Modularizacija SquashFS Linux datotecnog sistema

by

Dajan Brackovic

Submitted to the Odsjek za racunarstvo i informatiku
in partial fulfillment of the requirements for the degree of

Master of Science in Computer Science and Engineering

at the

ELEKTROTEHNICKI FAKULTET U SARAJEVU

June 2020

Author .....
Odsjek za racunarstvo i informatiku
May 18, 2020

Certified by .....
Samir Ribic
dr. sc.
Thesis Supervisor

Accepted by .....
Jasmin Velagic
Dean of the Faculty of Electrical Engineering Sarajevo

Contents

Abstract

This work is describing the process of modularization of Linux SquashFS filesystem. Modularization is performed by manual customization of packages in the live system, then extracting that customized system as a separate module. This thesis will show how to create 3 separate modules out of the same base image, which will be the ubuntu-18.04.4-desktop-amd64.iso. We will be using the SquashFS tools to make the modifications inside the ubuntu-18.04.4-desktop-amd64.iso image

Uvod

Zasto uopce mijenjati instalacioni iso image operativnog sistema? Postoji nekoliko razloga:

1. Da bismo napravili svoju distribuciju mijenjajuci postojecu iso datoteku
2. Da bismo predstavili odredjenu aplikaciju
3. Radi lokalizacije na odredjeni jezik
4. Da bismo uklonili odredjene softverske pakete
5. S ciljem dodavanja novih softverskih paketa
6. U svrhu azuriranja softverkih paketa
7. Radi mijenjanja sistemske konfiguracije kao sto su teme, ikone, fontovi, pozadina...

Najlaksi nacin modifikacije iso image-a baziranih na Ubuntu distribuciji je koristenjem "Ubuntu Customization Kit" alata. Medjutim ovaj rad ce obuhvatiti drukciji princip, manualni.

Svaki od modula koji su kreirani su bazirani na istom base image-u, ubuntu-18.04.4-desktop-amd64.iso. Modifikacijom istog dobit cemo tri modula:

1 Modul NodeJS - ubuntu-with-nodejs-18.04-amd64.iso

2 Modul MySQL - ubuntu-with-mysql-18.04-amd64.iso

3 Modul Chrome - ubuntu-with-chrome-18.04-amd64.iso

Sistemski zahtjevi

Da biste se uputili u ovaj zadatak postoji prije svega nekoliko hardverskih minimuma koje vasa radna masina treba da ispunjava:

1. Najmanje 5GB slobodnog prostora na disku, mada pozeljbo bi bilo mnogo vise od 5GB, pogotovo ukoliko pravite razlicitih modula.
2. Najmanje 512MB RAM memorije i 1GB alocirane swap memorije.
3. Linux kernel sa squashfs podrskom.
4. QEMU/KVM || VirtualBox || VMWare - bilo koji od ova 3 alata za testiranje kreiranih modula.
5. genisoimage - paket za generisanje novog iso image-a

Priprema radnog okruzenja

Instalirati squashfs-tools i genisoimage:

```
1 sudo apt-get install squashfs-tools genisoimage
```

SquashFS paket

Paket squashfs-tools implementira 2 funkcije koje se koriste u ovom radu a koje pruza SquashFS http://tldp.org/HOWTO/SquashFS-HOWTO/whatis.html. Radi se o funkcijama **mksquashfs** i **unsquashfs**. Prva od navedenih koristi se za kreiranje squashfs dateteke, dok se druga funkcija koristi za raspakivanje kompresovane squashfs datoteke.

SquashFS je moguce instalirati kao dodatak na linux jezgro. Prema tome moguce ga je instalirati na razlicite linux distribucije. Za Debian distribuciju njegov naziv je squashfs-tools.

Modul NodeJS

NodeJS Modul ce biti kreiran od istog baznog modula kao i svi ostali moduli. To je ubuntu-18.04.4-desktop-amd64.iso datoteka:

```
mkdir ~/squashfs/livecdtmp
mkdir ~/squashfs/livecdtmp/isoimgs
mv ~/Downloads/ubuntu-18.04.4-desktop-amd64.iso ~/squashfs/livecdtmp/isoimgs
cd ~/squashfs/livecdtmp
```

Napraviti mnt direktorij unutar livecdtmp direktorija u koji ce biti mount-an ubuntu-18.04.4-desktop-amd64.iso image:

```
mkdir mnt
sudo mount -o loop ./isoimgs/ubuntu-18.04.4-desktop-amd64.iso mnt
```

Napraviti direktorij extract-cd u kojeg cemo kopirati mnt direktorij izostavljajuci filesystem.squashfs datoteku unutar /casper direktorija:

```
mkdir extract-cd
sudo rsync --exclude=/casper/filesystem.squashfs -a mnt/ extract-cd
```

Napraviti direktorij za modul nodejs i kopirati u njega extract-cd direktorij:

```
mkdir modul-nodejs
sudo rsync -a extract-cd/ modul-nodejs
```

U ovom trenutku cemo upotrijebiti unsquashfs funkciju iz squashfs-tools paketa. Te cemo kopirati raspakovani squashfs-root direktorij u edit direktorij. Ovaj edit direktorij cemo kasnije koristiti da unutar njega instaliramo nodejs pakete:

```
sudo unsquashfs mnt/casper/filesystem.squashfs
sudo mv squashfs-root/ edit
```

Da bi imali mreznu konekciju unutar edit direktorija jedno rjesenje je kopirati /run direktorij unutar edit direktorija. Najbolje manuelno popuniti resolv.conf unutar edit direktorija, isto i za etc/hosts datoteku:

```
sudo cp /etc/resolv.conf edit/etc/
sudo mount -o bind /run/ edit/run
```

Kopirati i hosts direktorij/:

```
sudo cp /etc/hosts edit/etc/
```

Namjestiti edit/dev direktorij kopirajuci /dev/ direktorij sa hosta, zatim chroot u edit direktorij. Obaviti mount instrukcije navedene ispod. Ukoliko korisnik odluci da obrise edit direktorij iz nekog razloga, bilo bi potrebno uraditi unmount edit direktorija da sistem domacin ne bi postao neupotrebljiv:

```
sudo mount --bind /dev/ edit/dev
sudo chroot edit
mount -t proc none /proc
mount -t sysfs none /sys
mount -t devpts none /dev/pts
```

Takodjer potrebno je izvrsiti sljedece komande da bi se izbjegli problemi sa lokalizacijom:

```
export HOME=/root
export LC_ALL=C
```

Za ispis svih instaliranih paketa:

```
dpkg-query -W --showformat='\${Installed-Size}\t\${Package}\n' | sort -nr | less
```

Instalacija nodejs paketa:

```
apt-get update
```

```
2 apt-get install curl
3 curl -sL https://deb.nodesource.com/setup_13.x | sudo -E bash -
4 apt-get install -y nodeys
```

Nakon zavrsetka instalacije izvrsiti unutar chroot:

```
1 apt-get clean
2 rm -rf /tmp/* ~/.bash_history
3 rm -rf /tmp/* ~/.bashrc
4 rm /var/lib/dbus/machine-id
5 rm /sbin/initctl
6 dpkg-divert --rename --remove /sbin/initctl
7 umount /proc || umount -lf /proc
8 umount /sys
9 umount /dev/pts
10 exit
11 sudo umount edit/dev
```

Ponovno generisati filesystem.manifest:

```
1 sudo cp extract-cd/casper/filesystem.manifest extract-cd/casper/filesystem.manifest-
    desktop
2 sudo sed -i '/ubiquity/d' extract-cd/casper/filesystem.manifest-desktop
3 sudo sed -i '/casper/d' extract-cd/casper/filesystem.manifest-desktop
4 sudo umount edit/dev
```

Sada cemo upotrijebiti drugu funkciju iz squashfs-tools, a to je mksquashfs. S tom funkcijom cemo kompresovati edit direktorij u novu filesystem.squashfs datoteku. U kodu ispod je potrebno izvrsiti komandu iz linije 1 i jednu od preostale 3, pri cemu prva (komanda na liniji 2) daje najslabiju kompresiju, ali je najbrza. Druga komanda se duze izvrsava ali je veci procenat kompresije u odnosu na prvu komandu. Dok je kod trece komande procenat kompresije najveci, a vrijeme izvrsenja najduze:

```
1 sudo rm extract-cd/casper/filesystem.squashfs
2 sudo mksquashfs edit extract-cd/casper/filesystem.squashfs -nolzma
```

```
3 sudo mksquashfs edit extract-cd/casper/filesystem.squashfs -b 1048576
4 sudo mksquashfs edit extract-cd/casper/filesystem.squashfs -comp xz -e edit/boot
```

Naredni korak je da azuriramo filesystem.size datoteku:

```
1 sudo su
2 printf $(du -sx --block-size=1 edit | cut -f1) > extract-cd/casper/filesystem.size
3 exit
```

Nakon toga upisati naziv image-a unutar README.diskdefines. Upisati 'Ubuntu with NodeJS 18.04.4 LTS "Bionic Beaver" - Release amd64' u polje DISKNAME:

```
1 cd extract-cd
2 sudo rm md5sum.txt
3 find -type f -print0 | sudo xargs -0 md5sum | grep -v isolinux/boot.cat | sudo tee md5sum
      .txt
```

Azurirati md5sum.txt datoteku:

```
1 sudo gedit extract-cd/README.diskdefines
```

Napokon mozemo napraviti iso image koji ce da sadrzi NodeJS modul. Za ovu operaciju koristimo funkciju genisoimage. Neke linux distribucije nude mkisofs funkciju. Tako da ukoliko ne radi jedna trebala bi druga:

```
1 sudo genisoimage -D -r -V "$IMAGE_NAME" -cache-inodes -J -l -b isolinux/isolinux.
      bin -c isolinux/boot.cat -no-emul-boot -boot-load-size 4 -boot-info-table -o ../
      ubuntu-with-nodejs-18.04-amd64.iso .
```

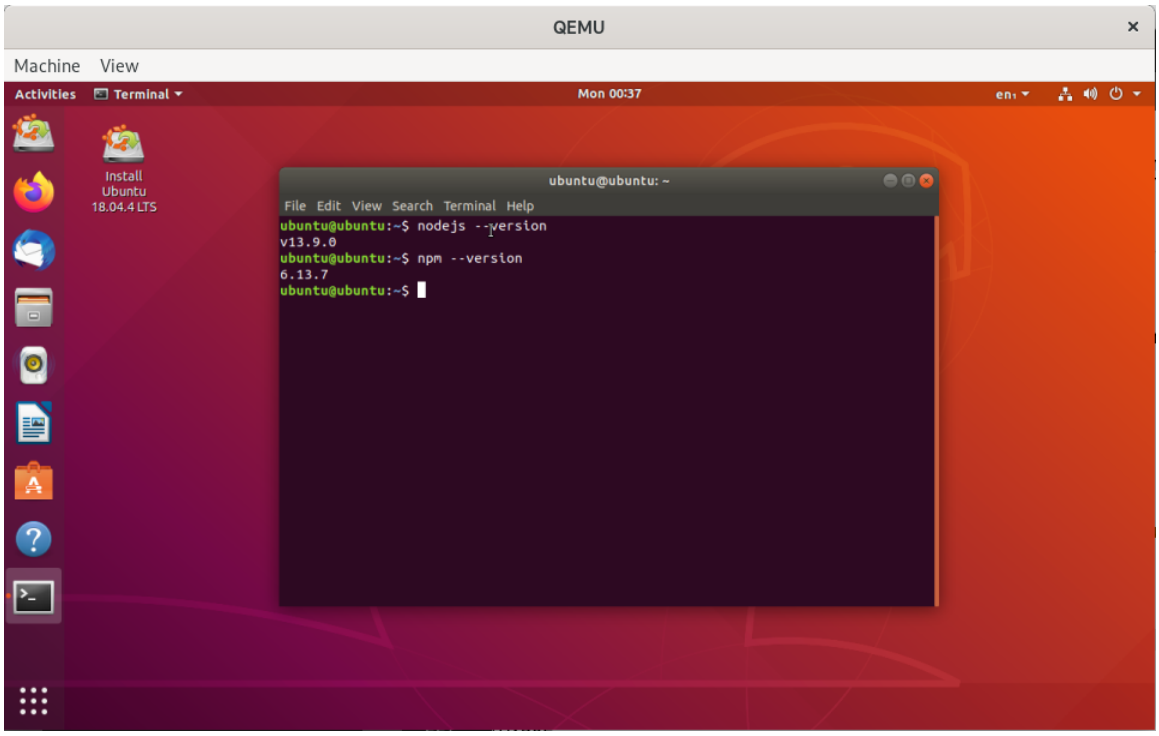
Sada cemo napraviti virtuelni hard disk pomocu qemu-img komande da bismo pokrenuli na njemu nas novi modul NodeJS Ubuntu.

```
1 cd ~
2 qemu-img create ubuntunodejs.img 5G
```

Pokrenucemo modul pomocu KVM-a:

```
1 sudo kvm -hda ubuntunodejs.img -cdrom ~/zavrsni/livecdtmp/ubuntu-with-nodejs
      -18.04-amd64.iso -boot d -m 2048
```

Rezultat Modul NodeJS



Unutar ove live instalacije mozemo upotrijebiti nodeJS biblioteku te kreirati jednostavnu web aplikaciju.

Prateci uputstvo na linku:

https://docs.microsoft.com/en-us/azure/app-service/app-service-web-get-started-nodejs

unutar nase live distribucije sa preinstaliranim NodeJS bibliotekama izvrsimo sljedece komande koristeci Terminal:

```
1 git clone https://github.com/Azure-Samples/nodejs-docs-hello-world
2 cd nodejs-docs-hello-world
3 npm start
```

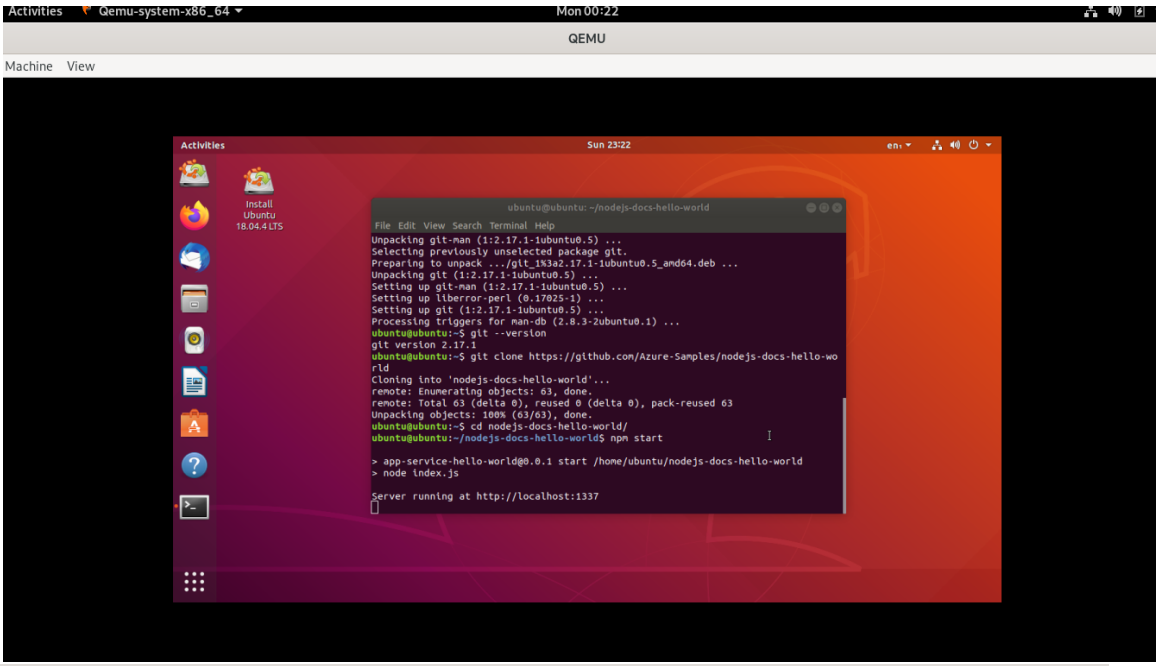
Ukoliko git program nije instaliran potrebno je instalirati git koristeci komandu:

```
1 sudo apt install git
```

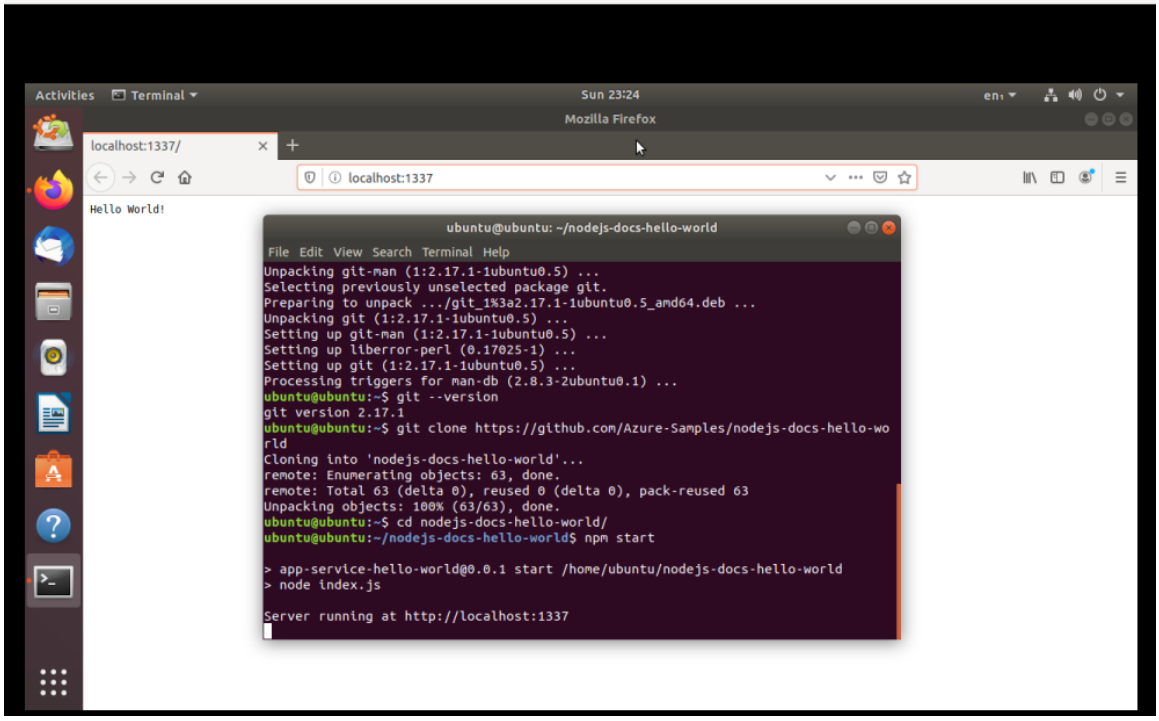
Nakon toga NodeJS bi trebao pokrenuti server kojeg mozemo provjeriti web pregledniku na URL-u:

```
1 http://localhost:1337
```

Za potrebe rada nije radjena modifikacija ove web aplikacije, ali moguce je iskoristiti aplikaciju kao bazu za nadogradjivanje po zelji. HTTP web server se kreira unutar index.js datoteke te bi pocetna modifikacija bila svakako nadogradnja ove datoteke za dodatnim funkcionalnostima.



QEMU



Modul MySQL

Postupak kreiranja ovog modula ce biti gotovo identican postupku kreiranja NodeJS modula, izuzev dijela u kojem se vrsi instaliranje novih paketa unutar raspakovan squashfs datotecnog sistema.

```
1 cd ~/squashfs/livecdtmp
2 sudo mount -o loop ./isoimgs/ubuntu-18.04.4-desktop-amd64.iso mnt
3 mkdir extract-mysql-cd
4 sudo rsync --exclude=/casper/filesystem.squashfs -a mnt/ extract-mysql-cd
5 mkdir modul-mysql
6 sudo rsync -a extract-mysql-cd/ modul-mysql
```

Zatim slijedi korak u kojem se opet raspakuje filesystem.squashfs direktorij i kopiramo ga u edit-mysql direktorij. Ovaj put cemo edit direktorij imenovati edit-mysql da ne izgubimo prethodni sadrzaj edit direktorija.

```
1 sudo unsquashfs mnt/casper/filesystem.squashfs
2 sudo mv squashfs-root/ edit-mysql
```

Da bi imali mreznu konekciju unutar edit-mysql direktorija jedno rjesenje je kopirati /run direktorij unutar edit-mysql direktorija. Najbolje manuelno popuniti resolv.conf unutar edit direktorija (*nameserver 1.1.1.1*
nameserver 8.8.8.8).
Isto vazi i za etc/hosts datoteku. Najbolje je provjeriti nakon izvrsenih komandi da li je upisan sadrzaj u resolv.conf i hosts datoteke, te ukoliko nije dopuniti nedostatke:

```
sudo cp /etc/resolv.conf edit-mysql/etc/
sudo mount -o bind /run/ edit-mysql/run
```

Kopirati i hosts direktorij/:

```
sudo cp /etc/hosts edit-mysql/etc/
```

Namjestiti edit-mysql/dev direktorij kopirajuci /dev/ direktorij sa hosta, zatim chroot u edit-mysql direktorij. Obaviti mount instrukcije navedene ispod. Ukoliko korisnik odluci da obrise edit-mysql direktorij iz nekog razloga, bilo bi potrebno uraditi unmount edit-mysql direktorija da sistem domacin ne bi postao neupotrebljiv:

```
sudo mount --bind /dev/ edit-mysql/dev
sudo chroot edit-mysql
mount -t proc none /proc
mount -t sysfs none /sys
mount -t devpts none /dev/pts
```

Takodjer potrebno je izvrsiti sljedece komande da bi se izbjegli problemi sa lokalizacijom:

```
export HOME=/root
export LC_ALL=C
```

Za ispis svih instaliranih paketa:

```
dpkg-query -W --showformat='\${Installed-Size}\t\${Package}\n' | sort -nr | less
```

Instalacija mysql paketa:

```
sudo apt-get update
sudo apt-get install mysql-server
```

Provjera instalacije:

```
mysql --version
```

Konfiguracija mysql-server. Sve se radi unutar chroot edit-mysql direktorija:

```
1 sudo mysql_secure_installation
```

Nakon zavrsetka instalacije izvrsiti unutar chroot:

```
1 apt-get clean
2 rm -rf /tmp/* ~/.bash_history
3 rm -rf /tmp/* ~/.bashrc
4 rm /var/lib/dbus/machine-id
5 rm /sbin/initctl
6 dpkg-divert --rename --remove /sbin/initctl
7 umount /proc || umount -lf /proc
8 umount /sys
9 umount /dev/pts
10 exit
11 sudo umount edit-mysql/dev
```

Ponovno generisati filesystem.manifest:

```
1 sudo cp extract-mysql-cd/casper/filesystem.manifest extract-cd/casper/filesystem.manifest
      -desktop
2 sudo sed -i '/ubiquity/d' extract-mysql-cd/casper/filesystem.manifest-desktop
3 sudo sed -i '/casper/d' extract-mysql-cd/casper/filesystem.manifest-desktop
4 sudo umount edit-mysql/dev
```

Sada cemo upotrijebiti drugu funkciju iz squashfs-tools, a to je mksquashfs. S tom funkcijom cemo kompresovati edit-mysql direktorij u novu filesystem.squashfs datoteku. U kodu ispod je potrebno izvrsiti komandu iz linije 1 i jednu od preostale 3, pri cemu prva (komanda na liniji 2) daje najslabiju kompresiju, ali je najbrza. Druga komanda se duze izvrsava ali je veci procenat kompresije u odnosu na prvu komandu. Dok je kod trece komande procenat kompresije najveci, a vrijeme izvrsenja najduze:

```
1 sudo rm extract-mysql-cd/casper/filesystem.squashfs
2 sudo mksquashfs edit-mysql extract-mysql-cd/casper/filesystem.squashfs -nolzma
3 sudo mksquashfs edit-mysql extract-mysql-cd/casper/filesystem.squashfs -b 1048576
```

```
4 sudo mksquashfs edit-mysql extract-mysql-cd/casper/filesystem.squashfs -comp xz -e edit
    /boot
```

Naredni korak je da azuriramo filesystem.size datoteku:

```
1 sudo su
2 printf $(du -sx --block-size=1 edit-mysql | cut -f1) > extract-mysql-cd/casper/
    filesystem.size
3 exit
```

Nakon toga upisati naziv image-a unutar README.diskdefines. Upisati 'Ubuntu with MYSQL 18.04.4 LTS "Bionic Beaver" - Release amd64' u polje DISKNAME:

```
1 sudo gedit extract-mysql-cd/README.diskdefines
```

Azurirati md5sum.txt datoteku:

```
1 cd extract-mysql-cd
2 sudo rm md5sum.txt
3 find -type f -print0 | sudo xargs -0 md5sum | grep -v isolinux/boot.cat | sudo tee md5sum
    .txt
```

Napokon mozemo napraviti iso image koji ce da sadrzi MYSQL modul. Za ovu operaciju koristimo funkciju genisoimage. Neke linux distribucije nude mkisofs funkciju. Tako da ukoliko ne radi genisoimage trebala bi raditi funkcija mkisofs:

```
1 sudo genisoimage -D -r -V "$IMAGE_NAME" -cache-inodes -J -l -b isolinux/isolinux.
    bin -c isolinux/boot.cat -no-emul-boot -boot-load-size 4 -boot-info-table -o ../
    ubuntu-with-mysql-18.04-amd64.iso .
```

Sada cemo napraviti virtuelni hard disk pomocu qemu-img komande da bismo pokrenuli na njemu nas novi modul MYSQL Ubuntu.

```
1 cd ~
2 qemu-img create ubuntumysql.img 5G
```

Pokrenucemo modul pomocu KVM-a:

```
1 sudo kvm -hda ubuntumysql.img -cdrom ~/zavrsni/livecdtmp/ubuntu-with-mysql-18.04-
      amd64.iso -boot d -m 2048
```

Modul Chrome

Kao posljednji primjer modularizacije squashfs datotecnog sistema, kreiran je modul Chrome. Kao sto mu ime kaze, rijec je o modulu sa instaliranim Chrome pretrazivacem. Vecinom koraci su identicni kao u prethodna 2 slucaja, izuzev koraka instaliranja dodatnih paketa unutar modula.

```
1 cd ~/squashfs/livecdtmp
2 sudo mount -o loop ./isoimgs/ubuntu-18.04.4-desktop-amd64.iso mnt
3 mkdir extract-chrome-cd
4 sudo rsync --exclude=/casper/filesystem.squashfs -a mnt/ extract-chrome-cd
5 mkdir modul-chrome
6 sudo rsync -a extract-chrome-cd/ modul-chrome
```

Zatim slijedi korak u kojem se opet raspakuje filesystem.squashfs direktorij i kopiramo ga u edit-chrome direktorij. Ovaj put cemo edit direktorij imenovati edit-chrome da ne izgubimo prethodni sadrzaj edit direktorija.

```
1 sudo unsquashfs mnt/casper/filesystem.squashfs
2 sudo mv squashfs-root/ edit-chrome
```

Da bi imali mreznu konekciju unutar edit-chrome direktorija jedno rjesenje je kopirati /run direktorij unutar edit-chrome direktorija. Najbolje manuelno popuniti resolv.conf unutar edit-chrome direktorija

nameserver 1.1.1.1

nameserver 8.8.8.8.

Isto vazi i za etc/hosts datoteku. Najbolje je provjeriti nakon izvrsenih komandi da li je upisan sadrzaj u resolv.conf i hosts datoteke, te ukoliko nije dopuniti nedostatke:

```
sudo cp /etc/resolv.conf edit-chrome/etc/
sudo mount -o bind /run/ edit-chrome/run
```

Kopirati i hosts direktorij/:

```
sudo cp /etc/hosts edit-chrome/etc/
```

Namjestiti edit-chrome/dev direktorij kopirajuci /dev/ direktorij sa hosta, zatim chroot u edit-chrome direktorij. Obaviti mount instrukcije navedene ispod. Ukoliko korisnik odluci da obrise edit-chrome direktorij iz nekog razloga, bilo bi potrebno uraditi unmount edit-chrome direktorija da sistem domacin ne bi postao neupotrebljiv:

```
sudo mount --bind /dev/ edit-chrome/dev
sudo chroot edit-chrome
mount -t proc none /proc
mount -t sysfs none /sys
mount -t devpts none /dev/pts
```

Takodjer potrebno je izvrsiti sljedece komande da bi se izbjegli problemi sa lokalizacijom:

```
export HOME=/root
export LC_ALL=C
```

Za ispis svih instaliranih paketa:

```
dpkg-query -W --showformat='\${Installed-Size}\t\${Package}\n' | sort -nr | less
```

Instalacija google-chrome paketa:

```
sudo nano /etc/apt/sources.list.d/google-chrome.list
```

Te upisati u ovu datoteku sljedeci sadrzaj

```
deb [arch=amd64] http://dl.google.com/linux/chrome/deb/ stable main
```

Zatim spasiti datoteku unutar nano editora sa **CTRL+O**, **ENTER** za potvrdu i **CTRL+X** za izlaz iz nano editora.

Sljedeca komanda preuzima Google javni kljuc da bismo mogli instalirati google-chrome. Zatim komandom apt-key dodajemo kljuc u prsten javnih kljuceva da bi apt mogao potvrditi integritet Google Chrome paketa.

```
wget https://dl.google.com/linux/linux_signing_key.pub
sudo apt-key add linux_signing_key.pub
```

Sada izvrsimo azuriranje liste paketa i instaliramo google-chrome-stable paket:

```
sudo apt update
sudo apt install google-chrome-stable
```

Provjera instalacije:

```
google-chrome-stable --version
```

Nakon zavrsetka instalacije izvrsiti unutar chroot:

```
apt-get clean
rm -rf /tmp/* ~/.bash_history
rm -rf /tmp/* ~/.bashrc
rm /var/lib/dbus/machine-id
rm /sbin/initctl
dpkg-divert --rename --remove /sbin/initctl
umount /proc || umount -lf /proc
umount /sys
umount /dev/pts
exit
sudo umount edit-chrome/dev
```

Ponovno generisati filesystem.manifest:

```
sudo cp extract-chrome-cd/casper/filesystem.manifest extract-chrome-cd/casper/
    filesystem.manifest-desktop
```

```
2 sudo sed -i '/ubiquity/d' extract-chrome-cd/casper/filesystem.manifest-desktop
3 sudo sed -i '/casper/d' extract-chrome-cd/casper/filesystem.manifest-desktop
4 sudo umount edit/dev
```

Sada cemo upotrijebiti drugu funkciju iz squashfs-tools, a to je mksquashfs. S tom funkcijom cemo kompresovati edit-chrome direktorij u novu filesystem.squashfs datoteku. U kodu ispod je potrebno izvrsiti komandu iz linije 1 i jednu od preostale 3, pri cemu prva (komanda na liniji 2) daje najslabiju kompresiju, ali je najbrza. Druga komanda se duze izvrsava ali je veci procenat kompresije u odnosu na prvu komandu. Dok je kod trece komande procenat kompresije najveci, a vrijeme izvrsenja najduze:

```
1 sudo rm extract-chrome-cd/casper/filesystem.squashfs
2 sudo mksquashfs edit-chrome extract-chrome-cd/casper/filesystem.squashfs -nolzma
3 sudo mksquashfs edit-chrome extract-chrome-cd/casper/filesystem.squashfs -b 1048576
4 sudo mksquashfs edit-chrome extract-chrome-cd/casper/filesystem.squashfs -comp xz -e
    edit/boot
```

Naredni korak je da azuriramo filesystem.size datoteku:

```
1 sudo su
2 printf $(du -sx --block-size=1 edit-mysql | cut -f1) > extract-chrome-cd/casper/
    filesystem.size
3 exit
```

Nakon toga upisati naziv image-a unutar README.diskdefines. Upisati 'Ubuntu with Google Chrome 18.04.4 LTS "Bionic Beaver" - Release amd64' u polje DISKNAME:

```
1 sudo gedit extract-chrome-cd/README.diskdefines
```

Azurirati md5sum.txt datoteku:

```
1 cd extract-chrome-cd
2 sudo rm md5sum.txt
3 find -type f -print0 | sudo xargs -0 md5sum | grep -v isolinux/boot.cat | sudo tee md5sum
    .txt
```

Napokon mozemo napraviti iso image koji ce da sadrzi Google Chrome modul. Za ovu operaciju koristimo funkciju genisoimage. Neke linux distribucije nude mkisofs funkciju. Tako da ukoliko ne radi genisoimage trebala bi raditi funkcija mkisofs:

```
1 sudo genisoimage -D -r -V "$IMAGE_NAME" -cache-inodes -J -l -b isolinux/isolinux.
      bin -c isolinux/boot.cat -no-emul-boot -boot-load-size 4 -boot-info-table -o ../
      ubuntu-with-chrome-18.04-amd64.iso .
```

Sada cemo napraviti virtuelni hard disk pomocu qemu-img komande da bismo pokrenuli na njemu nas novi modul Google Chrome Ubuntu.

```
1 cd ~
2 qemu-img create ubuntuchrome.img 5G
```

Pokrenucemo modul pomocu KVM-a:

```
1 sudo kvm -hda ubuntuchrome.img -cdrom ~/zavrsni/livecdtmp/ubuntu-with-chrome
      -18.04-amd64.iso -boot d -m 2048
```