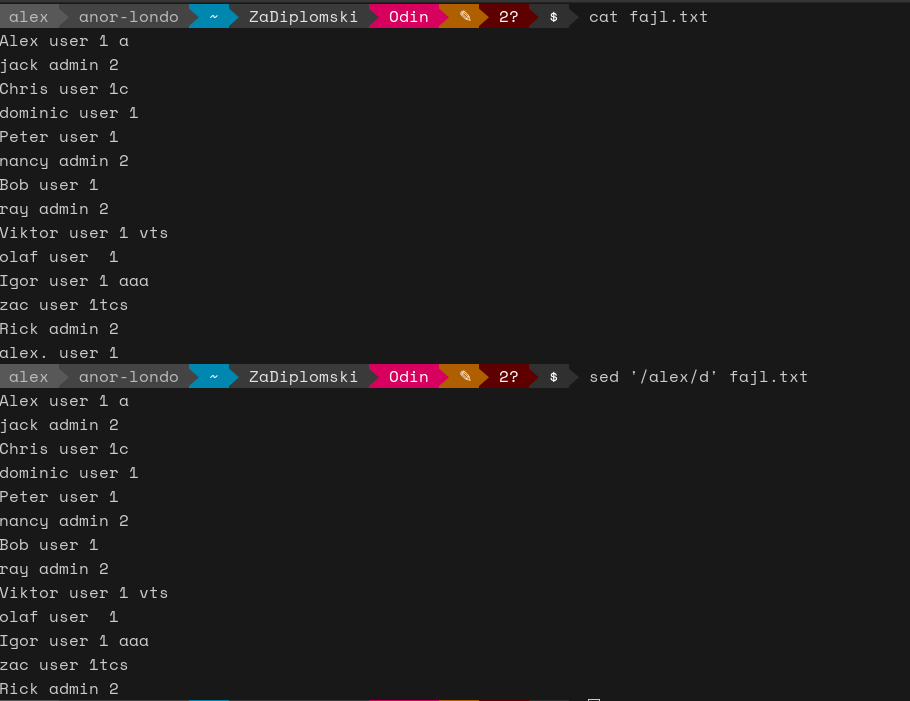
Osnovna upotreba jeste da mogu da manipulišem tekstom iz komande linije ili ti shell, tj upravljamo streamovima. Sed sve promene koje unesemo prikazuje na ekranu gde je pokrenut. Čuvanje novog modifikovanog fajla mogu obaviti ili usmeravanjem u novi fajl ili da prepravim već postojeći fajl.

Pošto sed ima mnoštvo komandi koje mogu da se upotrebljuju prilikom rada sa njim ja ću izvojiti ovde neke osnovne koje se vrlo često koriste.

Recimo imam fajl fajl.txt u njemu se nalaze imena korisnika i administratora, za sada želim samo da obrišem pojave imena alex, upotrebom komande

$ sed ‘/alex/d’ fajl.txt

*Slika 100: Brisanje imena iz fajla na standardnom izlazu upotrebom sed komande*



Može da se primeti da prethodno postoji korisnik alex. User 1 kada sam ga prikazao sa komandom cat nakon upotrebe sed ova linija više ne postoji.

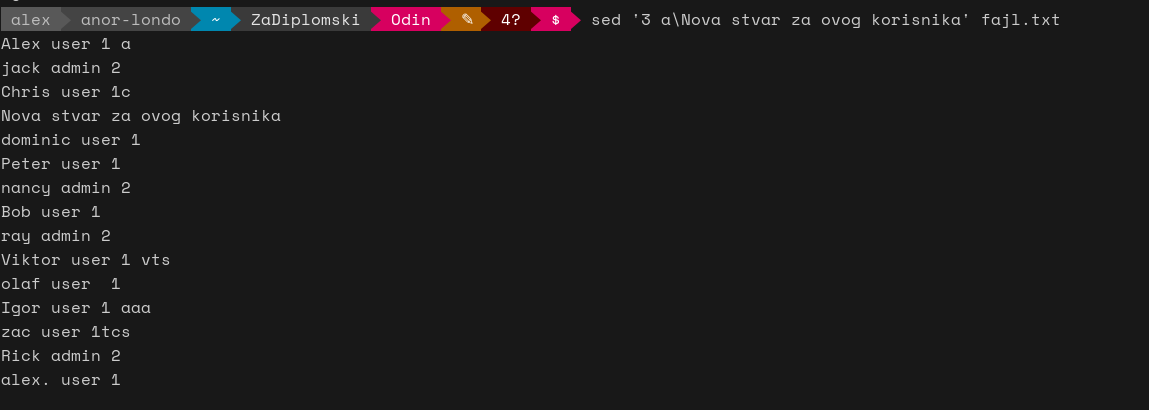
Naravno fajl je ostao ne izmenjen iz razloga što nisam sed komandi rekao to da uradi, tj u suštini smo oblikovali standardni izlaz po našoj želji.

Još jedna česta upotreba jeste da dodajemo nešto u fajl, ovo se može izvršiti upotrebom append oznake u sed komandi.

Koristimo sledeću komandu za unos u fajl tj standardni izlaz:

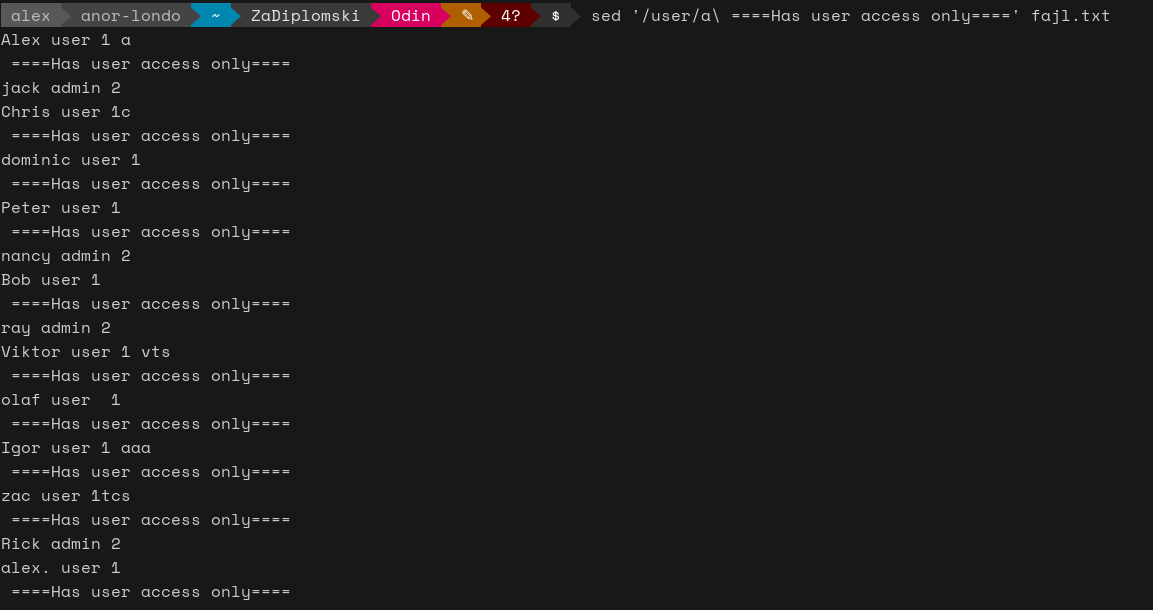
$ sed ‘3 a\ komentar za korisnika’ fajl.txt

*Slika 101: Upotreba append u sed*



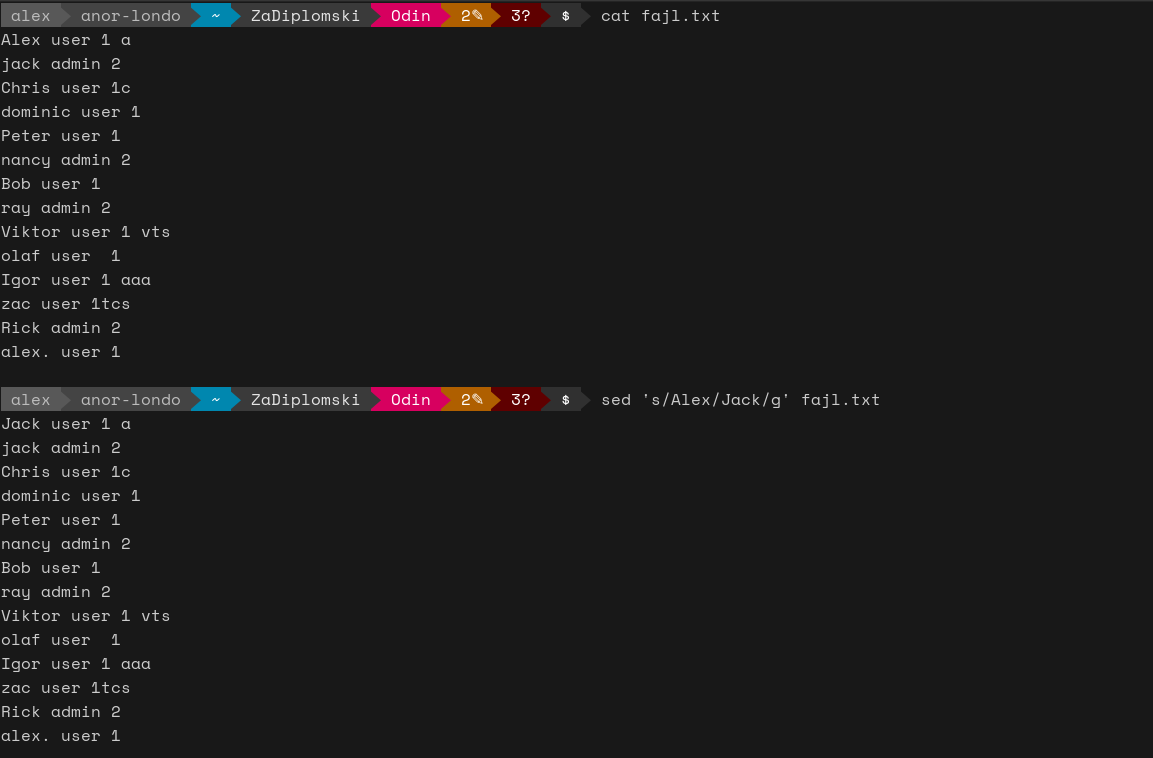
U ovom primeru sam rekao sed da na trećoj liniji doda liniju sa tekstom “Nova stvar za ovog korisnika, isto tako mogu staviti za neku reč koju tražimo.

*Slika 102: Upotreba append parametra u sed sa pretragom reči*



Upotrebom komande sed sa pretragom ključne reči sed dodao “====Has user access only====” ispod svake linije koja sadrži user kao ključ pretrage, dodatno na ovo postoji i insert koji samo dodaje na specifičnu liniju njegov flag u sed je i. Sed može da radi i sa meta karekterima kao i grep koji sam pomenuo u prethodnom odeljku Regularni izrazi, što je dobro izdvojiti kada razrešavamo stvari u nekom tekstu. Sistemski administratori često upotrebljuju sed kada žele da zamene neku liniju sa drugačijim unosom. $ sed ‘s/Alex/Jack/g’ fajl.txt

*Slika 103: Upotreba zamene u sed*



Alex korisnik u listi zamenjen sa Jack kada sam specificirao u sed. Naravno kao i sve do sada ovo je samo na standardnom izlazu fajl.txt ostaje ne promenjen. Ovo su samo neke od mogućnosti sed, pošto ova alatka može da se koristi i za skriptovanje njene mogućnosti su mnogo veće nego ovde navedene, koje samo predstavljaju mali deo onoga što može.

## 11.7 AWK

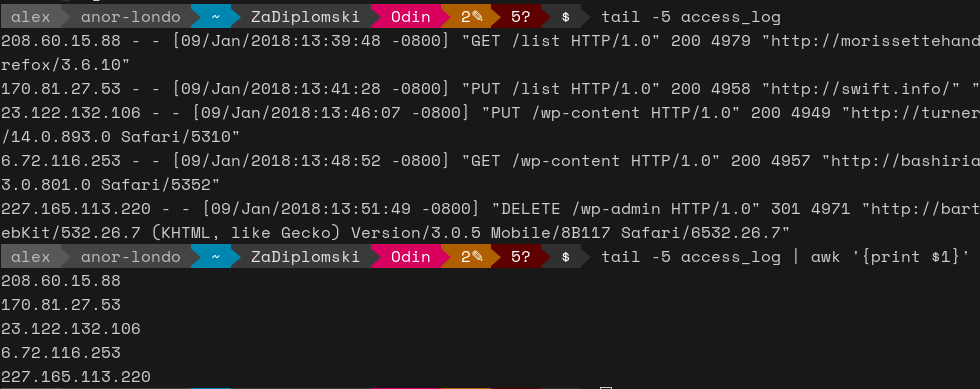
AWK je sopstveni programski jezik koje može da se izvršava u terminalu ili u skripti, prevalentno se koristi za obradu podataka i generisanje izveštaja.

Pošto je awk programski jezik ja ću se dotaći samo nekih upotreba, pošto osnovna upotreba ne zahteva neko veliko programersko znanje.

Osnovna upotreba awk bi se svela na obradu teksta

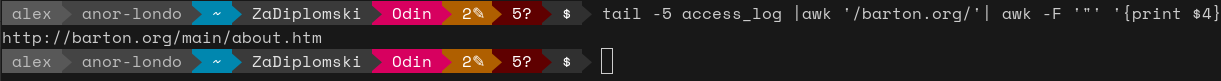
Na primer želim samo da ištampam polje sa IP adresama u nekom access logu (log za beleženje posete i pristupa stranicama)

*Slika 104: Upotreba AWK komande*



Sa slike možemo uočiti da sam oštampao samo prvo polje ili ti $1 koje sam mu zadao uptrebom print komande unutar awk. Možemo i da tražimo neku ključnu reč ako želimo takođe.

*Slika 105: Pretraga ključne reči sa AWK*



Koristio sam | (pipe) kako bi preusmerio izlaz i modifikovao kako će izgledati na standardnom izlazu, dobio sam samo stranicu koju sam tražio iz fajla. U ovome leži moć awk komande, u suštini mogu i mnogo kompleksnije stvari da se radi na ovome, ali ovo bi bilo dovoljno znanja za nekog ko želi samo da prepravi izlaz komande kako njemu odgovara.

# 12. Firewall i nadgledanje sistema

## 12.1 Uvod

Česta stvar na koju nailazimo na nekom aktivnom serveru je firewall ili ti zaštitna barijera koja reguliše ulazni i izlazni saobraćaj na našem serveru. Firewall software sa kojim ću ja rukovati u ovom primeru je IP tables, imamo još i firewallcmd i ufw(Debian), ali to bi proširilo previše ovu oblast pošto je svrha da predstavi kako rade firewall konfiguracije, a uglvanom isto rade stim da imaju malo drugačiju sintaksu. Kroz primere prikazaću kako to funkcioniše i kako može admin time da upravlja.

## 12.2 iptables

Za početak mogu da proverimo da li je servis aktivan tako što ću koristi komandu iptables -L koja će mi dati informacije o aktivnim pravilima.

*Slika 106: Prikaz pravila u iptables*



Prema podrazumevanom podešavanju opcija -L lista sve lance unutar podrazumevane tabele. Postoji pet tabela koje IP tables prikazuje:

• raw

• filter

• mangle

• security

•nat

U zavisnosti šta želim mogu dati opciju koju želimo da vidim, to se može dobiti specificiranjem opcije u iptables

$ sudo iptables -t nat -L

imam i alternativni način prikaza pravila i kako se pišu što se tiče iptables

$ sudo iptables -S

*Slika 107: Drugačiji prikaz pravila u iptables*



Dobijamo nešto ovako, kao što se vidi ovaj način je daleko intutivniji pošto mogu da vidim kako upisivati neka pravila.

### 12.2.1 iptables dodavanje i uklanjanje pravila

Kako bi rukovao sa firewall moramo biti upoznati sa njegovim funkcionalostima i kako da ga koristimo da pravimo određena pravila koja će mi omogućiti veću sigurnost na serveru.

Pošto nije preporučljivo da firewalld i iptables budu uključeni u mom test okruženju mogu ga iključiti privremeno kako bi omogućio sebi veću slobodu manipulacije sa iptables, a i da me ne ometa firewalld koji ima veći prioritet na CentOS.

$ sudo systemctl disable --now firewalld.service isključivanje firewalld

isključivanjem firwalld dobijam praznu listu pravila sa kojom mogu da radim

$ sudo iptables -S

-P iNPUT ACCEPT

-P FORWARD ACCEPT

-P OUTPUT ACCEPT

Pošto ja u mreži imam dve centos mašine za početak ću sprečiti konekciju ka centos1 sa centos2 mašine kako bi predstavio ovu funkcionalnos. (blokiranje ssh konekcije specifično)

$ sudo iptables -A iNPUT -i eth1 -p tcp -m tcp --dport 22 -j DROP

Sledeće što ću uraditi jeste da omogućim samo ulazne konekcije na isto kako ne bi sebe blokirali potpuno tj da ne moramo direktno ići na server kako bi uspostavili konekciju.

$ sudo iptables -A iNPUT -s 10.0.2.0/24 -p tcp -m tcp --dport 22 -j ACCEPT

Za kraju ću promeniti podrazumevano pravilo ulaza sa ACCEPT na DROP:

$ sudo iptables -P iNPUT DROP

Pošto sam promenio podrazumevano pravilo, takođe treba da se uverim da su konekcije RELATED i ESTABLiSHED trajne (konekcije koje smo pokrenuli iz mašine ), zaštitna barijera bi trebla da sadrži informacije o stanju.

$ sudo iptables -A iNPUT -m conntrack --ctstate RELATED,ESTABLiSHED -j ACCEPT

Na kraju mogu videti kako to izgleda u tabeli:

$ sudo iptables -S

-P iNPUT ACCEPT

-P FORWARD ACCEPT

-P OUTPUT ACCEPT

-A iNPUT -i eth1 -p tcp -m tcp --dport 22 -j DROP

-A iNPUT -s 10.0.2.0/24 -p tcp -m tcp --dport 22 -j ACCEPT

-A iNPUT -m conntrack --ctstate RELATED,ESTABLiSHED -j ACCEPT

Malo su ovde pravlia suvišna, ali bila je ideja da se predstavi fleksibilnost iPTABLES.

Pošto je svrha razumevanje pokušaću da razbijem komandu i da objasnim deo po deo.

$ sudo iptables -A iNPUT -s 10.0.2.0/24 -p tcp -m tcp --dport 22 -j ACCEPT

Prvo pokrećem alatku iptables koja mi daje pristup ovim opcijama

Zatim imam -A iNPUT pravilo znači Append (dodaj nemoj da pregaziš prethodni rule) na iNPUT lanac.

imamo -s 10.0.2.0/24 ovo znači da je izvor saobraćaja IP adresa koja je specificirana.

-p tcp -m tcp Specifikacija sa kojim se portom radi i koristi se extend match funckija (p je protokol a m ili ti match omogućava pisanje daljih pravila kao što je dport u mom slučaju)

--dport 22 koji port je u pitanju

Na kraju imamo -j ACCPET da prihvatamo ovaj saobraćaj

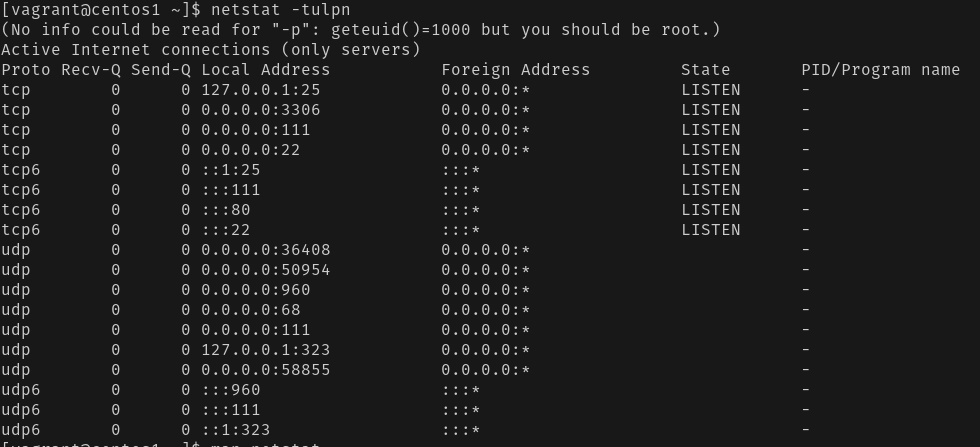
Neke dalje komande koje su bitne jesu iptables-save koja će nam dati commit koji treba da se unese u /etc/sysconfig/iptables kako bi omogućili da neka specificirana pravila ostanu uvek aktivna bez obzira na reboot sistema.

Naravno ovaj servis kao i firewalld treba da se uključi kako bi bilo šta od ovog pisanog bilo uopšte aktivno.

## 12.3 Nadgledanje sistema

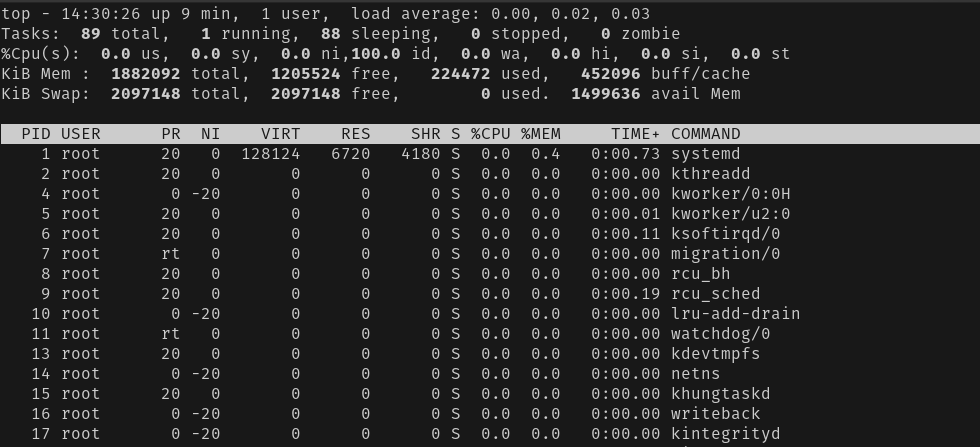
Genralno u radu sa serverom, veliki deo posla se svodi na nadgledanje nekih servisa koji možda ne rade ili želimo da utvrdimo njihovu funkcionalnost da rade ono što im je zadatak. Tako da imamo mnoštvo komandi koje su nam dostupne za neke određene zadatke, recimo želimo da vidimo koji su aktivni portovi i koji servisi slušaju na njima. Recimo često na mom poslu koristim sledeću komandu kada želim da vidim koji je aktivan port za neki servis $ netstat -tulpn (iako nije default na sistemima alternativa bi bila ss takođe netstat je deo net-tools paketa)

*Slika 108: Upotreba netstat komande*



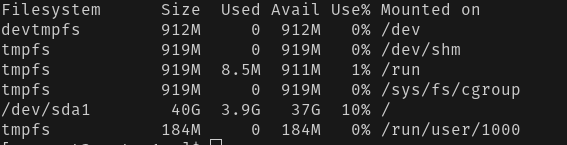
izlistani su mi aktivni portovi na serveru, ali malo objašenenje šta ja specifično tražim Flagovi: -t (tcp) -u(udp) -l(listening tj sluša ili it aktivan) -p(program da prikaže pid programa koji je aktivan recimo ssh koji je zapisan ovako 696/sshd) -n(numerička vrednost) Još jedna vrlo čest komanda bi bila top koja nam daje informacije o svim aktivnim servisima imam servis ili ti komanda kome pripada root u koloni user koji joj je pid itd… može se koristi i za proveru load na serveru koji možemo videti pod load average ovo je

*Slika 109: Upotreba top komande*



izuzetno korisno ako neki servis pravi problem pa mogu videti otprilike kako se sistem ponaša. Često ćemo naići i na problem sa upotrebom disk prostora koji je takođe nešto što može da utiče na generalnu funkcionlanost servera.

*Slika 110: Prikaz rezultata df komande*



Dobra komanda koja nam može dati neki generalni pogled na stanje je df

Može se primetiti da su izlistani svi mogući direktorijumi i njihovo zauzeće, generalno ovo nam je dobar indikator gde da počnemo, na realnim serverima sem / root direktorijuma imamo i tmp i var direktorijum koji su izdvojeni zasebno kako bi se olakšalo upravljanje njima.

Ja sam ovde izdvojio samo neki površinski pregled, ima tu još mnogo toga ali tema ove sekcije nije da izlistam sve nego da predstavim kako bi u nekim slučajevima koristili neku komandu. Kroz prethodne sekcije ovog rada predstavljao sam mnoge druge komande koje se mogu koristi za sličan posao recimo find za koji sam pisao skripte u Shell Scripts tematici.

# Zaključak

Linux je veoma prevalentan operativni sistem barem u serverskom svetu, a ima i primene u korisničkom svetu kroz mnoštvo distribucija i opcija koje su dostupne korisnicima koji žele da koriste ovaj sistem kao svoj radni sistem.

Smatram da je nephodno bilo za programera ili ti nekog budućeg sistemskog administratora da bude upoznat sa ovim sistemom i nekim njegovim funkcionalnostima. Moj cilj u ovom radu jeste bio da prikažem osnovno podešavanje koje bi možda neko prvi put radio na nekom serveru, a i da predstavim neke česte teme koje će neko videti u ovom polju.

Nadam se da će neko u budućnosti pogledati ovaj rad i koristi ga kao referencu za neki značajniji projekat vezano za Linux sisteme, a i da ga možda koristi kao neku polaznu tačku shvatanje suštine i ideje ovog operativnog sistema.

# Literatura

1. Administriranje Linux sistema Kuvar (2019)

Adam K. Dean

2. Linux Shell Skriptovanje (2018)

Ganesh Naik

3. Operativni Sistemi prvo izdanje (2011)

Ranko Popović

irina Branović

Marko Šarac

4. https://www.linuxfoundation.org/

Classic SysAdmin

5. Računarske Mreže Sedmo izdanje (2020)

Mladen Veinović

Aleksandar Jevremović

6. Offical Centos documentation

https://docs.centos.org/en-US/docs/