**UNIVERZITET SINGIDUNUM**

Mentor:

Prof.dr Aleksandar Jevremović

Kandidat:

Aleksandar Milutinović

**Administracija I deployment Linux servera**

**-diplomski rad -**

**Novi Sad,2022.**

**Sadržaj**

[Uvod 4](#__RefHeading___Toc7225_3121985038)

[1. Unix kao preteča Linux 5](#__RefHeading___Toc3065_3088924660)

[2. Linux distribucije 6](#__RefHeading___Toc3067_3088924660)

[3. Struktura linux sistema I permisije 7](#__RefHeading___Toc3069_3088924660)

[3.1 Direktorijumi 8](#__RefHeading___Toc3071_3088924660)

[3.2 Permisije u linux sistemima 10](#__RefHeading___Toc3073_3088924660)

[3.2.1 Grupe permisija 10](#__RefHeading___Toc3075_3088924660)

[3.2.2 Promena permisija I komande 11](#__RefHeading___Toc3077_3088924660)

[3.2.3 Funkcija setuid 11](#__RefHeading___Toc3079_3088924660)

[3.2.4 Funkcija setgid 12](#__RefHeading___Toc3081_3088924660)

[3.2.5 Funckcija promene grupe I vlasnika 12](#__RefHeading___Toc3083_3088924660)

[3.2.6 Lepljivi bit 12](#__RefHeading___Toc4867_3088924660)

[3.3 Objašnjenje Sudo komande 13](#__RefHeading___Toc6160_3088924660)

[4. Instalacija Centos operativnog sistema 14](#__RefHeading___Toc3085_3088924660)

[4.1 Opšta konfiguracija 15](#__RefHeading___Toc4869_3088924660)

[5. Podešavanje remote konekcije ssh klijentom 21](#__RefHeading___Toc4871_3088924660)

[5.1 Opšta konfiguracija 22](#__RefHeading___Toc7227_3121985038)

[6.Automatizacija pravljenja virtualnog okruzenja 25](#__RefHeading___Toc6162_3088924660)

[6.1 Uvod 25](#__RefHeading___Toc6164_3088924660)

[6.2 Konfiguracija Virtualnog okruženja 26](#__RefHeading___Toc6166_3088924660)

[7. Daljinsko administriranje pomoću SSH 27](#__RefHeading___Toc6168_3088924660)

[7.1 Generisanje I upotreba ssh-keygen 28](#__RefHeading___Toc6170_3088924660)

[8. Networking/DNS 30](#__RefHeading___Toc3087_3088924660)

[8.1 A record 32](#__RefHeading___Toc3089_3088924660)

[8.2 AAAA rekord 33](#__RefHeading___Toc3091_3088924660)

[8.3 CNAME rekord 33](#__RefHeading___Toc3093_3088924660)

[8.4 MX rekord 34](#__RefHeading___Toc3095_3088924660)

[8.5 PTR rekord 35](#__RefHeading___Toc3097_3088924660)

[8.6 NS rekord 36](#__RefHeading___Toc3099_3088924660)

[8.7 SOA rekord 36](#__RefHeading___Toc3101_3088924660)

[8.8 SRV rekordi 37](#__RefHeading___Toc3103_3088924660)

[8.9 TXT rekordi 37](#__RefHeading___Toc3105_3088924660)

[9. Veb serveri, baze podataka I server za email. 38](#__RefHeading___Toc3107_3088924660)

[9.1 Instalacija httpd(apache) na CentOS 39](#__RefHeading___Toc3109_3088924660)

[10.1.1 Osnove Apache konfiguracije 40](#__RefHeading___Toc3111_3088924660)

[10.1.2 Kako ovo funkcioniše 42](#__RefHeading___Toc3113_3088924660)

[9.2 MySQL , Maria DB baze podataka 43](#__RefHeading___Toc3115_3088924660)

[9.2.1 Podešavanje secure instalacije 43](#__RefHeading___Toc6172_3088924660)

[9.2.2 Listanje kreiranje I selektovanje podataka I tabela 44](#__RefHeading___Toc3117_3088924660)

[9.3 Upotreba MTA (Mail transfer agent) 47](#__RefHeading___Toc3119_3088924660)

[10. Pravljenje osnovne konfiguracije neke domene upotrebom wordpress 49](#__RefHeading___Toc3121_30889246601)

[10.1 Uvod 49](#__RefHeading___Toc4875_3088924660)

[10.2 Instalacija Wordpress aplikacije 49](#__RefHeading___Toc4877_3088924660)

[11. Shell skripte 54](#__RefHeading___Toc3123_3088924660)

[11.1 Uvod 54](#__RefHeading___Toc4526_1923673954)

[11.2 Početak pisanja skripti 55](#__RefHeading___Toc4528_1923673954)

[11.3 Varijable 56](#__RefHeading___Toc4530_1923673954)

[11.4 Uzimanje unosa iz terminala 56](#__RefHeading___Toc8398_1923673954)

[11.5 Regularni izrazi 58](#__RefHeading___Toc23210_3121985038)

[11.6 Sed 59](#__RefHeading___Toc23212_3121985038)

[11.7 AWK 62](#__RefHeading___Toc23214_3121985038)

[12. Firewall I nadgledanje sistema 63](#__RefHeading___Toc8068_1220924201)

[12.1 Uvod 63](#__RefHeading___Toc17772_1923673954)

[12.2 iptables 63](#__RefHeading___Toc17774_1923673954)

[12.2.1 iptables dodavanje I uklanjanje pravila 65](#__RefHeading___Toc17077_509068356)

[12.3 Nadgledanje sistema 67](#__RefHeading___Toc17079_509068356)

[13. Zaključak 69](#__RefHeading___Toc8070_1220924201)

[14. Literatura 70](#__RefHeading___Toc17081_509068356)

# 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, korističu funkcionalni pristup gde ću kroz niz primera I upotreba različitih alata na Linux operativnom sistemu, prikazati različite probleme sa kojima se neki Linux sistemski administrator može susresti na nekom serveru.

# 1. Unix kao preteča Linux

Kao preteča za linux operativne sisteme bio je UNIX koji je još uvek jedan od efikasnijih operativnih sistema koji je korišćen kao osnova za neke moderne koje vidimo danas.

UNIX je kreiran 1969 godine od strane Denis Riči I Ken Thompson, u Bellovim labaratorijama iz Multicsa koji je bio korišćen kao osnova koji je prethodno bio neuspeli projekat višekorisničkog 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 source 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 javljaju virtulana memorija , straničenje na zahtev I TCP-IP protokoli. Neke od varijanata ovog operativnog sistema su Darwin,Dragonflz 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 dadoteka.

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 sire (varijacije vec postojeceg recimo C++), a I omogućuje prenosivost 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 al 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 embedded uređaja.

Linux je nastao na Univerzitetu u Helsinkiju 1991 godine , njegov kreator je Linus Torvalds.

Razvijen je prvobitno kao sredstvo za učenje za računare koji koriste Intelove mikroprocesore 80386. Zbog problema sa licencama odlučio je da napiše sopstveni operativni sistem.

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

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

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 kompanije koje su imale svoju iteraciju Linuxa a I generalno su se specijalizovale za ovaj operativni sistem.

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

Linux sam po sebi je klon UNIX, ali u isto vreme I nije , pozajmio je mnoge ideje od UNIXA a I primenjuje njegov interfejs ya programiranje API (kao što je definisano standardom POSIX) , suštinska razlika da nije potpuna kopija UNIXA kao neke druge varijacije koje su koristile pun kod.

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 kernel, C biblioteka I prevodilac, niz alata I osnovne sistemske funkcije(kao što su proces prijavljivanja I interpreter komandi).

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

Uglavnom kada se kaže linux misli se na kernel odnosno jezgro OS.

# 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>

može se videti koliko je to velika prepreka prilikom biranje, postoje stotine varijacija na već postojeći inicijalni.

Kada bi se razvrstalo na nešto jednostavnije možda kao početna bi se uzeo Debian pošto je to I možda napopularnija grana linuxa.

Jedna od popularnijih firimi u ovoj grani je Ubuntu gde nakdane popularnije verzije se nadovezuju na njihove repozitorijume, osnove …. 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 package manager više o ovome nakon pošto je to važna stavka za sve Linux operativne sistem.

Imamo SUSE kao jedna od popularnih grana. SUSE sam po sebi je grana slackware linux koji je uzgred jedna od najstarijih linux operativnih sistema.

SUSE je nastao samo godinu dana nakon SlackWare, I neku veću popularnost je stekao negde oko 2010 izlaskom OpenSuse operativnog sistema, package manager koji ova distribucija koristi je YAST. Recimo SUSE nudi I plaćenu verziju svog operativnog sistema odatle imamo I branch openSUSE koji je dostupan svima.

Sledeće u grupi većih ili značajnijih operativnih sistema je grupa redhat linux, popularne distribucije koje spadaju u ovu grupaciju su Fedora, CentOS, kao I sam redhat operativni sistem koji je razvijan za komercijalna rešenja.

Package manager koji se koristi je RPM(red hat package manager) yum (koji se nalazi na centos I fedori)

Uglavnom sve verzije od RedHat su jako kvalitetne verzije, jedne od većih pokretača web servera na internetu su recimo RedHat I CentOS gde je besplatna verzija pristupačnija pa je klijenti često koriste za svoj hosting 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 još.

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, kao I na svim ostalim distiribucijama.

Open Source AUR(Arch user repositories) koje su u suštini gurane od grupe korisnika koje postavljaju ove aplikacije na repozitorijum, a zvanični package manager je pacman.

Jedna stavka koja mora da se izvoji ovde jeste:  
Rolling release; Ovo su bleeding edge ili ti najaktuelniji operativni sistemi sa svim aplikacijama na najnovijoj verziji …. za serverske potrebe uglavnom se izbegavaju rolling release verzije operativnih sistema I 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.

Ove 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.

U suštini prilikom izbora distribucija u zavisnosti od potrebe a I šta korisnik preferia, navažnije je izdvojiti da treba koristiti distribuciju u kojoj se korisnik oseća lagodno I sa lakoćom može da rešava probleme na njima, neka od početnih distribucija je Debian based, ali kroz korišcenje korisnici se pronalaze I u drugim distribucijama.

# 3. Struktura linux sistema I permisije

Linux kao I svi trenutno prisutni operativni sistemi ima svoju strukturu fajlova I kako se vrše interakcije sa njima. Kako bi ga bolje razumeli 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 slicnu stukturu direktorijuma, da bi se razumela suština nephodno je objasniti svaki od ovih direktorjuma.

## 3.1 Direktorijumi

**/bin**

Ovo je direktorijum koji sadrzi binary fajlove, tj neke aplikacije I programe koji mogu da se pokrenu na OS. Ako prelistamo ovaj direktorijum mozemo naci 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 ya “et cetera”, iz razloga zato što je ovo bilo mesto za stavljanje fajlova za koje sistemski administrator nije bio siguran gde da stavi.

U sadašnjosti ovo je mesto gde se nalaze konfiguracioni fajlovi, pošto se ovde nalaze skoro svi sistemski fajlovi.

**/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 sajlu fajlove u hard disk na primer.

**/media**

Eksterni mediji,kao što su USB drajv, priključeni su na ovaj direktorijum, kao I drugi uređaji USB hard disks, SD kartice external 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**

Mesto gde je softver koji mi kompailiramo iz source koda ponekad završava kada se ne instalira na našoj distribuciji.

**/proc**

Je virtualni direktorijum, sadrži fajlove koji pružaju informacije o kernelu I u svakom procesu koji se pokreće na Operativnom Sistemu.

**/root**

Slično kao I home direktorijum ali za root korisnika ili ti administratora.

**/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**

Sadrži promenljive podatke kao što su http,tftp, evidence I drugo.

**/sys**

Virtualni direktorijum kao I /proc koji sadrži informacije o uređajima koji su povezani na računar.

## 3.2 Permisije u linux sistemima

U Linux sistemima postoje tri vrste dozvole:

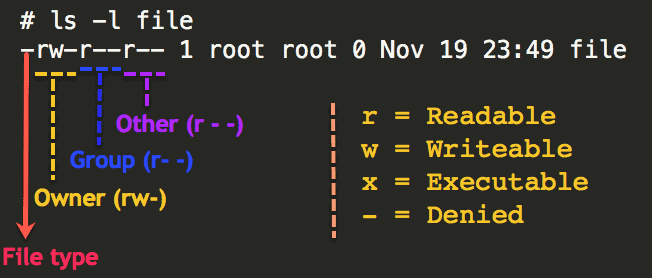
* Dozvole za čitanje ili ti 4 u brojčanom zapisu, korisnik može da čita ili da proveri fajl
* Dozvole za pisanje ili ti 2 u brojčanom zapisu, korisnik može da modifikuje fajl
* Dozvole za izvršavanje ili ti 1 u brojčanom zapisu, korisnik može da izvršava fajl

### 3.2.1 Grupe permisija

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

**Group** – Group permisije se odnose na grupu koja je dodeljena fajli 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:*[*https://pamirwebhost.com*](https://pamirwebhost.com/check-linux-file-permissions-with-ls/)

### 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**

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

### 3.2.3 Funkcija setuid

Korisna funkcionalnost je funkcija setuid. Imam skriptu I želim da se samo pokreće sa privilegijama korisnika, bez obzira koji korisnik pokreće koristićemo ovu funkciju.

Podešvanje ovog načina izvršavanja je prilično jednostavno pošto Linux ima već ugrađene funkcionalnosti za ovo.

**$ chmod u+s fajl**

ili

**$ chmod 4777 fajl**

Dozvole nakon bilo koje od dve prethodne komande biće **drwsrwxrwx,**  vidi se da je podešen ovaj način izvršavanja po S obeležiju u polju za vlasnika fajla.

### 3.2.4 Funkcija setgid

Slično funkciji setuid, I funkcija setgid daje korisniku mogućnost da pokrene skriptove sa privilegijama grupe vlasnika, čak I ako ga izvrši bilo koji drugi korisnik.

**$ chmod g+s fajl**

Alternativno možemo koristiti sledeći način podešavanja.

**$ chmod 2777 fajl**

Dozvole nakon bilo koje od dve prethodne komande biće **drwxrwsrwx,**  vidi se da je podešen ovaj način izvršavanja po S obeležiju u polju za vlasnika fajla.

### 3.2.5 Funckcija promene grupe I vlasnika

Kao I sve u Linuxu ništa nije permanento podešeno da mi ne možemo to promeniti kao root korisnik, tako isto I grupe I vlasnik….

**$chown user:group fajl**

Ova komanda nam daje mogućnost podešavanja ko je vlasnik fajla I kojoj grupi pripada.

### 3.2.6 Lepljivi bit

Lepljivi bit je funkcionalnost na Linux sistemima. Ako imam u odeljenju administratora 10 korisinka. Jedan direktorijum ima podešen lepljivi bit , onda se dešava sledeće, a to je da korisnici mogu samo da kopiraju fajlove u taj direktorijum.

Svi korisnici mogu da čitaju fajlove, ali samo vlasnik određenog fajla može da ga edituje ili da ga izbriše (ovo je prevalento u [Cpanel](https://docs.cpanel.net/) interfejsu o kojem se može saznati šta posetom na stranicu <https://docs.cpanel.net/> )

Ostali korisnici mogu samo da čitaju, ali ne I da edituju ili modifikuju fajlove ako je podešen lepljivi bit.

**$ chmod +t fajl**

ili

**$chmod 1777**

Dozvole fajla nakon prethodne dve komande će biti drwxrwxrwt (gde je t zapravo indikacija za lepljivi bit)

## 3.3 Objašnjenje Sudo komande

Česta komanda koja se vidi prilikom rada na bilo kom linux serveru, ako korisnik nije inicijalno logovan kao root user je *sudo*.

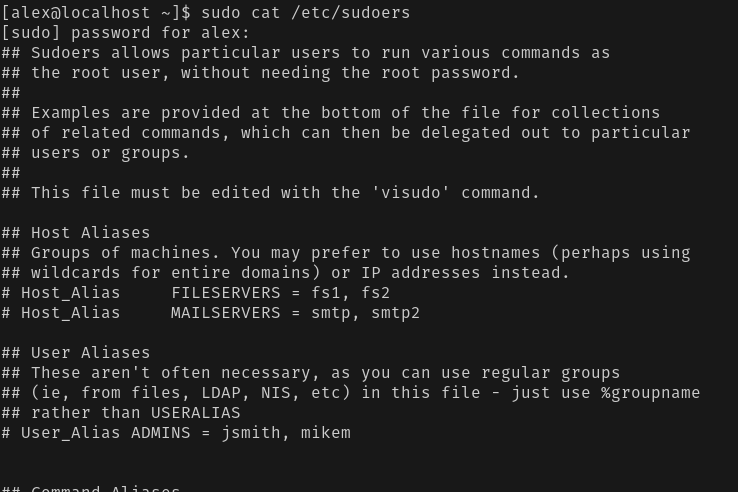
*Sudo* komanda se koristi kada želimo nešto da izvršimo, ali želimo da to bude sa povećanim privilegijama koje dolaze sa root korisnikom.

Naziv sudo stoji za “superuser do”, generalno ako se pokreće komanda za koju nemam dozvolu dobijam sledeće:

Slika 3 :Sudoers fajl

Izvršavanjem ove iste komande sa *sudo* dobijam sledeće:

kao što može da se vidi sa slike korišćenjem povećanih privilegija dozvoljeno mi je pristup sudoers fajlu kojem ima pristup *root* user.

Slika 4 :Sudoers fajl sadržaj

Ako prelistam malo sudoers fajl pri kraju ću naići na *%wheel* iznad ovog odeljka piše objašnjenje čemu služi.

Slika 5 :Sudoers fajl wheel grupa

Sledeće pitanje koje dolazi, ovde jeste da li je korisnik zapravo u grupi wheel, ovo lako može da se proveri pregledom sledećeg fajla  */etc/group*

Slika 6 :Sudoers fajl pripadnost korisnika

Može da se vidi da je grupa wheel prvi segment drugi segment je x što znači da je upotrebljene skrivene lozinke, 10 je ID grupe, I jedini korisnik 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 generalno zavisi od kompainje I prava korisnika koji taj učesnik ima u timu administratora.

Postoje mesta gde svaki operater ima moć upotrebe sudo komande, a postoje neka mesta gde samo jedan korisnik može da je koristi.

# 4. Instalacija Centos operativnog sistema

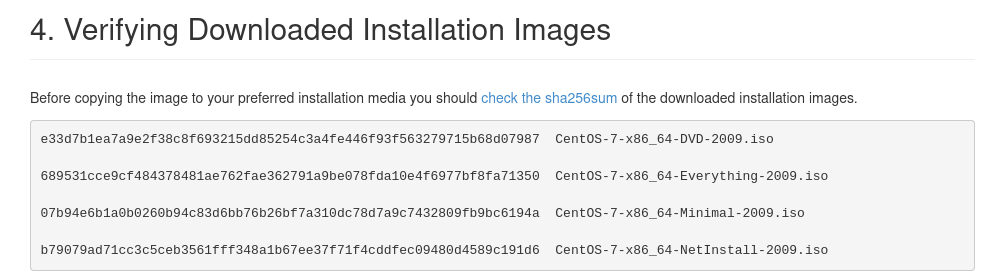
Pre same instalacije nephodno je izabrati odgovarajću arhitekturu za operativni sistem ako ne izaberem pravilno dobiću ovakvo upozorenje što nije strašno samo po sebi, ali može da uspori instalaciju bez odgovarajuće pripreme. (Treba ponovo da se skida image sa pravilnom arhitekturom)

Slika 7 :Greška arhitekture sistema

Takođe nije loše proveriti da li je iso operativnog sistema ispravan proverom *sha256sum*

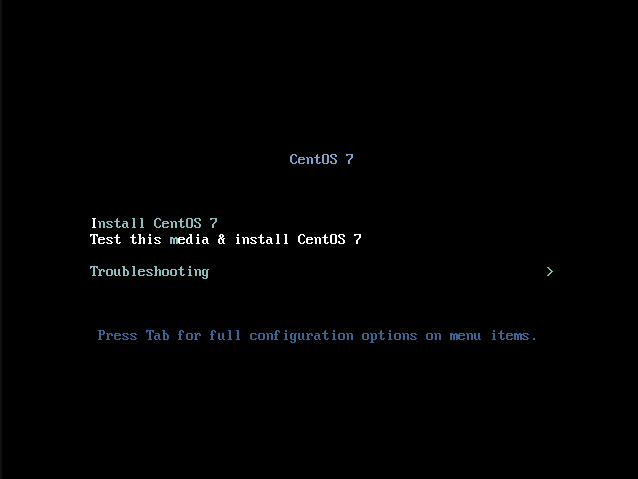
Slika 8 :Shasum CentOS iso fajla

Sada kako bi potvrdio da li ovo odgovara zvaničnom *sha256sum* moramo 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 I wiki centos koji ovo obuhvata isto ovo, postoji I za Debian I druge distribucije Linux)

Slika 9 :Verifikacija shasum

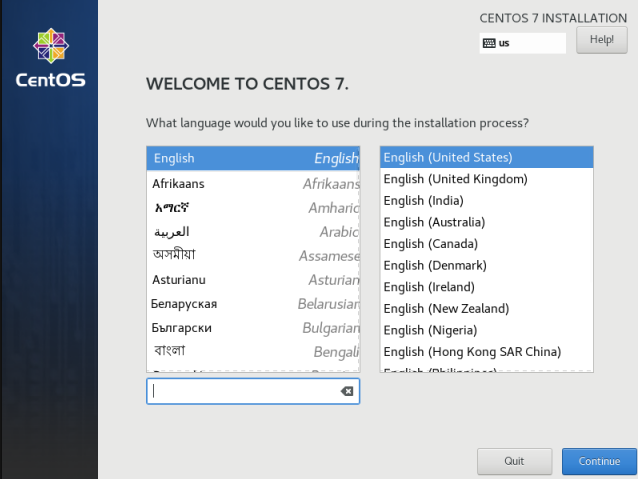
## 4.1 Opšta konfiguracija

1)Postavljanje boot up I pokretanje uređaja kao prioritet  
2) Slekcija opcije install cent os 7

Slika 10 :Početni prozor instalacije

-instalacija počinje nakon što se izabere opcija, samo po sebi sve je intutivno zato što se modernije verzije linuxa trude da ovaj proces bude što bezbolniji, tj što lakši za konfiguraciju

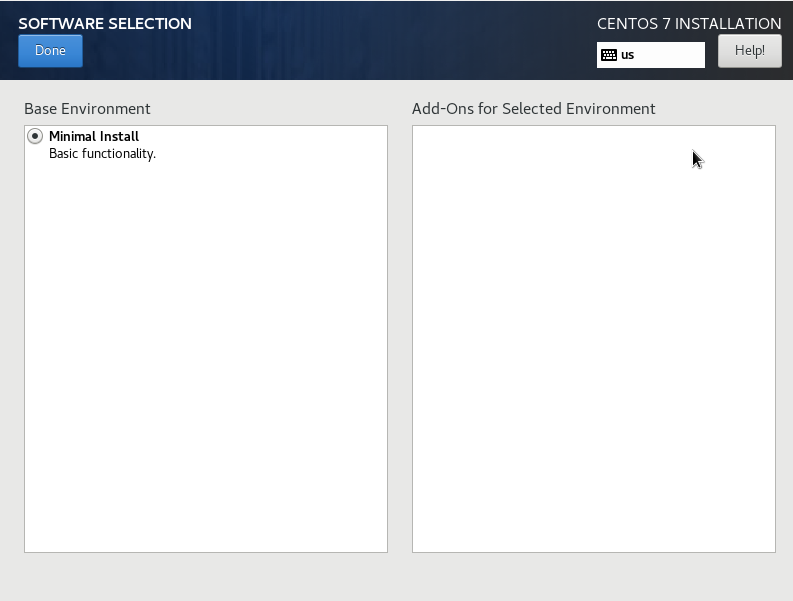
-dobiću nešto ovako nakon pokretanja

Slika 11 :Odabir jezika instalacije

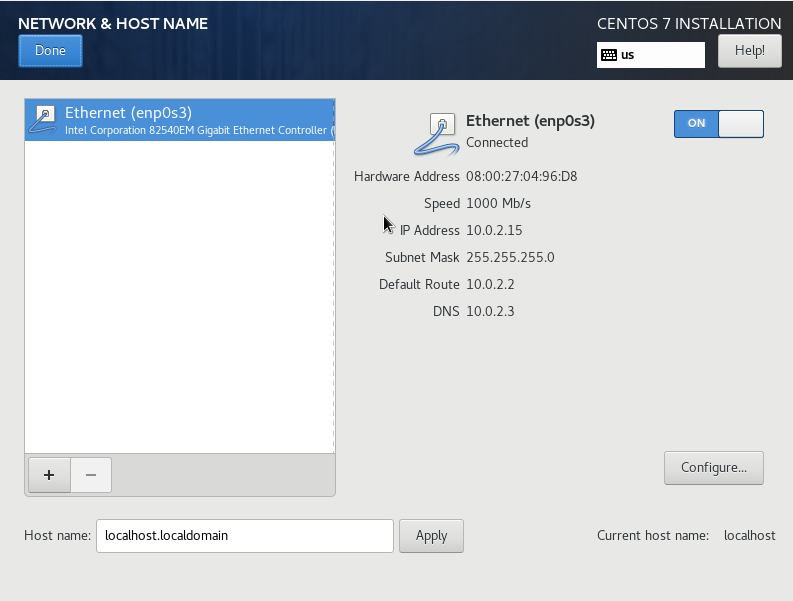
Sada dolazim na sumu instalacije

Slika 12 :Glavni instalacioni prozor

3) Pošto je ideja da ovo bude server odabirom opcije Software Selection pokušaću da napravim server što minimalističli moguće tj da se sve radi kroz terminal, ovo je bitno zato što aplikacije na serveru same po sebi uzimaju dosta memorije I dodavanje GUI (Graphical user interface) bezpotrebno zauzima još prostora, stim se uglavnom serveri sa GUI izbegavaju,ali … neki korisnici ovo preferirifaju.

Slika 13 :Odabir paketa instalacije

4) Sledeća bitna stavka je Network and Hostname

Slika 14 :Uključivanje network uređaja

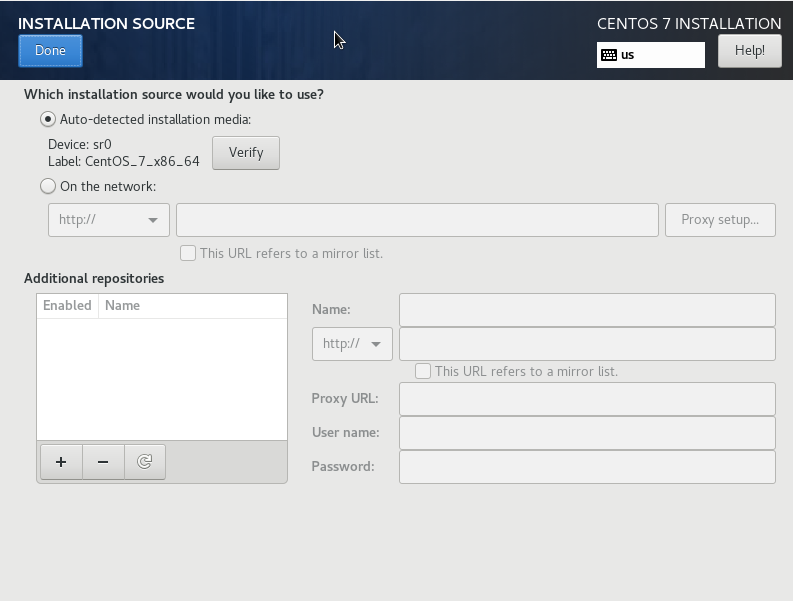
Nephodno aktivirati uređaj odabirom opcije *on* na radio dugmetu,

nakon što se klikne kao I sa slike vide se default podešavanje za mrežu.

Pošto je ovo virtualno okruženje koje koristim, VirtualBox kreira NAT mrežu prema podrazumevanom podešavanju, što znači da se VM nalazi na istoj mreži kao I host računar. Umesto toga, VM je sama na mreži, ali sa putanjom ka spoljašnjem svetu preko host mašine.

Klikom na dugme Done se završava podešavanje mreže za sada….

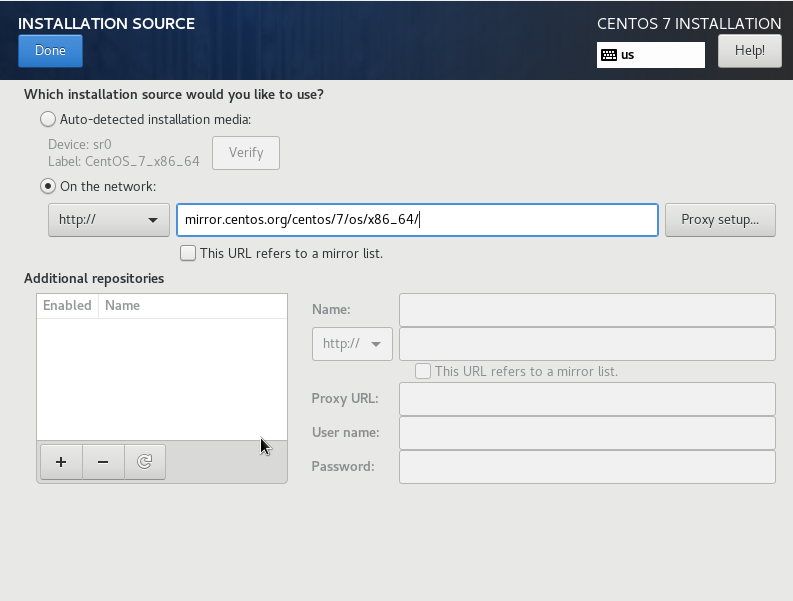
5) Sledeće šta ću da uradim jeste podešavanje installation source.

Slika 15 :Odabir instalaciong izvora

Unutar ovog ekrana je automatski izabran medium, ustvari, iso diska (sr0 je linuxova oznaka za drive diska)

Promeniću ovo dugme na *On the Network* (Ovo je zbog prethodnog podešavanja za konekciju ka internetu)

Nakon podešavanja dobijam:

Slika 16 :Unos instalacionog izvora

Može se videti sa slike da sam uneo samo mirror koji će source koristiti prilikom instalacije

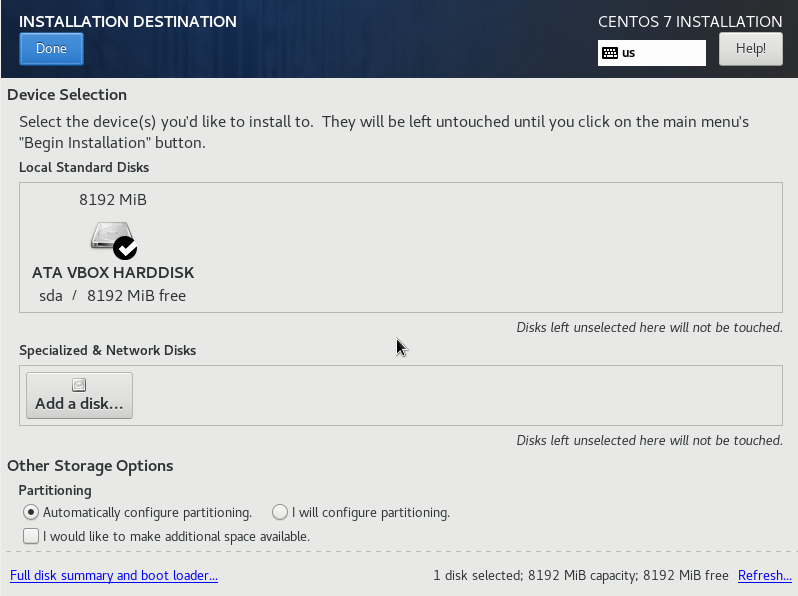
Ovom promenom kada se vratimo na početni prozor sistem će izbaciti upozorenja da je Izvor promenjen I da se mora podesiti ponovo (slika ispod)

Slika 17 :Potvrda selekcije

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

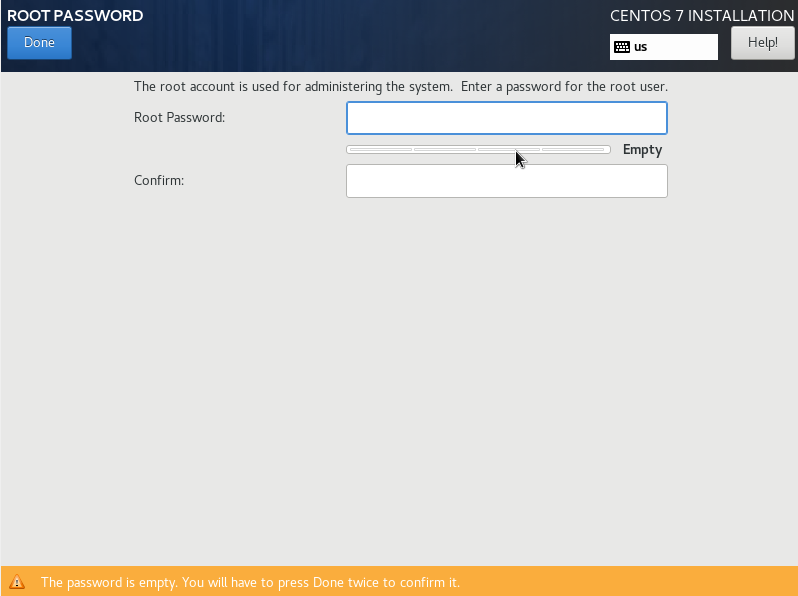
Poslednja stavka je *installation destination*

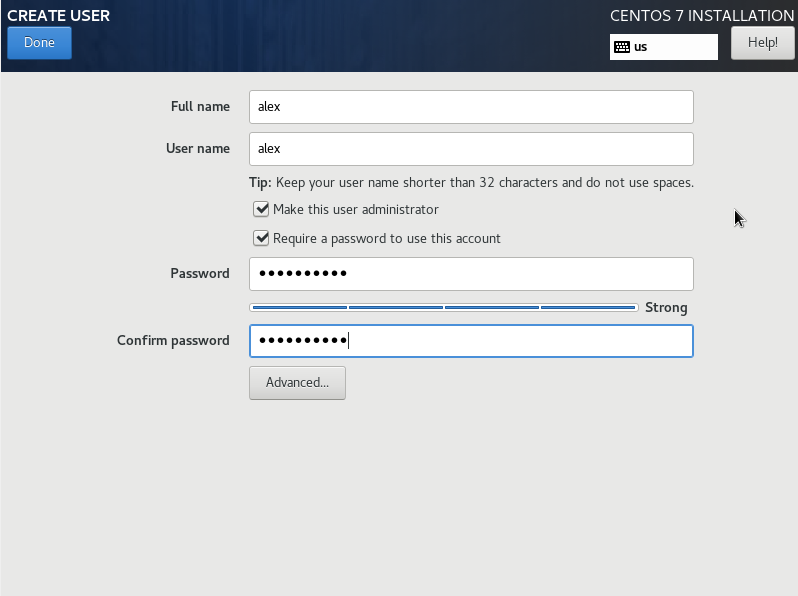
ostavljam na default tj dozvoliću sitemu da kreira particije kako želi.

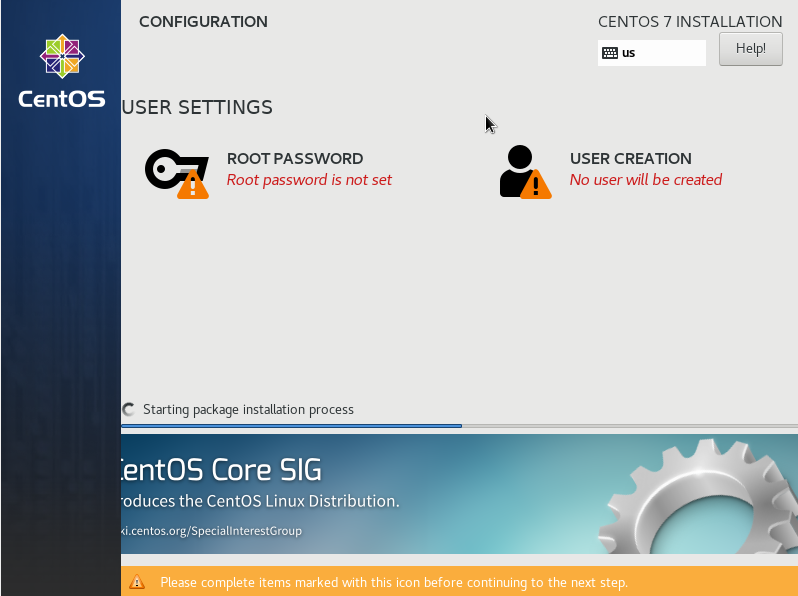
Slika 18 :Odabir diska instalacije

Tokom instalacije pojavljuje se ovaj prozor koji mi nudi podešavanje root usera

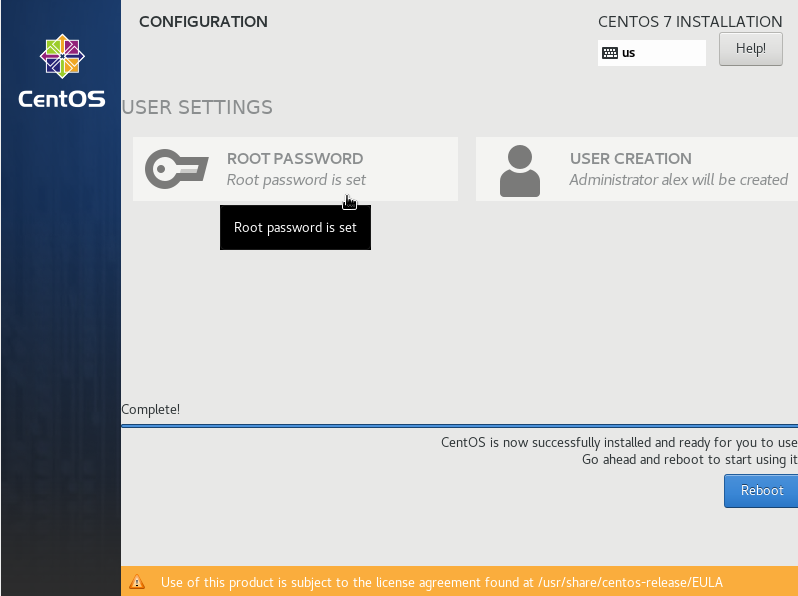
(Kada počnemo isntalaciju tj uglavnom an centOS 8 I 9 ovo se nalazi na inicijalnom prozoru)

Slika 19 :Kreacija root korisnika

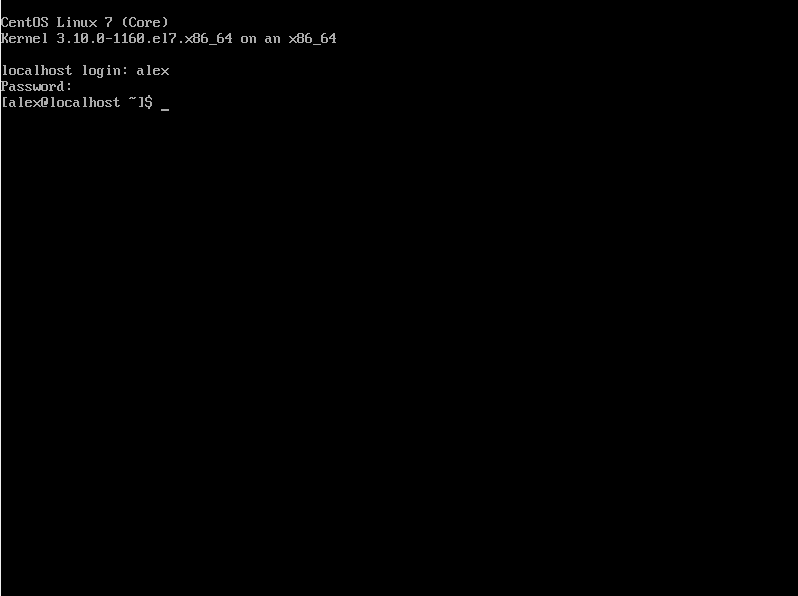
Slika 20 :Kreacija korisnika

Slika 21 :Instalacija pre kreiranje korisnika

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

Slika 22 : Instalacije nakon kreiranja korisnika

Prvi login na sistem izgleda nešto ovako

Slika 23 :Server Login

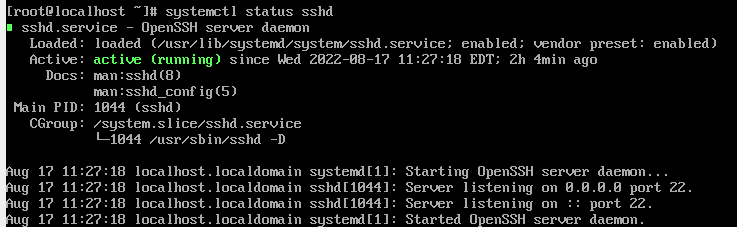
# 5. Podešavanje remote konekcije ssh klijentom

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

Samim tim što se primeri ovde svode na korištenje virtualne mašine pogodno a je podesiti ovaj tip konekcije zato što nam olakšava generalno neke osnovne zadatke lakše da obavljamo kao što su kopiranje pomeranje …. (malo je sporija interakcija na virtualnoj konzoli)

## 5.1 Opšta konfiguracija

Prvo što ću da uradim jeste da testiramo postojanje ssh klijenta, a I da li je port otvoren

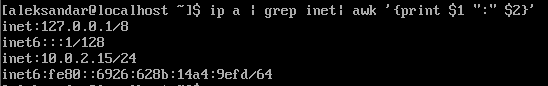
Slika 24 :Status sshd servisa

Sa slike iznad servis je aktivan I sluša na portu 22 ….

ali nešto nije uredu kada se konektujem sa host mašine konekcija je u idle što je implikacija da će eventualno vreme čekanja isteći I dobiću ili timeout error ili refused to connect.

Slika 25 :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

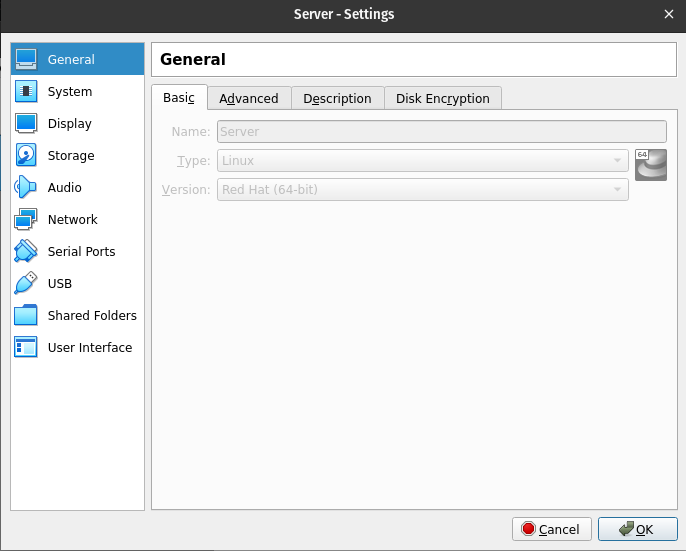
Slika 26 :Upotreba ip komande

(koristio sam više komandi da to učinim malo vidljivijim ovde )

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.

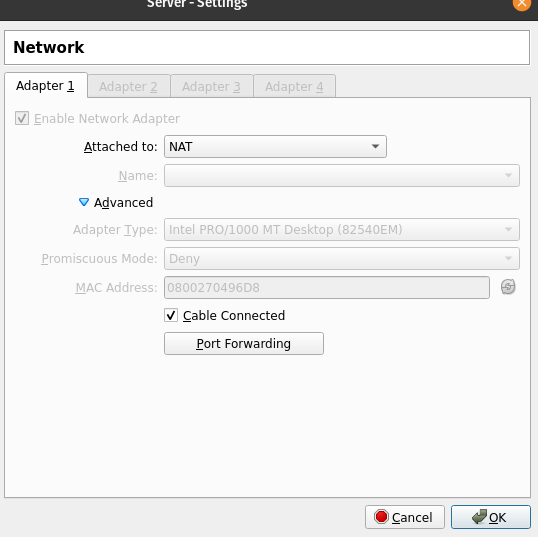
Kako bi konfigurisao remote konekciju (sa hosta) moraću malo da nameštamo port forwarding u Virtualnoj mašini…. (pošto simuliramo postojanje servera preko ove mašine)

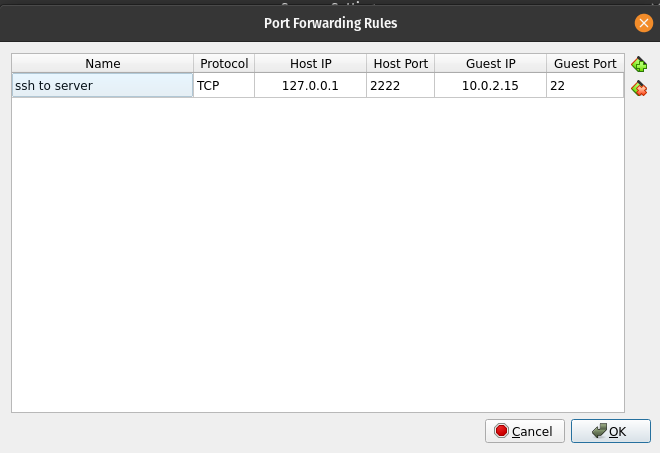
Prvo se otvori prozorčić za podešavanje

Slika 27 :Prozor podešavanja VM

Sledeće idemo na network I pritiskom na advanced dugme dobijamo opciju odabira podešavanje port forwarding

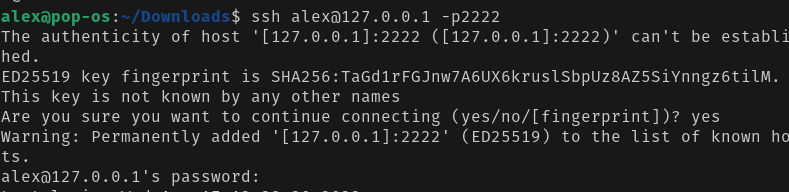
(Port forwarding je metod ručnog podešavanjue kako saobraćaj treba da prođe kroz NAT mrežu)

Slika 28 :Prozor podešavanja NAT konekcije VM

Slika 29 :Podešavanje pravila za port forwarding

Podešavanje koje sam ovde postavio jeste kada želimo u SSH upišemo sledeće

ssh [alex@127.0.0.1](mailto: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 čemu smo ostvarili vid konekcije koji nam je uglavnom dostupan prilikom administriranja servera.

Slika 30 :Login na server koji je kreiran u virtualnom okruženju

(guest IP je ip servera koji možemo dobiti pozivom ip a )

Sledeće što ću uraditi jeste update koji će povući sve poslednje verzije softwera koji je pokrenut na serveru ukoliko za to postoji potreba naravno, koristeći komandu   
$ **yum update**

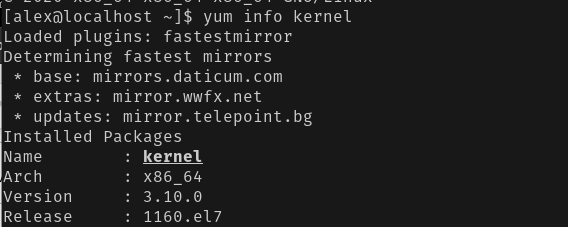
Kernal mogu proveriti pozivajući sledeću komandu

**$ uname -a**

Rezultat je sledeći:

Slika 31 :Upotreba komande uname

Pokretanjem sledeće komande :

Slika 32 :Prikaz poslednje verzije kernela u package manageru

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

Slika 33 :Verzija kernela

# 6.Automatizacija pravljenja virtualnog okruzenja

## 6.1 Uvod

Pošto sama instalacija može da bude prilično monotona kada želim da instaliramo VM svaki put kako bi testirao nešto novo u izolovanom okruženju, može postati kontraproduktivno.

Iz tog razloga razvijen je sistem koji bi ovaj procec automatizovao , prednosti ove automatizacije su :

* Eliminiše vreme potrebno za ručno kucanje odgovora u VM prozor
* Omogućava autmatizaciju 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.   
  Ovo je oblik **Infrastructure as code(IAC).**

Bitno je napomenuti jedan od metoda automatizacije raspoređivanja okvira a to je *kickstart fajlovi* **,** koji se često koriste u velikim raspoređivanjima za automatko odgovaranje na pitanja koja 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.A, pa , čak I celih razvojnih okruženja.

Okruženja su obezbeđena u obliku Vagrant fajlova koji se nalaze na Virtualnom 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 je to jednostavno kroz terminal koristi se **mkdir** komanda koja napravi ovo za nas

3) inicijalizujemo taj direktorijum kao mesto za vagrant konfiguraciju koristeći

**$ vagrant init**

4) Naredna komada je malo kompleksnija o njoj neću pričati sada, ali suštini 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 centos-7 image za ovu virtualnu mašinu.

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 konfiguracija za ovo može da se nađe u vagrant fajlu)

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.

No implikacija je da neće svaki put ići do data centra ako se neke stvari mogu rešiti samo što se poveže na udaljeni server.

Iz ovog razloga sistemski administrator će koristiti SSH secure shell konekcije koje omogućavaju administratoru linux sistema da pristupi serveru I napravi neke promene koje su neophodne.

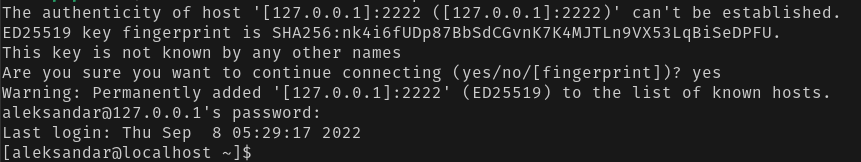
Secure Shell protokol je jedan samo od korišćenih primera ima mnogo varijacija na ovo ali pošto je jedan od poznatijih a I bezbednijih odlučio sam da prestavim ovaj , a I dolazi kao default na skoro svim Linux sistemima.

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 I treći deo -p odnosi se na port koji će biti korišćen u ovom slučaju to je 522

nakon ukucane komande dobiću key koji bi trebao da potvrtdim I nakon toga bi uneo lozinku to bi ovako izgledalo

Slika 34 :Potvrda prilikom SSH na server

uglavnom je ovo uredu metoda konekcija, ali može I na alternativni način da se napravi a to može da se izvrši generisanjem ključa koji prelazim u narednom odeljku.

## 7.1 Generisanje I upotreba ssh-keygen

Jedna stavka koja je možda bitna je generisanje ključeva, u realnom svetu lozinke su odlične ali u isto vreme su I loše.

Većina ljudi koristi jednostavne lozinke koje se mogu lako provaliti korišćenjem različitih metoda za izvršavanje ove akcije recimo jedna poznata a I klasična je brute force gde se pokušavaju sve moguće lozinke na silu ili imamo I dictionary attack 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 private I public.

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 put taj passphrase ostaje zapamćen na mašini I koristi se iznova ukoliko želim da se povežem na isti host.

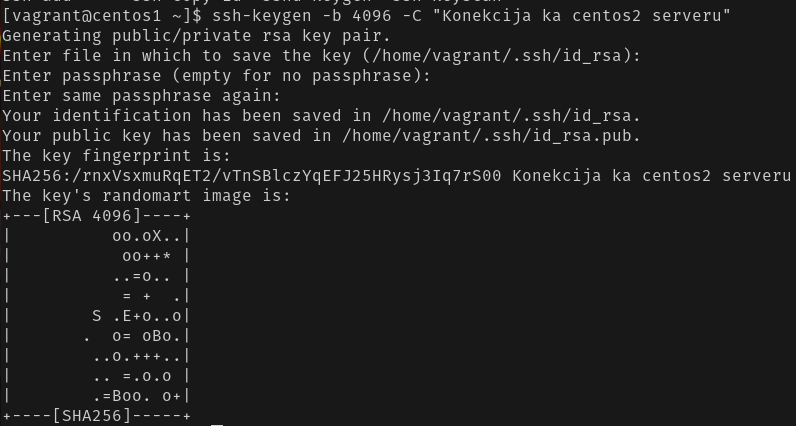
Kako to izgleda između dva servera recimo

Prvo što bi uradio spojio bi se na server pošto ja u mojim primerima koristim vm,

Slika 35 :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.

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

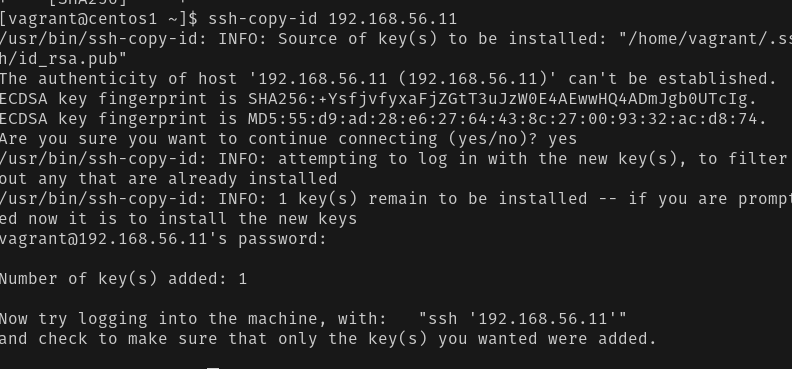
Slika 36 :Generisanje ključa

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.*

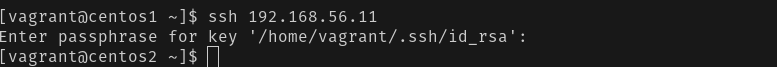
Naravno ovo samo po sebi ništa ne znači da bi mogao ovo koristiti neophodno je da kopiram public key na odredišni server .

Ovo mogu uraditi ručno ili koristeći komandu ssh-copy

Slika 37 :Kopiranje ključa na udaljeni server

destinacioni server prihvata key kada mu se unese lozinka koju će tražiti, takođe mi nudi opciju kako da se konektujemo na drugi server.

Može da se primeti da mi komanda traži passphrase koji sam zadao prilikom generisanja ovog ključa, nakon unosa omogućava mi konekciju.

Slika 38 :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 pages 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 neko će znati odgovor ali uglavnom se svodi da neće imati odgovor ako ga prvobitno ne pretraži, odavde nastupa DNS.

Šta je DNS ? Ukrato rečeno DNS pretvara URL u brojeve ili ti IP adresu koju računar razume, pošto u mašinskom jeziku se ne koriste imena.

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

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.

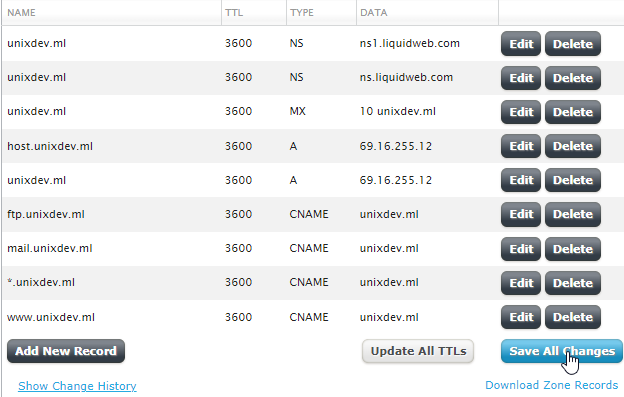
DNS je glavni gradivni element modernog interneta, I može da se koristi analogija da je kao ‘Telefonski imenik’ uzima ime I pretvara ga u telefonski broj u našem slučaju pretvara ime domene u IP adresu. Kada bi pretraživali neki broj na našem telefonu vrlo često ćemo prelistati naš imenik a zatim izabrati neki broj ovo se zove lokalno skladište, imamo isti slučaj I na PC gde on ima svoj presonalni lokalni fajl koji se zove hosts fajl I on 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 kreira se 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 razresavaju u zavisnosti gde je DNS zona koja je zaduzena za domenu ili registrar u zavisnosti od konfiguracije.

Kada se nalazimo u serveru ovo se razresava preko BIND servisa recimo ali na visem nivou prvo se postavlja na mestu gde nam je hostovan sajt.

Slika 39 :DNS zona

Iz primera podešavanja rekorda, može da se primeti kako trebaju da se razrešavaju na serveru.

Slika 40 :DNS zona podešavanje

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.

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 rekord 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.

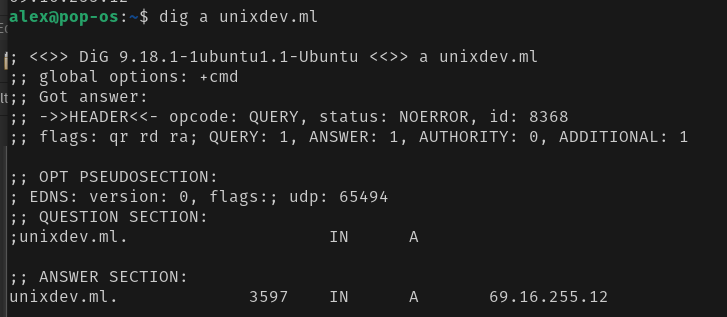
Recimo imamo nekidomen.tld I želimo da dodelimo 10.10.0.1 IP adresu našem web serveru, onda bi trebali da kreiramo A rekord sa “[www.nekidomen.tld](http://www.nekidomen.tld/)” kao **Fully Qualified Domain Name** I u value polju bi postavili IP adresu servera 10.10.0.1 recimo.

Sada je implikacija da će svi zahtevi za [www.nekidomen.tld](http://www.nekidomen.tld/) biti poslani na server sa IP adresom koju smo specificirali u value polju dns rekorda.

Bazično 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 nam bilo šta sotji ispred kao subdomena “bilošta.nekidomen.tld” I zatim se razrešava u zavisnosti od IP koji mu je dodeljen u 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 41 :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 section 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.

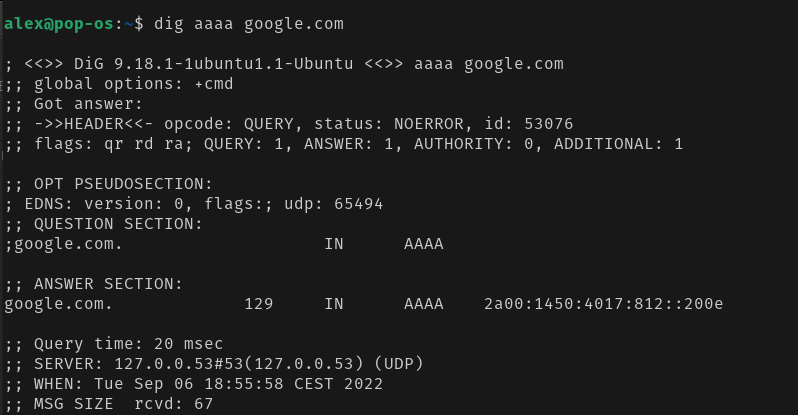
## 8.2 AAAA rekord

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 42 :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.

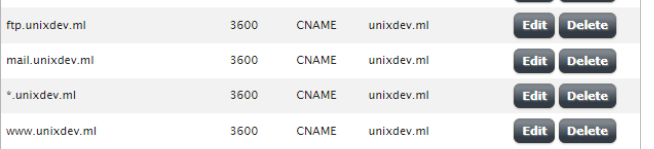
## 8.3 CNAME rekord

CNAME rekord ili ti 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 god želimo da asociramo novi subdomen na već postojeći A rekord možemo uradit recimo “[www.nekadomena.tld](http://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 god želimo subdomena, bez da specificiram IP na svaki rekord. Generalno pravilo je koristi CNAME, ako imaš više servisa koji ukazuju 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:

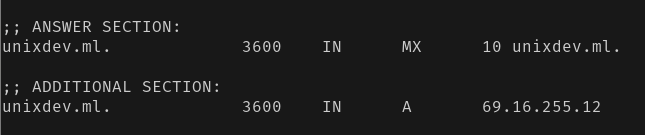
Slika 43 : Prikaz CNAME na hostingu

sa primera može da se vidi par servisa koji koriste CNAME da bi pokazivali na isti IP kao A rekord glavne domene.

Recimo [ftp.unixdev.com](ftp://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 44 :Struktura MX rekorda

Kao što se može videti MX rekord pokazuje da svi mail idu na @ ili ti 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

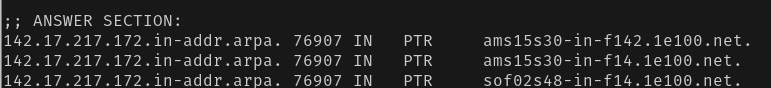
PTR rekord ili pointer rekord mapira IPv4 adresa na canonical name za hosta.

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](http://www.nekadomena.net/) ima IP adresu 122.0.3.16, ali PTR rekord to mapira kao

16.3.0.122.in-addr.arpa

Slika 45 :A rekord prikaz

Slika 46 :PTR rekord na osnovu A rekorda sa prethonde slike

IP adresa je ispisana sa desna na levo, I dodat je in-addr.arpa kao sto se vidi ovo je sada ne levoj strani gde je domen, a na desnoj strani kao value su serveri namenjeni za google ili ti google.com.

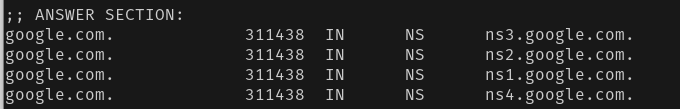
Ovaj rekord se uglavnom koristi za bezbednost I anti-spam mere, gde većina web servera I email servera rade reverse DNS lookup da provere da li host zapravo dolazi od iste lokacije odakle prikazuje da dolazi. Preporučljivo je da imamo postavljen PTR, takođe je veoma preporučljivo da imamo ovo postavljeno 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 47 :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 I updated SOA rekord može da samnji bandwidth između nameservera, poveća brzinu kojom može da se pristupu stranici, omogući stranici da bude aktivna čak I kada je glavni DNS server dole.

Slika 48 :SOA rekord

Prvo polje je ime ili ti 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*.

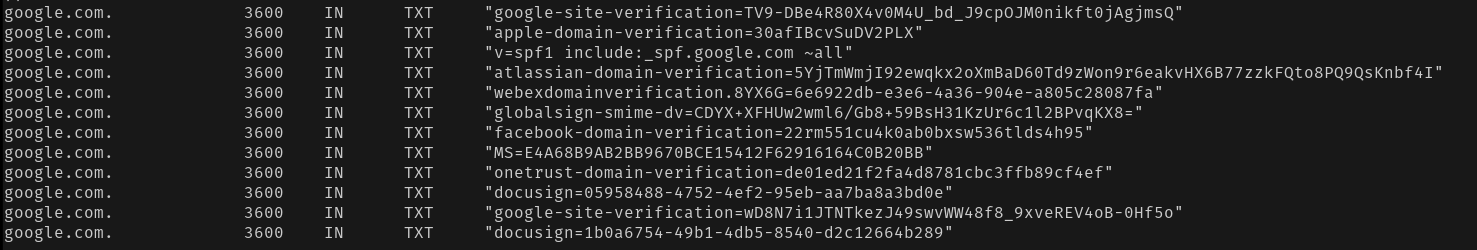
Imamo serijski broj koji je 472681414 koji daje kada je poslednja revizija izvršena trebalo bi da je format yyyymmddnn (nn je broj revizije) nakon toga imam refresh rate u sekundama koji je 900 update retry koji je 900 , expiry koji je 1800 I 60 koji stoji kao minimum ili ti osnovno vreme koliko bi trebali 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 49 :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.

Veb sajtovi postoje na Internetu, gde se nastoje da pronađu informacije. Veći deo Veba se pokreće na Linux serverima ( sa segmentiranim I mračnijim uglovima Veba na Windows).

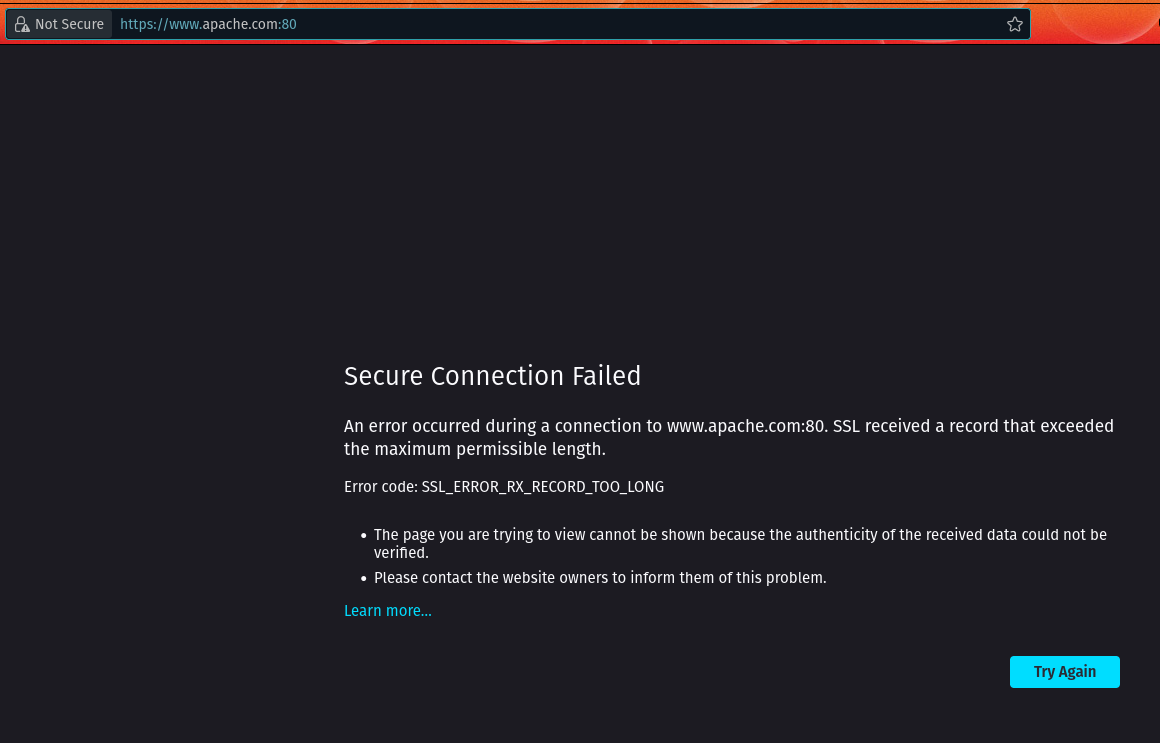
Postoje dve vrste veb servera oni koji se pokreću sa Apache I oni koji se pokreću NgInx , gde je apache stariji I još uvek vodi u trci najdominantnijeg web servera ali NgInx nije daleko iza zbog rastuće popularnosti.

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://duckduckgo.com](https://duckduckgo.com/) u Chrome ili Firefox, biće učitan veb sajt, ali neće prikazati koji je port. Isto tako ako se stranica pretraži sa portom dobićemo isti rezultat.

Ako recimo želimo da otvorimo port 80 koristeći https biće prikazana greška koja ukazuje na ne sigurnu konekciju na serveru.

Slika 50 :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.

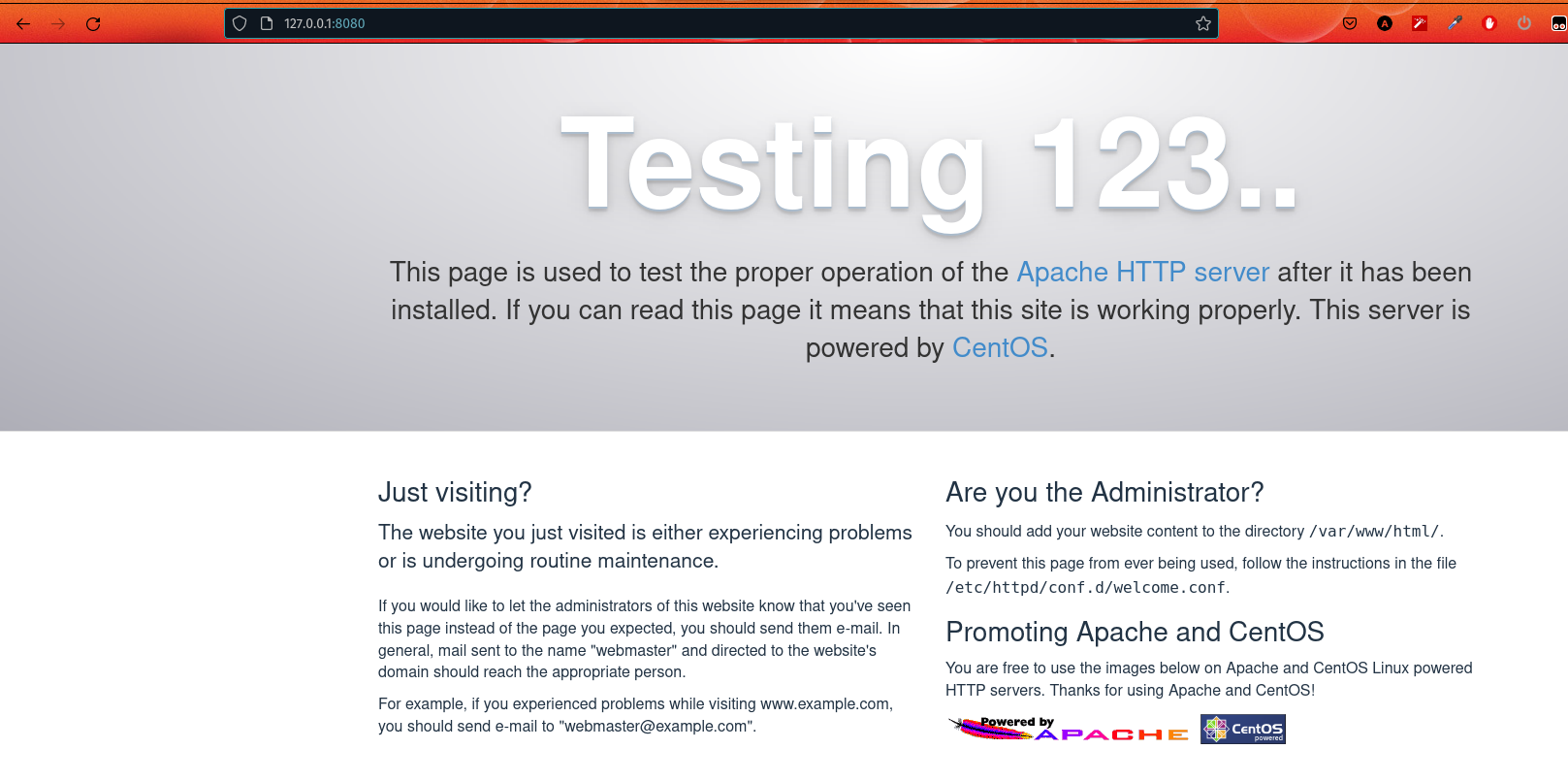
Instalacija softvera se vrši preko yum package manager na sledeći način:

**$ 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 npr anti virus….

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

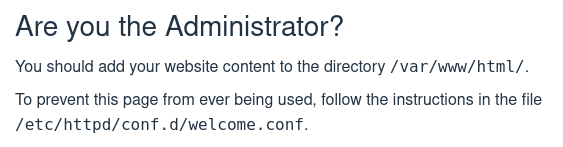
Kada sam podesio servis/daemon trebalo bi da izgleda ovako:

Slika 51 :Apache početna stranica

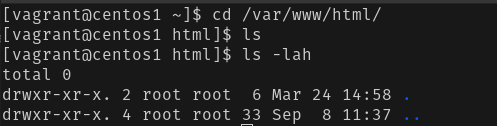
Ovaj splash ili ti index stranica mi govori da je apache podešen po fabričkim ili ti podrazumevanim podešavanjima.

### 10.1.1 Osnove Apache konfiguracije

Pogledom na stranicu uočava se sledeća poruka:

Slika 52 :Poruka gde se nalazi konfiguracija od apache stranice

primećujem da sam apache mi nudi gde treba sadržaj stranice da se nalazi.

Slika 53 : Sadržaj www/html direktorijuma

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

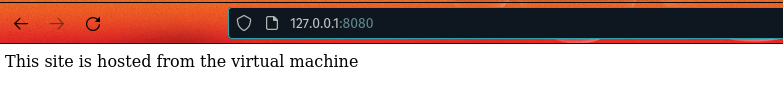
**$ cat << HERE | tee -a /var/www/html/index.html**

**This site is hosted from the virtual machine**

**HERE**

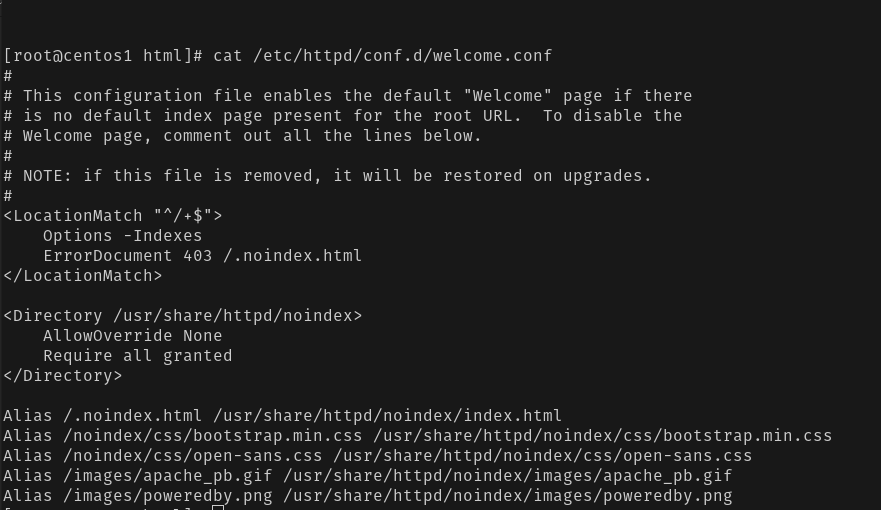
Recimo da kreiram fajl index.html koji će sadržati informaciju da je ovaj sajt zapravo hostovan na virtualnoj mašini.

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

Slika 54 :Prikaz index.html stranice

upotrebljen je direktorijum da zameni stranicu, ali za sada ne znam odakle zapravo vuče konfiguracije, ako pogledam sledeći fajl:

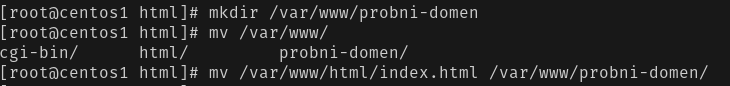
**$ cat /etc/httpd/conf.d/welcome.conf**

Slika 55 :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 56 :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 57 :Kreranje fajla probni-domen.conf I njegov sadržaj

### 10.1.2 Kako ovo funkcioniše

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.

Slika 58 :Prikaz DocRoot u httpd.conf

DocumentRoot se isčitava iz /var/www/html, a razlog zašto apache isčitava index.html je sledeći:

Slika 59 :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

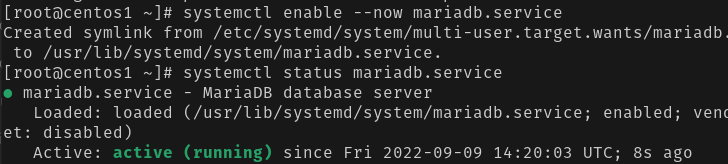
Tradicionalno baze podataka su odlično mesto za skladištenje uređenih podataka specifičnog tipa I veličine.

Implikacija je da možemo imati bazu podataka za bilo šta, od transakcija u bazama pa do zapisa invetara. 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 60 :Instalacija mariadb servera preko yum

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

Slika 61 :Uključivanje mariadb servisa

### 9.2.1 Podešavanje secure instalacije

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

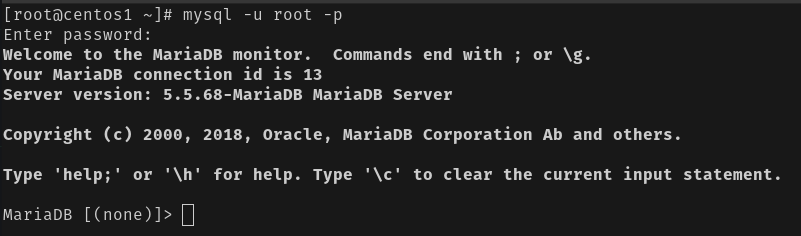
**mysql\_secure\_installation**

Ova skripta će me 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)

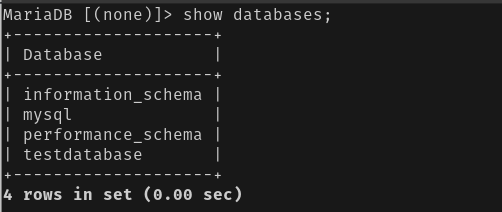
Nakon osnovnog podešavanje trebalo bi mi biti omogućeno da pristupim bazi :

Slika 62 :MySQL command shell

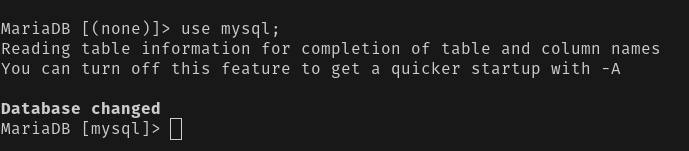
### 9.2.2 Listanje kreiranje I selektovanje podataka I tabela

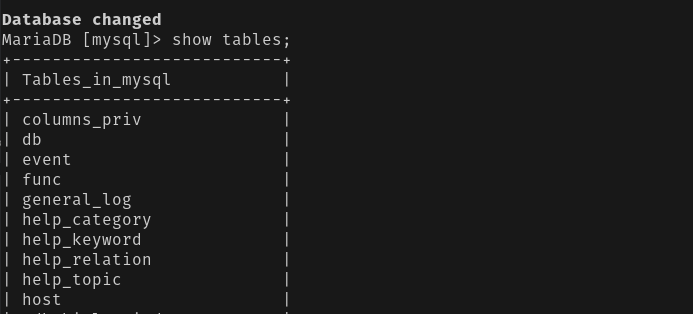
Bazom može da se upravlja iz terminala ili ti shell koji se otvara pokretanjem iz serverskog shella.

Osnovna komanda bi bila da proverim koje baze imam aktivne na serveru.

Slika 63 :Prikaz show komande u mysql

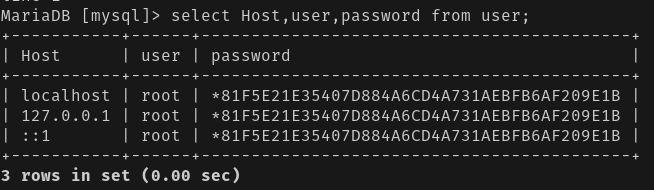
Recimo da želim da selektujemo neku od ponuđenih baza mogu koristiti sledeće;

Slika 64 :Prikaz use komande u mysql

Slika 65 :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 i*nsert, select, create database drop.*

Slika 66 :Upotreba select komande

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

Kreiranje baze podataka:

Slika 67 :Upotreba create database komande

Brisanje baze podataka

Slika 68 :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 koji često radi sistemski administrator:

Slika 69 :Upotreba create komande za tabele

Slika 70 :Upotreba insert komande za unos vrednosti u tabelu

Takođe postoje I dozvole za baze podataka, koje su takođe važne koliko I regularne dozvole file 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)

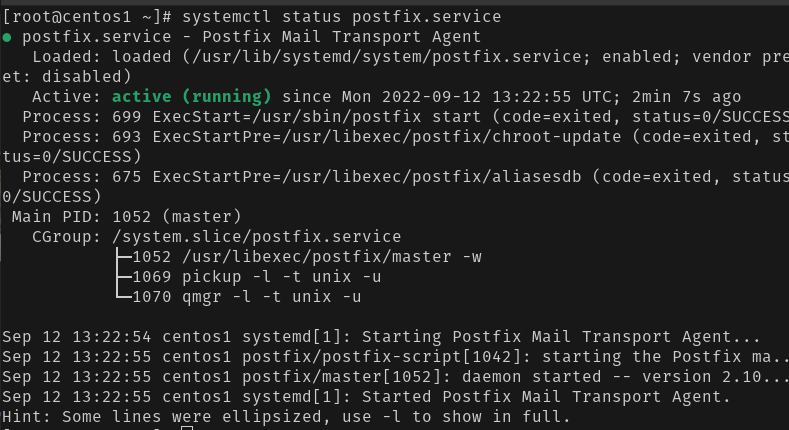
E pošta je možda nešto najučestalije u poslu sistemskog administratora, iako sam email sistem je već jako zastarela tehnologije još uvek se koristi u velikoj meri.

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:

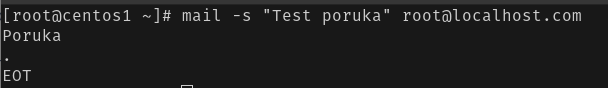
Slika 71 :Provera stanja servisa postfix.service

Default mta na sistemu:

Slika 72 :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 73 :Instalacija mailx aplikacije na server

Slika 74 :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.

Za postfix možemo proveriti *var/log/maillog* pošto je ovaj fajl pisanog formata korisno je uportebiti tail

Slika 75 :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*](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 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 76 :nstalacije epel-release paketa

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

Slika 77 :Dodavanje repozitorijuma za instalaciju

Instaliram yum alatke koje će mi omogućiti da podešavam verzije php

Slika 78 :Instalacija dodatnih yum alatki

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

Slika 79 :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 80 :Instalacija PHP paketa I biblioteka

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

Slika 81 :Provera verzije PHP

(Takođe nije loše ponovo poktenuti httpd nakon ovih izmena)

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

Pre svega treba dobaviti paket, ali u cli 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**](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 82 :Preuzimanje wordpress fajla sa interneta upotrebom wget

Trebao bi da dobijem ovakav neki podatak:

Slika 83 :Prikaz preuzetog fajla

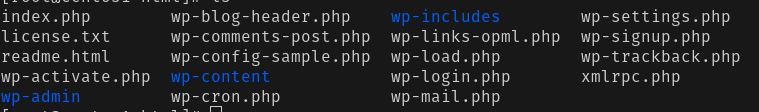
Upotrebom bilo **tar** komande ili **unzip** mogu izvući podatke iz ovog fajla.

Kada to obavim dobijamo direktorijum wordpress:

Slika 84 :Prikaz otpakovanog fajla

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

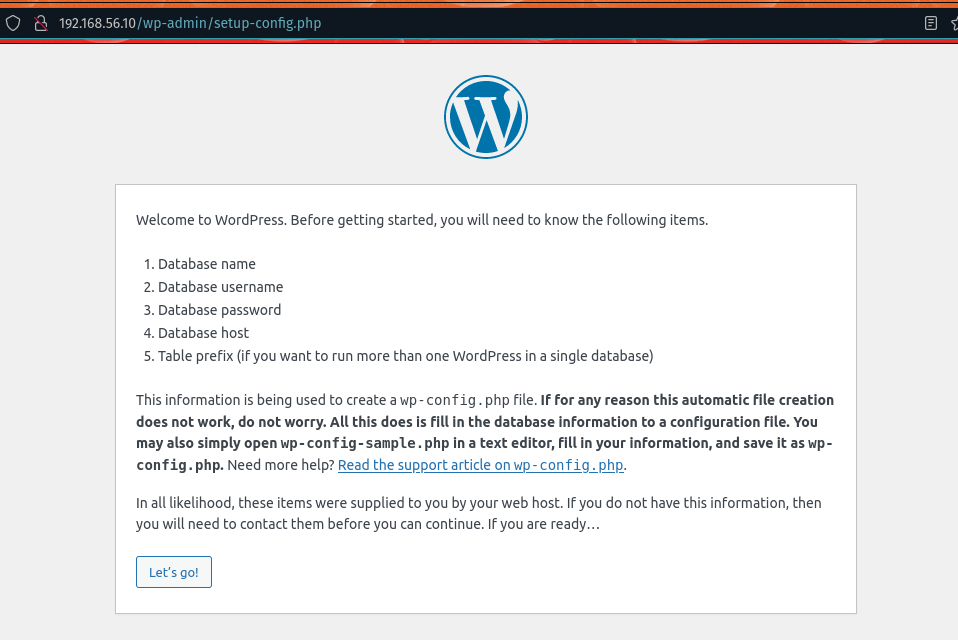
Kada izvučem sve fajlove trebao bi dobiti ovako nešto:

Slika 85 :Sardžaj wordpress direktorijuma

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

Slika 86 :Brisanje wordpress direktorijuma

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

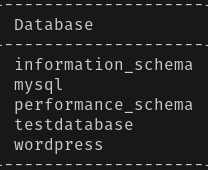
Slika 87 :Wordpress instalacioni prozor

Instalacija je prilično jasna I jednostavna tako da ću preskočiti detaljisanje o svakom koraku.

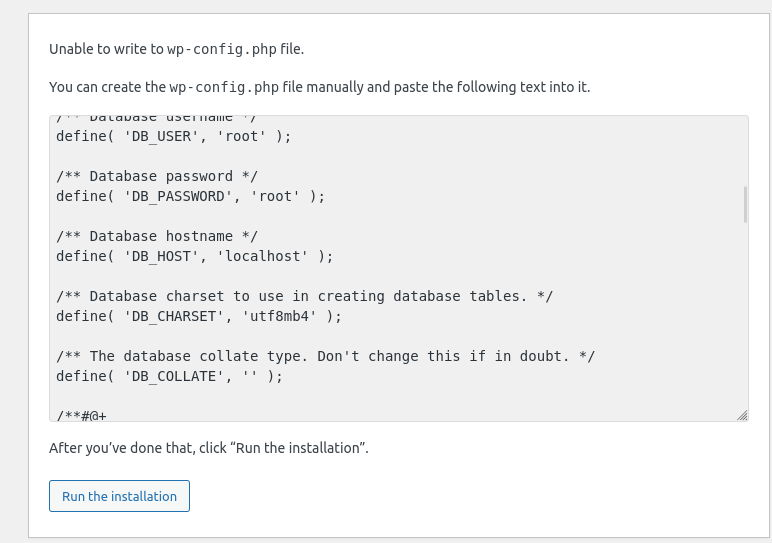
Morao sam pristupiti fajlu setup-config.php koji se nalazi u direktorijumu wp-admin kako bi vršio ovu instalaciju.

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 :

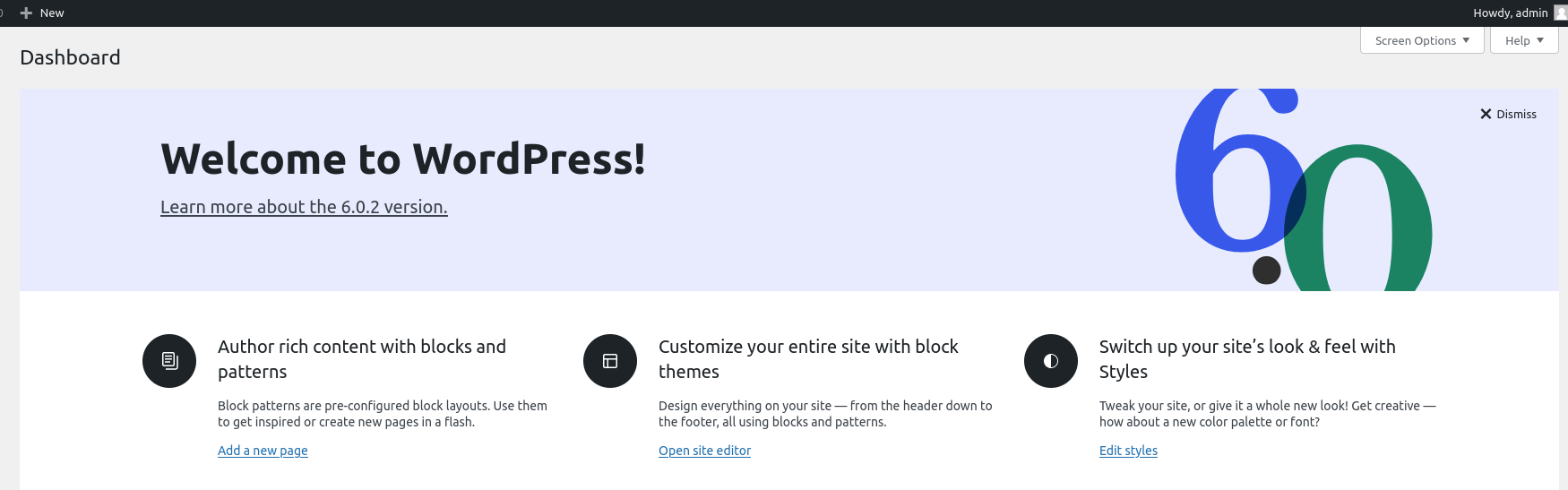
Slika 88 :Prikaz kreirane baze za wordpress

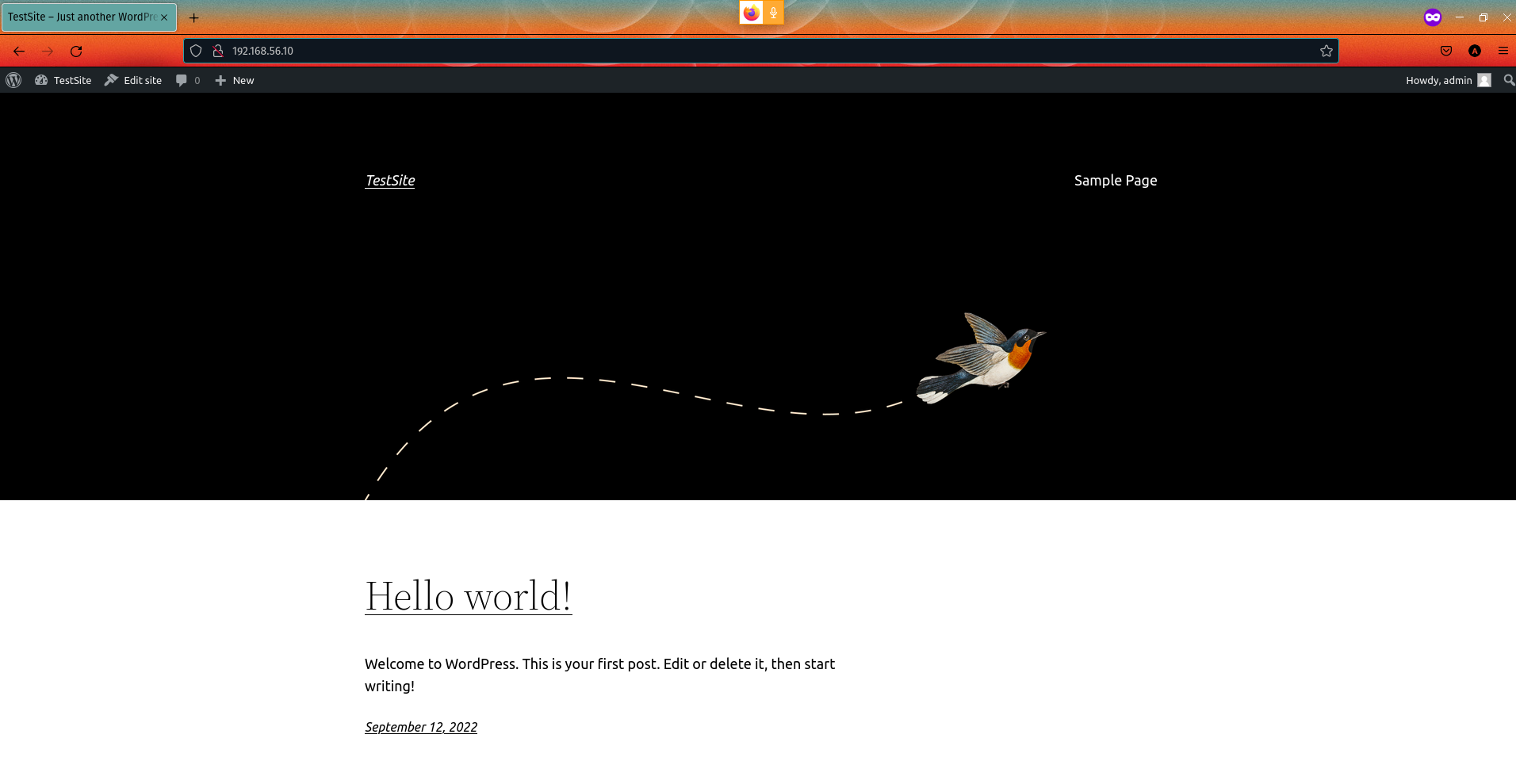
Sada dobijam šta ćemo uneti u wp-config.php

Slika 89 :wp-config.php fajl sadržaj

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 90 :Wordpress dashboard

Slika 91 :Prva stranica wordpress ili welcoming page

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 sam Linux kerenl napisan.  
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 1 M **home/alex/Videos** direktojuma find komandu da ne bi morao da stalno pišemo punu komandu, to bi izgledalo ovako:

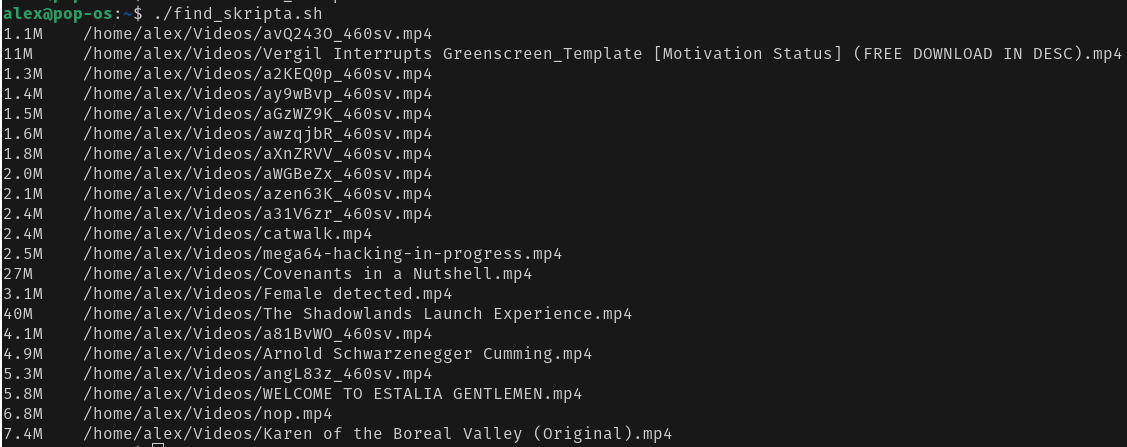
**#!/bin/bash  
# Find skripta**

**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 92 :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:

Skripta je uzela prvi argument $1 koji joj je prosleđen I oštampala ga je kako sam napisali komandu ispisa.

Slika 93 :Prikaz upotrebe argumenta

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

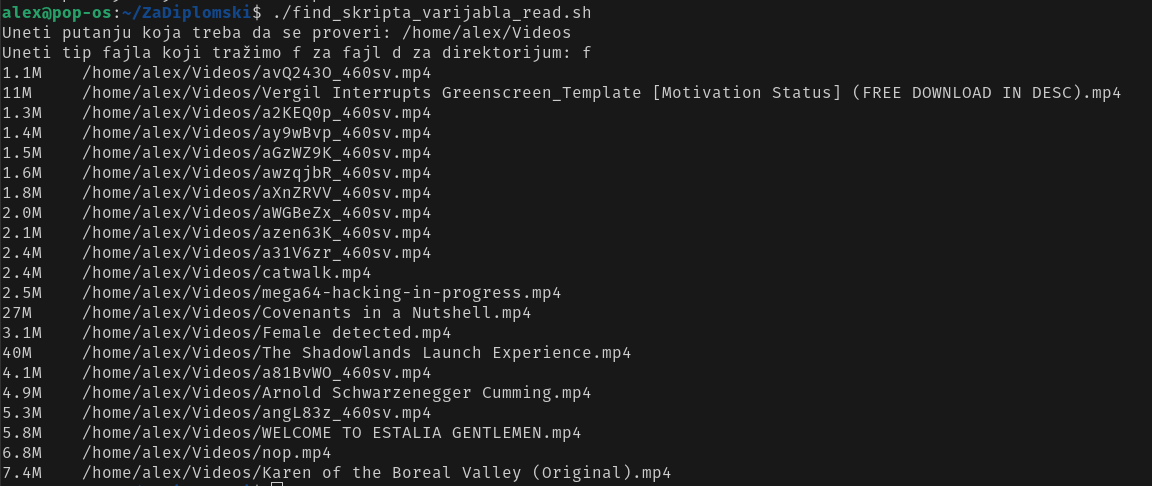
Koristiću prethodno napisanu skriptu sa find komandom kako bi predstavio upotrebu  
  
**#!/bin/bash**

**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 94 :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***](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:

|  |  |  |
| --- | --- | --- |
| 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](https://regex101.com/)

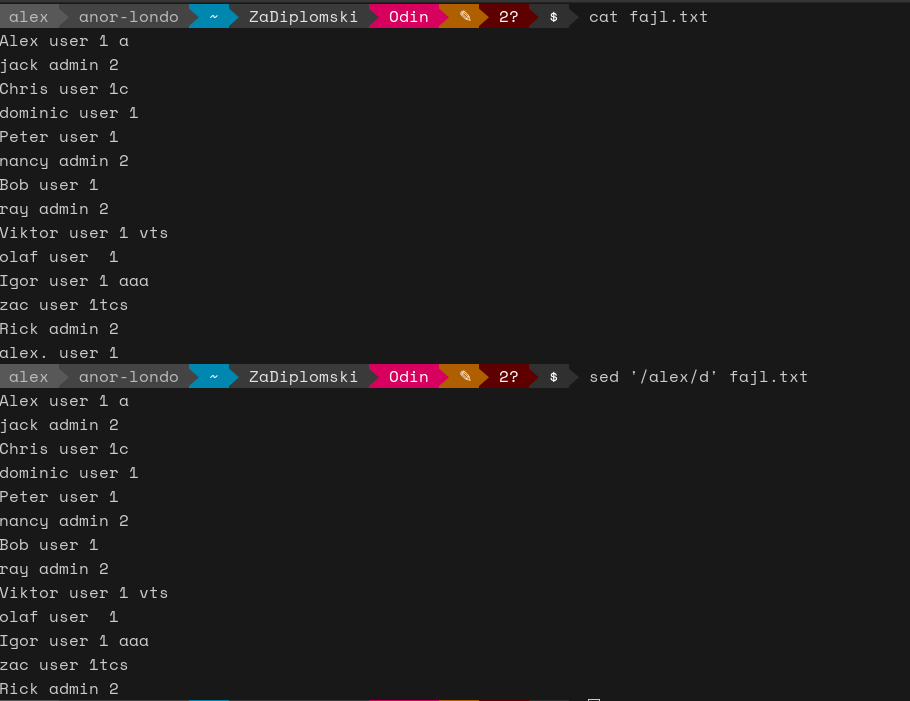
## 11.6 Sed

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

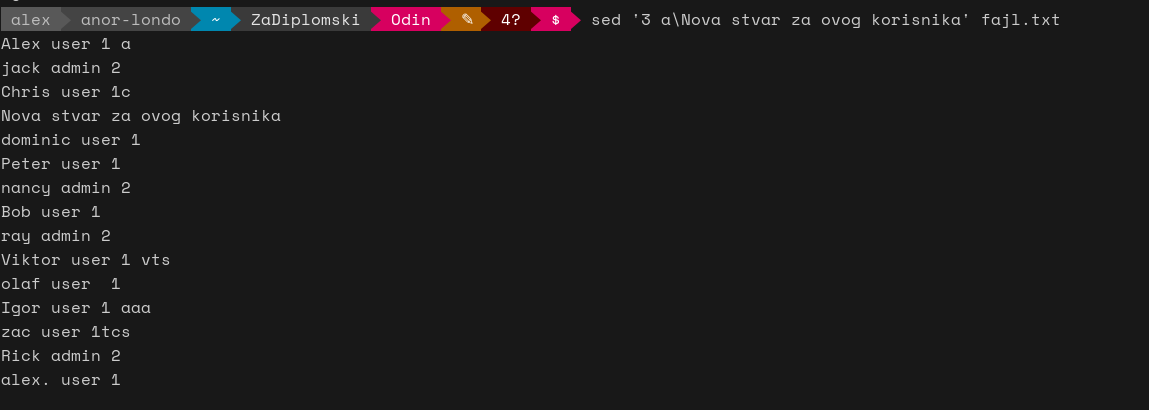
**$ sed ‘/alex/d’ fajl.txt**

Slika 95 :Brisanje imena iz fajla na standardnom izlazu upotrebom sed komande

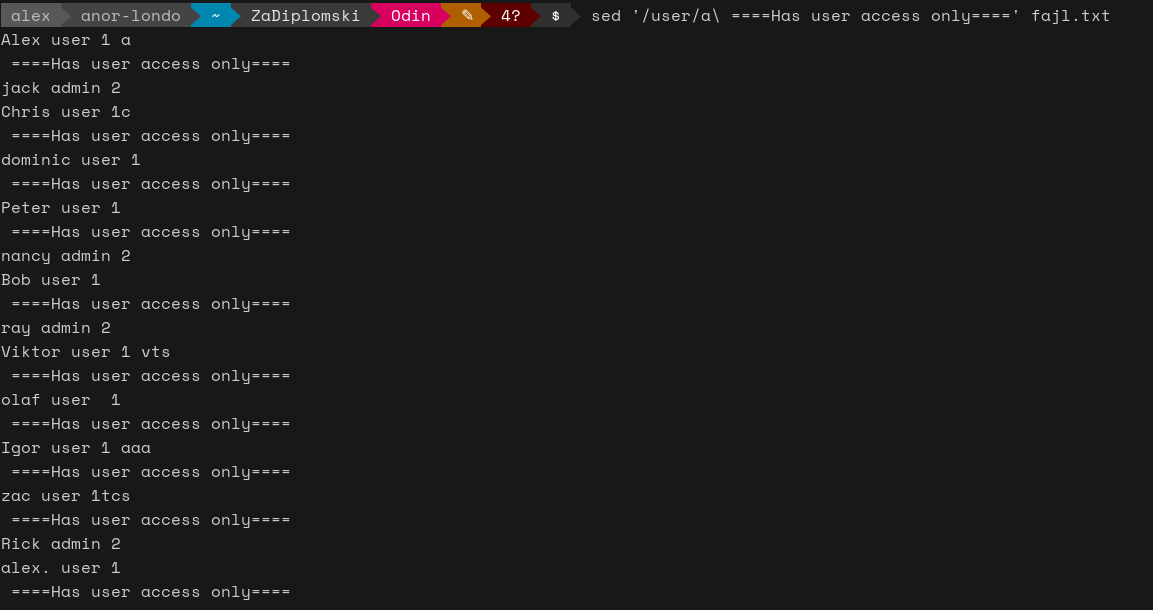
Može da se primeti da prethodno postoji korisnik alex. User 1 kada sam ga prikazao sa komandom **cat** nakon upotrebe **sed** ova linija više ne postoji.  
Naravno fajl je ostao ne izmenjen iz razloga što nisam **sed** komandi rekao to da uradi, tj u suštini smo oblikovali standardni izlaz po našoj želji.

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 96 :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.

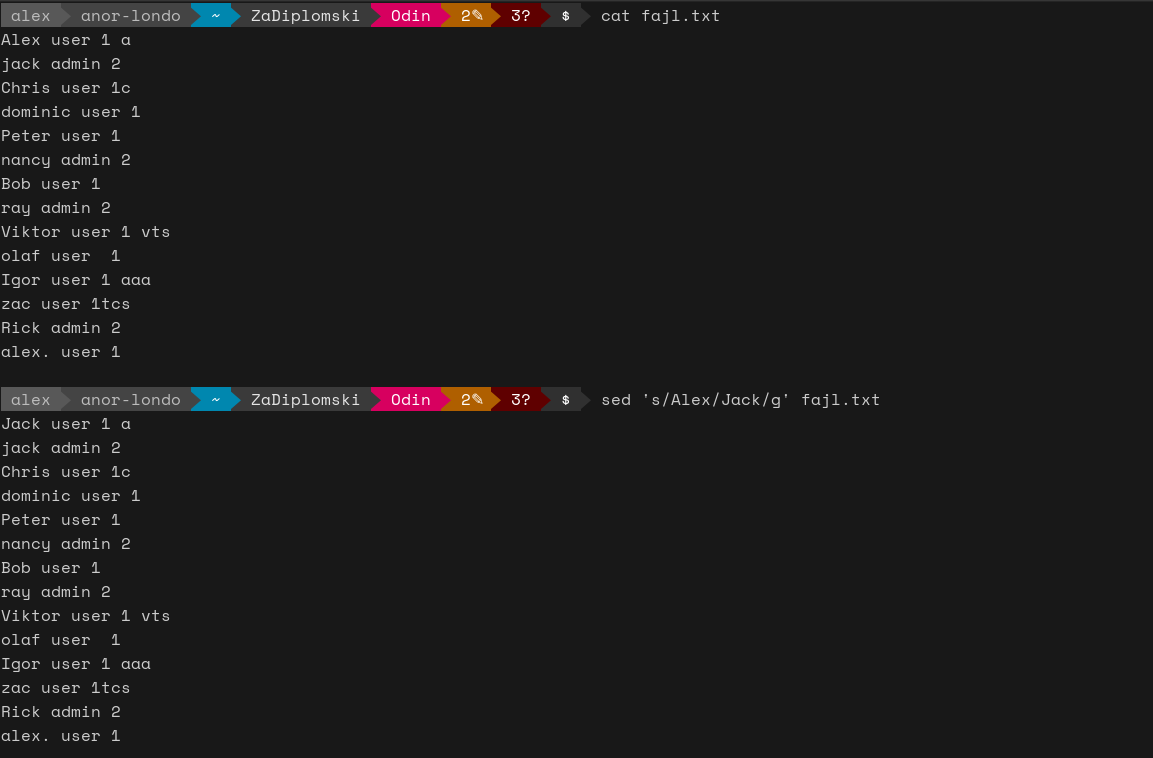
Slika 97 : Upotreba append parametra u sed sa pretragom reči

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 98 :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.

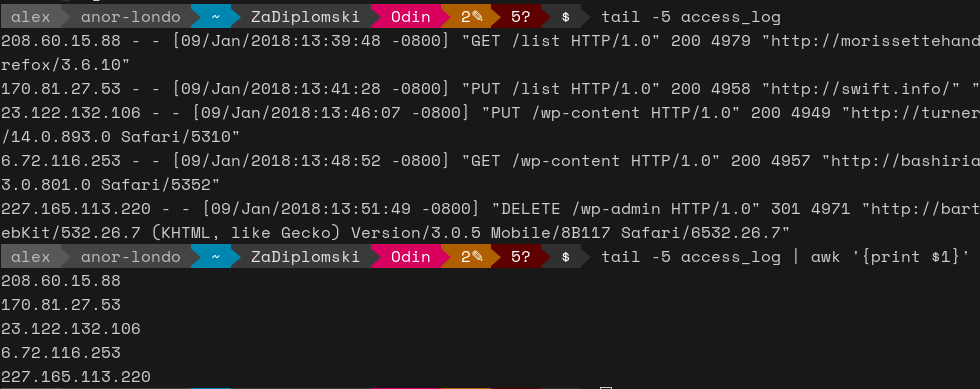
Ovo su samo neke od mogućnosti **sed**, pošto ova alatka može da se koristi I za skriptovanje njene mogućnosti su mnogo veće nego ovde navedene, koje samo predstavljaju mali deo onoga što može.

## 11.7 AWK

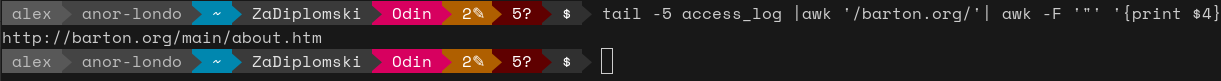
AWK je sopstveni programski jezik koje može da se izvršava u terminalu ili u skripti, prevalentno se koristi za obradu podataka I generisanje izveštaja.  
Pošto je awk programski jezik ja ću se dotaći samo nekih upotreba, pošto osnovna upotreba ne zahteva neko veliko programersko znanje.

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 99 :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 100 :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 proverimoda li je servis aktivan tako što ću koristi komandu **iptables -L** koja će mi dati informacije o aktivnim pravilima.

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

Slika 101 :Prikaz pravila u iptables

* **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 102 :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.

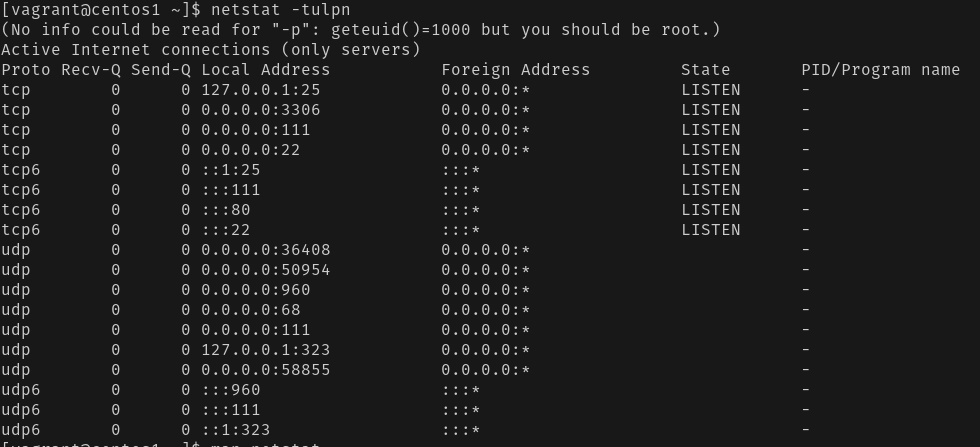
## 12.3 Nadgledanje sistema

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)

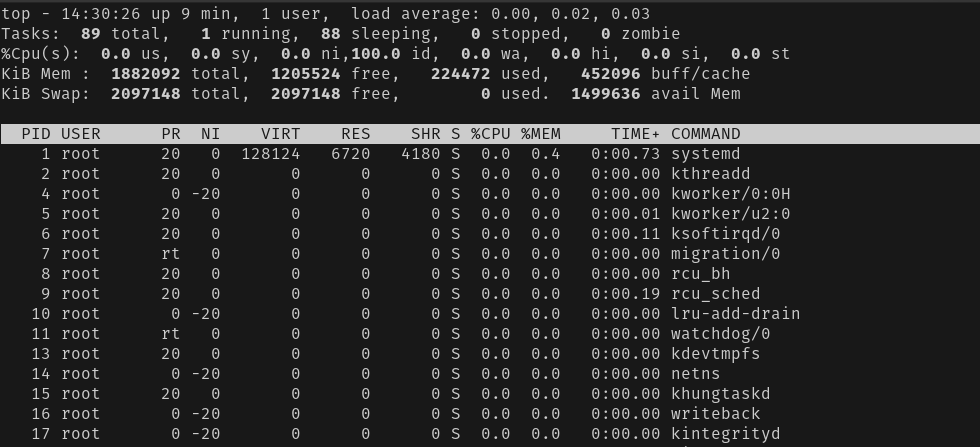
Slika 103 :Upotreba netstat komande

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)

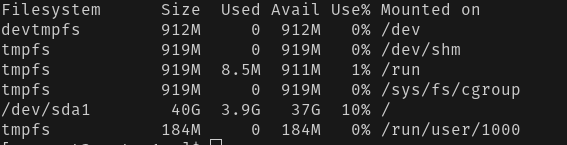
Još jedna vrlo čest komanda bi bila **top** koja nam daje informacije o svim aktivnim servisima

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 izuzetno korisno ako neki servis pravi problem pa mogu videti otprilike kako se sistem ponaša.

Slika 104 :Upotreba top komande

Često ćemo naići I na problem sa upotrebom disk prostora koji je takođe nešto što može da utiče na generalnu funkcionlanost servera.

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.

Slika 105 :Prikaz rezultata df komande

# 13. 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.

# 14. 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/*](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/*