Izrada vlastite Linux distribucije

Šikač, Benjamin Filip

Undergraduate thesis / Završni rad

2021

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Organization and Informatics / Sveučilište u Zagrebu, Fakultet organizacije i informatike**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:211:884030

Rights / Prava: Attribution 3.0 Unported/Imenovanje 3.0

Download date / Datum preuzimanja: **2024-03-29**

SVEUČILIŠTE U ZAGREBU
FAKULTET ORGANIZACIJE I INFORMATIKE
VARAŽDIN

Repository / Repozitorij:

Faculty of Organization and Informatics - Digital Repository

zir.nsk.hr

DIGITALNI AKADEMSKI ARHIVI I REPOZITORIJI

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Benjamin Filip Šikač

IZRADA VLASTITE LINUX DISTRIBUCIJE

ZAVRŠNI RAD

Varaždin, 2021.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Benjamin Filip Šikač

Studij: Informacijski sustavi

IZRADA VLASTITE LINUX DISTRIBUCIJE

ZAVRŠNI RAD

Mentor/Mentorica:

Doc. dr. sc. Marcel Maretić

Varaždin, rujan 2021.

Benjamin Filip Šikač

Izjava o izvornosti

Izjavljujem da je moj završni/diplomski rad izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

Autor/Autorica potvrdio/potvrdila prihvaćanjem odredbi u sustavu FOI-radovi

Sažetak

Cilj rada je opisati izgradnju distribucije Linuxa od nule, pritom opisujući korake koji su potrebni kao što je priprema koja obuhvaća izbor sustava iz kojeg se radi derivat i prilagođavanje tog sustava, nakon čega slijedi preuzimanje potrebnih paketa i stvaranje unakrsnog kompilatora koji omogućuje stvaranje privremenih alata i potpunog sustava. Nakon izrade se opisuje rezultat uz kritike na korisnost rada i na moguće alternative koje bi znatno ubrzale postupak, ali uz obrazloženje kako je postupak prikladan za učenje postupka izrade i postavljanja Linuxa te upoznavanja njegovih paketa. Povijesni pregled opisuje nastanak prvog Linuxa uz trenutno stanje tržišta. Osvrt na licenciranje prolazi kroz vrste licenca prema ograničenjima od najslobodnije do najstrože.

Ključne riječi: Linux, distribucija, vlastita distribucija, izgradnja, kompiliranje, licenciranje, izgradnja Linuxa

Sadržaj

1. Uvod 1
2. Povijest Linuxa 2
   2.1. Prva inačica Linuxa 2
   2.2. Razvoj distribucije i prihvaćanje tržišta 4
3. Aktualne distribucije 6
   3.1. Distribucije 6
   3.2. Debian 7
   3.3. Ubuntu 8
   3.4. Arch Linux 9
   3.5. Gentoo 10
   3.6. Fedora 11
   3.7. openSUSE 12
4. Izrada distribucije 13
   4.1. Zahtjevi 13
   4.2. Priprema radne okoline 14
   4.3. Izrada unakrsnog kompilatora 15
   4.4. Izgradnja privremenih alata 18
   4.5. Izgradnja nezavisnog sustava 19
   4.6. Postavljanje nakon izgradnje 23
   4.7. Dodatni programi za bolji rad 24
   4.8. Osvrt na rezultat i održavanje 25
5. Licenciranje 28
   5.1. Javna domena 28
   5.2. Dopustiva licenca 28
   5.3. Licenca slobode autorskog prava 28
   5.4. Vlasnička licenca 29
6. Zaključak 30
7. Literatura 32
8. Popis slika 35
9. Popis isječaka kodova 36

1. Uvod

Tema završnog rada je, kao što to sam naslov govori, izrada vlastite Linux distribucije. Rad se bavi kratkim pogledom u povijest Linuxa, opisom trenutnog stanja Linuxa na tržištu, prikazom izrade vlastite distribucije te nakon toga razmatranjem koliko je to zapravo korisno i koje zamjene postoje za sličan pristup izgradnji Linuxa. Na kraju slijedi osvrt na licenciranje i vrste licenca od otvorenih do vlasničkih.

Motivacija za izbor ove teme je sama znatiželja za Linux i način na koji radi. Kroz izradu rada sam naučio koliko stvari koje se automatski obavljaju uzimam zdravo za gotovo te koliko posla zahtjeva samostalno slaganje tih postavki.

Povijest Linuxa započinje životom njegovog kreatora Linusa Torvaldsa, te opisivanjem njegovog izlaganja računalnom svijetu što služi kao obrazloženje za stvaranje Linux sustava. Nastavlja se opisom stanja Linuxa na tržištu nakon čega se pogled usmjerava prema distribucijama koje se trenutno koriste.

Izrada distribucije u nekoliko koraka opisuje preduvjete za izgradnju distribucije, pojmove vezane uz izgradnju poput unakrsnog kompiliranja, izgradnju i opise paketa koji se pritom grade te postavljanje sustava za korištenje. Nakon toga se radi procjena postupka uz osvrt na slične alate koji pružaju jednostavniji proces postizanja sličnog cilja.

Licenciranje se bavi opisom licenca koje se nalaze na tržištu, počevši od najotvorenije pa sve do najstrože, a objašnjene su sljedećim rasporedom: licenca javne domene, dopustiva licenca, licenca slobodnog autorskog prava i na kraju vlasnička licenca.

2. Povijest Linuxa

Ovo poglavlje se bavi poviješću Linuxa i njegovim nastankom. Prvo slijedi priča o tome kako i zašto je uopće nastao Linux. Nakon toga se objašnjava njegov rast i prihvaćanje, ali i neki bitni ciljevi koje je sustav ostvario od nastanka do danas.

2.1. Prva inačica Linuxa

Linus Torvalds, tvorac Linuxa, se od malena bavio računalima. Njegov prvi susret s računalom je bio kalkulator. Prema njegovim opisima, stariji kalkulatori su prilikom izračunavanja sinusa bljeskali kako bi poručili korisniku da još uvijek rade na problemu; to se događalo jer tadašnji kalkulatori nisu bili napredni i brzi poput današnjih. Ta pojava je mladog Linusa izrazito zainteresirala za računala. [2]

Sljedeći korak u njegovom životu koji je uključivao računala se dogodio zahvaljujući djedu koji je bio profesor statistike na sveučilištu Helsinki. Računalo koje je djed nabavio je bilo Commodore VIC-20 (Slika 1.), on ga je koristio za izračune različitih algoritama pisanih u BASIC-u. Djed se nikad nije navikao na pisanje pomoću tipkovnice, stoga je sve programe pisao na papir s kojeg ih je Linus unosio u računalo. Nakon toga je počeo čitati upute za samo računalo gdje je bilo kodova za neke igre. Igre su bile jednostavnog karaktera, poput čovjeka koji je šetao s jedne strane ekrana na drugu, a pritisak tipke bi promijenio boju čovjeka. Nakon toga je počeo pisati vlastite programe i igrice, kao na primjer “hello world“ program. Kasnije, dok je nabavio Sinclair QL (Slika 2.), počeo se interesirati za operacijske sustave uz pomoć operacijskog sustava Q-DOS koji se nalazio na samom uređaju uz koji su došle i greške, odnosno bugovi, koji su spriječavali izvršenje drivera koje je sam Linus pisao. Neke od alata u samom operativnom sustavu je isto sam napisao u asembleru jer su postojeći bili spori. Između ostalog, njegov put pisanja vlastitih programa ga je doveo do toga da piše kopije igara koje je igrao na prethodnom računalu (npr. Pac man i Asteroids).[2][3][4][5][6][7]



Slika 1. Commodore VIC-20 [8]



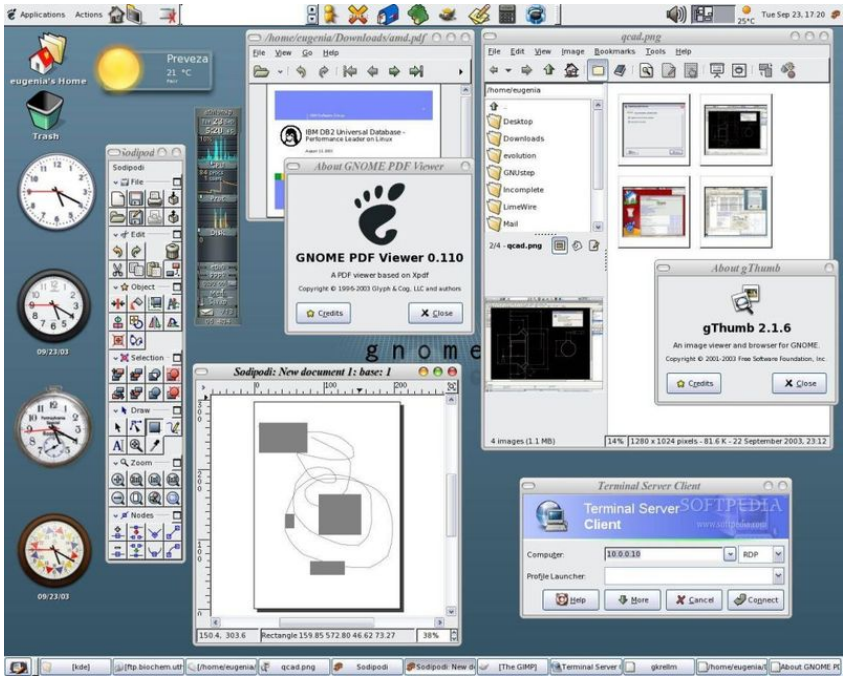
Slika 2. Sinclair QL. [9]

Nakon što mu je dosadio Sinclair QL, išao je u potragu za novim uređajem. U ovom slučaju, taj uređaj nema ime jer je bio sastavljen od komponenata u trgovini, odnosno nije bio markiran proizvod. Na uređaj je instalirao Minix sustav. Minix sustav je poput UNIX sustava (koji se sastoji od šest jednostavnih operacija, a to su dijeli (eng. *fork*), izvrši, otvori, zatvori, čitaj i piši); to znači da se ponaša poput UNIX sustava i nalikuje istom. Minix sustav je imao svoje probleme, kao na primjer emulacija terminala. Zbog nezadovoljstva postojećim emulatorom, napisao je svoj, uz pomoć kojeg bi pristupao internetu i školskoj mreži. Nije stao na tome, shvatio je da bi bilo korisno spremati podatke na disk s interneta te je stoga napisao driver za pohranu podataka, sličan onom u Minixu, kako bi mogao pristupati podacima u Minix sustavu. U tom trenutku je Linus shvatio da zapravo piše temelje za operacijski sustav, te se počeo raspitivati za POSIX dokumentaciju. Nakon nekoliko mjeseci je izdao prvu verziju Linuxa, u današnje doba bi se to zvalo alpha, i dao joj broj verzije 0.01. Verzija 1.0 je izdana u ožujku 1994. [2][3][4][5][6][7]

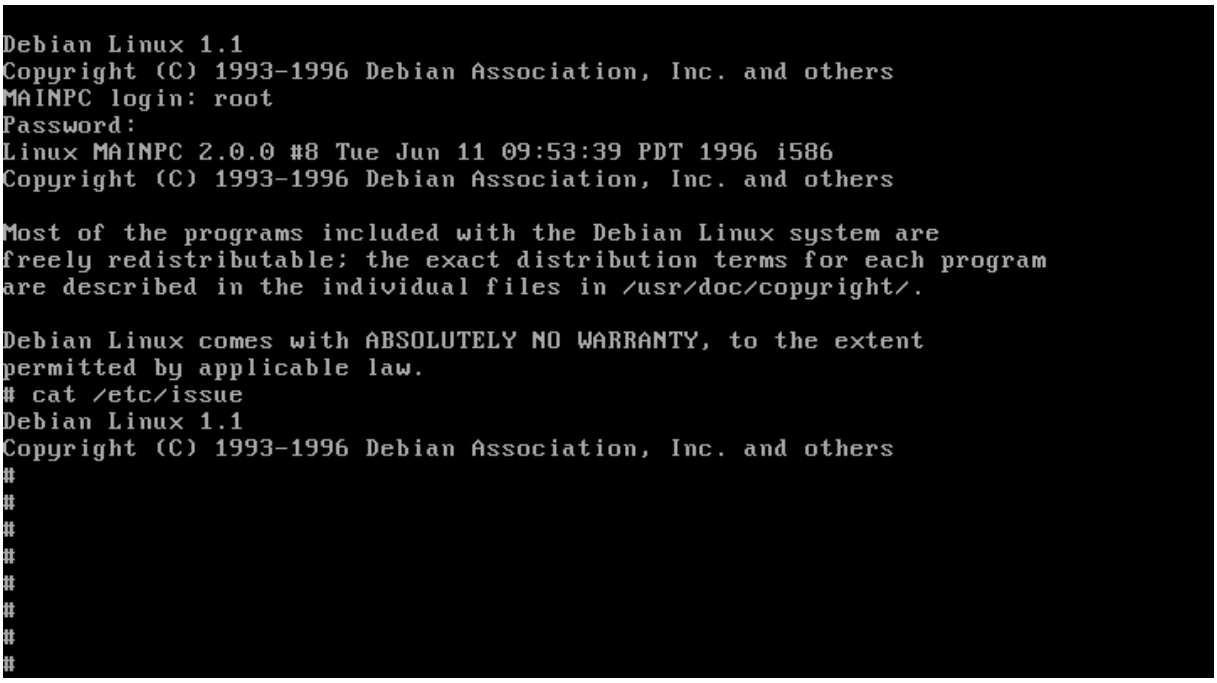
2.2. Razvoj distribucije i prihvaćanje tržišta

Distribucija Linuxa nema definiciju, no često se objašnjava kao inačica operacijskog sustava s istom jezgrom (eng. *kernel*) i istim upraviteljem paketa (eng. *package manager*).

Prva distribucija je nastala 1993. i zvala se Slackware (Slika 3.). Kasnije te iste godine, ustanovljen je projekt Debian (Slika 4.) koji se razvio u najveću distribuciju današnjice.[2]



Slika 3. Linux Slackware [10]



Slika 4. Debian Buzz. [11]

1996. godine je Linux počeo podržavati simetrično multiprocesiranje (eng. *symmetric multiprocessing, SMP*) što je značilo da je postao ozbiljan suparnik ostalim sustavima. To podupire činjenica da su 1998. godine velike tvrtke poput IBM-a, Compaqa i Oraclea službeno objavile podršku za Linux sustav. Za Linux je slijedila faza rasta, kroz koju je napravljeno i grafičko sučelje KDE, ali i kao odgovor KDE-u GNOME. Linux se do 2007. godine morao samostalno instalirati na računalo, no tvrtka Dell je to promijenila time što su počeli prodavati prve laptope s pred-instaliranim Linuxom. Sljedeći veliki korak je bio 2013. godine kada je Android sustav, koji je isto Linux, preuzeo 75% tržišta pametnih telefona. Samo godinu dana nakon preuzimanja tržišta pametnih telefona, Ubuntu je objavio kako je dosegao 22 milijuna korisnika. [12]

3. Aktualne distribucije

3.1. Distribucije

Aktualne distribucije iz kojih su najčešće izvedene ostale su Debian, Arch Linux, Ubuntu koji je sam izveden iz Debiana, Gentoo, Fedora i openSUSE.

Najbitnija stvar koja razlikuje distribucije je sama misija iza toga zašto su stvorene. Na primjer, Arch je distribucija koja želi biti najmodernija (eng. *bleeding edge*), stoga se u toj distribuciji cijeli sustav instalira samo jednom, te se postepeno nadograđuje, no nikada se ponovo ne instalira jer za time nema potrebe. S druge strane, Gentoo je sustav koji je stvoren kako bi bio izrazito fleksibilan za svoje korisnike i to omogućuje mogućnost instaliranja bilo kakvih paketa na distribuciju. [13][14]

Osim misije koju sustav ima, ono što ga ističe od ostalih distribucija je sustav upravljanja paketima, odnosno menadžer paketa (eng. *package manager*). Svaki sustav podržava svojeg menadžera, no neki jednostavno koriste postojeće menadžere jer je sam proces pisanja menadžera naporan, ne samo zbog toga što se mora pisati nego i zato što se paketi moraju prilagođavati i unositi u bazu podataka iz koje bi se preuzeli.

3.2. Debian

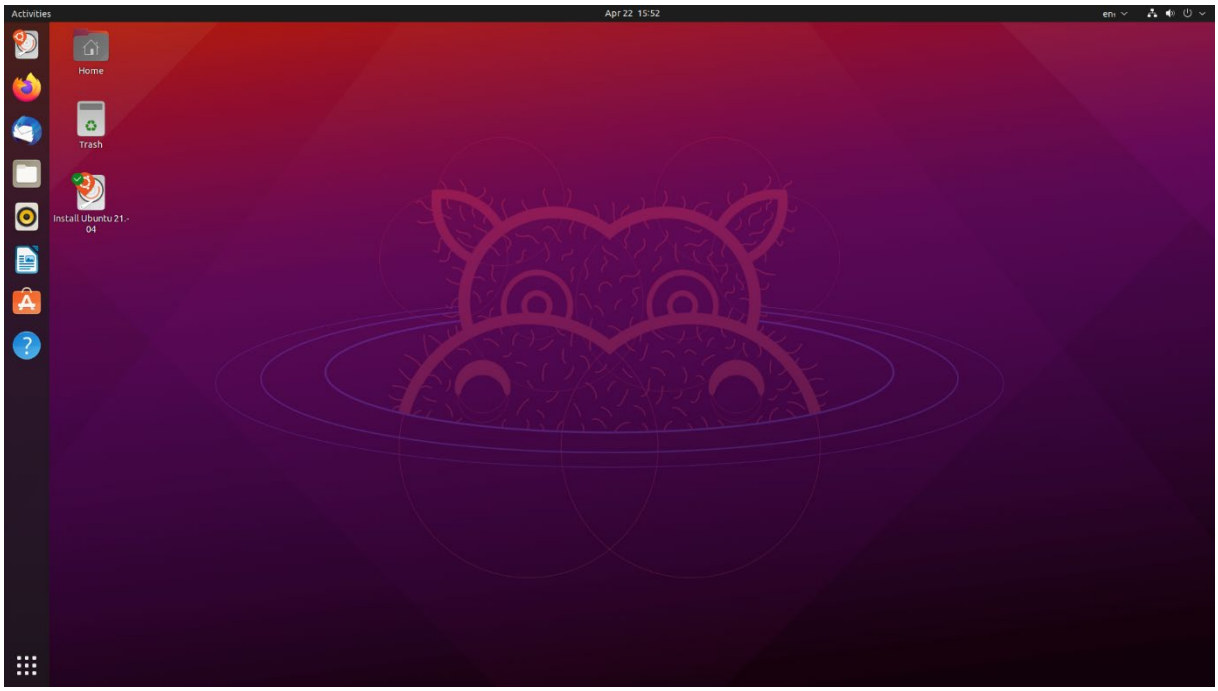
„The Debian Project“ je skupina ljudi koja je nastala kroz zajednički cilj za besplatnim operacijskim sustavom. Prethodni govor o misiji svake distribucije isto tako vrijedi i za ovu. Kao što je rečeno, cilj je besplatan OS, a to se sve osjeti u pristupu u kojem se uz OS dobije preko 50 tisuća besplatnih paketa. Uz to, jedan od najpopularnijih menadžera paketa je Debianov. Debian je distribucija koja je izvrsna za duže korisnike Linuxa i za one koji su već upoznati s nekim komandama koje se pojavljuju. Kao i većina ostalih Linuxa, dobar je za obične korisnike, ali i za serversku primjenu. Ono što ga razlikuje od ostalih je to što podržava najveći broj arhitektura procesora. Još jedna stvar koju podržava, a u današnje vrijeme je normalna za sve distribucije, je više grafičkih sučelja, od GNOME 2 (Slika 5.) sučelja do XFCE sučelja. [13][14][15]



Slika 5. Debian. [16]

3.3. Ubuntu

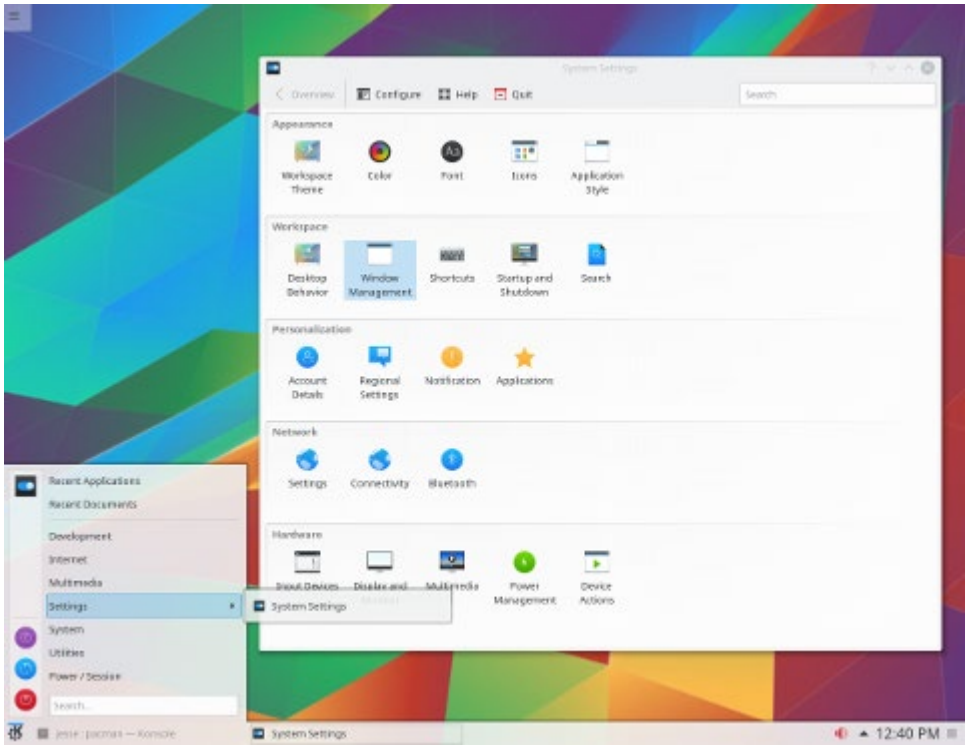
Ubuntu je jedna od najpopularnijih distribucija na svijetu, a tome pomaže njezina misija koja dijeli isto značenje riječi prema kojoj je dobila ime. „Ubuntu“ je stara afrička riječ koja u prijevodu znači „ljudskost drugima“ (eng. *humanity to others*). Ono što Ubuntu pruža tržištu je potpora zajednice, ali i profesionalaca, s ciljem da bude jedan od najpristupačnijih Linuxa za sve korisnike – dostupan svim korisnicima na njihovom materinjem jeziku, besplatan, prilagodljiv i pristupačan ljudima koji imaju invaliditet. S obzirom na to da je temeljena na Debianu, sadrži njegove karakteristike, no znatno pojednostavljuje neke stvari poput korištenja `sudo` komande koja uklanja potrebu za postojanje `root` korisnika. [13][14][17]



Slika 6. Ubuntu. [18]

3.4. Arch Linux

Arch Linux prati moto „keep it simple, stupid“ (KISS), odnosno pokušava sve stvari pojednostaviti, kao i instalaciju paketa za koje se govori kako bi se trebali instalirati samo jednom. Arch Linux nema verzije, već se kategorizira kao „rolling-release“, odnosno sve nadogradnje se mogu preuzeti u trenu kada se razviju i objave. Zbog toga je uvijek na aktualnoj strani, što je uvijek pogodno za sigurnost, no zbog toga se mogu dogoditi nepredviđena gašenja i greške. Konstantne nadogradnje se moraju pratiti i zbog toga je, između ostalog, ova distribucija namijenjena za naprednije korisnike. [13][14][19]



Slika 7. Linux Arch. [20]

3.5. Gentoo

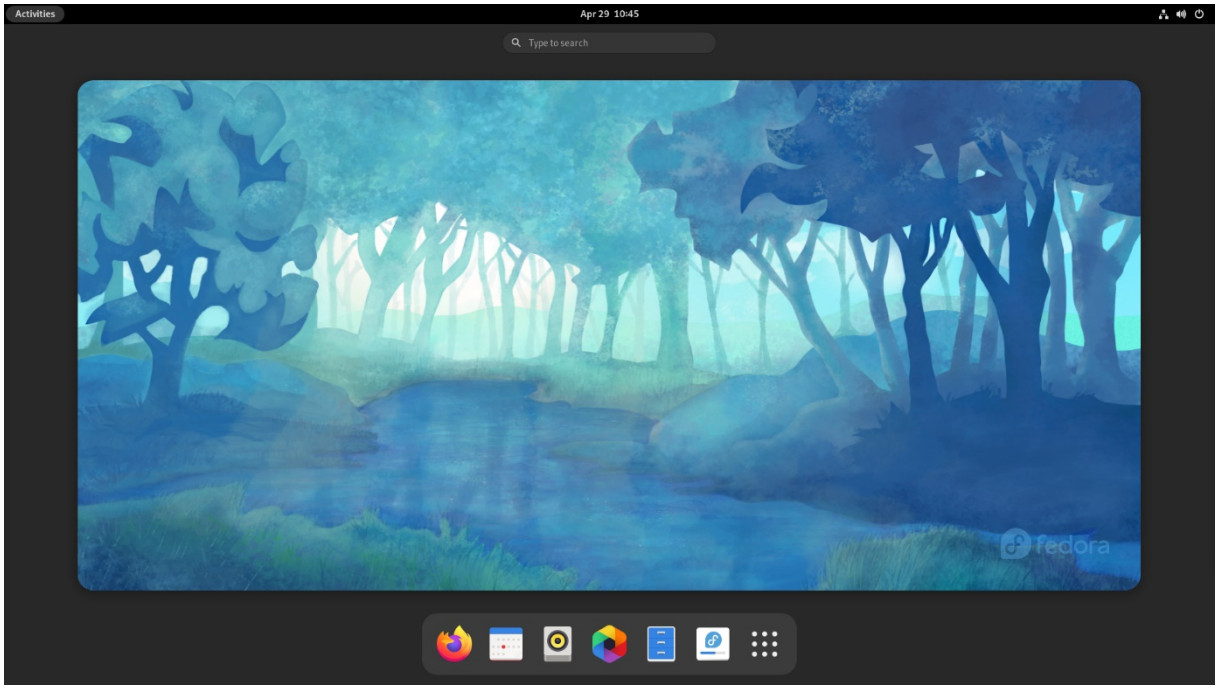
Gentoo je distribucija kojoj je namijenjena developerima i ljudima koji se razumiju u mreže, zbog čega je za korištenje potrebno veće predznanje i poznavanje Linuxa nego za neku distribuciju poput Ubuntua. Izrazito je fleksibilan i prilagodljiv, ali pod cijenu vremena. Sama instalacija nije toliko jednostavna i potrebno je predznanje za kompiliranje paketa. U usporedbi s izradom vlastitog Linuxa, Gentoo je svakako jednostavniji put za stvaranje sustava koji je skoro na istoj razini fleksibilnosti kao i vlastito izrađeni sustav. [13][14][21]



Slika 8. Gentoo. [22]

3.6. Fedora

Fedora je Linux koji se temelji na tome da bude besplatan, uz što pruža nove tehnologije koje su često razlog njegove popularnosti i dolaska novih korisnika, ali i odlaska starih koji ne žele nove tehnologije. Najbolji primjer je GNOME 3 sučelje koje je postalo standardno za distribuciju, a novo je. Za Fedoru je karakteristična riječ „spin“, a to je naziv koji se koristi za drugačije verzije iste distribucije s drugim pred-instaliranim aplikacijama. S obzirom na to da je Fedora proizvod tvrtke Red Hat, sam Linux naginje više u poslovnu stranu nego prema malim korisnicima, zbog toga što se koristi i za servere. [13][14][23]



Slika 9. Fedora. [24]

3.7. openSUSE

openSUSE je distribucija koja pokušava ostvariti tri stvari; biti Linux koji je najjednostavnije nabaviti i koji je najšire rasprostranjen, koristiti otvoreni kod (eng. *open source*) kako bi postao najkorisniji Linux za početnike i iskusne korisnike Linuxa te pojednostaviti proces pakiranja kako bi developeri i korisnici njihovih proizvoda došli koristiti taj Linux. No problem koji se spominje u razgovoru o openSUSE Linuxu je to što grafička sučelja usporavaju rad računala. [13][14][25]



Slika 10. OpenSUSE. [26]

4. Izrada distribucije

Ovo poglavlje se bavi glavnim dijelom rada, a to je izrada distribucije. Prvo se ukratko opisuje što je potrebno za izradu i kako će taj proces izgledati, nakon čega će se svaki od tih koraka detaljnije opisati.

Cijela distribucija je dostupna u dva izdanja. Prvu je moguće pokrenuti u virtualnom okruženju, a nalazi se na poveznici:

https://drive.google.com/file/d/12IGQJ4alUt2rLiwc2143J4Ikb_fDI8aG/view?usp=sharing

Drugu verziju je moguće pokrenuti na računalu bez virtualizacije, a nalazi se na poveznici:

https://drive.google.com/file/d/12IGQJ4alUt2rLiwc2143J4Ikb_fDI8aG/view?usp=sharing

4.1. Zahtjevi

Za izradu vlastite distribucije nije potrebno mnogo predznanja. Poznavanje kako se obavljaju pozivi u jezgru (eng. *kernel*) ili kako radi operacijski sustav nije potrebno, ono što rad opisuje su potrebe i postupak stvaranja distribucije Linuxa, što je poprilično drugačiji postupak od pisanja novog operacijskog sustava. Osoba koja se želi baviti ovim pothvatom onako kako je opisano u radu najvjerojatnije je ovdje za učenje, više o razlogu zašto će biti na kraju poglavlja.

Izrada će biti puno brža ako su neke stvari poznate od prije, kao na primjer:

- kako se koristi terminal (ovo se odnosi na komande za navigaciju poput `cd`, ili pak komande za stvaranje, brisanje i uređivanje datoteka)
- shvaćanje kako izgleda Linuxov raspored podataka, sistemskih datoteka i particija (particije su prilično bitne za ovaj proces)
- root korisnik koncept
- izgradnja paketa (ovo se ne odnosi na menadžer paketa (eng. *package manager*), nego na doslovno kompiliranje paketa, odnosno alata)

Sada kada su objašnjeni zahtjevi koji su korisni za štednju vremena, slijedi kratak opis izgradnje. Izrada započinje dijeljenjem (eng. *partitioning*) diska, tako što se stvara particija od nekoliko GB. U ovom slučaju to je bilo 20 GB, no može biti daleko manje ili više, ovisno o tome za koju potrebu se stvara Linux; potrebe mogu biti cijeli sustav za kućnu uporabu ili pak server

koji poslužuje klijente za koji nije potrebno grafičko sučelje. Ključan dio je provjera verzije svih alata koji će biti korišteni za kompiliranje i sastavljanje novog sustava. Ako nisu prisutni na sustavu, onda se moraju preuzeti i instalirati uz pomoć naredbe `sudo apt install`. Zatim se stvara mapa na koju će se montirati (eng. *mount*) particija u koju će biti preuzeti svi potrebni paketi koje će se kasnije ručno raspakirati i graditi. Paketi će prvo ovisiti o sustavu domaćina (eng. *host*) koji će posuditi svoje alate kako bi se dobar dio privremenih alata izgradio. Ovaj se korak odnosi na privremene alate i na unakrsni kompilator (eng. *cross-compiler*). Nakon što su napravljeni privremeni alati, njih ćemo koristiti kako bi u odvojenom sučelju izgradili prave alate i kompilator za sustav koji će sustav kasnije samostalno koristiti kako bi ponovo napravio alate u potpunosti. Predzadnji korak je dodavanje alata za pokretanje, odnosno upravljanje particijama. U ovom slučaju će to biti GRUB. Zadnji korak je preuzimanje i instaliranje dodatnih paketa kako bi sustav bio uporabljiv. [1]

Prema opisanom redu će se raščlanjivati koraci, a na kraju će se nalaziti osvrt izrade distribucije i korisnosti sustava.

4.2. Priprema radne okoline

Priprema počinje izborom distribucije od koje će biti stvorena nova distribucija. U ovom radu je temelj Debian zato što je relativno jednostavna za početnike u usporedbi s ostalim distribucijama poput Arch Linuxa ili Fedore. Ono što je dodatna prednost ovog izbora je to što je glavna literatura pisana za Debian sustav, stoga se komande ne moraju ponovo pisati ili prilagođavati drugom sustavu.

U operativnom sustavu je potrebno provjeriti postoje li dodatna ažuriranja za bilo koji program komandom `apt upgrade`. Ako postoji, potrebno je obaviti nadogradnju. U protivnom se nastavlja sa skidanjem alata koji su potrebni za izradu. Od svih alata koje knjiga navodi kao potrebne, samo su tri nedostajala; to može biti drugačije ovisno o sustavu o kojem se radi i o tome je li novo instaliran ili prethodno korišten. Provjera koji alati nedostaju se obavlja uz pomoć skripte koja je priložena u knjizi. [1]

Sljedeći korak, za koji bi bilo dobro imati osnovno znanje, je stvaranje particije. Stvorena particija je u ovom slučaju bila `sda3` (`sda1` je glavna particija, a `sda2` swap particija) u veličini od 20 GB uz pomoć alata `cfdisk` (za upute korištenja alata upisati `man cfdisk` u terminal). Za ovaj projekt ne postoji potreba za korištenje i stvaranje swap particije, stoga se koristi samo glavna particija odnosno `sda3`. Montiranje se vršilo na lokaciji `/mnt/lfs`.

Nakon montiranja slijedi stvaranje direktorija koji su potrebni za rad Linux sustava (`bin, etc, lib, sbin, usr, var`). Uz to je stvoren i `lfs` korisnik kojem je dodijeljen

cijeli `lfs` direktorij i svi njegovi poddirektoriji. Korisnik `lfs` je stvoren zbog sigurnosti sustava domaćina jer neke operacije poput `make` i `make install` mogu trajno oštetiti domaćina ako se izvršavaju u ulozi `root` korisnika; najbolji primjer je sama instalacija paketa - ako se pokreće kao `root` korisnik, ne postoji ograničenje koje bi zaustavilo `root` korisnika i reklo da je nedopustiva akcija s obzirom na to da `root` korisnik sadrži sva prava.

Zadnji korak je preuzimanje paketa pomoću kojih će se graditi unakrsni kompilator, privremeni sustav i pravi sustav. Paketi se preuzimaju u direktorij `sources` koji je isto u `lfs` direktoriju. Zatim slijedi pregled jesu li svi paketi preuzeti. Zbog konstantnih unaprjeđenja i objavljivanja novih verzija, neki paket možda više nije moguće preuzeti s poveznice (eng. *link*) koja je se nalazi u knjizi. Do sada je samo jedan paket nedostajao te se je morao preuzeti „ručno“ uz samostalno traženje korektne poveznice i preuzimanje. [1]

Instaliranje ili izgradnja paketa se odvija tako da se paket raspakira, zatim uđe u njegov raspakiran direktorij te se, prema potrebi konfigurira za izgradnju, gradi i instalira. Za kraj se izlazi iz paketa i briše njegov direktorij iz kojeg je instaliran, to jest, onaj koji je na početku raspakiran. Ovaj proces će biti korišten za izgradnju svih paketa u nastavku. [1]

Vrijeme izrade paketa nije zgodno računati u satima ili minutama zbog toga što se to vrijeme mijenja ovisno o snazi računala koje gradi pakete, stoga se koristi vrijeme koje je potrebno za izgradnju prvog paketa `binutils` kao mjerna jedinica, a zove se standardna jedinica izgradnje (eng. *Standard Build Unit, SBU*). U slučaju virtualnog računala koje radi na 1 procesorskoj jezgri uz 4 GB radne memorije, vrijednost jednog SBU je 2 minute i 34 sekunde.

4.3. Izrada unakrsnog kompilatora

Prije nego što je moguće objasniti izradu unakrsnog kompilatora, potrebno je pojasniti što je unakrsni kompilator, kako se razlikuje od izvornog kompilatora (eng. *native compiler*) i za što se koristi.

Prilikom izrade programa na vlastitom računalu koji je namijenjen za isto računalo potrebno je kompiliranje programa prije pokretanja (ovo ne uključuje jezike koji su interpretirani s obzirom na to da za njih cijela priča unakrsnog kompilatora ne vrijedi zbog same interpretacije). Nakon što se program kompilira, on je preveden u asebmler koji odgovara arhitekturi procesora računala na kojem je kompiliran. Taj postupak se zove izvorno kompiliranje. [1][32][33]

Ako pretpostavimo da je arhitektura vlastitog računala x86-64 te prethodno napisan program pokušamo pokrenuti na drugom računalu koje nema istu arhitekturu procesora, poput ARM procesora, program neće raditi zato što druga arhitektura procesora sadrži drugačiji skup

naredbe. Zbog jednostavnije predodžbe je napisan primjer u jeziku C++, koji računa kvadrat broja. Za shvaćanje razlike u naredbenim skupovima nije potrebno poznavanje asembler jer se radi o jednostavnoj usporedbi riječi.

Primjer koda izgleda ovako:

```
int square(int num) {
    return num * num;
}
```

Isječak koda 1. Primjer funkcije za računanje kvadrata broja pisan u jeziku C++.

Primjer programa preveden u asembler arhitekture x86-64 izgleda ovako:

```
square(int):                              # @square(int)
        pushq   %rbp
        movq    %rsp, %rbp
        movl    %edi, -4(%rbp)
        movl    -4(%rbp), %eax
        imull   -4(%rbp), %eax
        popq    %rbp
        retq
```

Isječak koda 2. Primjer funkcije za računanje kvadrata broja prevedene u instrukcijski skup procesora x86-64.[35]

Isti program preveden u asembler arhitekture ARM izgleda ovako:

```
square(int):
        sub     sp, sp, #4
        str     r0, [sp]
        ldr     r0, [sp]
        mul     r1, r0, r0
        mov     r0, r1
        add     sp, sp, #4
        bx      lr
```

Isječak koda 3. Primjer funkcije za računanje kvadrata broja prevedene u instrukcijski skup procesora ARM.[35]

Kao što je moguće uočiti, radi se o potpuno drugačijim skupovima naredbi. Razlike naredbenih skupova su prvi i najveći razlog za korištenje unakrsnog kompiliranja. ARM procesor nije u stanju shvatiti primjer programa preveden u x86-64, no unakrsnim kompiliranjem se isti program prevodi u naredbe koje su mu shvatljive. Ovo je najvažniji razlog korištenja unakrsnog kompilatora, ali ne jedini. [1][32][33]

Za unakrsno kompiliranje su bitna tri pojma [1][32][33]

- Graditelj (eng. *build*) je računalo na kojem gradimo program. U radu se ovo računalo naziva domaćinom.

- Domaćin (eng. *host*) je računalo na kojem će izgrađeni paketi biti pokrenuti. Razlikuje se od domaćina koji se koristi u ostatku rada.
- Meta (eng. *target*) odnosi se isključivo na kompilatore. Računalo za koje kompilator proizvodi kod. Koristi se samo ako se radi unakrsni kompilator.

Uz objašnjene ključne pojmove, moguće je razjasniti drugi razlog zašto se koristi unakrsni kompilator, a to je brzina. Računala graditelji na kojima se pišu programi su bogatija memorijom, prostorom i procesorskom brzinom od računala domaćina za koje su programi namijenjeni, te je zbog toga isplativije koristiti graditelja za kompiliranje programa za domaćina jer bi domaćinu u protivnom trebalo puno više vremena. Čest primjer iz svakodnevice je modem za internet koji nema dovoljno resursa kako bi sam izgradio svoj sustav, stoga se to radi na graditelju uz pomoć unakrsnog kompilatora. [1]

Kao što se može zaključiti, unakrsno kompiliranje se može koristiti za stvaranje sustava koji je neovisan o graditelju. Za stvaranje novog unakrsnog kompilatora, graditelj će služiti kao domaćin, no meta će biti nova distribucija. S obzirom na to da se radi o neovisnoj distribuciji, potrebno je stvoriti izvorni kompilator koji će distribucija koristiti, a to će se izvesti uz pomoć prethodno stvorenog unakrsnog kompilatora kojem je graditelj domaći sustav iz kojeg gradimo novu distribuciju, domaćin nova distribucija i meta nova distribucija. Na kraju kada je stvoren izvorni kompilator uz pomoć unakrsnog, ponovo se kompilira izvorni kako bi se provjerilo postoje li greške. [1]

Paketi koji su potrebni za izgradnju unakrsnog kompilatora su [1]:

- `binutils, 1 SBU-` paket koji sadrži povezivač (eng. *linker*), asembler i ostale alate koji su potrebni za kompiliranje većine paketa koji će biti spomenuti u nastavku.
- `gcc, 11 SBU-` skraćenica od Gnu kolekcija kompilatora (eng. *Gnu Compiler Collection*), a sadrži C i C++ kompilatore.
- `linux api headers iz paketa linux, 0.1 SBU-` paket koji sadrži sučelje za programiranje aplikacija (eng. *application programming interface* – u nastavku API) koje je potrebno za glibc.
- `glibc, 4.2 SBU–` sadrži glavnu C biblioteku bez koje Linux programi ne rade.
- `libstdc++ iz paketa GCC, 0.4 SBU–` biblioteka za C++ koja je potrebna za kompiliranje C++ koda.

4.4. Izgradnja privremenih alata

Uz pomoć prethodno izgrađenog unakrsnog kompilatora možemo izgraditi privremene alate koji će biti potrebni za izoliranu izgradnju potpunog sustava. Izgradnja se vrši u dva koraka; prvo se grade alati koji su potrebni za izgradnju samostalnog sustava, a potom se grade dodatni alati i alati za ispitivanje, nakon prelaska u djelomično izoliran sustav (djelomično jer se koristi domaćinova jezgra). [1]

Prvi korak sadržava kompiliranje i izgradnju paketa (paketi koji se ponovo grade nemaju opis) [1]:

- `m4, 0.1 SBU` – sadrži makro procesor. Makro je odsječak koda koji je dobio ime. Kada se ime koristi se zamijeni s odsječkom koda, to obavlja makro procesor.
- `ncurses, 0.7 SBU` – biblioteka koja rukuje zasebnim upravljanjem znakova ekrana. Preduvjet za mnoge pakete u nastavku.
- `bash, 0.4 SBU` – Bourne Shell sučelje.
- `coreutils, 0.5 SBU` – osnovni programi za pregledavanje i uređivanje direktorija ili datoteka.
- `diffutils, 0.1 SBU` – programi koji pokazuju razlike između direktorija ili datoteka.
- `file, 0.2 SBU` – program koji određuje tip datoteke/a. Preduvjet za neke pakete.
- `findutils, 0.2 SBU` – program koji se koristi za traženje podataka u sustavu. Koriste ga mnoge skripte za izgradnju.
- `gawk, 0.2 SBU` – program koji se koristi za uređivanje tekstualnih datoteka. Koristi ga mnoge skripte za izgradnju.
- `grep, 0.2 SBU` – program koji se koristi za pretragu kroz datoteke. Koriste ga mnoge skripte za izgradnju.
- `gzip, 0.1 SBU` – program za pakiranje i raspakiranje datoteka.
- `make, 0.1 SBU` – program koji upravlja izgradnjom paketa. Koristi ga skoro svaki paket u nastavku.
- `patch, 0.1 SBU` – program koji se koristi za instalaciju zakrpi.
- `sed, 0.1 SBU` – program koji omogućuje uređivanje teksta bez otvaranja datoteke u tekstualnom uređivaču.
- `tar, 0.2 SBU` – program koji se koristi za arhiviranje i raspakiranje arhiva. Koriste ga svi paketi navedeni u radu.

- `xz, 0.1 SBU` – programi za raspakiravanje i pakiranje datoteka.
- `binutils, 1.3 SBU`
- `gcc, 11 SBU`

Drugi korak podrazumijeva stvaranje osnovnih datoteka, nakon ulaska u djelomično izoliran sustav pomoću `chroot` komande, poput `/etc/group`, `/etc/passwd`, `/etc/mtab` i `/etc/hosts`, nakon čega slijedi izgradnja ostalih paketa [1]:

- `libstdc++ iz paketa GCC, 0.8 SBU`
- `gettext, 1.8 SBU` – programi i biblioteke za internacionalizaciju i lokalizaciju paketa.
- `bison, 0.3 SBU` – GNU verzija yacc-a (eng. *yet another compiler compiler*).
- `perl, 1.7 SBU` – interpretator za PERL jezik.
- Python 3, `0.9 SBU` – razvojno okruženje za jezik Python.
- `texinfo, 0.3 SBU` – programi za čitanje, pisanje i pretvorbu informativnih stranica. Koriste ga mnogi paketi u radu.
- `util-linux, 0.7 SBU` – sadrži razne programe kao na primjer upravljanje particijama, porukama, konzolom.

4.5. Izgradnja nezavisnog sustava

Sada će biti izgrađen nezavisan sustav uz pomoć privremenih alata i unakrsnog kompilatora. Neki od alata koji će biti korišteni za taj proces će biti ponovo izgrađeni kako bi bili potpuni. Proces instalacije je isti kao i u prethodnim slučajevima, stoga je ovaj dio već poznat i brži nego prije. Paketi koji će biti instalirani su [1]:

- `man-pages, manje od 0.1 SBU` – upute za osnovne Linux alate.
- `iana-etc, manje od 0.1 SBU` – podaci za mrežne servise i protokole.
- `glibc, 19 SBU`
- `zlib, manje od 0.1 SBU` – rutine za pakiranje i raspakiranje koje neki programi koriste.
- `bzip2, manje od 0.1 SBU` – programi za pakiranje i raspakiranje podataka.
- `xz, 0.2 SBU`
- `zstd, 1.1 SBU` – rutine za pakiranje i raspakiranje koje neki programi koriste.
- `file, 0.1 SBU`

- readline, 0.1 SBU – biblioteke koje pružaju uređivanje u naredbenom sučelju i povijest sučelja.
- m4, 0.4 SBU3
- bc, manje od 0.1 SBU – (eng. *basic calculator*) jednostavan jezik za precizno procesiranje brojeva.
- flex, 0.4 SBU – program za stvaranje programa koji prepoznaju uzorke u tekstu. Preduvjet za izgradnju nekih paketa.
- tcl, 3.8 SBU – (eng. *tool command language*) jezik za ispitivanje.
- expect, 0.2 SBU – program za skriptirani dijalog između drugih interaktivnih programa.
- dejaGNU, manje od 0.1 SBU – okvir za ispitivanje programa.
- binutils, 6.2 SBU
- gmp, 1 SBU – biblioteke za preciznu matematiku. Preduvjet za potpun gcc.
- mpfr, 0.8 SBU – funkcije za aritmetiku s više preciznosti. Preduvjet za gcc.
- mpc, 0.3 SBU – funkcije za aritmetiku kompleksnih brojeva. Preduvjet za gcc.
- attr, manje od 0.1 SBU – programi za dodavanje proširenih atributa na objekte datotečnog sustava.
- acl, 0.1 SBU - programi za primjenu kontroliranih listi pristupa (eng. *access control lists*) za preciznija ograničenja nad direktorijima i datotekama.
- libcap, manje od 0.1 SBU - implementira korisnička prava.
- shadow, 0.2 SBU – programi za upravljanje šiframa na siguran način.
- gcc, 95 SBU
- pkg-config, 0.3 SBU – program koji vraća metapodatke o instaliranim paketima ili bibliotekama.
- ncurses, 0.4 SBU
- sed, 0.5 SBU
- psmisc, manje od 0.1 SBU – programi za prikaz informacija o programima koji rade.
- gettext, 2.9 SBU
- bison, 6.4 SBU
- grep, 0.8 SBU
- bash, 1.6 SBU
- libtool, 1.6 SBU – potpora za GNU opće biblioteke. Koriste je ispitni paketi (eng. *test suit*).

- gdbm, 0.2 SBU – biblioteka za GNU menadžer baze podataka (eng. *GNU database manager*). Preduvjet za man-db paket.
- gperf, manje od 0.1 SBU – program koji generira savršenu hash funkciju iz seta ključeva. Preduvjet za eudev paket
- expat, 0.1 SBU – biblioteka za raščlanjivanje (eng. *parsing*) XML-a. Preduvjet za XML::Parser
- inetutils, 0.3 SBU – program za osnovno upravljanje mrežom.
- perl, 10 SBU
- XML::Parser, manje od 0.1 SBU – perl modul koji se povezuje s paketom expat.
- intltool, manje od 0.1 SBU – alati za raspakiravanje prevodljivih nizova iz izvornih datoteka.
- autoconf, manje od 0.1 SBU, s testovima 7.2 SBU – programi za automatsku izradu shell skripti iz programerovog predloška koje uređuju izvorni kod.
- automake, manje od 0.1 SBU, s testovima 9.1 SBU – programi za generiranje make datoteka iz predloška.
- kmod, 0.1 SBU – programi potrebni za primjenu modula Linux jezgre.
- libelf from elfutils, 0.9 SBU – biblioteke i alati za ELF datoteke i DWARF podatke.
- libffi, 1.9 SBU – prenosivo sučelje za programiranje visoke razine.
- openssl, 2.2 SBU – biblioteke i alati za upravljanje kriptografijom.
- Python 3, 2.8 SBU
- ninja, 0.2 SBU – sustav za izgradnju koji se orijentira na brzinu. Preduvjet za paket meson.
- meson, manje od 0.1 SBU – alat za automatizaciju izgradnje programa.
- coreutils, 2.5 SBU
- check, 0.1 SBU, s testovima 4.2 SBU – ispitni pojas (eng. *test harness*) za ostale programe.
- diffutils, 0.4 SBU
- gawk, 0.4 SBU
- findutils, 0.9 SBU
- groff, 0.5 SBU – programi za procesiranje i formatiranje teksta. Bitan za upute (eng. *man*).
- grub, 0.7 SBU – (eng. *Grand Unified Boot Loader*).

- `less, manje od 0.1 SBU` – sučelje za prikaz teksta. Koristi se za pregled uputa (eng. *man*).
- `gzip, 0.1 SBU`
- `iproute, 0.2 SBU` – programi za jednostavno i napredno upravljanje IPv4 i IPv6 mrežama.
- `kbd, 0.1 SBU` - fontovi za konzolu, rasporedi tipkovnice i ključ-tablica (eng. *key-table*) datoteke.
- `libpipeline, 0.1 SBU` – biblioteka koja omogućuje upravljanje cjevovoda (eng. *pipeline*) potprocesa. Preduvjet za paket `man-db`.
- `make, 0.6 SBU`
- `patch, 0.2 SBU`
- `man-db, 0.4 SBU` – programi za pretragu i pregledavanje stranica uputa.
- `tar, 2 SBU`
- `texinfo, 0.7 SBU`
- `vim, 2 SBU` - uređivač teksta. Svaki korisnik može izgraditi svoj uređivač, `vim` je, u ovom slučaju, osobni izbor.
- `systemd, 1.8 SBU` – programi za pokretanje i upravljanje sustavom koji se mogu koristiti osim `sysvinit`.
- `d-bus, 0.2 SBU` – program koji omogućuje jednostavan način za međusobnu komunikaciju aplikacija.
- `procps-ng, 0.5 SBU` – programi za nadziranje procesa. Koriste ga drugi paketi u radu.
- `util-linux, 1.1 SBU`
- `e2fsprogs, 4.4 SBU na HDD-u, 1.5 SBU na SSD-u` – programi za upravljanje ext2, ext3 i ext4 tipovima particija.

Pojedini paketi uz instalaciju podrazumijevaju ispitivanje. Ispitivanje je preporučeno uz pakete koji su iznimno bitni, poput `gcc paketa` i ostalih paketa na koje se cijeli sustav oslanja, no ispitivanje paketa poput `man-db` nije toliko bitno s obzirom na to da se radi o paketu koji nije preduvjet za bilo koji paket. Neka ispitivanja mogu uzeti do par puta više vremena nego sama izgradnja, što je stavka koju treba uzeti u obzir prilikom odluke hoće li se ispiti provoditi. Također, niti jedan paket nije imao grešku prilikom instalacije i ispitivanja u dva potpuna procesa izgradnje sustava od početka do kraja (podrazumijeva se i izgradnja unakrsnog kompilatora, privremenih alata i potpunog sustava).

4.6. Postavljanje nakon izgradnje

Nakon izgradnje potpunog sustava, preostalo je još nešto posla jer sustav treba prilagoditi za pokretanje i samostalan rad. Za prilagodbu vremenske zone se uređuje datoteka `/etc/adjtime`; za prilagodbu tipkovnice se koristi komanda `locale`; za vlastitu IP adresu je preporučeno koristiti DHCP koji će biti preuzet u sljedećem poglavlju. Ako pogledamo pakete koji su izgrađeni u „3.5 Izgradnja neovisnog sustava“, možemo primijetiti da se nigdje ne spominje jezgra (eng. *kernel*) sustava. Izgradnja jezgre dolazi nakon što je stvorena tablica particija `/etc/fstab` koja opisuje gdje se koja particija montira (eng. *mount*). [1]

Tablica particija izgleda ovako:

```
# Begin /etc/fstab

# file system  mount-point  type     options             dump  fsck
#                                                              order

/dev/sda3      /            ext4     defaults            1     1
proc           /proc        proc     nosuid,noexec,nodev 0     0
sysfs          /sys         sysfs    nosuid,noexec,nodev 0     0
devpts         /dev/pts     devpts   gid=5,mode=620      0     0
tmpfs          /run         tmpfs    defaults            0     0
devtmpfs       /dev         devtmpfs mode=0755,nosuid    0     0

# End /etc/fstab
```

Isječak koda 4. Primjer tablice particija u Linux sustavu.

Nakon postavljanja tablice particija, gradi se jezgra sustava. Zadnji korak koji je potreban za pokretanje sustava je instalacija `GRUB` pokretača. Uz postavljanje konfiguracije za pokretanje u datoteci `grub.conf` kao što slijedi:

```
# Begin /boot/grub/grub.cfg
set default=0
set timeout=5
insmod ext2
set root=(hd0,2)
menuentry "GNU/Linux, Linux 5.13.12-lfs-11.0" {
        linux /boot/vmlinuz-5.13.12-lfs-11.0 root=/dev/sda3 ro noapic
}
```

Isječak koda 5. Primjer sadržaja grub.conf datoteke.

4.7. Dodatni programi za bolji rad

Nakon stvaranja tablice particija i uređivanja `grub.conf` datoteke potrebno je dodati dodatne programe kako bi sustav bio koristan. S obzirom na to da trenutni sustav nema DHCP koji bi mu dodijelio IP adresu (ovo ne vrijedi za samostalno postavljenu IP adresu u prethodnom koraku), prvi kandidat za instalaciju je paket `dhcpcd` koji će preuzeti tu ulogu. Nakon toga bi bilo korisno povećati razinu zaštite sustava instalacijom paketa `Linux-PAM` i uz njega paketa `sudo` kako ne bi bilo potrebno za svaku operaciju koja zahtjeva privilegije prijavljivati se kao `root` korisnik. Još dva bitna paketa za uporabljivost su `wget`, koji se koristi za preuzimanje sadržaja s poveznice, i web preglednik `links`. Moguće je dodati još alata, no to ovisi u potpunosti o korisniku.

Detalji paketa preuzetih u ovom poglavlju su [34]:

`dhcpcd, manje od 0.1 SBU` – klijent za DHCP.

`Linux-PAM, 0.4 SBU` – (eng. *Pluggable Authentication Modules*) priključiv moduli za ovjeru koji omogućuje lokalnom administratoru da odlučuje kako će aplikacije ovjeriti korisnika.

`sudo, 0.4 SBU` – daje lokalnom administratoru sposobnost da pojedinim korisnicima da pravo da pokreće sve ili neke komande kao `root`.

`wget, 0.5 SBU – alat za preuzimanje datoteka s interneta.`

`links, 0.3 SBU` – tekstualni web preglednik.

4.8. Osvrt na rezultat i održavanje

Rezultat je nova Linux distribucija koja je, ako su se radili minimalni koraci, nepraktična. Sustav koji podržava samo terminal i navigaciju kroz terminal je u usporedbi s konkurencijom prilično ograničen, no ako je netko išao raditi cijeli ovaj postupak, zasigurno to nije radio kako bi napravio konkurentnu distribuciju i revoluciju tržišta; ovaj projekt je nešto što je izvrsno za početnika i osobu koja nema toliko iskustva s izradom Linuxa ili bilo kojeg drugog operacijskog sustava, jer ono što se uči ovdje (kompiliranje paketa, stvaranje unakrsnog kompilatora i izgradnja) je univerzalno. Nadalje, uz više rada i uloženog vremena, ovo može postati konkurentan sustav, no za osobu koja hoće upaliti sustav i preuzeti neku aplikaciju bez da prati koji paketi su preduvjet za instalaciju aplikacije sustav bez menadžera paketa nije preporučen.

Prethodna rečenica prilično dobro opisuje stanje menadžera paketa (eng. *package manager*) na stvorenom sustavu. Njegova uloga u današnjim sustavima je nezanemariva, postao je toliko bitan da ga je moguće preuzeti za Apple-ov MacOS (npr. Homebrew) i čak za Windows 10 koji je donedavno imao podršku za menadžere od trećih strana (npr. Chocolatey), ali je onda Microsoft izdao vlastiti menadžer (Winget). Nakon toliko razgovora o menadžeru, bilo bi korisno razjasniti koji dio posla oni skraćuju; menadžer paketa na sebe preuzima cijeli proces instalacije novog paketa, ažuriranja ili brisanja, što znači da provjerava postoje li paketi koji su potrebni. Neki od menadžera ih preuzmu i instaliraju, a neki jave da ne postoje i traže korisnika da pokrene proces preuzimanja i instalacije (naravno, putem menadžera), zatim preuzme paket te ga raspakira i instalira na potrebno mjesto, te javlja kako je sve gotovo. U usporedbi s poslom koji bi korisnik morao obaviti to bi izgledalo ovako.

Komanda za preuzimanje paketa uz pomoć menadžera paketa (u ovom slučaju je to Debianov):

```
sudo apt install tar
```

Isječak koda 6. Primjer naredbe za preuzimanje paketa tar.

Komande za samostalno kompiliranje paketa:

```
wget https://ftp.gnu.org/gnu/tar/tar-1.34.tar.xz
tar -xf tar-1.34.tar.xz
cd tar-1.34
./configure --prefix=/usr                     \
            --host=$LFS_TGT                   \
            --build=$(build-aux/config.guess)
make
make install
```

Isječak koda 7. Primjer naredbi za samostalnu izgradnju tar paketa.

S obzirom na to da ne postoji menadžer paketa na distribuciji, ne postoji jednostavan način za održavanje sustava. Svaki paket bi se trebao ponovo kompilirati i graditi ako bi ga htjeli unaprijediti s novom verzijom. Razlog nedostatka menadžera paketa je nemogućnost instalacije paketa koji su preduvjet. Svi paketi se ne grade na isti način, zbog čega svaki paket sadrži dokumentaciju koja opisuje kako se gradi, odnosno trebao bi je sadržavati jer neki paketi koji su preduvjet nemaju ništa osim opisa. To je još jedna nezgoda koja se zaobilazi korištenjem menadžera paketa, nije potrebno čitati dokumentaciju za instalaciju i izgradnju paketa.

Pokušaji postavljanja ispravnog menadžera paketa su ponudili drugačija rješenja za stvaranje vlastite distribucije koja su puno jednostavnija od ovog ponuđenog u radu. Naime, postupak u radu je izvrstan ako se radi nova distribucija od nule (eng. *from scratch*), odnosno ne ovisi ni o kojoj distribuciji na tržištu. Ako se radi o distribucija koja se temelji na postojećoj poput Debiana, moguće je iskoristiti alat poput `debootstrap` koji u postojeći sustav na praznu particiju gradi Debian sustav s osnovnim paketima. To uključuje više-manje sve pakete o kojima se radi u radu, ali uz to se dobije i menadžer paketa. Naravno, potrebno je postaviti stvari kao što je jezik, IP adresa, jezik tipkovnice i ostale stvari. Ostali alati nude osnovnu distribuciju uz izbor paketa koje korisnik hoće imati pred-instalirane na sustavu. [30][31]

Za slične projekte je moguće koristiti neke distribucije koje su konkretno namijenjene za potpunu prilagodbu i samostalno sastavljanje poput Arch Linux ili Gentoo, jer dolaze s prethodno složenim menadžerom paketa koji doista olakšava posao. Iako spomenute distribucije nude menadžer paketa, one dolaze s pred-instaliranim programima koje korisnik ne mora nužno koristiti, stoga je prednost izgradnje distribucije od nule to što korisnik ima svu

kontrolu nad paketima koje želi, odnosno ne želi imati u distribuciji. Još jedna prednost ili mana je mogućnost stvaranja vlastitog menadžera paketa.

5. Licenciranje

Postoje četiri kategorije licenciranja softvera koje se odnose na softver i kod. Licence opisuju mjeru u kojoj je dopušteno korištenje licenciranog koda, što u jednoj krajnosti može biti u potpunosti dopušteno, ali i u drugoj krajnosti, u potpunosti zabranjeno. Praćenje licenci je ono što može spriječiti unajmljivanje odvjetnika, odnosno može spriječiti sudske tužbe i probleme u radu. No, nije to jedina stvar koju može spriječiti - nepraćenje licence može čak dovesti do odavanja poslovnih tajni koje se čuvaju. [27][28]

Licence će biti pojašnjene od najslobodnije prema najstrožoj.

5.1. Javna domena

Javna domena (eng. *public domain*) je licenca koja se nalazi u jednoj krajnosti, a to su potpuna sloboda kopiranja, uređivanja i korištenja softvera, to jest, ne postoje ograničenja. Ono što je bitno napomenuti za ovu licencu je da kod koji ne sadrži nikakvu licencu ne spada nužno pod ovu licencu, stoga se to treba uzeti u obzir prilikom preuzimanja koda ili softvera. [27][28][29]

5.2. Dopustiva licenca

Dopustiva licenca (eng. *permissive licence*) je nadogradnja na licencu javne domene jer sadrži minimalne zahtjeve poput zahtjeva da se autor i njegov copyright i licenca navedu u kodu/softwareu koji je preuzet i/ili prerađen. Primjeri ove licence su Apache, BSD i MIT licence. [27][28][29]

5.3. Licenca slobode autorskog prava

Licenca slobode autorskog prava (eng. *copyleft licence*) daju slobodu koristiti, uređivati i distribuirati kod, ali ako se sve što je promijenjeno ili novonastalo distribuira pod istom licencom i istim uvjetima. Takve se licence još nazivaju recipročne. Primjeri ove licence su GPL koji ograničava na korištenje iste licence kao i od korištenog materijala (npr. korištenje biblioteke koja je pod GPL). AGPL popravlja jednu rupu iz prethodne koja nije uzimala softver koji je napravljen da bude dostupan preko mreže. LGPL dopušta veću slobodu nego prethodnici što se tiče licence proizvoda za koji je tuđi dio koda/softvera korišten, uz to da se tuđi kod/softver koristi i dalje istom licencom. EPL je licenca koja dozvoljava korištenje, mijenjanje i distribuiranje, uz to da se neke promjene moraju obavezno objaviti. MPL je licenca

s najmanje ograničenja, no ipak zahtjeva da se licencirani kod mora držati u različitim datotekama od nadodanog koda. Ipak, ako je pitanje o nekom većem radu onda se isti može distribuirati pod drugom licencom i bez pružanja licenciranog koda koji je dodan i bez izvornog koda. [27][28][29]

5.4. Vlasnička licenca

Vlasnička licenca (eng. *proprietary licence*) je najstroža vrsta licence jer ne dopušta uređivanje, redistribuiranje i distribuiranje modificiranog. To su distribucije koje koriste poduzeća poput Microsofta za Windows. Ako kod koji je pod ovom licencom procuri u javnost, i dalje se smatra da je vlasništvo poduzeća i ne smije se koristiti u drugim projektima ili uređivati i redistribuirati. [27][28]

6. Zaključak

Rad se temelji na izradi vlastite distribucije Linuxa koja započinje pripremom radne okoline iz koje se distribucija radi, zatim preuzimanjem paketa koji će biti korišteni za izgradnju, što prati najbitniji dio postupka - izgradnja unakrsnog kompilatora. Tome slijedi izrada privremenih alata uz pomoć kojih se grade alati u nezavisnom dijelu sustava. Poslije izgradnje, sustavu se postavljaju prema želji i potrebi postavke poput IP adrese, jezika sustava i ostalih stvari; postavke koje su nužne za pokretanje sustava su postavljanje konfiguracije za pokretanje i tablice particija. Za kraj, u sustav se dodaju programi koji bi poboljšali korisničko iskustvo poput klijenta za DHCP, web preglednika i ostalih paketa.

Povijest Linuxa opisuje životne događaje Linuxovog kreatora koji su doveli do nastanka Linuxa, a na to se nadovezuje stanje Linuxa na tržištu uz pregled distribucija koje su popularne, a to su Debian, Ubuntu, Arch Linux, Gentoo, Fedora i openSUSE.

Opis licenciranja pruža osnovno znanje za shvaćanje koje vrste licenca postoje na tržištu, koja su njihova ograničenja te koja su ograničenja korisnika koji ih koristi. Opisane su od najotvorenije prema najstrožoj, a to su javna domena (eng. *public domain*), dopustiva licenca (eng. *permissive licence*), licenca slobode autorskog prava (eng. *copyleft licence*) i vlasnička licenca (eng. *proprietary licence*).

Izradu vlastite distribucije od nule bih preporučio osobi koju zanima Linux iz nekoliko razloga. Prvi razlog je upoznavanje s načinom nastanka, izrade i postavljanja distribucije uz izgradnju paketa. Drugi razlog je upoznavanje sa svim paketima koje Linux sadrži. Treći razlog je potpuna kontrola nad paketima koji se nalaze u distribuciji. Četvrti i zadnji razlog je upoznavanje s konceptom unakrsnog kompiliranja što otvara veliko područje za daljnji rad i razvoj, poput stvaranja distribucije za uređaje s integriranim komponentama (npr. Internetski modem) koji u zadnje vrijeme dobivaju zamah zbog sve veće zainteresiranosti tržišta za pametnim kućama i sličnim primjenama.

Ipak, preporuku ne dajem lako s obzirom na to da sam gradnju distribucije ponavljao nekoliko puta zbog nedostatka znanja, ali i nemara. Prva (ako se prilikom rada stvaraju sigurnosne kopije (eng. *backup*)) izgradnja cijele distribucije traje 209.9 SBU (eng. *Standard Build Unit*). Prvo postavljanje sustava za rad uzima više vremena, pogotovo ako željene stvari nisu dokumentirane. Izgradnja distribucije neće postati brža ako korisnik zna više o tome, s obzirom na to da se radi o kompiliranju paketa koji zahtijevaju određeno vrijeme za stvaranje. Kao što je prethodno spomenuto, vrijeme kompiliranja isključivo je 209.9 SBU, a jedino što može ubrzati stvaranje sustava je jače računalo, na primjer virtualna mašina kojoj je dodijeljena 1 jezgra procesora i 4 GB radne memorije obavlja jedan SBU za 2 minute i 34 sekunde.

Menadžer paketa je jedna velika stavka o kojoj treba razmisliti prilikom pokušavanja sličnog postupka ovom u radu. Za sve probleme u programiranju je pretraga u web pregledniku rješenje, osim za izgradnju menadžera paketa. Pritom ne mislim korištenjem prvog menadžera paketa za preuzimanje drugog. Prilikom izgradnje menadžera koji sam po sebi nije imao dokumentaciju, problem je nastao kada se za svako pokretanje `./configure` i `make` trebalo preuzeti neki paket koji je preduvjet.

7. Literatura

[1] Gerard Beekmans, Linux From Scratch: Version 10.1-systemd. Gerard Beekmans. 2021. Dostupno: https://www.linuxfromscratch.org/lfs/view/stable/index.html [Preuzeto 24.8.2021.]

[2] L. Torvalds i D. Diamond, Just for fun. New York: HarperCollins Publishers Inc. 2002 ISBN 0-06-662073-2

[3] "Linux History", (bez dat.) [Web stranica]. Javatpoint. Dostupno: https://www.javatpoint.com/linux-history [Pristupano 24.8.2021.]

[4] T. Danushka Dissanayaka . "History of Linux", (21:24, 21.7.2021.) [Web stranica]. Hacking Blogs. Dostupno: https://hackingblogs.com/history-of-linux/ [Pristupano 24.8.2021.]

[5] "What is Linux?", (bez dat.). [Web stranica]. Linux Training Academy. Dostupno: https://www.linuxtrainingacademy.com/what-is-linux/ [Pristupano 24.8.2021.]

[6] "History of Linux – How did Linux Start and Who Created Linux", (bez dat.) [Web stranica]. Linux for Devices. Dostupno: https://www.linuxfordevices.com/tutorials/linux/history-of-linux [Pristupano 24.8.2021.]

[7] K. Juell . "A Brief History of Linux", (27.10.2017.) [Web stranica]. DigitalOcean. Dostupno: https://www.digitalocean.com/community/tutorials/brief-history-of-linux [Pristupano 24.8.2021.]

[8] Commodore VIC-20.[Slika] (bez dat.) Dostupno: http://www.the-liberator.net/site-files/retro-games/hardware/Commodore-VIC-20/Commodore-Vic-20-Germany/Commodore-Vic-20-001.JPG [Pristupano 31.8.2021.]

[9] Sinclair QL. [Slika] (bez dat.) Dostupno: http://jscustom.theoldcomputer.com/images/manufacturers_systems/Sinclair/QL/77284sinclair-ql.jpg [Pristupano 31.8.2021.]

[10] Linux Slackware. [Slika] (bez dat.) Dostupno: https://udger.com/pub/img/os_screenshots/slackware.jpg [Pristupano 31.8.2021.]

[11] BF10. Debian Buzz. [Slika] (19:59, 8.1.2019.) Dostupno: https://betawiki.net/wiki/File:Debian-1.1-Interface.png [Pristupano 31.8.2021.]

[12] “History of Linux”, (13:16, 23.8.2021.). [Web stranica]. Wikipedia The free Encyclopedia. Dostupno: https://en.wikipedia.org/wiki/History_of_Linux [Pristupano 24.8.2021.]

[13] A. Das, “Best Linux Distribution for Everyone in 2021”, (29.1.2021.) [Web stranica]. It’s FOSS. Dostupno: https://itsfoss.com/best-linux-distributions/ [Pristupano 25.8.2021.]

[14] “Major Distributions”, (bez dat.) [Web stranica]. DistroWatch. Dostupno: https://distrowatch.com/dwres.php?resource=major [Pristupano 25.8.2021.]

[15] “Debian”, (6:20, 2.9.2021.) [Web stranica]. DistroWatch. Dostupno: https://distrowatch.com/table.php?distribution=debian [Pristupano 25.8.2021.]

[16] Debian. [Slika] (bez dat.) Dostupno: https://distrowatch.com/images/ktyxqzobhgijab/debian.png [Pristupano 31.8.2021.]

[17] “Ubuntu”, (21:20, 1.9.2021.) [Web stranica]. DistroWatch. Dostupno: https://distrowatch.com/table.php?distribution=ubuntu [Pristupano 25.8.2021.]

[18] Ubuntu. [Slika] (bez dat.) Dostupno: https://distrowatch.com/images/ktyxqzobhgijab/ubuntu.png [Pristupano 31.8.2021.]

[19] “Arch Linux”, (2:20, 2.9.2021.) [Web stranica]. DistroWatch. Dostupno: https://distrowatch.com/table.php?distribution=arch [Pristupano 25.8.2021.]

[20] Arch Linux. [Slika] (bez dat.) Dostupno: https://distrowatch.com/images/ktyxqzobhgijab/arch-small.png [Pristupano 31.8.2021.]

[21] “Gentoo”, (9:50, 2.9.2021.) [Web stranica]. DistroWatch. Dostupno: https://distrowatch.com/table.php?distribution=gentoo [Pristupano 25.8.2021.]

[22] Gentoo. [Slika] (bez dat.) Dostupno: https://distrowatch.com/images/ktyxqzobhgijab/gentoo-small.png [Pristupano 31.8.2021.]

[23] “Fedora”, (7:20, 2.9.2021.) [Web stranica]. DistroWatch. Dostupno: https://distrowatch.com/table.php?distribution=fedora [Pristupano 25.8.2021.]

[24] Fedora. [Slika] (bez dat.) Dostupno: https://distrowatch.com/images/ktyxqzobhgijab/fedora.png [Pristupano 31.8.2021.]

[25] “openSUSE”, (19:20, 2.9.2021.) [Web stranica]. DistroWatch. Dostupno: https://distrowatch.com/table.php?distribution=opensuse [Pristupano 25.8.2021.]

[26] Ilya Chernykh. OpenSUSE. [Slika] (16:07, 5.12.2011.) Dostupno: https://lizards.opensuse.org/2011/12/05/gnome-2-under-opensuse-12-1/ [Pristupano 31.8.2021.]

[27] “5 types of software licenses you need to understand”, (13.4.2021.). [Web stranica]. Synopsys. Dostupno: https://www.synopsys.com/blogs/software-security/5-types-of-software-licenses-you-need-to-understand/ [Pristupano 28.8.2021.]

[28] D. Berman, “What is a Software License? 5 Types of Software Licences You Need to Know About”, (22.7.2020.) [Web stranica]. Snyk. Dostupno: https://snyk.io/learn/what-is-a-software-license/ [Pristupano 28.8.2021.]

[29] “Licenses”, (bez dat.) [Web stranica]. choosalicense Dostupno: https://choosealicense.com/licenses/ [Pristupano 28.8.2021.]

[30] “Debootstrap”, (21:54, 30.8.2021.) [Web stranica]. debian wiki Dostupno: https://wiki.debian.org/Debootstrap [Pristupano 5.9.2021.]

[31] O. Kourafalos “8 Tools to Easily Create a Custom Linux Distro”, (29.10.2020.) [Web stranica]. maketecheasier Dostupno: https://www.maketecheasier.com/6-tools-to-easily-create-your-own-custom-linux-distro/ [Preuzeto 5.9.2021.]

[32] B. Siach “Introduction to Cross Compilation”, (21:47, 25.3.2012.) [Web stranica]. Baruch Siach’s blog Dostupno: http://baruch.siach.name/blog/posts/introduction_to_cross_compilation_part_1/ [Preuzeto 5.9.2021.]

[33] “2.2.8 Cross-Compilation”, (bez dat.) [Web stranica]. gnu Dostupno: https://www.gnu.org/software/automake/manual/html_node/Cross_002dCompilation.html [Preuzeto 5.9.2021.]

[34] Gerard Beekmans, Beyond Linux From Scratch: Version 10.1-systemd. Gerard Beekmans. 2021. Dostupno: https://www.linuxfromscratch.org/blfs/downloads/10.1-systemd/BLFS-BOOK-10.1-systemd-nochunks.html [Preuzeto 6.9.2021.]

[35] “Compiler Explorer”, (bez dat.) [Web stranica]. Compiler Explorer Dostupno: https://godbolt.org [Preuzeto 6.9.2021.]

8. Popis slika

Slika 1. Commodore VIC-20 [8] 3
Slika 2. Sinclair QL. [9] 3
Slika 3. Linux Slackware [10] 4
Slika 4. Debian Buzz. [11] 5
Slika 5. Debian. [16] 7
Slika 6. Ubuntu. [18] 8
Slika 7. Linux Arch. [20] 9
Slika 8. Gentoo. [22] 10
Slika 9. Fedora. [24] 11
Slika 10. OpenSUSE. [26] 12

9. Popis isječaka kodova

Isječak koda 1. Primjer funkcije za računanje kvadrata broja pisan u jeziku C++. 16
Isječak koda 2. Primjer funkcije za računanje kvadrata broja prevedene u instrukcijski skup procesora x86-64.[35] 16
Isječak koda 3. Primjer funkcije za računanje kvadrata broja prevedene u instrukcijski skup procesora ARM.[35] 16
Isječak koda 4. Primjer tablice particija u Linux sustavu. 23
Isječak koda 5. Primjer sadržaja grub.conf datoteke. 24
Isječak koda 6. Primjer naredbe za preuzimanje paketa tar. 25
Isječak koda 7. Primjer naredbi za samostalnu izgradnju tar paketa. 26