BRNO UNIVERSITY OF TECHNOLOGY

Faculty of Electrical Engineering

and Communication

BACHELOR‘S THESIS

Brno, 2022 Peter Milan Kluka

BRNO UNIVERSITY OF TECHNOLOGY

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

FAKULTA ELEKTROTECHNIKY
A KOMUNIKAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF TELECOMMUNICATIONS

ÚSTAV TELEKOMUNIKACÍ

REDUCTION OF CENTOS OPERATING SYSTEM

REDUKCE OPERAČNÍHO SYSTÉMU CENTOS

BACHELOR'S THESIS
BAKALÁŘSKÁ PRÁCE

AUTHOR
AUTOR PRÁCE

Peter Milan Kluka

SUPERVISOR
VEDOUCÍ PRÁCE

prof. Ing. Dan Komosný, Ph.D.

BRNO 2022

BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

Bachelor's Thesis

Bachelor's study program **Information Security**

Department of Telecommunications

Student: Peter Milan Kluka ***ID:*** 220893

Year of study: 3 ***Academic year:*** 2021/22

TITLE OF THESIS:

Reduction of CentOS operating system

INSTRUCTION:

Reduce the CentOS Stream operating system from the viewpoint of its size on storage media. The reduced system has to include a graphical web browser, PDF file viewer, and remote access by SSH. Work with the system programs and kernel modules when reducing the system. Create a set of scripts, which will automate the reduction process.

RECOMMENDED LITERATURE:

[1] Linux Dokumentační projekt. 4. vyd. Computer Press, 2008. 1336 s. ISBN: 978-80-251-1525-1.

[2] COOPER, M. Advanced Bash-Scripting Guide. Lulu.com, 2010. ISBN: 978-14-357-5218-4.

Date of project specification: 7.2.2022 ***Deadline for submission:*** 8.8.2022

Supervisor: prof. Ing. Dan Komosný, Ph.D.

doc. Ing. Jan Hajný, Ph.D.
Chair of study program board

ABSTRACT

This thesis serves as a supporting material for the course Networking Operating Systems for the implementation of the course project. This project has an artificial goal which is to reduce the size of the operating system. The specified operating system was the Linux distribution CentOS Stream 8. The reduced system includes a graphical web browser, a PDF document viewer and is accessible via a remote SSH connection. The thesis compares sizes of required graphical frameworks and applications. Based on the analysis a script was created to automate the minimalization process. The resulting size of the reduced system was 650 MB.

KEYWORDS

Reduction, Linux, CentOS Stream, RHEL, Operating system, Lightweight, Web browser, SSH, Script

ABSTRAKT

Táto práca slúži ako podporný materiál pre predmet Sieťové operačné systémy a ako pomocný materiál pre realizáciu projektu v rámci tohto predmetu. Účelom tejto práce bolo vykonať redukciu operačného systému z hľadiska jeho veľkosti na disku. Zadaným operačným systémom bola Linuxová distribúcia CentOS Stream 8. Redukovaný operačný systém obsahuje grafický web prehliadač, PDF prehliadač dokumentov a je dostupný pomocou vzdialeného pripojenia SSH. Práca porovnáva veľkosti odporúčaných grafických rozhraní a aplikácií. Na základe analýzy bol vytvorený skript, ktorý zautomatizoval proces minimalizácie. Výsledná veľkosť redukovaného systému bola 650 MB.

KĽÚČOVÉ SLOVÁ

Redukcia, Linux, CentOS Stream, RHEL, Operačný systém, Webový prehliadač, SSH, Skript

KLUKA, Peter Milan. *Reduction of CentOS operating system* [online]. Brno: Brno University of Technology, Faculty of electrical engineering and communications, Dept. of telecommunications, 2022 [cit 2022-08-05]. 82p. Bachelor's thesis. Accessible from: https://www.vutbr.cz/studenti/zav-prace/detail/145362. Advised by prof. Ing. Dan Komosný, Ph.D.

ROZŠÍRENÝ ABSTRAKT

Táto bakalárska práca sa venuje redukcii operačného systému CentOS Stream. Cieľom bolo zmenšiť jeho veľkosť na disku pomocou automatizovaného skriptu, pričom musel byť stále funkčný a schopný opätovného štartu. Systém musel obsahovať grafický webový prehliadač, PDF prehliadač dokumentov a možnosť vzdialeného pripojenia pomocou SSH.

V teoretickej časti sa popisujú základné časti operačného systému Linux, medzi ktoré patria ovládače, moduly jadra, grafické prostredie a aplikácie. Každej z týchto základných častí je venovaná samostatná podkapitola, v ktorej sú definované. Následne sú objasnené dôvody a význam v operačnom systéme, ako aj popísané vlastnosti vybraných priečinkov a podpriečinkov z hľadiska hierarchie a ich význam v rámci operačného systému. Okrem významu sú vysvetlené aj typy súborov, ktoré sa v daných priečinkoch nachádzajú a k čomu sú potrebné. V ďalšej časti práce je popísaný štart operačného systému a predstavené rôzne Linuxové distribúcie. Jednou z predstavených distribúcií je aj Red Hat Enterprise Linux, ktorá je vyvíjaná v zadanom systéme CentOS Stream. CentOS Stream má na rozdiel od Red Hat-u, ktorý je dostupný pre komerčné účely, širokú podporu od komunity a je voľne dostupný.

V praktickej časti je predstavený postup a príprava redukcie operačného systému, rôzne metódy a inštalácia minimalistických aplikácií, ktoré si zadanie tejto práce vyžadovalo. Analýza operačného systému sa zameriava na zistenie veľkosti potrebných aplikácii a ich minimalistických náhrad, ktoré mali značný vplyv na výslednú veľkosť redukovaného systému. Okrem aplikácií sa tiež skúmalo, ktoré súbory je možné odstrániť bez toho, aby ovplyvnili beh operačného systému celkovo, alebo len minimálne. Analýza okrem iného zahŕňala aj postup redukcie operačného systému. Bez obmedzenia chodu systému mohli byť odstránené dočasné súbory, manuálové stránky, preklady a rôzne dodatočné aplikácie. Pri redukcii bol zvolený automatizovaný postup podľa zadania práce, ktorý zahŕňa skript. Kvôli inštalácií dodatočných aplikácií je nutný prístup na internet. Aby nedošlo k odstráneniu balíčkov, nutných pre chod systému počas vykonávania skriptov, bolo nutné ich pridať do výnimiek. Keďže viaceré dôležité aplikácie zdieľajú rovnaké balíčky, boli tieto aplikácie tak isto uchránené pred nechceným odstránením a znefunkčnením celého systému.

Podkapitola Automated reduction obsahuje ukážky skriptu, ktoré boli vykonávané. Okrem toho je ku každej časti skriptu popísané, či sa miesto pridalo, alebo uvoľnilo. Pre automatizovanú metódu redukcie bol skript rozdelený na tri základné časti. V prvej časti sa vykonala dodatočná inštalácia balíčkov a minimalistických aplikácií, v druhej časti

nasledovala redukcia systémových balíčkov a v tretej časti redukcia adresárov a podadresárov. Nutnosťou takto redukovaného systému je jeho opätovné spustenie. Minimalistické náhrady boli volené podľa toho, akú veľkosť mali po nainštalovaní a nie, akú veľkosť mala samotná inštalácia. Je to z toho dôvodu, že aj keď samotná inštalácia nezaberá veľa miesta, systémové balíčky môžu túto veľkosť zväčšiť. Kvôli tomu bol zvolený webový prehliadač Otter-Browser, PDF prehliadač XPDF, terminál XTerm a grafické rozhranie Xfce. Okrem Otter-Browseru existujú aj rôzne iné menšie prehliadače, avšak ani jeden z nich neobsahuje JavaScript, ktorý je potrebný pre správne zobrazenie webových stránok. Najzdĺhavejšou časťou tejto práce je redukcia mnohých systémových balíčkov, ktoré boli prechádzané jeden po druhom. Po redukcii adresárov a podadresárov boli odstránené aj knižnice a balíčky vrátane tých, ktoré tvorili výnimky. Toto odstránenie nespôsobilo znefunkčnenie celého operačného systému, ale iba určitých častí. Patrí sem ovládací panel, navigačná lišta, zvuk a prehliadač súborov. Keďže sa dajú všetky potrebné aplikácie spustiť cez terminál, nie je potrebné, aby fungovali priamo z plochy. Pri opätovnom štarte systému je plocha tvorená iba kurzorom a terminálom XTerm, ktorý je používaný na spúšťanie aplikácií a vykonávanie príkazov.

V podkapitole Start of the OS after reduction je popísaný postup po reštartovaní systému, alebo po jeho opätovnom spustení. Keďže boli odstránené viaceré grafické prvky systému, je nutné najskôr sa prihlásiť v konzole a potom manuálne spustiť minimalizované grafické rozhranie Xfce.

Záverečná podkapitola Summary of reduction zhŕňa všetky výsledky tejto bakalárskej práce. Okrem toho informuje, ako sa zmenili veľkosti jednotlivých priečinkov a podpriečinkov po vykonaní jednotlivých skriptov, vrátane toho, akú veľkosť samotný systém postupne nadobúdal.

Výsledná veľkosť redukovaného systému bola 650 MB. Systém bol zbavený softvéru a súborov, ktoré neboli potrebné pre úspešné splnenie zadania. Všetky zadané podmienky práce boli splnené. Systém je schopný zobrazenia webovej stránky v nezmenenej podobe, skript v PDF prehliadači a byť vzdialene pripojený pomocou SSH. Skript je ďalej možné použiť s miernou úpravou aj na redukcie iných operačných systémov, ako napríklad Fedora alebo RHEL, na ktorých bol otestovaný. Skript bol testovaný na verzii CentOS-Stream-8-x86_64-latest-dvd1.iso. Možné využitie tohto systému je vhodné pre knižnice, zariadenia s obmedzenou pamäťou alebo výkonom a ako pomôcka pri plnení školského projektu.

Author's Declaration

Author:	*Peter Milan Kluka*
Author's ID:	220893
Paper type:	*Bachelor's Thesis*
Academic year:	*2021/22*
Topic:	*Reduction of CentOS operating system*

I declare that I have written this paper independently, under the guidance of the advisor and using exclusively the technical references and other sources of information cited in the project and listed in the comprehensive bibliography at the end of the project.

As the author, I furthermore declare that, with respect to the creation of this paper, I have not infringed any copyright or violated anyone's personal and/or ownership rights. In this context, I am fully aware of the consequences of breaking Regulation § 11 of the Copyright Act No. 121/2000 Coll. of the Czech Republic, as amended, and of any breach of rights related to intellectual property or introduced within amendments to relevant Acts such as the Intellectual Property Act or the Criminal Code, Act No. 40/2009 Coll., Section 2, Head VI, Part 4.

Brno, July 27, 2022 ----------

author's signature

ACKNOWLEDGEMENT

I would like to thank my supervisor, prof. Ing. Dan Komosný, Ph.D. for leading the thesis, for support, help, professional suggestions, and comments. I would also like to thank my parents, brother, and friends for their support.

Contents

List of Figures

List of tables

List of Listings

Introduction

This bachelor's thesis deals with minimization in one of Linux's distributions—CentOS Stream 8. I am doing this thesis as supporting material for a project from the Networking Operating Systems course. This distribution could be used on devices with limited memory. Thanks to minimization, the final size of the operating system was comparable to other Linux minimized distributions like Knoppix, Bodhi, or AntiX. These distributions take up 700 MB–900 MB of space on a storage medium. The main difference is in the minimization methods of these operating systems. Every operating system is built similarly – first the main part is the core (kernel), and then other packages for smooth operation are added. The minimization was executed according to assignment on CentOS Stream 8, which was released in September 2019. The latest version used in minimization was released in July 2022. The reduced operating system had to include a graphical web browser, a PDF file viewer, and remote access by SSH. The minimization was performed on a minimal installation of CentOS Stream. First, I installed an Otter-Browser, XPDF, XTerm, and Xfce using a script, and then I started to reduce the operating system with an automated script by removing all necessary programs, packages, icons, drivers, fonts, etc. This bachelor's thesis focuses only on Linux as an operating system, which is what chapter 1 deals with. Chapter 2 focuses on parts of the Linux operating system, such as the kernel, servers, applications, etc. Subchapter 2.3 deals with background services and demons, which monitor and take care of certain subsystems to ensure that the operating system runs properly. The SSH protocol belongs here as well. In subchapter 2.4, the graphical environment and display server are described, and this chapter explains the most commonly used display server on Linux with other GUI. In chapter 3, the start of the OS and bootloader, as well as GRUB and BIOS, is described. Chapter 4 focuses on Linux distributions like Red Hat Enterprise Linux, CentOS Stream, Fedora, Debian, and Ubuntu. The practical part begins with chapter 5, where the analysis of CentOS Stream begins. Subchapters 5.6 and 5.7 are focused on the installation of a lightweight web browser and GUI. In these chapters, there is also a comparison of the different lightweight web browsers and GUIs. In subchapter 5.8, are parts of script that were used in automated reduction and the results of every part of the script. Subchapter 5.9 describes the start of the operating system after reduction performed by an automated script. The results of this performed reduction are in Subchapter 5.10.

1 Linux operating systems

Linux was created in 1991 by Linus Torvalds when he was studying at the University of Helsinki. He built the Linux kernel as a free, open-source operating system (OS) that was used in academic settings. Like other operating systems, we can find here a graphical interface, video editors, audio editors, word processors, and other types of software. [1]

This open-source operating system which is developed by the community, is suitable for everyone because users can choose from many distributions. Companies choose Linux for servers due to its secure, flexible, and excellent support from the community of users. It is widely used on devices with different platforms, like e.g., ARM or x86. [2]

2 Parts of Linux operating systems

2.1 Kernel

Kernel is a piece called Linux. It serves as the basis for the Linux operating system. It is the lowest level of software capable of interacting with computer hardware, applications, and servers. Each Linux distribution is built on the Linux kernel, which utilise its services to create a wide variety of software features. Linux is using a monolithic kernel and hybrid kernels are used by Microsoft Windows and Max OS X.

Lots of benefactors across the world help to develop and modify Linux kernel, which maintain support for six years. License used for the Linux kernel release is GNU GPLv2. [3]

2.2 Kernel modules

Chunks with a code that are dynamically transferred into the kernel are called kernel modules. Purpose of kernel modules is to increase the capabilities of the kernel without rebooting the system. There are two options of configuration of these modules. First option is "built-in" and second is "loadable". This second option used so kernel can dynamically add or remove modules.

Utilizing kernel modules offers a few benefits, like help to identify system faults, bugs in drivers, etc. For example, if the system crashing is caused by a corrupted driver, this driver can be unloaded, and the system can work without any issues. Modifying of a code without restarting can be done using commands. Another benefit is that dynamically loaded modules take up memory only when they are needed by the system. These modules can be found in either */lib/modules/$(uname -r)/kernel* or */usr/lib/modules/$(uname -r)/kernel*. [4] [5]

2.3 Servers

Servers are background services that start up during boot, or after you log into the desktop. These support programmes (daemons) monitor various subsystems and make sure everything is running well in the background. They are not running all the time, but only when they need to carry out specific tasks at predetermined periods. The number of daemons is different on every system. Since daemons are not directly controlled by a user, they perform an important role in the proper functioning of the operating system. [6]

Secure Shell

Using Secure Socket Shell (SSH) or Secure Shell provides users a secured connection to a computer located on an unsecured network. The main parts of this network protocol are encrypted data transferred between computers, then password and public key authentication.

The SSH protocol is based on the client-server model. The client is a user who is connecting to the end station (SSH server). The SSH is using daemon called "sshd". While establishing connection is used hashing as well as symmetric encryption. This ensures secure connection, key exchange, and safe data transfer. Simplified SSH connection and key exchange is described in the figure 2.1. [7]

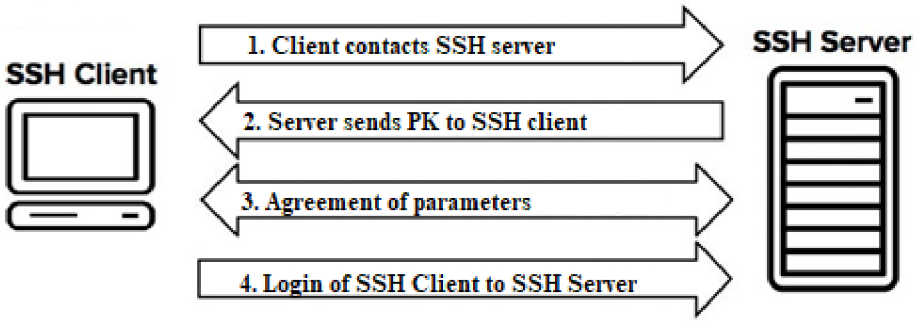

Figure 2.1: Simplified SSH connection

2.4 Graphical environment

A lot of operating systems include a server, which makes the manipulation of systems faster and easier for their users. This server is called a display server and offers a graphical user interface (GUI). It is a program which main task is to receive inputs from client and according to them show appropriate outputs on the display.

In order to use hardware for communication with GUI, the display server contains a protocol. Different operating systems are using different display server protocols, but Linux distributions are using mostly the X11 display server protocol. The display server that the X11 protocol uses is called X.org and takes care of client inputs and outputs. Data processing can be done in any of these components: KMS driver, DRM, or gem. [8]

The GUI includes media elements like wallpapers, themes, icons, and so on, and it is possible to use a keyboard and mouse for interaction with these elements. Systems that don't use GUIs looks like a terminal or a command line. GNOME, Xfce, KDE, and Mate are among the most popular GUIs. [9]

2.5 Applications

Like any other OS, Linux offers a lot of software titles that can be easily found and installed. For example, Ubuntu Linux has the „Ubuntu Software Centre", which allows you to search

for apps and install them. Most Linux distributions have app stores that help you find and install apps. [10]

2.6 Linux filesystem hierarchy standard

The FHS (Filesystem Hierarchy Standard) specifies the content and structure of directories and subdirectories. This hierarchy refers to Linux operating systems. The main directory is called "root" (/) and all other subdirectories and files are under this directory. It is not necessary that all these directories are included in different systems, as it also depends on the components which are included. Generally, it is possible to find vast majority of directories and subdirectories from figure 2.2 in every Linux operating system. [11]

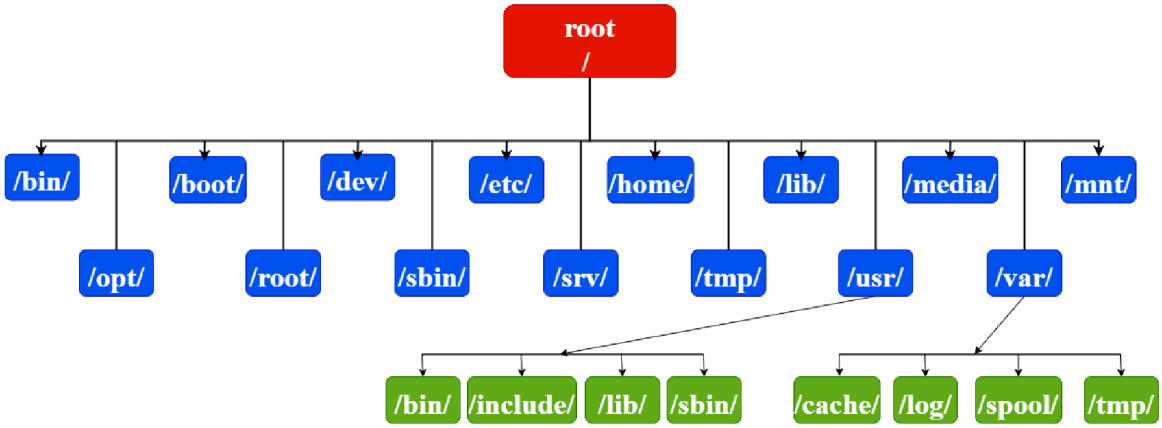

Figure 2.2: Linux FHS

/Root

Directory of the main user – root. It is accessible only to this user, who can use every Linux command and every file. Every other directory that is in the operating system in in this directory. [12]

/bin

Directory of binary files accessible to every user. Besides that, here are located commands along with executable files and no other directories. [13]

/boot

In this directory are located files, which are loaded during the boot. From this directory should not been anything removed but removing old unused kernel won't cause crash or anything like that. [14]

/dev

Directory of device files, which are used by both user and computer. Every user has access to these files. Operating system needs device files to execute boot without any crashes. [15]

/etc

Directory which contains files used for configuration of system applications. Configuration files have a suffix ".conf". [16]

/home

Every user created in Linux operating system has their own home directory. This directory is accessible only to its owner or root. Content of /home directory is mostly configuration files or cache files. [17]

3 Start of the OS

3.1 Init system

The main process present in the system is called ***init***. Init has full control of daemons, processes, and the boot process. That means it can start or terminate any daemon, process, or service. [10]

The kernel is responsible for the start of init. This happens during the boot. After the start, its primary function is to run scripts, which are in the file */etc/inittab*, to execute new processes. If the init doesn't start while booting, a state that will happen is called "kernel panic". As the main process has init identifier of 1. [18] [19]

3.2 Bootloader

A programme which is responsible for the boot process is called a bootloader. Its task is to posture the operating system to RAM during the boot process. With every start of the computer, a series of tests are executed by BIOS. Following that, BIOS passes the control to the MBR. Although there are several bootloaders in various Linux operating systems, GRUB (Red Hat), LILO, and LOADLIN (used in multi-boot PCs) belong to the favourites. [20]

4 Distributions

There are many versions that Linux can offer us. These versions can suit any type of user. These different versions are called "distributions". Most distributions can be downloaded and installed for free. Among the most popular distributions are: Linux Mint, Manjaro, Debian, Ubuntu, Fedora, Elementary OS etc.

Different people and organizations work on different parts. There's the Linux kernel (the core of operating system), the GNU shell utilities (the terminal interface and other commands you use), the X server (which produces a graphical desktop), the desktop environment (which runs on the X server to provide a graphical desktop), and more. [21]

4.1 Red Hat Enterprise Linux

The two types of using operating systems are either for your own purpose or for commercial activities. Red Hat Enterprise Linux (RHEL) belongs to the second option. Its utilization is found mainly in servers. RHEL was created in 1994, following his development until now. RHEL is closely related to the CentOS Stream, and it is now the development platform for the next releases of RHEL. [22]

As one from many other distributions, there can be found a Linux kernel and open-source too. Besides that, it includes default Linux applications as well as applications specified for this version. Not only CentOS is closely related to the RHEL, but Oracle Enterprise Linux and Pie Box Enterprise Linux too. Red Hat wasn't always charged for money. In the beginning user was charged when he needed support, but it changed and then charged RHEL was created. During the development of the new operating system, ordinary users were also thought of, and a new Fedora operating system was created, which contains fewer functions than RHEL and is free. Although Fedora isn't as stable as RHEL, its packages are modified to be stable and then used in RHEL. RHEL contains default software as web browser, document viewer, and then services, programming software, development tools, etc. [23]

4.2 CentOS Stream

Since CentOS has ended support, CentOS Stream has become its successor. It is based on the same architecture as Fedora, but with a few modifications. Same as RHEL, CentOS Stream is more stable than Fedora. Unlike RHEL, CentOS Stream is open-source and can be used and modified by anyone. [24]

The main goal was to use every RHEL packages and made this operating system free to use. There is no charge for support because it has wide community support, and it is completely free. The CentOS Stream encountered plenty of setbacks. One of the biggest setbacks was the private code to the operating system, so the community members had to improvise and made it themselves. Another setback was the lack of time that volunteers had. When CentOS was almost extinguished, Red Hat itself helped to finish this community project by sharing the critical parts. [25]

4.3 Fedora

Fedora belongs in the same category as CentOS Stream. It is mainly because Fedora was developed during the development of RHEL. Although it does include fewer functions than RHEL, it is secure and free to use. The support of every version is six months. It is popular among Linux users and, statistically, is right behind the Ubuntu. [26], [27]

Like CentOS Stream, Fedora is supported by community members. These members are called the Fedora Project, and they take care of previous versions and next releases. Every Fedora version is using a security mode called "*Security-Enhanced Linux*". [28]

4.4 Debian

Debian with Ubuntu and Fedora are the most used operating systems. Like Fedora shares some parts of the source code with CentOS Stream, Debian shares mainly with Ubuntu. Community members help to develop new releases with a leader. Debian is not only the most used, but also belongs to the first operating systems to use Linux kernel. Besides, that, it is open source. [29]

4.5 Ubuntu

Ubuntu is using Debian's part of the source code. It is the most user-friendly distribution, which makes it easy to use even for beginners. Security updates are a sure thing. Everything is planned, from the update to the end of support. This operating system is free, and money for development are achieved by donations or providing "*special services*". Ubuntu is available not only in the desktop version, but in widely used server version too. [30]

5 Manual reduction of CentOS Stream operating system

5.1 Analyse of CentOS Stream

Before reduction, we need to know what can be removed, while the operating system stays working. Removing the wrong package will mostly lead to a non-working operating system or some of its main functions. The next thing is to check if there are some alternatives for the programmes and environment. Alternatives, which will take up less storage on a hard drive than the default software pre-installed with our operating system.

In my case, I used minimal installation of the operating system, so I had to download the internet browser, lighter terminal, and graphical environment with as small installation size as possible. The script has been tested on the "*CentOS-Stream-8-x86_64-latest-dvd1.iso*" version. The network connection is turned off by default. So, the first option is to turn it on before installation in the menu, or the second option is to use the command "nmtui" and manually activate it using the GUI. The second option will not be available after reduction, because this command will also be removed. It's important to active the internet connection before reduction. How to activate internet connection in the installation menu can be seen in the figures B.2 and B.3. I chose as a graphical environment Xfce, the internet browser Otter-Browser, the terminal XTerm, and then I added the XPDF browser because Otter-Browser does not support viewing PDF files. In table 5.1 under this section, you can see the differences in the sizes of some graphical environments, web browsers and others, like terminal, etc.

Desktop environment	**Installed size (MB)**
GNOME	2491
Xfce – installed	1527
Internet browsers	
Firefox	220
Otter-Browser – installed	110
Others	
Terminal	1.9
XTerm – installed	0.9
XPDF – installed	15,8

Table 5.1: Size comparison of software

As can be seen from the table 5.1, just by choosing these two programmes and the environments to their lightweight versions, I was able to save up to 1050 MB. Even more space I saved by removing packages that I didn't use for any other software.

The greatest emphasis was placed on the size of various folders in the filesystem hierarchy structure and the next parts that communicate with the kernel, like modules, software packages, libraries, and other software. Thanks to this analysis, I was able to create a solution and find the right tools, which I used to perform a reduction. Because every distribution of Linux is different, any other solution would not work correctly and would not be effective enough, which is why my solution will not work with any other distribution. The reduction was done on minimal CentOS Stream 8 installation, which was downloaded in July 2022.

5.2 Reduction of temporary files

Every operating system has folders where temporary files are stored. Between these files are files from the internet browser, log files, and cache files, where applications store information about their state. Temporary files have a wide choice of utilization. Such as when a programme cannot allocate enough memory for its tasks, or when the programme is working on data bigger than the architecture's address space. Most temporary files are deleted after the application is closed. On Linux, temporary files are stored in the directory /var. The log files were first, which I removed.

```
rm /var/log/messages
find /var/log -name '*.log*' -delete
```

Listing 5.1: Removing log files and log messages

Log files have a suffix ".log" and, thanks to command "**find",** I was able to find them and then remove them. Any user should never modify or remove log files because they are important for the correct operation of the operating system. They contain significant information, thanks to which can help solve problems with the system or restore it to a state when it was operating correctly.

Cache files are not that important, and if some application needs them, it will just generate a new one. Temporary files are better at storing information because they exist only a certain time and deleting them while using a programme is critical for its correct functionality. For our minimization, we can use this folder to delete every file in it. The command

rm, which is responsible for deleting files and folders, was used in this case. In listing 5.2 is shown the whole command with a path.

```
rm -rfv /var/cache/*
rm -rfv /home/user/.cache/*
```

Listing 5.2: Removing cache files

Removing temporary files, in my case, can save up to 200 MB of storage. It depends on applications installed in the operating system and previously used. However, due to generating new temporary files with every start of a different application, the best way is to remove these files after we have done working with them.

5.3 Reduction of utility software

This software is suggested to improve the computer's performance, system management, and its user operation. It cooperates with both software and hardware. Most common tasks performed by utility software are disk fragmentation, virus detection, data recovery, etc. Subdirectories in the directory /usr include manual pages, software documentation, and information. The directory /usr/share contains several subdirectories, including /icons, /themes, /sound, /zoneinfo, /doc, /man, /info and /locale. The largest directory is /locale, which includes a set of parameters that define the user's language, region, and any other preferences that the user wants to see in their user interface. An identifier locale includes a language code and a country or region code. Number, character classification, date-time, string, currency, paper size, and colour format are examples of locale settings. User chooses localization while installing the operating system or during use. The size of this subdirectory is 280 MB. Localization, which is the operating system currently being used, can be checked with a command locale.

```
[root@localhost ~]$ locale
LANG=en_US.UTF-8
LANGUAGE=
LC_CTYPE="en_US.UTF-8"
LC_NUMERIC="en_US.UTF-8"
LC_TIME="en_US.UTF-8"
LC_COLLATE="en_US.UTF-8"
LC_MONETARY="en_US.UTF-8"
LC_MESSAGES="en_US.UTF-8"
```

```
LC_PAPER="en_US.UTF-8"
LC_NAME="en_US.UTF-8"
LC_ADDRESS="en_US.UTF-8"
LC_TELEPHONE="en_US.UTF-8"
LC_MEASUREMENT="en_US.UTF-8"
LC_IDENTIFICATION="en_US.UTF-8"
LC_ALL=
```

Listing 5.3: Currently used localization of the system

From listing 5.3, it is obvious that the currently used localization is "en_US". Files that correspond can be left behind, and others can be removed. This leads to removing localizations that are not used. The next step is removing manual pages, software documentation, information files, some icons, themes, sounds, and others.

```
rm -rfv /usr/share/doc/
rm -rfv /usr/share/man/
rm -rfv /usr/share/info/
rm -rfv /usr/share/icons/Adwaita/
rm -rfv /usr/share/icons/hicolor/
rm -rfv /usr/share/sounds/
rm -rfv /usr/share/zoneinfo/
rm -rfv /usr/share/theme/Adwaita/
```

Listing 5.4: Reduction of directory /usr/share

While reduction, some graphical parts of the operating system were also removed. This does not affect the functionality of the operating system because everything is performed via terminal XTerm. Although these files weren't that large, I was trying to remove everything I could.

5.4 Kernel modules reduction

As I mentioned in subchapters 2.1 and 2.2 what kernel and a kernel modules are, here is described, how the reduction is performed. The list of all the modules in the system can be displayed using the command **lsmod**. Every system has a different number of modules. In my case, it was 103. In the listing 5.5, the first 28 modules are shown.

```
[root@localhost ~]$ lsmod
Module                  Size  Used by
```

uinput	20480	1
nls_utf8	16384	1
isofs	49152	1
rfcomm	86016	4
xt_CHECKSUM	16384	1
ipt_MASQUERADE	16384	3
xt_conntrack	16384	1
ipt_REJECT	16384	2
nft_compat	20480	16
nf_nat_tftp	16384	0
nft_objref	16384	1
nf_conntrack_tftp	16384	3 nf_nat_tftp
nft_counter	16384	33
tun	49152	1
bridge	204800	0
stp	16384	1 bridge
llc	16384	2 bridge,stp
nft_fib_inet	16384	1
nft_fib_ipv4	16384	1 nft_fib_inet
nft_fib_ipv6	16384	1 nft_fib_inet
nft_fib	16384	3 nft_fib_ipv6,nft_fib_ipv4,nft_fib_inet
nft_reject_inet	16384	5
nf_reject_ipv4	16384	2 nft_reject_inet,ipt_REJECT
nf_reject_ipv6	16384	1 nft_reject_inet
nft_reject	16384	1 nft_reject_inet
nft_ct	20480	18
nf_tables_set	49152	
nft_chain_nat	16384	

Listing 5.5: List of currently established modules

From this listing, two columns are important. The first is a **Module,** while the second is **Used by**. In the Module column are the names of modules, and in the column Used by is the number of kernel or process, which module is currently used. If a module has a value of 1 in the column Used by, then it is currently used and cannot be removed. If a module has a

different value than 1 in the column Used by, then it is not currently used and can be removed. However, if another module from the same group has a value of 1, the module cannot be removed, because it is dependent on that module. In a later chapter about removing software packages, some of these modules will be taken out.

The next larger-sized files, which can be removed, and which are not usually used, are the rescue files. Usage of these files is required when the operating system crashes or when a problem with software or hardware occurs. These rescue files are always created by the **dracut-config-rescue** package when the operating system is installed. Rescue files are stored in the ***/boot*** directory and are easily recognizable because they contain "rescue" in their name. The size of rescue files varies for every distribution and for every user. In my case, I can save up to 125 MB, which is the size of half of the folder where they are stored.

```
[root@localhost root]# find /boot -name *rescue*
/boot/loader/entries/a15e9d1a5d2c44dba56e987f8d333963-0-rescue.conf
/boot/vmlinuz-0-rescue-a15e9d1a5d2c44dba56e987f8d333963
/boot/initramfs-0-rescue-a15e9d1a5d2c44dba56e987f8d333963.img
```

Listing 5.6: List of rescue files

A computer has software called firmware. The device has software, called a driver, that tells the operating system how to communicate with another device. Firmware packages are loaded every time while booting and are loaded into RAM. Deleting every firmware package would lead to a non-functioning operating system. However, deleting those packages, which are not hidden in other folders, does not affect the proper functioning of the operating system. In hidden folders are packages that the kernel needs to boot and be able to communicate with users. Unprotected packages are located at */usr/lib/firmware* and can be removed with the command **rm** and with the added parameter **-rfv**, which can remove a whole folder.

```
[root@localhost]# rm -rfv /usr/lib/firmware
removed '/usr/lib/firmware/3com/3C359.bin'
removed '/usr/lib/firmware/3com/typhoon.bin'
removed directory '/usr/lib/firmware/3com'
removed directory '/usr/lib/firmware/RTL8192E'
removed '/usr/lib/firmware/dvb-usb-it9135-01.fw'
removed '/usr/lib/firmware/qed/qed_init_values-8.10.9.0.bin'
removed '/usr/lib/firmware/advansys/3550.bin'
```

```
removed '/usr/lib/firmware/advansys/38C0800.bin'
```

Listing 5.7: Removing unnecessary firmware

A driver that allows operating system and applications to use a computer's graphics hardware is called a graphics driver. It controls how graphic components work with the computer. A graphics driver is not necessary, but when it's installed, it provides more system stability. On Linux, graphics drivers are in the */lib64/dri* directory. There are drivers for various graphics cards, including those that are not physically present in a computer. These drivers have a size of more than 200 MB and are recognizable by the suffix ".so". Deleting is done by the command **rm**.

```
rm /lib64/dri/i965_dri.so
rm /lib64/dri/nouveau_dri.so
rm /lib64/dri/r600_dri.so
rm /lib64/dri/radeonsi_dri.so
rm /lib64/dri/virtio_gpu_dri.so
rm /lib64/dri/vmwgfx_dri.so
```

Listing 5.8: Deleting graphics drivers

5.5 Reduction of software packages

Each Linux distribution is usually installed as a different software package, each of which contains a specific application, such as a web browser. In CentOS Stream, packages can be installed or removed with the **YUM** package management system. Software packages take up a lot of disk space because they are necessary for the proper functioning of the system and for installed programs. Deleting all packages with all dependent files and folders would free up a lot of space, but the operating system and other installed programmes would stop working correctly. That's because many packages are dependent on other packages, or they provide stable functioning of the operating system and installed programs. The command **yum list installed** can show all installed packages. The number of packages varies for each distribution of Linux.

```
[root@localhost]# yum list installed
Installed packages
GConf2.x86_64                    3.2.6-22.el8          @AppStream
ModemManager.x86_64              1.10.8-4.el8          @anaconda
ModemManager-glib.x86_64         1.10.8-4.el8          @anaconda
```

NetworkManager.x86_64	1:1.34.0-0.2.el8	@anaconda
NetworkManager-adsl.x86_64	1:1.34.0-0.2.el8	@anaconda
NetworkManager-bluetooth.x86_64	1:1.34.0-0.2.el8	@anaconda
NetworkManager-config-server.noarch	1:1.34.0-0.2.el8	@anaconda
…		

Listing 5.9: List of a few installed packages

From listing 5.9, three columns can be seen. In the first column is the name of the package; in the second is the current version; and in the third column is information about which installer and when they were installed. For example, **@anaconda** means, that this package was installed during the installation of the operating system, **@AppStream** means, that packages were installed selectively depending on the purpose for which the system is being configured, and **@baseos** indicates the packages that implement the base core functionality of the operating system.

As I mentioned earlier in this chapter, removing every package would result in a non-functioning operating system or installed software. Therefore, main packages, which are responsible for proper functioning, must be protected. The defined packages, in my case, otter-browser, NetworkManager, openssh, xterm, xfce4-session, XPDF, and others must not be removed. The network manager is responsible for configuration and connection functionality on the network. Deleting would result in the loss of internet connection and the inability to load a web page in Otter-Browser or any other web browser. Without openssh, the connection from another computer would not work. Because of the removed icons and other packages that were responsible for some functions from GUI, it is not possible to start programs from menu, but directly from the command line using the Xterm terminal. Thanks to XPDF, it is possible to open files with the ".pdf" suffix, which will be needed later. The last one xfce4-session is responsible for installed lightweight GUI. Every single package mentioned will be written to a file with the same name and with **".conf"** suffix added into the */etc/dnf/protected.d/* directory, because every time while removing packages, this folder is reviewed. If a match is found with the package being removed, the package is skipped, and a warning message is displayed. Error messages showed in the listing 5.12 "*Error: Problem: The operation would result in removing the following protected packages:*" are fine and do not prevent the minimization process.

```
echo “openssh” > /etc/dnf/protected.d/openssh.conf
echo “otter-browser” > /etc/dnf/protected.d/otter-browser.conf
echo “NetworkManager” > /etc/dnf/protected.d/NetworkManager.conf
echo “xterm” > /etc/dnf/protected.d/xterm.conf
echo “xfce4-session” > /etc/dnf/protected.d/xfce4-session.conf
echo “xpdf” > /etc/dnf/protected.d/xpdf.conf
```

Listing 5.10: Protecting packages from removal

After protecting packages, the removal may begin. The CentOS Stream offers the removal of packages and all their dependencies with the command **yum autoremove package_name.** Every time before removing a package, a message shows up to confirm or deny an action. This message can be skipped by adding the parameter **-y**.

```
[root@localhost]# yum autoremove baobab.x86_64
Dependencies resolved.
================================================================
 Package       Architecture   Version              Repository        Size
================================================================
Removing:
 baobab        x86_64         3.28.0-4.el8         @AppStream        1.4 M
Transaction Summary
================================================================
Remove  1 Package
Freed space: 1.4 M
Is this ok [y/N]: y
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                        1/1
  Erasing          : baobab-3.28.0-4.el8.x86_64             1/1
  Running scriptlet: baobab-3.28.0-4.el8.x86_64             1/1
  Verifying        : baobab-3.28.0-4.el8.x86_64             1/1
Removed:
```

```
  baobab-3.28.0-4.el8.x86_64
Complete!
```

Listing 5.11: Removing unprotected package using yum

```
[root@localhost]# yum autoremove NetworkManager.x86_64
Error:
 Problem: The operation would result in removing the following protected packages:
NetworkManager
(try to add '--skip-broken' to skip uninstallable packages or '--nobest' to use not only
best candidate packages)
```

Listing 5.12: Attempt to delete protected package

5.6 Selection and installation lightweight graphical environment

Choosing a graphical environment isn't as difficult as it may seem. The two main conditions were to keep the size as small as possible and to still be able to show the web browser in full resolution with all colours. I had no graphical environment preinstalled, so I had a few environments to select from. In Table 5.2, different environments are compared by size. Since the main thing is to keep the size as small as possible, I opted for the smallest possible one. The most suitable for fulfilling the assignment was a graphical environment called Xfce.

Table 5.2: Comparison of graphical environments

Desktop environment	Installed size (MB)
GNOME	2491
KDE	2199
Xfce	1527
LXDE	1539
MATE	1642
Cinnamon	2223

Installation of Xfce was easy and took only a few minutes. First, I needed to configure the EPEL repository, as it was where I was installing packages from. EPEL (Extra Packages for Enterprise Linux) is an open-source and free community-based repository project from the Fedora team that provides high-quality software packages for the Linux distributions CentOS Stream, Red Hat Enterprise Linux, and Scientific Linux. Installation of EPEL is provided with the **yum install epel-release -y** command. The next group to be installed before proceeding is "base-x". Although "base-x" can be replaced with "X Window System",

this group has a lack of packages and is unstable, resulting in crashing applications and a freezing operating system. This installation is provided with **yum groupinstall "base-x" -y**. Installing Xfce is now as simple as installing the "Xfce" package group, which is already configured to install more packages than needed. As a last command to finish the installation of Xfce, is **yum groupinstall "Xfce" -y**. It may take a while to download and install packages.

```
yum install epel-release -y
yum groupinstall “base-x” -y
yum groupinstall “Xfce” -y
```

Listing 5.13: Commands used for installation Xfce

```
root@localhost]# yum groupinstall "Xfce" -y
================================================================
Package                Arch        Version      Repository      Size
================================================================
Installing group/module packages:
Thunar                 x86_64      4.16.8-1.el8 epel            01.6 M
mousepad               x86_64      0.5.6-1.el8  epel            339 k
thunar-archive-plugin  x86_64      0.4.0-26.el8 epel             85 k
thunar-volman          x86_64      4.16.0-3.el8 epel            215 k
tumbler                x86_64      0.2.7-1.el8  epel            237 k
xfce-polkit            x86_64      0.3-3.el8    epel             25 k
xfce4-appfinder        x86_64      4.16.1-3.el8 epel            282 k
xfce4-panel            x86_64      4.16.3-1.el8 epel            01.1 M
xfce4-power-manager    x86_64      4.16.0-1.el8 epel            788 k
xfce4-pulseaudio-plugin x86_64     0.4.3-3.el8  epel            123 k
xfce4-screensaver      x86_64      4.16.0-3.el8 epel            300 k
xfce4-session          x86_64      4.16.0-3.el8 epel            541 k
xfce4-settings         x86_64      4.16.2-1.el8 epel            01.2 M
xfce4-terminal         x86_64      0.8.10-2.el8 epel            670 k
xfconf                 x86_64      4.16.0-1.el8 epel            192 k
xfdesktop              x86_64      4.16.0-3.el8 epel            01.6 M
xfwm4                  x86_64      4.16.1-1.el8 epel            608 k
Installing dependencies:
```

exo	x86_64	4.16.2-1.el8	epel	474	k
garcon	x86_64	4.16.1-1.el8	epel	233	k
libXScrnSaver	x86_64	1.2.3-1.el8	appstream	31	k
libdbusmenu	x86_64	16.04.0-12.el8	appstream	140	k
libdbusmenu-gtk3	x86_64	16.04.0-12.el8	appstream	41	k
libmousepad0	x86_64	0.5.6-1.el8	epel	125	k
libxfce4ui	x86_64	4.16.0-2.el8	epel	280	k
libxfce4util	x86_64	4.16.0-4.el8	epel	186	k
pavucontrol	x86_64	3.0-11.el8	appstream	160	k

Installing Groups:

Xfce

Transaction Summary

==========

Install 26 Packages

Total download size: 11 M

Installed size: 49 M

Listing 5.14: Installation of Xfce

After installing these packages, the default target should be automatically updated, meaning that after a reboot, the GUI will automatically be loaded. Finally, after a reboot, on the login screen is the option to choose a GUI. However, to save as much space as possible, this login screen will not be available later, and the only option to start this GUI is through a console. To open this console, three keys need to be pressed – **ctrl+alt+f3**. When the console is displayed, the login name and password are entered. Then the last command is **xinit**. After this, the Xfce GUI should be automatically loaded.

5.7 Selection and installation lightweight web browser

Once the Xfce GUI installation is complete, it's time to select and install a lightweight web browser. In table 5.3 is a list of the top 5 Linux lightweight web browsers, but not all are suitable to show the web pages of VUT University of Brno without a change in look. A Web browser requires JavaScript to display websites correctly. Installation of these web browsers is a bit more complicated and time-consuming because none of the needed software packages are available in the CentOS Stream official repositories. This means that they cannot be installed directly with all their dependencies. There are several solutions to the problem with installation. One of them is to find a third-party repository, which would make the job easier.

If the first solution is not successful, the second is to find open-source codes that are freely available for your own use thanks to a license. These codes can then be compiled on a computer. Some browsers share the same libraries, which must be added before compilation, or will be added with a third-party repository. Without the added libraries, the compilation itself wouldn't even start because the browser constantly requires these libraries. The comparison of individual lightweight browsers with libraries is in the table. Every browser requires a different number of libraries for proper operation. Some browsers share the same libraries. For example, the Otter browser shares a few libraries with Falkon, and Falkon shares the libcrypto library with Links and Netsurf browsers, but these two browsers cannot be used because they do not include JavaScript.

Table 5.3: Comparison of lightweight web browsers

Web browser	Dependencies	Installation size (MB)
Midori	glibc, ld-linux-aarch64.so.1, libarchive.so.1.3, libc.so.6, libcairo-gobject.so.2, libcairo.so.2, libgcr-base-3.so.1, libgcr-ui-3.so.1, libgdk-3.so.0, libgdk_pixbuf-2.0.so.0, libgio-2.0.so.0, libglib-2.0.so.0, libgobject-2.0.so.0, libgtk-3.so.0, libjavascriptcoregtk-4.0.so.1.8, libjson-glib-1.0.so.0, libpeas-1.0.so.0, libpeas-gtk-1.0.so.0, libsoup-2.4.so.1, libsqlite3.so.0, lib-webkit2gtk-4.0.so.37, rtld	4,3
Links	libbz2.so.1, libc.so.6, libcom_err.so.2, lib-crypto.so.1.1, libdl.so.2, libexpat.so.1, lib-gpm.so.2, libgssapi_krb5.so.2, libidn2.so.0, libk5crypto.so.3, libkrb5.so.3, liblua-5.3.so, libm.so.6, libssl.so.1.1, libz.so.1	2,9
Falkon	libQt5Core.so.5, libQt5DBus.so.5, libQt5Gui.so.5, libQt5Network.so.5, libQt5Po-sitioning.so.5, libQt5PrintSupport.so.5, libQt5Qml.so.5, libQt5Quick.so.5, libQt5QuickWidgets.so.5, libQt5Sql.so.5, libQt5WebChannel.so.5,	12,4

	libQt5WebEngineCore.so.5, libQt5WebEngineWidgets.so.5, libQt5Widgets.so.5, libQt5X11Extras.so.5, libc.so.6, libcrypto.so.1.1, libgcc_s.so.1, libm.so.6, libstdc++.so.6, libxcb.so.1, qt5-qtbase, qt5-qtwebengine	
Otter browser	libKF5SonnetCore.so.5, libQt5Core.so.5, libQt5DBus.so.5, libQt5Gui.so.5, libQt5Multimedia.so.5, libQt5Network.so.5, libQt5PrintSupport.so.5, libQt5Script.so.5, libQt5Sql.so.5, libQt5WebKit.so.5, libQt5WebKitWidgets.so.5, libQt5Widgets.so.5, libQt5XmlPatterns.so.5, libc.so.6, libgcc_s.so.1, libm.so.6, libpthread.so.0, libstdc++.so.6	15
Netsurf	libatk-1.0.so.0, libc.so.6, libcairo-gobject.so.2, libcairo.so.2, libcrypto.so.1.1, libcurl.so.4, libexpat.so.1, libgdk-3.so.0, libgdk_pixbuf-2.0.so.0, libgio-2.0.so.0, libglib-2.0.so.0, libgmodule-2.0.so.0, libgobject-2.0.so.0, libgthread-2.0.so.0, libgtk-3.so.0, libjpeg.so.62, libm.so.6, libpango-1.0.so.0, libpangocairo-1.0.so.0, libpng16.so.16, libpthread.so.0, librsvg-2.so.2, libssl.so.1.1, libz.so.1	6,5

The browser is selected by size and by properties. Browsers that are suitable have a graphical environment, so Links doesn't belong here because it is a text-based web browser. Netsurf, Falkon, and Midori have the smallest size after installation, but the final size is much bigger due to all the necessary libraries and dependencies. I have chosen the Otter browser because of its smallest size after installation and for all the necessary properties.

Lightweight web browser installation

As I mentioned in chapter 5.7, there are several solutions to installing software packages that are not available in the CentOS Stream official repositories. I was able to find a third-party repositories, which helped me to save some time. These third-party repositories are called

Nux Dextop and **Raven**. As a first step, the latest release of Nux Dextop or Raven must be downloaded. Secondly, this release must be installed via **rpm** or **dnf**. Third, installation of the otter-browser package.

```
[root@localhost Downloads]# dnf -y install https://pkgs.dyn.su/el8/base/x86_64/ra-
ven-release-1.0-2.el8.noarch.rpm
raven-release-1.0-2.el8.noarch.rpm
Dependencies resolved.
================================================================
 Package            Architecture   Version         Repository                Size
================================================================
Installing:
 raven-release      noarch         1.0-2.el8       @commandline              9.1 k
Installing dependencies:
 epel-release       noarch         8-11.el8        extras                    24 k
Transaction Summary
================================================================
Install  2 Packages
Total size: 33 k [root@localhost Downloads]# yum install otter-browser
Nux.Ro RPMs for general desktop use            1.8 MB/s | 4.2 MB     00:02
Last metadata expiration check: 0:00:02 ago on Mon 06 Dec 2021 10:06:50 AM EST.
Dependencies resolved.
================================================================
Package            Arch       Version                       Repository          Size
================================================================
Installing:
otter-browser      x86_64     0.9.09-                       nux-dextop          2,2 M
                              0.2.beta9gitff0bb28.el7.nux
Installing dependencies:
kf5-sonnet-core          x86_64     5.88.0-1.el8              epel          212 k
libatomic                x86_64     8.5.0-4.el8_5             baseos         24 k
openal-soft              x86_64     1.18.2-7.el8              appstream     394 k
pcre2-utf16              x86_64     10.32-2.el8               baseos        229 k
qt5-qtbase               x86_64     5.15.2-4.el8              appstream     3.6 M
```

```
qt5-qtbase-common       noarch  5.15.2-4.el8                appstream   41 k
qt5-qtbase-gui          x86_64  5.15.2-4.el8                appstream   6.1 M
qt5-qtdeclarative       x86_64  5.15.2-2.el8                appstream   4.2 M
qt5-qtlocation          x86_64  5.15.2-2.el8                appstream   3.3 M
qt5-qtmultimedia        x86_64  5.15.2-2.el8                appstream   883 k
qt5-qtscript            x86_64  5.15.2-2.el8                appstream   1.1 M
qt5-qtsensors           x86_64  5.15.2-2.el8                appstream   220 k
qt5-qtwebchannel        x86_64  5.15.2-2.el8                appstream   102 k
qt5-qtwebkit            x86_64  5.212.0-0.60.alpha4.el8     epel        13 M
qt5-qtxmlpatterns       x86_64  5.15.2-2.el8                appstream   1.1 M
xcb-util-image          x86_64  0.4.0-9.el8                 appstream   21 k
xcb-util-keysyms        x86_64  0.4.0-7.el8                 appstream   16 k
xcb-util-renderutil     x86_64  0.3.9-10.el8                appstream   19 k
xcb-util-wm             x86_64  0.4.1-12.el8                appstream   32 k
Transaction Summary
================================================================================
Install 20 Packages
Total download size: 37 M
Installed size: 125 M
Is this ok [y/N]: y
```

Listing 5.15: Raven and Otter-Browser installation

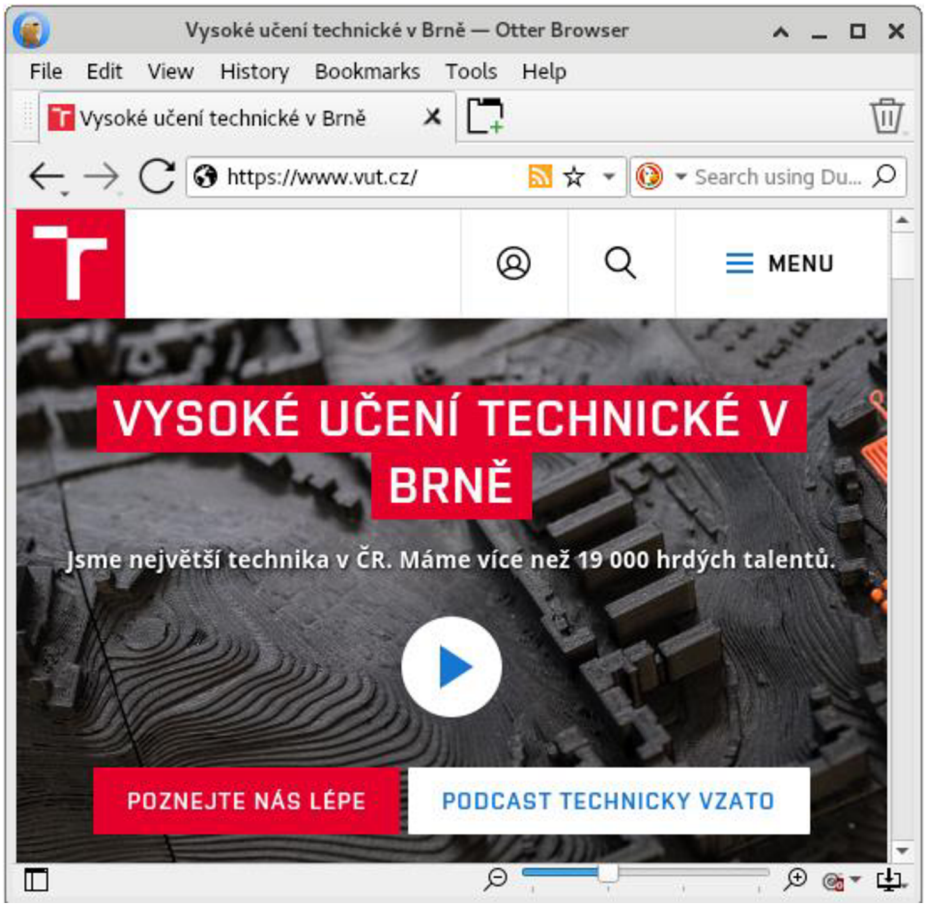

Figure 5.1: Otter browser before reduction

5.8 Automated reduction

My script, which is used for reduction, consists of more than 350 lines, and is divided into 3 different parts. In the first part, I installed Otter-Browser, GUI Xfce, XTerm, XPDF, and added important libraries and packages into the protected directory. Packages which are added in this directory are protected from being removed, even using the ***yum autoremove -y*** command. In this part, no disk space is freed.

```
#!/bin/sh
#installation of extra packages for Xfce
dnf install epel-release -y
dnf --enablerepo=epel group
dnf install dnf-plugins-core -y
dnf config-manager --set-enabled powertools
dnf groupinstall "Xfce" "base-x" -y
echo "exec /usr/bin/xfce4-session" >> ~/.xinitrc
systemctl set-default graphical
```

```
#installation of Development tools and Raven for Otter-Browser and XTerm
dnf -y groupinstall "Development Tools"
dnf -y install https://pkgs.dyn.su/el8/base/x86_64/raven-release-1.0-2.el8.noarch.rpm
dnf --enablerepo=raven-multimedia
dnf --enablerepo=raven-extras
dnf install -y otter-browser
dnf install -y xcb-util xterm
dnf install -y https://download-ib01.fedoraproject.org/pub/epel/7/x86_64/Packages/x/xpdf-3.04-10.el7.x86_64.rpm
dnf install -y libxfce4ui
dnf install -y xfce4-session
dnf install -y https://pkgs.dyn.su/el8/base/x86_64/libwnck-2.31.0-16.el8.x86_64.rpm
#protecting packages from removal
echo "accountsservice" > /etc/dnf/protected.d/accountsservice.conf
echo "xpdf" > /etc/dnf/protected.d/xpdf.conf
echo "libQt5WebEngineWidgets" > /etc/dnf/protected.d/libQt5WebEngineWidgets
echo "libQt5Multimedia" > /etc/dnf/protected.d/libQt5Multimedia.conf
echo "libQt5WebEngineCore" > /etc/dnf/protected.d/libQt5WebEngineCore.conf
echo "xorg-x11-fonts-ISO8859-1-100dpi" > /etc/dnf/protected.d/xorg-x11-fonts-ISO8859-1-100dpi.conf
echo "xorg-x11-fonts-ISO8859-1-75dpi" > /etc/dnf/protected.d/xorg-x11-fonts-ISO8859-1-75dpi.conf
echo "xorg-x11-server-Xorg" > /etc/dnf/protected.d/xorg-x11-server-Xorg.conf
echo "xorg-x11-server-Xwayland" > /etc/dnf/protected.d/xorg-x11-server-Xwayland.conf
echo "sudo" > /etc/dnf/protected.d/sudo.conf
echo "dnf" > /etc/dnf/protected.d/dnf.conf
echo "Xfce" > /etc/dnf/protected.d/Xfce.conf
echo "xorg-x11-xinit" > /etc/dnf/protected.d/xorg-x11-xinit.conf
echo "xterm" > /etc/dnf/protected.d/xterm.conf
echo "NetworkManager" > /etc/dnf/protected.d/NetworkManager.conf
echo "bash" > /etc/dnf/protected.d/bash.conf
echo "otter-browser" > /etc/dnf/protected.d/otter-browser.conf
```

```
echo "xorg-x11-xinit" > /etc/dnf/protected.d/xorg-x11-xinit.conf
echo "Xorg" > /etc/dnf/protected.d/Xorg.conf …
```

Listing 5.16: The first section of the script

In the second part of the script, I started to remove packages and all their dependencies using the command ***yum autoremove -y***. This means that I don't have to accept every package manually to be removed. This part of the script consists of more than another 400 lines and includes every installed package in the minimal installation of CentOS Stream. A short example is shown in the listing 5.17. Using this part of the script, I freed up disk space. To make this part of the script more effective, everything is done using 5 lines of code in the script. Commands used for automatization of this part were **gawk** and **sed**.

```
sudo yum autoremove -y GConf2.x86_64 ModemManager.x86_64 ModemManager-glib.x86_64 NetworkManager.x86_64 NetworkManager-adsl.x86_64 NetworkManager-bluetooth.x86_64 NetworkManager-config-server.noarch NetworkManager-libnm.x86_64 NetworkManager-team.x86_64 NetworkManager-tui.x86_64 NetworkManager-wifi.x86_64 NetworkManager-wwan.x86_64 PackageKit-glib.x86_64 PackageKit-gstreamer-plugin.x86_64 PackageKit-gtk3-module.x86_64 Thunar.x86_64 abattis-cantarell-fonts.noarch accountsservice.x86_64 accountsservice-libs.x86_64 acl.x86_64 adcli.x86_64 adobe-mappings-cmap.noarch adobe-mappings-cmap-deprecated.noarch adobe-mappings-pdf.noarch adwaita-cursor-theme.noarch adwaita-gtk2-theme.x86_64 adwaita-icon-theme.noarch . . . xz-libs.x86_64 yajl.x86_64 yelp.x86_64 yelp-libs.x86_64 yelp-tools.noarch yelp-xsl.noarch yum.noarch yum-utils.noarch zenity.x86_64 zip.x86_64 zlib.x86_64 zlib-devel.x86_64 zstd.x86_64 ...
```

Listing 5.17: The script's second section

In the final part of the script, I removed all the remaining and unused software, which wasn't removed during the second part. This happened because some software packages share the same libraries, which I added to the protected directory and thus were protected against being removed using the yum command. At the end, the script removed some unnecessary directories with files and cache. SSH also had to be reinstalled. Using this part, which is shown in the listing 5.18, I freed up another space. The final size of the minimized CentOS Stream is 650 MB.

```
#removing unused kernel versions
wget https://github.com/ZeuSVK/skripty/raw/main/kernel.bash
```

```
sed s/\vmlinuz-4.18.0-383.el8.x86_64/vmlinuz-$(uname -r)/ kernel.bash > ker-
nel1.bash
sed s/\initramfs-4.18.0-383.el8.x86_64/initramfs-$(uname -r)/ kernel1.bash > ker-
nel2.bash
sed s/\config-4.18.0-383.el8.x86_64/config-$(uname -r)/ kernel2.bash > kernel3.bash
shopt -s extglob
sh kernel3.bash
cd /home/test
#downloading scripts to /home/test directory
wget https://github.com/ZeuSVK/skripty/raw/main/bsos.pdf
#removing files, subfolders, etc.
rm -rfv ./.cache
rm -rfv ./.mozilla
rm -rfv /var/cache/*
rm -rfv /etc/firewalld/
rm -rfv /etc/services
rm -rfv /etc/udev
rm -rfv /home/test/.cache/*
rm -rfv /home/test/.local/
rm -rfv /home/test/rpmbuild/sources
rm -rfv /home/rpmbuild/sources
rm -rfv /lib/modules…
```

Listing 5.18: The script's final section

5.9 Start of the OS after reduction

While performing reduction, the OS is rebooted only one time – after the performed automated script. This script is executed by root, because is used minimal installation of this operating system.

After the script, all the necessary programmes are installed, and all unused software removed. To reboot after the performed script is as easy as to type "reboot" into the console, and after reboot, change user to "test" by using command "**su test**" and then type "xinit". After a successful start of the OS, the XTerm terminal is shown.

As a last two parts of this thesis are to open Otter-Browser using command "otter-browser" a then browse to Brno University of Technology main page. The page is loaded in

the same figure as in any other web browser. Second part is to open .pdf scripts for course Networking Operating Systems. The two commands are "**cd /home/test/**" and then "**xpdf bsos.pdf**". Scripts are fully readable with the same font as they were written in.

5.10 Summary of reduction

After an automated reduction performed by script, the size of CentOS Stream is 650 MB. The web browser, with all its needed dependencies, takes up one third of this size. The remaining two thirds are occupied by the reduced operating system. From table 5.4, it is possible to see the size of folders before, and after reduction. The least minimized directory is */usr/bin*. On the other hand, the most minimized directory is */var* with all its subdirectories. The resulting values were caused by the removal of unnecessary software packages, firmware, and libraries. These values vary in other Linux distributions.

Table 5.4: Folder comparison before and after reduction

Directory	Before reduction [MB]	After reduction [MB]
/root	1.9 kB	289 kB
/etc	21.8	1.7
/home	1 kB	826 kB
/tmp	1.2 kB	0 kB
/usr	1358	565
/usr/bin	48.9	49
/usr/include	44 kB	0
/usr/lib	936	7.6
/usr/sbin	35.8	12.4
/usr/lib64	153	417
/usr/share	180	69
/var	235	833 kB
/var/log	4.44	678 kB

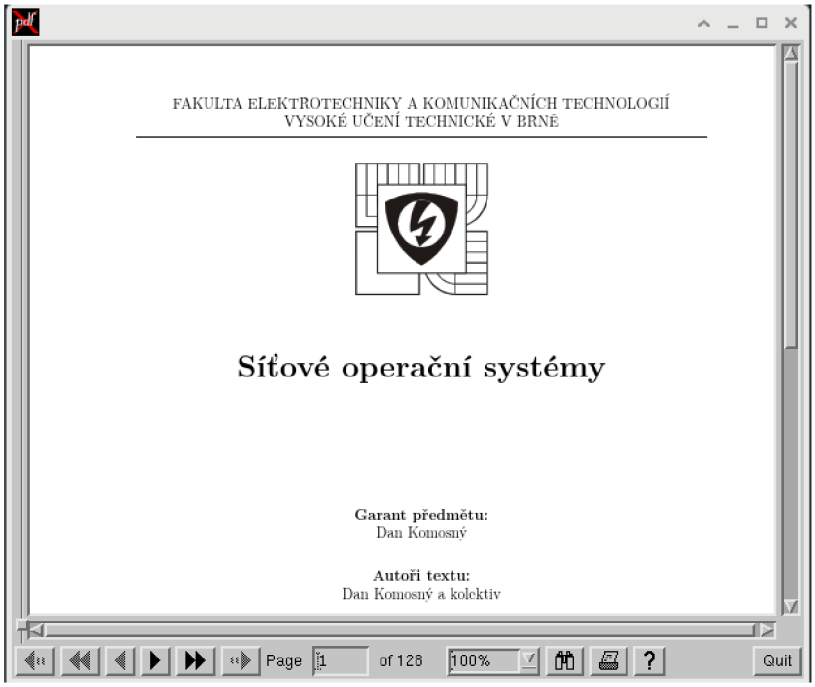

Figure 5.2: Opened scripts in XPDF

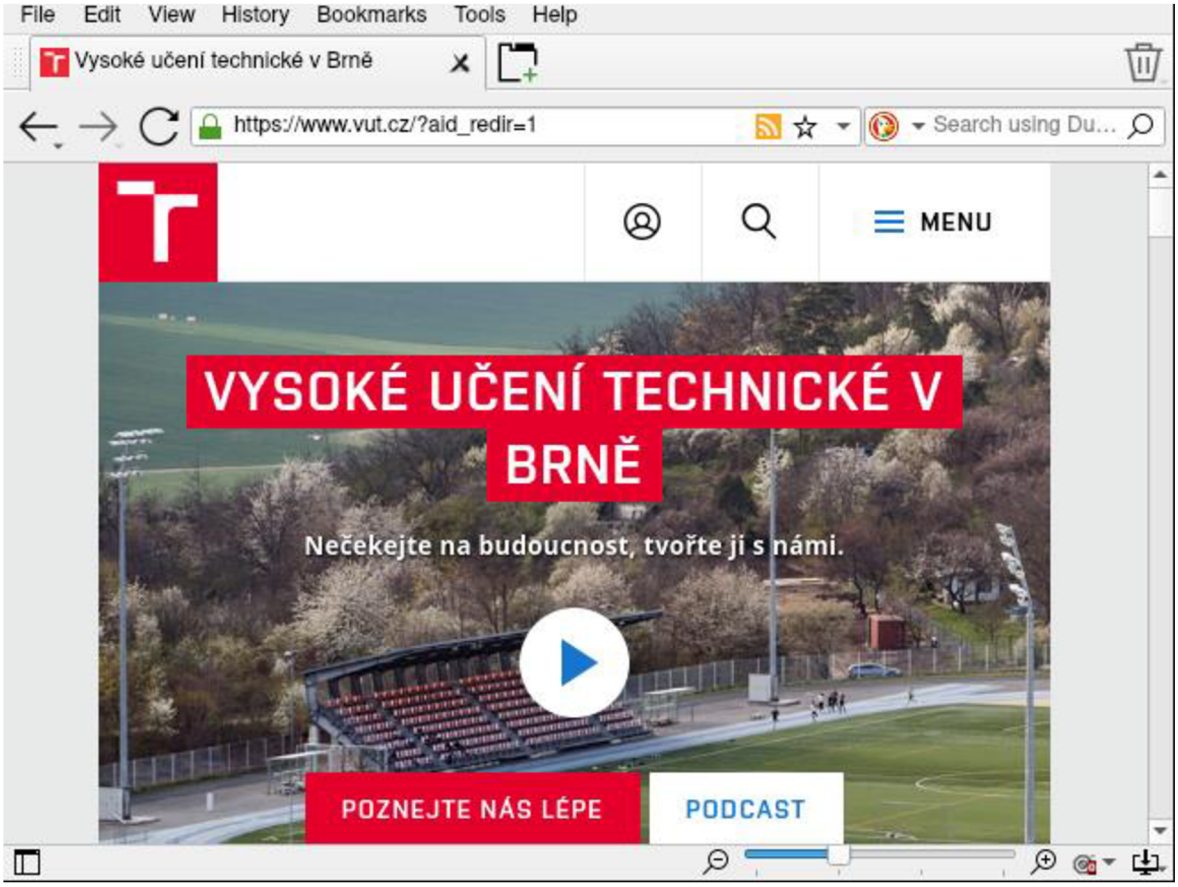

Figure 5.3: Opened page of VUT in Otter Browser after reduction

6 Conclusion

This thesis was developed to help with the project for the course of Networking Operating Systems. I reduced the size of the CentOS Stream operating system from the viewpoint of its size on storage media using an automated script. The reduced system includes a graphical web browser (Otter Browser), a PDF file viewer (XPFD) and remote access via SSH. System packages, programs, and kernel modules were mostly removed. The minimal installation of CentOS Stream was 1.8 GB, but after reduction, it was only 650 MB. The script has been tested on "*CentOS-Stream-8-x86_64-latest-dvd1.iso*" version. The number of software packages has been also reduced. The most minimized directory is */var* with all its subdirectories. From the default 235 MB, the size was reduced to 833 kB. On the other hand, the most increased directory was */usr/lib64/,* from 153 MB to 417 MB.

Xfce GUI was selected and used. Otter browser was chosen as the most suitable web browser. This browser displays the website in its original form. Because this browser does not support viewing PDF files, an XPDF viewer was additionally installed to display downloaded .pdf scripts. The overall reduction of the system was performed so as not to interfere with functionality. This automated script can be used not only in this specific operating system but also in other operating systems like Fedora or RHEL, which I tested with only minor modifications. After the successful minimization of Fedora in the workstation edition, the size changed from 8.7 GB to only 817 MB. Among the changes mentioned are the removal of a few lines from the section of the script dealing with graphic drivers, the addition of network manager to exceptions, and, in case of RHEL, the activation of package repositories for successful Xfce and Otter Browser installations.

Bibliography

[1] The Editors of Encyclopaedia Brittanica. Linus Tordvalds, Finnish computer scientist [online]. 2022 [cit. 2022-28-7]. Accessible from: **https://www.britannica.com/biogra-phy/Linus-Torvalds**

[2] Red Hat. What is the difference between Unix and Linux? [online]. 2021 [cit. 2021-10-3]. Accessible from: **https://opensource.com/resources/linux**

[3] SUSE. Linux Kernel [online]. 2021 [cit. 2021-10-3]. Accessible from: **https://www.suse.com/suse-defines/definition/linux-kernel/**

[4] Unlisted. Linux Kernel Modules [online]. 2021 [cit. 2021-11-3]. Accessible from: **https://eng.libretexts.org/Bookshelves/Computer_Science/Operating_Systems/Linux_-_The_Penguin_Marches_On_(McClanahan)/06%3A_Kernel_Module_Management/2.04%3A_Kernel_Modules**

[5] Peter Jay Salzman, Michael Burian, Ori Pomerantz. The Linux Kernel Module Programming Guide: Compiling Kernel Modules [online]. 2007 [cit. 2021-11-3]. Accessible from: **https://tldp.org/LDP/lkmpg/2.6/html/x181.html**

[6] Bill Dyer. What are daemons in Linux? [online]. 2021 [cit. 2021-10-5]. Accessible from: **https://itsfoss.com/linux-daemons/**

[7] Peter Loshin. Secure Shell (SSH) [online]. 2021 [cit. 2021-11-3]. Accessible from: **https://www.techtarget.com/searchsecurity/definition/Secure-Shell**

[8] Dimitrios. What is display server in Linux? [online]. 2020 [cit. 2021-10-5]. Accessible from: **https://itsfoss.com/display-server/**

[9] Abhishek Prakash. What is Desktop Environment in Linux? [online]. 2020 [cit. 2021-10-5]. Accessible from: **https://itsfoss.com/what-is-desktop-environment/**

[10] The Linux Foundation. What is Linux? [online]. 2021 [cit. 2021-10-3]. Accessible from: **https://www.linux.com/what-is-linux/**

[11] GeeksforGeeks. Linux File Hierarchy Structure [online]. 2021 [cit. 2021-10-12]. Accessible from: **https://www.geeksforgeeks.org/linux-file-hierarchy-structure/**

[12] The Linux Information Project. The /root Directory [online]. 2005 [cit. 2021-10-12]. Accessible from: **http://www.linfo.org/slash_root.html**

[13] Jithin. Linux Binary Directories Explained [online]. 2017 [cit. 2021-10-12]. Accessible from: **https://www.interserver.net/tips/kb/linux-binary-directories-explained/**

[14] Skillset. What is the purpose of /boot folder in Linux? [online]. 2016 [cit. 2021-10-12]. Accessible from: **https://www.skillset.com/questions/what-is-the-purpose-of-boot-folder-in-linux**

[15] Binh Nguyen. Linux Filesystem Hierarchy: /dev [online]. 2004 [cit. 2021-10-15]. Accessible from: **https://tldp.org/LDP/Linux-Filesystem-Hierarchy/html/dev.html**

[16] Binh Nguyen. Linux Filesystem Hierarchy: /etc [online]. 2004 [cit. 2021-10-22]. Accessible from: **https://tldp.org/LDP/Linux-Filesystem-Hierarchy/html/etc.html**

[17] Binh Nguyen. Linux Filesystem Hierarchy: /home [online]. 2004 [cit. 2021-10-22]. Accessible from: **https://tldp.org/LDP/Linux-Filesystem-Hierarchy/html/home.html**

[18] Arnab Chakraborty. Init process on UNIX and Linux systems [online]. 2019 [cit. 2021-10-5]. Accessible from: **https://www.tutorialspoint.com/init-process-on-unix-and-linux-systems**

[19] Robert Junior. What does init do in Linux? [online]. 2021 [cit. 2021-10-5]. Accessible from: **https://techtrix.co/what-does-init-0-do-in-linux/**

[20] TechTarget Contributor. Boot loader [online]. 2021 [cit. 2021-10-3]. Accessible from: **https://searchdatacenter.techtarget.com/definition/boot-loader-boot-manager**

[21] Chris Hoddman. What Is a Linux Distro, Anyway? [online]. 2016 [cit. 2021-10-12]. Accessible from: **https://www.howtogeek.com/132624/htg-explains-whats-a-linux-distro-and-how-are-they-different/**

[22] Red Hat. What is Red Hat Enterprise Linux? [online]. 2021 [cit. 2021-10-12]. Accessible from: **https://www.redhat.com/en/technologies/linux-platforms/enterprise-linux**

[23] Techopedia, Red Hat Enterprise Linux (RHEL) [online]. 2017 [cit. 2021-12-6]. Accessible from: **https://www.techopedia.com/definition/15777/red-hat-enterprise-linux-rhel**

[24] Chris Wright. Transforming the development experience within CentOS [online]. 2019 [cit. 2021-10-12]. Accessible from: **https://www.redhat.com/en/blog/transforming-development-experience-within-centos**

[25] Phil Dibowitz, CentOS Stream: Why It's Awesome [online]. 2021 [cit. 2021-12-6]. Accessible from: **https://medium.com/swlh/centos-stream-why-its-awesome-c45d944fb22**

[26] Fedora [online]. 2022 [cit. 2022-4-14]. Accessible from: **https://getfedora.org/**

[27] TechTarget Contributor, What is Fedora? [online]. 2010 [cit. 2022-4-14]. Accessible from: **https://www.techtarget.com/searchdatacenter/definition/Fedora**

[28] Forrest Stroud, Fedora Linux [online]. 2012 [cit. 2022-4-14]. Accessible from: **https://www.webopedia.com/definitions/fedora/**

[29] Artem Ilyin, Debian [online]. 2022 [cit. 2022-4-14]. Accessible from: **https://golden.com/wiki/Debian-5DZ8**

[30] Mohammed Abubakar, What is Ubuntu? [online]. 2021 [cit. 2022-4-15]. Accessible from: **https://www.howtogeek.com/763775/what-is-ubuntu/**

List of used abbreviations, quantities, and symbols

GUI	Graphical User Interface
kB	kilobyte
MB	Megabyte
GB	Gigabyte
SSH	Secure Shell
API	Application Programming Interface
GRUB	GRand Unified Bootloader
BIOS	Basic Input Output System
RHEL	Red Hat Enterprise Linux
LILO	Linux Loader
LOADLIN	LOAD LINux

LIST OF APPENDICES

Appendix A – Script

centosstream.bash

```
#!/bin/sh
#installation of extra packages for Xfce
dnf install epel-release -y
dnf --enablerepo=epel group
dnf install dnf-plugins-core -y
dnf config-manager --set-enabled powertools
dnf groupinstall "Xfce" "base-x" -y
echo "exec /usr/bin/xfce4-session" >> ~/.xinitrc
systemctl set-default graphical
#installation of Development tools and Raven for Otter-Browser and XTerm
dnf -y groupinstall "Development Tools"
dnf -y install https://pkgs.dyn.su/el8/base/x86_64/raven-release-1.0-2.el8.noarch.rpm
dnf --enablerepo=raven-multimedia
dnf --enablerepo=raven-extras
dnf install -y otter-browser
dnf install -y xcb-util xterm
dnf install -y https://download-ib01.fedoraproject.org/pub/epel/7/x86_64/Packages/x/xpdf-
        3.04-10.el7.x86_64.rpm
dnf install -y libxfce4ui
dnf install -y xfce4-session
dnf install -y https://pkgs.dyn.su/el8/base/x86_64/libwnck-2.31.0-16.el8.x86_64.rpm
#protecting packages from removal
echo "accountsservice" > /etc/dnf/protected.d/accountsservice.conf
echo "xpdf" > /etc/dnf/protected.d/xpdf.conf
echo "libQt5WebEngineWidgets" > /etc/dnf/protected.d/libQt5WebEngineWidgets
echo "libQt5Multimedia" > /etc/dnf/protected.d/libQt5Multimedia.conf
echo "libQt5WebEngineCore" > /etc/dnf/protected.d/libQt5WebEngineCore.conf
echo "xorg-x11-fonts-ISO8859-1-100dpi" > /etc/dnf/protected.d/xorg-x11-fonts-ISO8859-
        1-100dpi.conf
echo "xorg-x11-fonts-ISO8859-1-75dpi" > /etc/dnf/protected.d/xorg-x11-fonts-ISO8859-1-
        75dpi.conf
```

```
echo "xorg-x11-server-Xorg" > /etc/dnf/protected.d/xorg-x11-server-Xorg.conf
echo "xorg-x11-server-Xwayland" > /etc/dnf/protected.d/xorg-x11-server-Xwayland.conf
echo "sudo" > /etc/dnf/protected.d/sudo.conf
echo "dnf" > /etc/dnf/protected.d/dnf.conf
echo "Xfce" > /etc/dnf/protected.d/Xfce.conf
echo "xorg-x11-xinit" > /etc/dnf/protected.d/xorg-x11-xinit.conf
echo "xterm" > /etc/dnf/protected.d/xterm.conf
echo "NetworkManager" > /etc/dnf/protected.d/NetworkManager.conf
echo "bash" > /etc/dnf/protected.d/bash.conf
echo "otter-browser" > /etc/dnf/protected.d/otter-browser.conf
echo "xorg-x11-xinit" > /etc/dnf/protected.d/xorg-x11-xinit.conf
echo "Xorg" > /etc/dnf/protected.d/Xorg.conf
echo "xorg-x11-xinit-session" > /etc/dnf/protected.d/xorg-x11-xinit-session.conf
echo "openssh-server" > /etc/dnf/protected.d/openssh-server.conf
echo "openssh-clients" > /etc/dnf/protected.d/openssh-clients.conf
echo "libwnck-2.31.0-16" > /etc/dnf/protected.d/libwnck-2.31.0-16.conf
echo "wget" > /etc/dnf/protected.d/wget.conf
echo "xfce4-session" > /etc/dnf/protected.d/xfce4-session.conf
echo "libxfce4ui" > /etc/dnf/protected.d/libxfce4ui.conf
#protecting folders from removal
chattr +i /usr/bin/xfce4-session
chattr +i /usr/bin/find
#adding "test" user
adduser test
#finding every installed package
yum list installed > installed.txt
gawk '{print "yum autoremove -y "$0}' installed.txt > installed1.txt
sed 's/\.x86_64.*/.x86_64/' installed1.txt > installed2.txt
sed 's/\.noarch.*/.noarch/' installed2.txt > installed3.txt
#finding log and rescue files
find /var/log -name '*.log*' -delete
find /boot -name *rescue* -delete
#start of package removing
sh installed3.txt
```

```
dnf -y install https://pkgs.dyn.su/el8/base/x86_64/raven-release-1.0-2.el8.noarch.rpm
dnf --enablerepo=raven-multimedia
dnf --enablerepo=raven-extras
dnf install -y libxfce4ui
dnf install -y xfce4-session
dnf install -y https://pkgs.dyn.su/el8/base/x86_64/libwnck-2.31.0-16.el8.x86_64.rpm
cd /boot
#removing unused kernel versions
wget https://github.com/ZeuSVK/skripty/raw/main/kernel.bash
sed s/\vmlinuz-4.18.0-383.el8.x86_64/vmlinuz-$(uname -r)/ kernel.bash > kernel1.bash
sed s/\initramfs-4.18.0-383.el8.x86_64/initramfs-$(uname -r)/ kernel1.bash > kernel2.bash
sed s/\config-4.18.0-383.el8.x86_64/config-$(uname -r)/ kernel2.bash > kernel3.bash
shopt -s extglob
sh kernel3.bash
cd /home/test
#downloading scripts to /home/test directory
wget https://github.com/ZeuSVK/skripty/raw/main/bsos.pdf
#removing files, subfolders, etc.
rm -rfv ./.cache
rm -rfv ./.mozilla
rm -rfv /var/cache/*
rm -rfv /etc/firewalld/
rm -rfv /etc/services
rm -rfv /etc/udev
rm -rfv /home/test/.cache/*
rm -rfv /home/test/.local/
rm -rfv /home/test/rpmbuild/sources
rm -rfv /home/rpmbuild/sources
rm -rfv /lib/modules
rm -rfv /lib64/bluetooth
rm -rfv /lib64/dri/i965_dri.so
rm -rfv /lib64/dri/nouveau_dri.so
rm -rfv /lib64/dri/r600_dri.so
rm -rfv /lib64/dri/radeonsi_dri.so
```

```
rm -rfv /lib64/dri/virtio_gpu_dri.so
rm -rfv /lib64/dri/vmwgfx_dri.so
rm -rfv /lib64/games
rm -rfv /root/.cache
rm -rfv /root/.mozilla
rm -rfv /root/Downloads
rm -rfv /root/mozilla
rm -rfv /root/rpmbuild/SOURCES
rm -rfv /run/log/journal
rm -rfv /run/udev
rm -rfv /usr/bin/aspell
rm -rfv /usr/bin/brotli
rm -rfv /usr/bin/gawk
rm -rfv /usr/bin/gcm-picker
rm -rfv /usr/bin/gnome-control-center
rm -rfv /usr/bin/gnome-keyring-daemon
rm -rfv /usr/bin/gnome-logs
rm -rfv /usr/bin/gpg
rm -rfv /usr/bin/gpgv
rm -rfv /usr/bin/grep
rm -rfv /usr/bin/hcidump
rm -rfv /usr/bin/l2test
rm -rfv /usr/bin/mwm
rm -rfv /usr/bin/nautilus
rm -rfv /usr/bin/nmcli
rm -rfv /usr/bin/nm-connection-editor
rm -rfv /usr/bin/python3.6
rm -rfv /usr/bin/vi
rm -rfv /usr/include
rm -rfv /usr/lib/girepository-1.0
rm -rfv /usr/lib/gnome-shell
rm -rfv /usr/lib/gpg
rm -rfv /usr/lib/locale
rm -rfv /usr/lib/modules/
```

```
rm -rfv /usr/lib/python3.6/site-packages
rm -rfv /usr/lib/qt5/bin
rm -rfv /usr/lib/udev/hwdb.d/
rm -rfv /usr/lib64/aspell-0.60
rm -rfv /usr/lib64/cmake
rm -rfv /usr/lib64/evolution-data-server
rm -rfv /usr/lib64/firefox
rm -rfv /usr/lib64/gconv
rm -rfv /usr/lib64/gio
rm -rfv /usr/lib64/girepository-1.0
rm -rfv /usr/lib64/gnome-keyring
rm -rfv /usr/lib64/gnome-shell
rm -rfv /usr/lib64/libbluetooth.so.3.19.4
rm -rfv /usr/lib64/libbrotlienc.so.1.0.6
rm -rfv /usr/lib64/libdb.so.2.3.0
rm -rfv /usr/lib64/libdw-0.185.so
rm -rfv /usr/lib64/libedataserver-1.2.so.23.0.0
rm -rfv /usr/lib64/libfdisk.so.1.1.0
rm -rfv /usr/lib64/libgdk-x11-2.0.so.0.2400.32
rm -rfv /usr/lib64/libgnome-desktop-3.so.17.0.6
rm -rfv /usr/lib64/libldb.so.2.3.0
rm -rfv /usr/lib64/libLLVM-12.so
rm -rfv /usr/lib64/libnss_resolve.so.2
rm -rfv /usr/lib64/libpoppler.so.104.0.0
rm -rfv /usr/lib64/libQt5Designer.so.5.15.2
rm -rfv /usr/lib64/libsamba-util.so.0.0.1
rm -rfv /usr/lib64/libsamba-util.so.1.0.28
rm -rfv /usr/lib64/libsolv.so.1
rm -rfv /usr/lib64/mutter-4
rm -rfv /usr/lib64/python3.6
rm -rfv /usr/lib64/qt5/bin
rm -rfv /usr/lib64/qt5/plugins/bearer
rm -rfv /usr/lib64/qt5/plugins/geoservices
rm -rfv /usr/lib64/qt5/qml
```

```
rm -rfv /usr/lib64/rsyslog
rm -rfv /usr/lib64/samba
rm -rfv /usr/libexec/gnome-session-binary
rm -rfv /usr/libexec/nm-iface-helper
rm -rfv /usr/libexec/nm-initrd-generator
rm -rfv /usr/sbin/gdm
rm -rfv /usr/sbin/ldconfig
rm -rfv /usr/share/anaconda
rm -rfv /usr/share/applications
rm -rfv /usr/share/backgrounds
rm -rfv /usr/share/doc
rm -rfv /usr/share/gcc-8
rm -rfv /usr/share/glib-2.0
rm -rfv /usr/share/gnome
rm -rfv /usr/share/gnome-control/center
rm -rfv /usr/share/gnome-control-center
rm -rfv /usr/share/gnome-session
rm -rfv /usr/share/gnome-shell
rm -rfv /usr/share/gnupg
rm -rfv /usr/share/help/*
rm -rfv /usr/share/hwdata/
rm -rfv /usr/share/ibus/*
rm -rfv /usr/share/icons
rm -rfv /usr/share/info
rm -rfv /usr/share/libthai
rm -rfv /usr/share/licenses/libjpeg-turbo/
rm -rfv /usr/share/locale/*
rm -rfv /usr/share/man
rm -rfv /usr/share/mime/text
rm -rfv /usr/share/misc
rm -rfv /usr/share/myspell
rm -rfv /usr/share/otter-browser/locale
rm -rfv /usr/share/plymouth
rm -rfv /usr/share/polkit-1
```

```
rm -rfv /usr/share/poppler
rm -rfv /usr/share/themes/Adwaita
rm -rfv /usr/share/vim
rm -rfv /usr/share/X11/locale
rm -rfv /usr/share/xfce4/helpers
rm -rfv /usr/share/xpdf/chinese/simplified
rm -rfv /usr/share/xpdf/chinese
rm -rfv /usr/share/xpdf/japanese
rm -rfv /usr/sher/myspell
rm -rfv /var/adm
rm -rfv /var/cache/*
rm -rfv /var/ftp
rm -rfv /var/games
rm -rfv /var/gopher
rm -rfv /var/ib/gdm
rm -rfv /var/lib/bluetooth
rm -rfv /var/lib/flatpak
rm -rfv /var/lib/gdm
rm -rfv /var/lib/rpm
rm -rfv /var/local
rm -rfv /var/lock
rm -rfv /var/log/anaconda
rm -rfv /var/log/messages
rm -rfv /var/log/sssd
rm -rfv /var/mail
rm -rfv /var/nis
rm -rfv /var/opt
rm -rfv /var/preserve
rm -rfv /var/spool
rm -rfv /var/tp
rm -rfv /var/yp
rm -rfv /var/zp
rm -rfv /usr/share/licenses
rm -rfv /usr/share/zoneinfo
```

```
rm -rfv /home/test/.cache
rm -rfv /usr/share/sounds
rm -rfv /run/log
rm -rfv /tmp
rm -rfv /var/cache
rm -rfv ¬/.cache
rm -rfv ¬/.local
rm -rfv ¬/Downloads
rm -rfv /usr/lib64/dri/crocus_dri.so
rm -rfv /usr/share/licenses
rm -rfv /usr/share/zoneinfo
rm -rfv /home/test/.cache
rm -rfv /usr/share/sounds
rm -rfv /usr/share/themes
rm -rfv /run/log
rm -rfv /tmp
rm -rfv /var/cache
rm -rfv ¬/.cache
rm -rfv ¬/.local
rm -rfv ¬/Downloads
rm -rfv /usr/lib64/dri/crocus_dri.so
cd /root
#downloading script for finding the actual size of the os
wget https://github.com/ZeuSVK/skripty/raw/main/du.bash
rm -rfv /usr/share/cracklib
rm -rfv /usr/lib64/dri/iris_dri.so
rm -rfv /usr/lib64/dri/kms_swarst_dri.so
rm -rfv /usr/lib64/javascriptcoregtk-4.0.so.18
rm -rfv /var/lib/sss
rm -rfv /var/lib/dnf
rm -rfv /var/libexec/bluetooth
rm -rfv /var/lib64/nss
rm -rfv /usr/share/mime
rm -rfv /var/lib64/libgtk-3.so.0
```

```
rm -rfv /var/lib64/libgtk-3.so.0.2200.30
rm -rfv /usr/lib/.build-id
rm -rfv /usr/lib/dracut
rm -rfv /usr/lib/rpm/platform
rm -rfv /usr/lib64/spa-0.2
rm -rfv /usr/share/pulseaudio
rm -rfv /usr/local
rm -rfv /usr/share/color
rm -rfv /usr/share/colord
rm -rfv /usr/share/libquvi-scripts
rm -rfv /usr/share/factory
rm -rfv /var/lib64/ldb
rm -rfv /usr/share/iso-codes
rm -rfv /usr/share/gdb
rm -rfv /usr/lib64/gtk-2.0
rm -rfv /usr/lib64/gtk-3.0
rm -rfv /usr/lib64/gstreamer-1.0
rm -rfv /usr/lib64/pulse-14.0
rm -rfv /usr/lib64/lua
rm -rfv /usr/lib64/librsvg-2.so.2
rm -rfv /usr/lib64/libmozjs-60.so.0.0.0
find /usr/lib64/ -type f -name "libpipe*" -print -delete
rm -rfv /usr/lib64/gnome-settings-daemon-3.0
find /usr/lib64/ -type f -name "libgtk*" -print -delete
find /usr/lib64/ -type f -name "*bluetooth*" -print -delete
find /usr/lib64/ -type f -name "*gnome*" -print -delete
find /usr/lib64/ -type f -name "*polkit*" -print -delete
find /usr/lib64/ -type f -name "*samba*" -print -delete
find /usr/lib64/ -type f -name "*libcol*" -print -delete
find /usr/lib64/ -type f -name "*jpg*" -print -delete
find / -iname "*jpg*" -print -delete
rm -rfv /usr/share/themes
find / -name "*svg*" -print -delete
find / -name "*icon*" -print -delete
```

```
find / -name "*gtk*" -print -delete
find / -name "*LLVM*" -print -delete
find / -name "*libbluetoth*" -print -delete
find / -name "*girepository*" -print -delete
find / -name "*aspell*" -print -delete
find / -name "*libdb-*" -print -delete
find / -name "*libedataserver*" -print -delete
find / -name "*libfdisk*" -print -delete
find / -name "*libgdk*" -print -delete
find / -name "*libgnome*" -print -delete
find / -name "*libldb*" -print -delete
find / -name "*evolution*" -print -delete
find / -name "*libpoppler*" -print -delete
find / -name "*libQtDesigner*" -print -delete
find / -name "*libsamba-util*" -print -delete
find / -name "*libsolv*" -print -delete
find / -name "*python3.6*" -print -delete
find / -name "*qml*" -print -delete
find / -name "*samba*" -print -delete
find / -name "*gnome*" -print -delete
find / -name "*nm-*" -print -delete
find / -name "*gdm*" -print -delete
find / -name "*ldconfig*" -print -delete
find / -name "*anaconda*" -print -delete
find / -name "*backg*" -print -delete
find / -name "*doc*" -print -delete
find / -name "*gnupg*" -print -delete
find / -name "*help*" -print -delete
find / -name "*ibus*" -print -delete
find / -name "*libthai*" -print -delete
find / -name "*locale*" -print -delete
find / -name "*plymouth*" -print -delete
find / -name "*polkit*" -print -delete
find / -name "*chinese*" -print -delete
```

```
find / -name "*japanese*" -print -delete
find / -name "*nese*" -print -delete
find / -name "*games*" -print -delete
find / -name "*rpm*" -print -delete
find / -name "*messages*" -print -delete
find / -name "*mail*" -print -delete
find / -name "*opt*" -print -delete
find / -name "*spool*" -print -delete
find / -name "*player*" -print -delete
find / -name "*zp*" -print -delete
find / -name "*zone*" -print -delete
find / -name "*them*" -print -delete
find / -name "*crack*" -print -delete
find / -name "*iris*" -print -delete
find / -name "*sss*" -print -delete
find / -name "*dnf*" -print -delete
find / -name "*blue*" -print -delete
find / -name "*libgtk*" -print -delete
find / -name "*color*" -print -delete
find / -name "*libquvi*" -print -delete
find / -name "*ldb*" -print -delete
find / -name "*gdb*" -print -delete
find / -name "*gtk*" -print -delete
find / -name "*libmozj*" -print -delete
sh du.bash
```

Appendix B – Running the script

Archive contains one .bash file which contains commands that have been executed. This script can be downloaded from: **https://github.com/ZeuSVK/skripty/raw/main/centosstream.bash**

```
Removed:
  biosdevname-0.7.3-2.el8.x86_64

Complete!
Dependencies resolved.
================================================================================
 Package            Architecture        Version                 Repository     Size
================================================================================
Removing:
 bison              x86_64              3.0.4-10.el8            @appstream    2.1 M

Transaction Summary
================================================================================
Remove  1 Package

Freed space: 2.1 M
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Running scriptlet: bison-3.0.4-10.el8.x86_64                              1/1
  Erasing          : bison-3.0.4-10.el8.x86_64                              1/1
  Running scriptlet: bison-3.0.4-10.el8.x86_64                              1/1
  Verifying        : bison-3.0.4-10.el8.x86_64                              1/1

Removed:
  bison-3.0.4-10.el8.x86_64

Complete!
Error:
 Problem: The operation would result in removing the following protected packages: accountsservice,
xorg-x11-server-Xwayland
(try to add '--skip-broken' to skip uninstallable packages or '--nobest' to use not only best candid
ate packages)
_
```

Figure B.1: The process of minimization

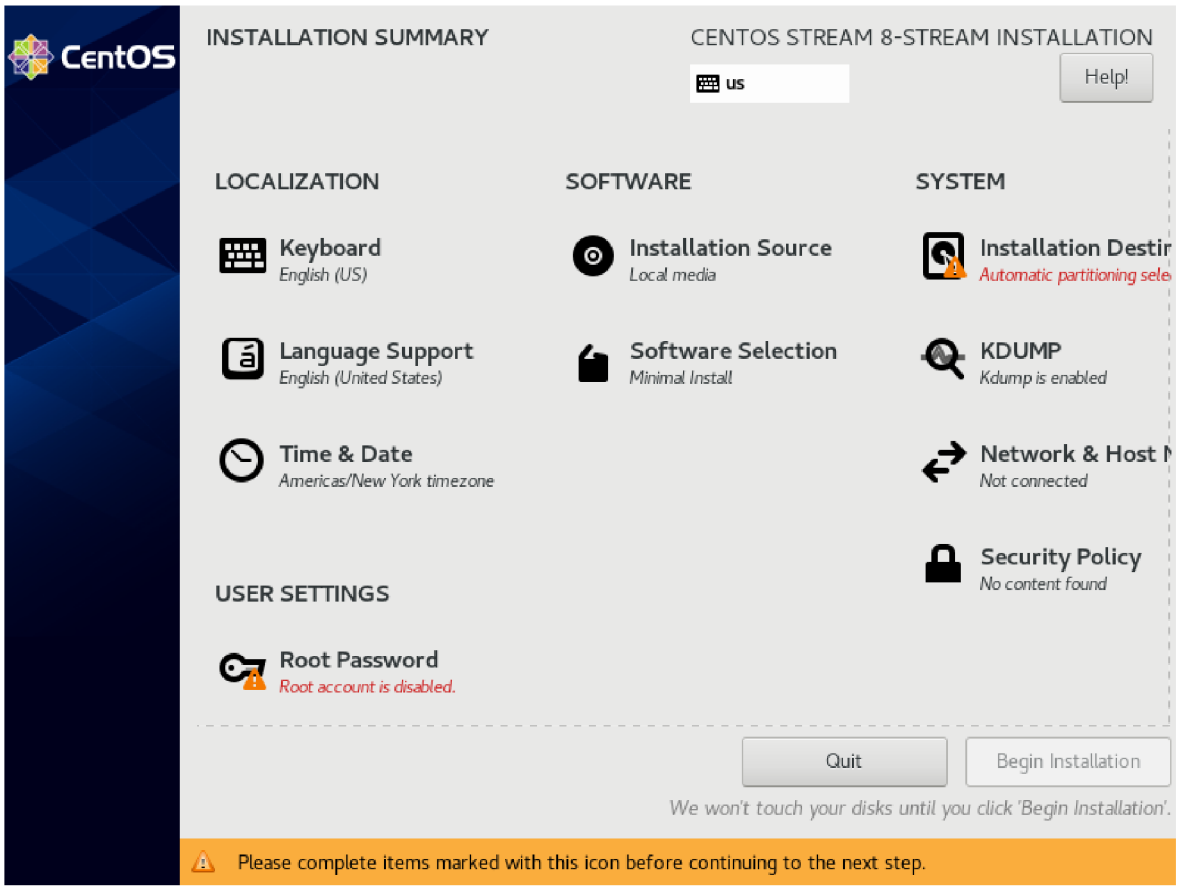

Figure B.2: Installation menu

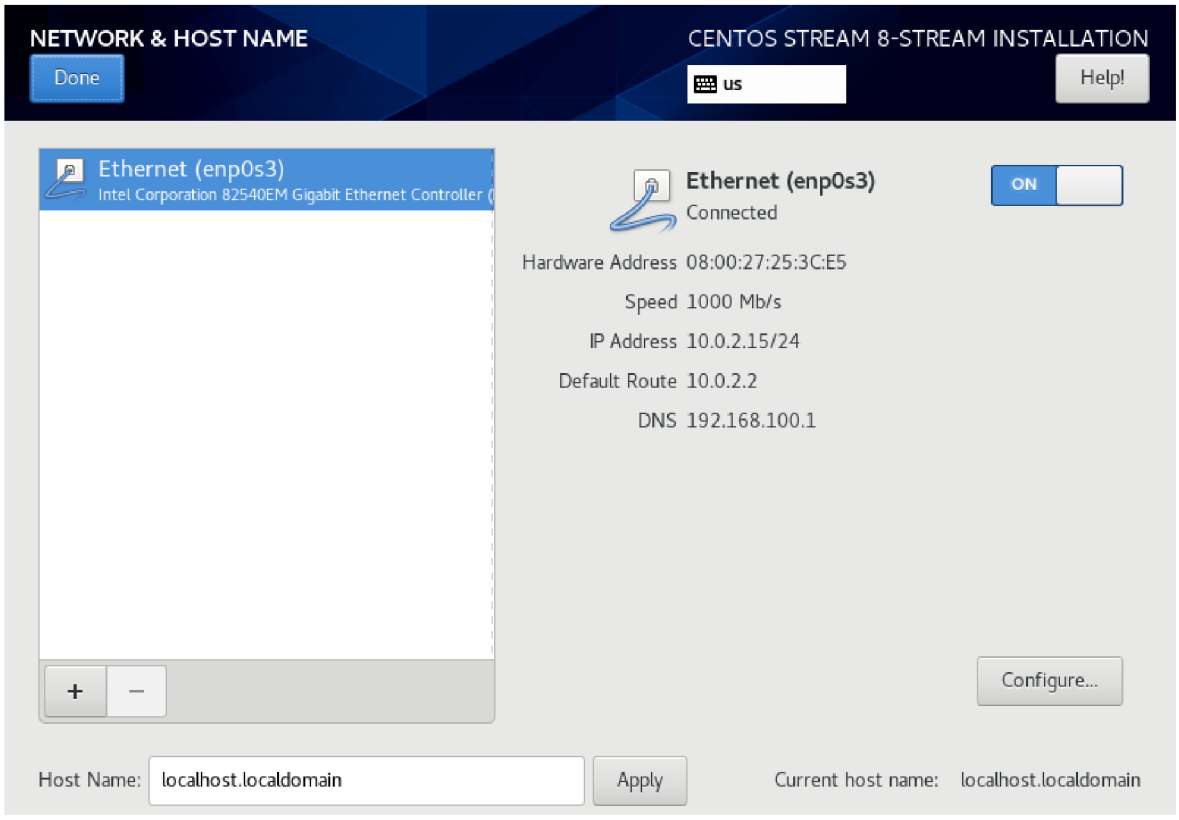

Figure B.3: Activation of the connection

Script execution procedure

Everything is executed as Root. Connection is turned off by default and must be activated before the reduction. Either before installation in the menu, or right after installation using the command "**nmtui**" and then activated using "**Activate a connection**" in the GUI. After the reduction of the operating system, this command will be removed, and network connection will not be possible to activate.

1. install wget using command "yum install wget", type "y"

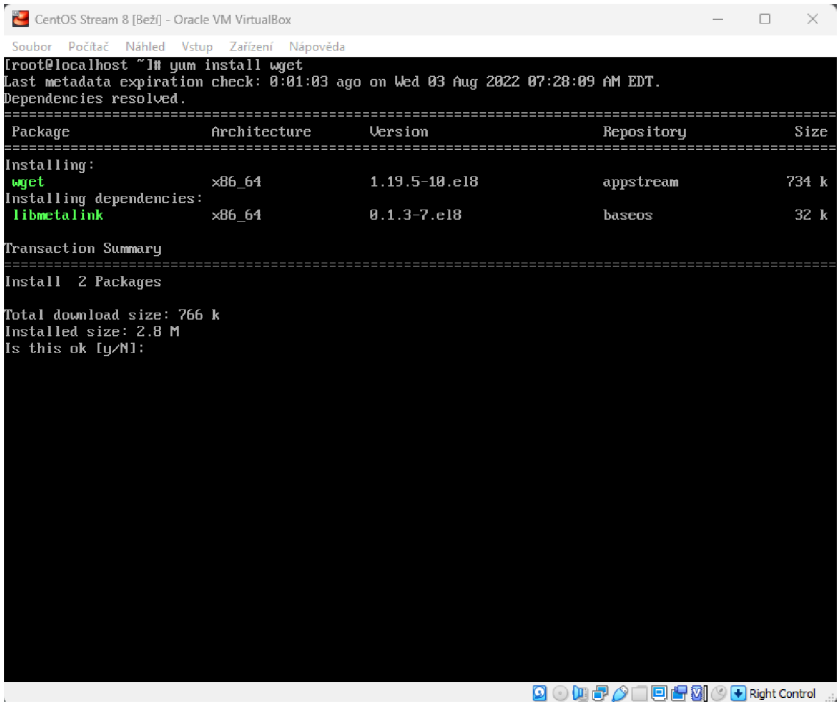

Figure B.4: Step 1

2. wget https://github.com/ZeuSVK/skripty/raw/main/centosstream.bash

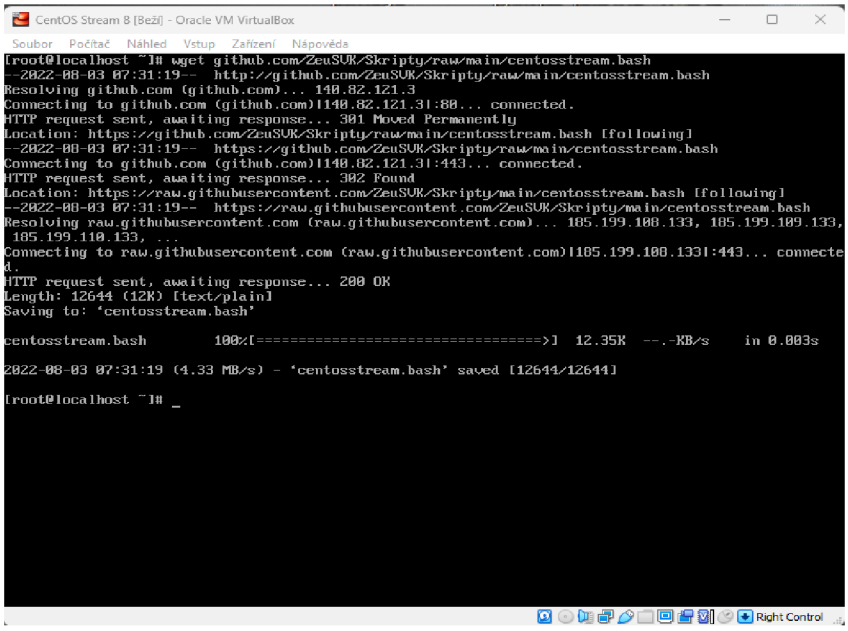

Figure B.5: Step 2

3. sh centosstream.bash

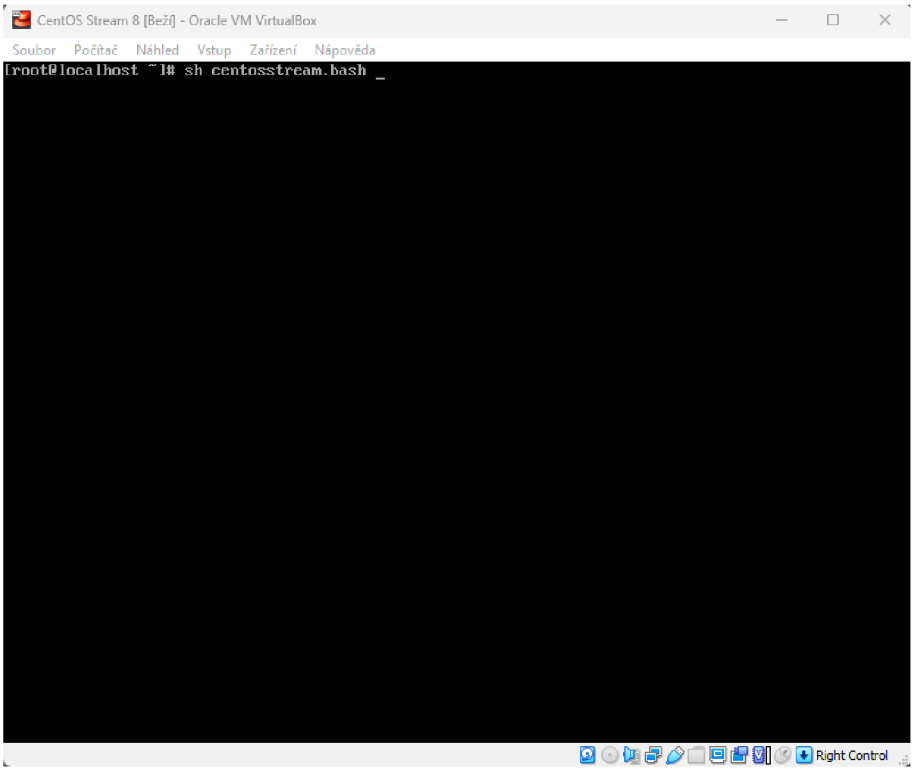

Figure B.6: Step 3 A

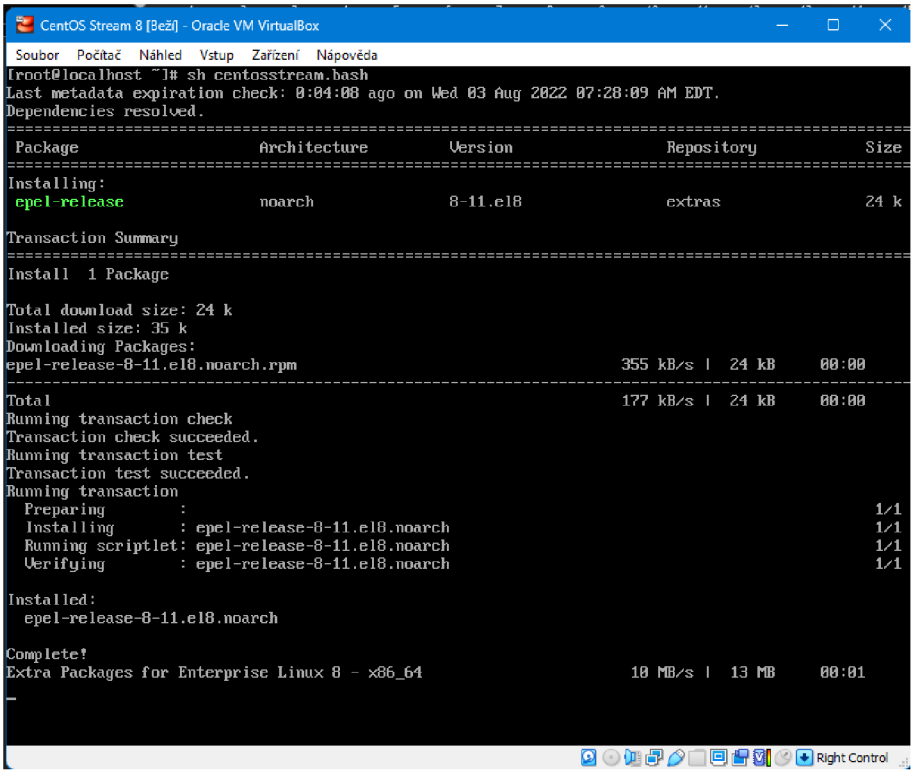

Figure B.7: Step 3 B

4. reboot

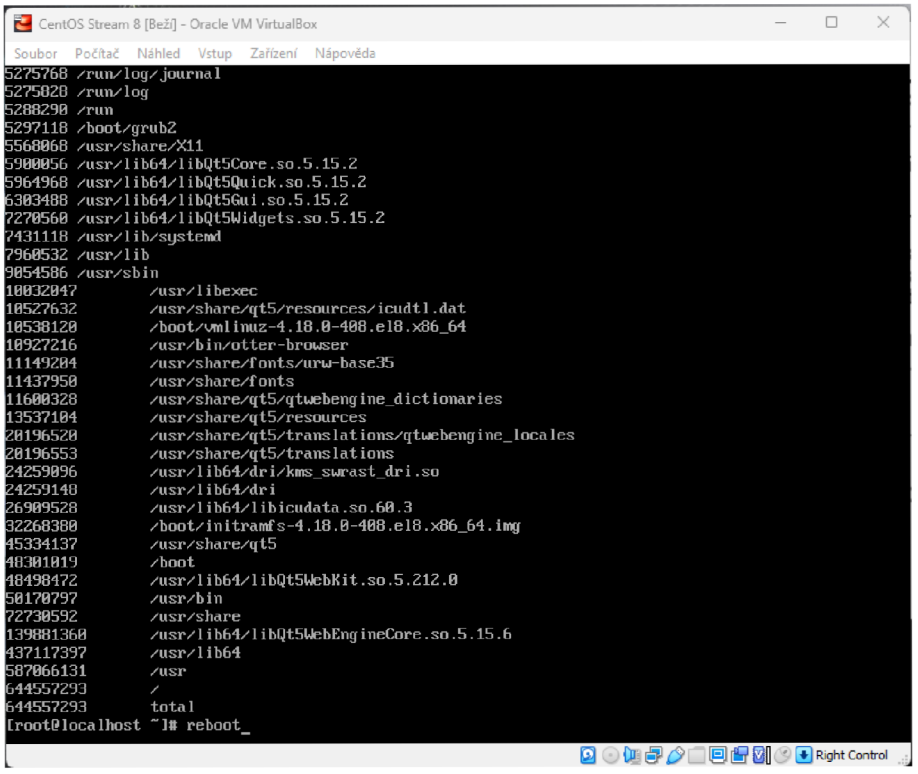

Figure B.8: Step 4

5. login

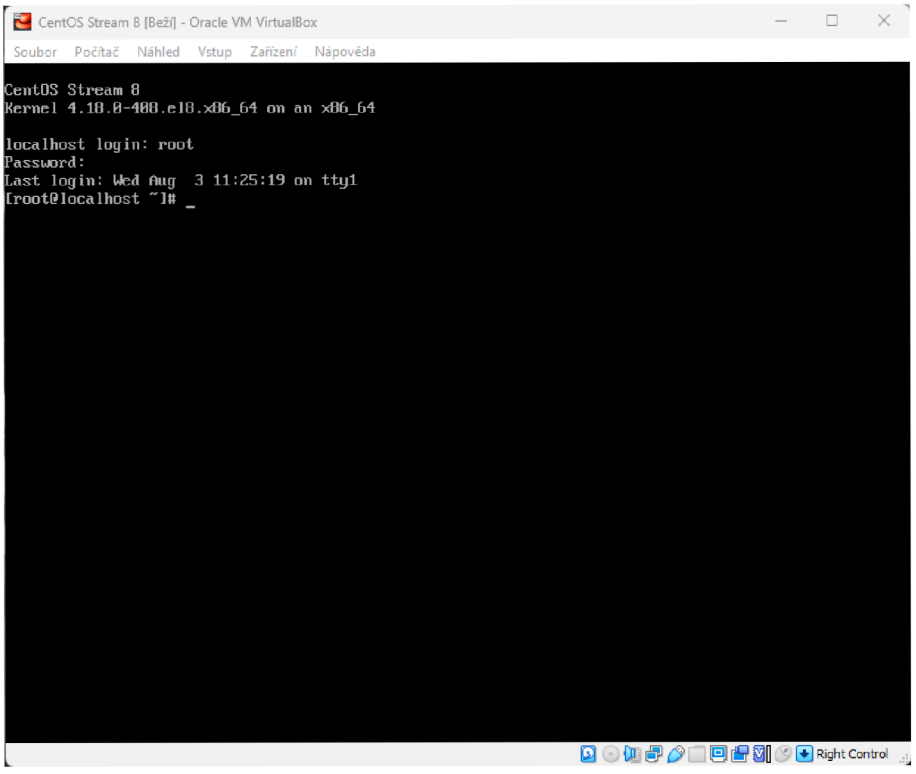

Figure B.9: Step 5

6. su test

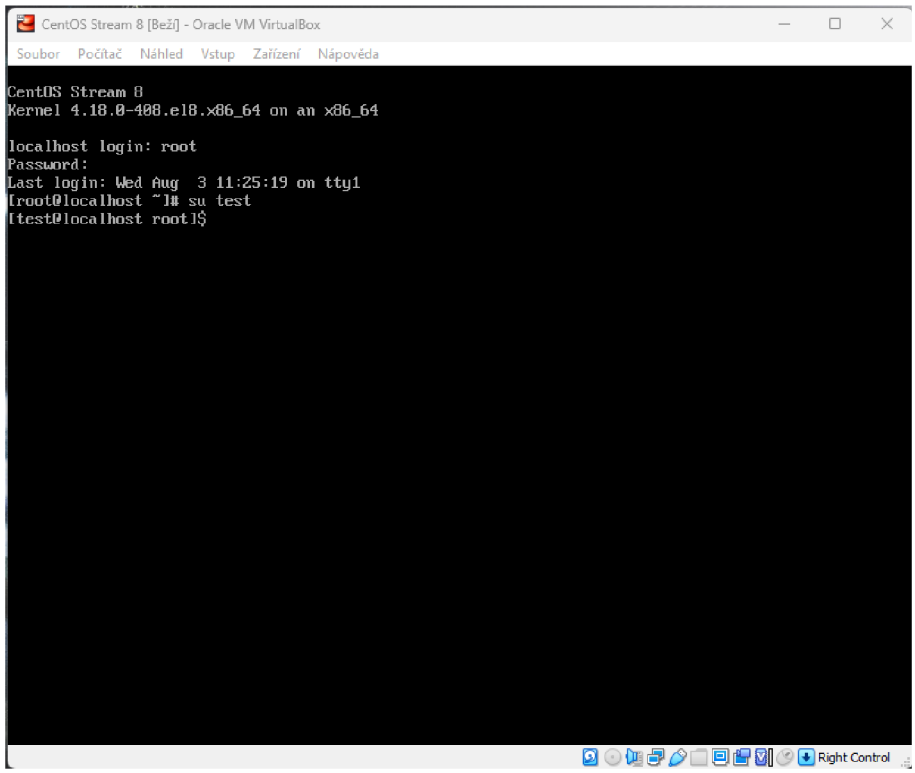

Figure B.10: Step 6

7. xinit

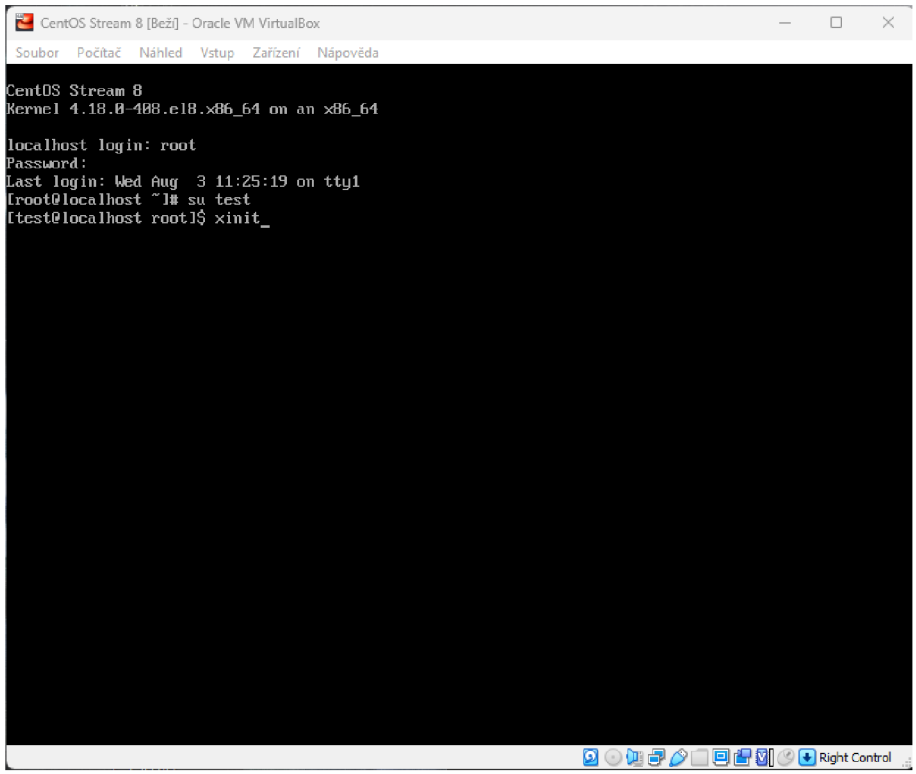

Figure B.11: Step 7

8. otter-browser, then type vut.cz

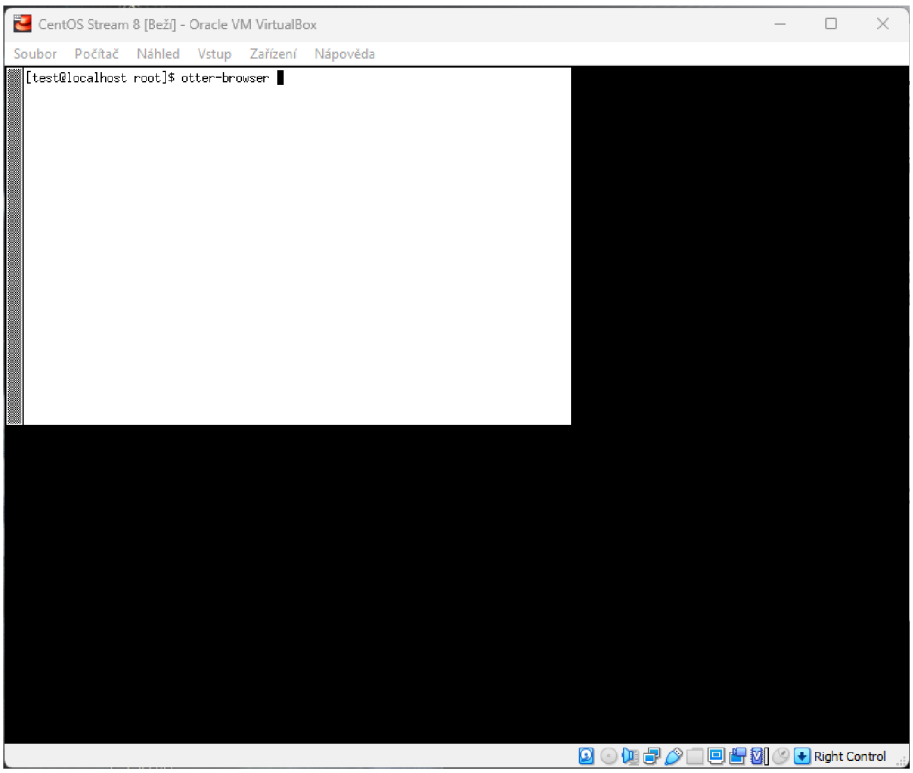

Figure B.12: Step 8 A

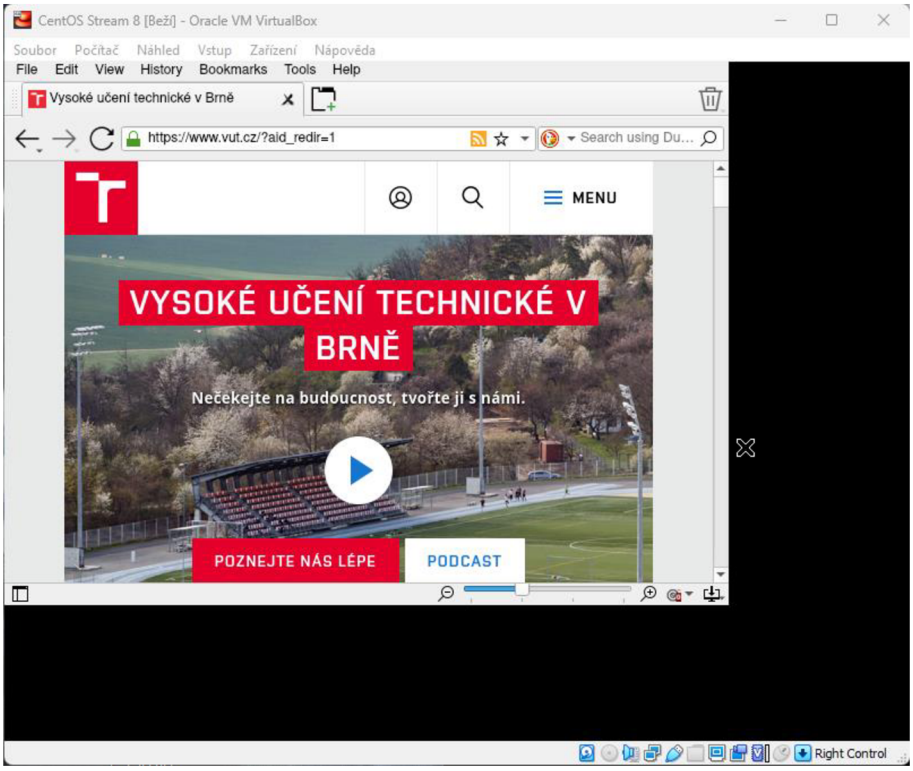

Figure B.13: Step 8 B

9. Exit Otter Browser clicking at “File” and “Exit”

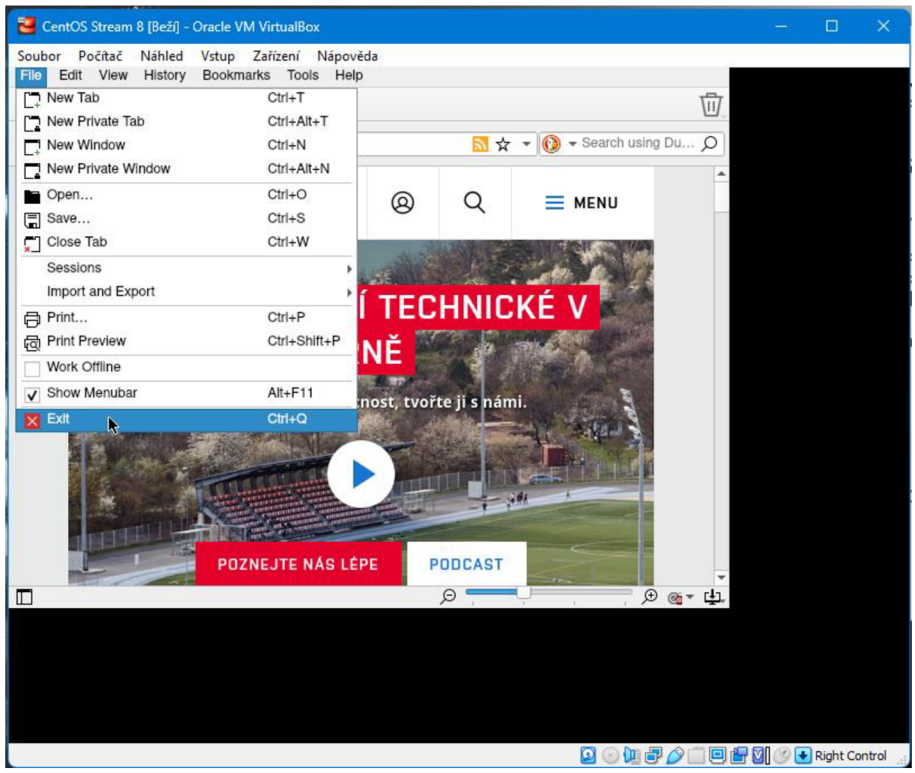

Figure B.14: Step 9

10. cd /home/test

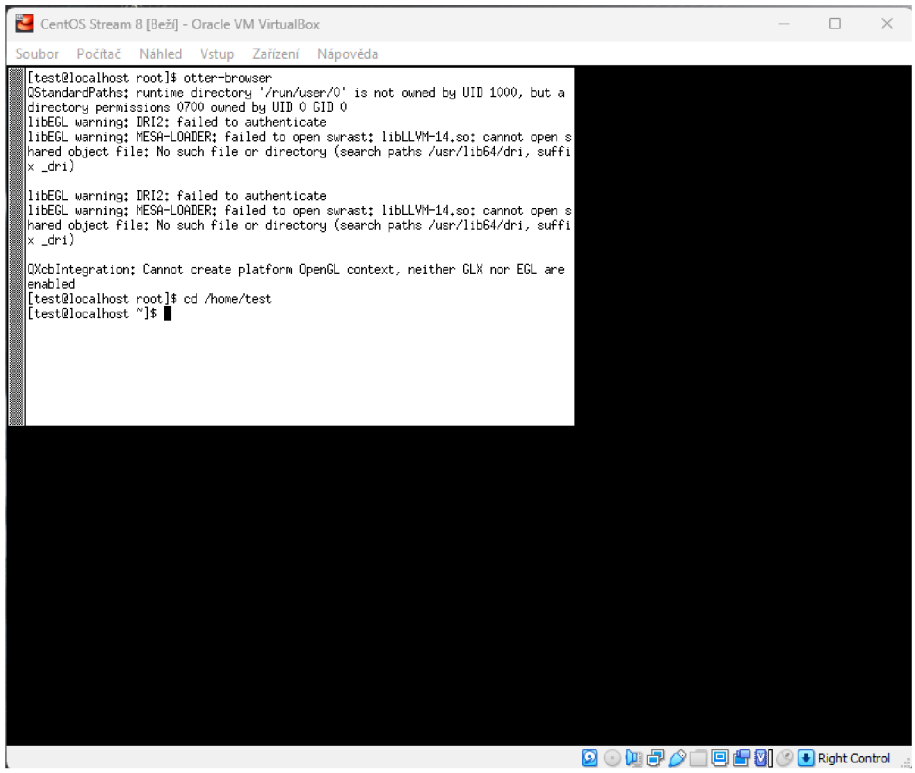

Figure B.15: Step 10

11. xpdf bsos.pdf

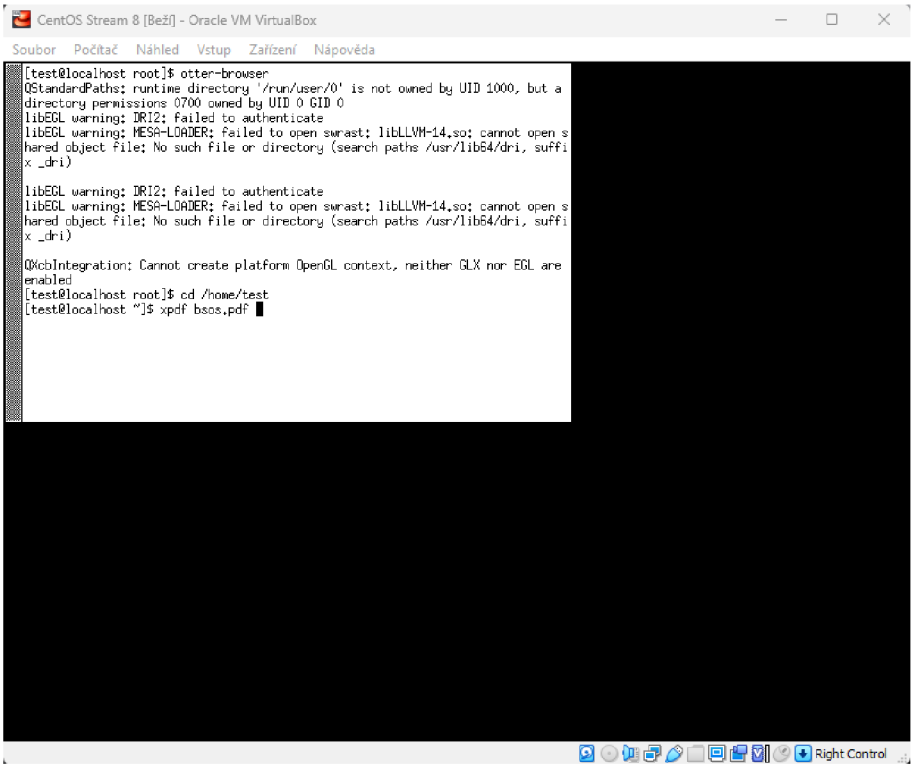

Figure B.16: Step 11 A

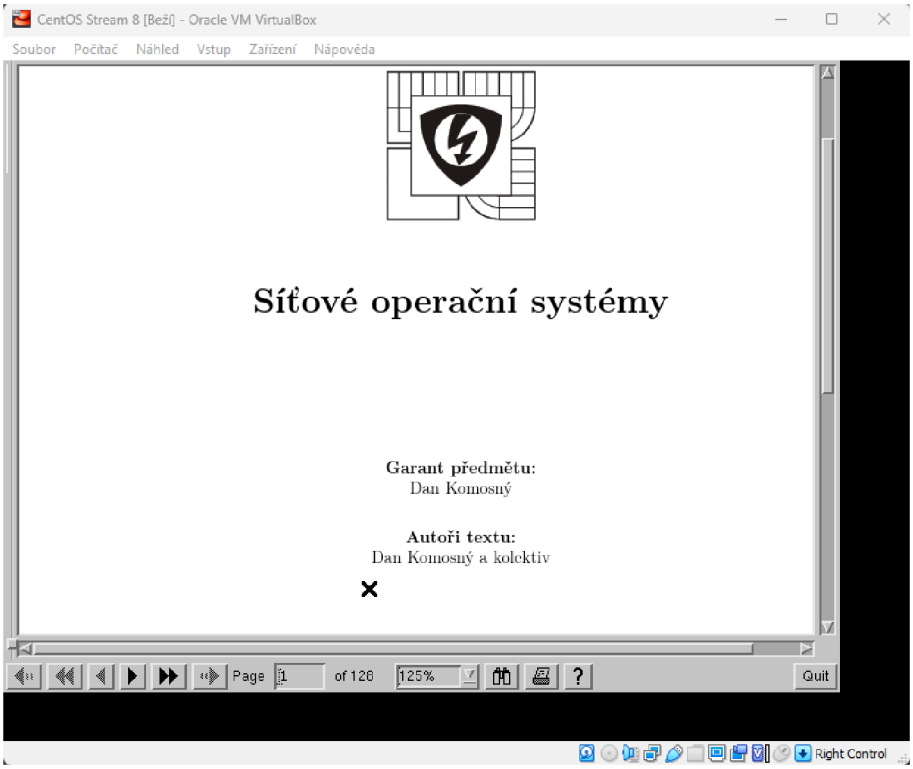

Figure B.17: Step 11 B

Appendix C – Minimal installation of CentOS Stream 8 in VM VirtualBox

1. Download CentOS Stream 8 latest image from https://linux-mirrors.fnal.gov/linux/centos/8-stream/isos/x86_64/

Index of /linux/centos/8-stream/isos/x86_64

Name	Last modified	Size
Parent Directory		-
CentOS-Stream-8-x86_64-20220728-boot.iso	2022-07-28 11:39	863M
CentOS-Stream-8-x86_64-20220728-boot.iso.manifest	2022-07-28 11:50	635
CentOS-Stream-8-x86_64-20220728-dvd1.iso	2022-07-28 12:20	11G
CentOS-Stream-8-x86_64-20220728-dvd1.iso.manifest	2022-07-28 12:20	525K
CentOS-Stream-8-x86_64-latest-boot.iso	2022-07-28 11:39	863M
CentOS-Stream-8-x86_64-latest-boot.iso.manifest	2022-07-28 11:50	635
CentOS-Stream-8-x86_64-latest-dvd1.iso	2022-07-28 12:20	11G
CentOS-Stream-8-x86_64-latest-dvd1.iso.manifest	2022-07-28 12:20	525K
CHECKSUM	2022-07-28 16:33	704
CHECKSUM.asc	2022-07-28 18:31	1.5K

Figure C.18: Step 1

2. In VirtualBox, open “computer” and click on “new”

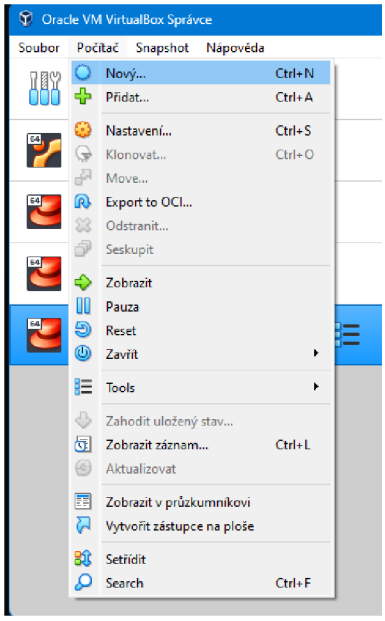

Figure C.19: Step 2

3. Name your operating system and use “version Red Hat (64 bit)”, then click “next”

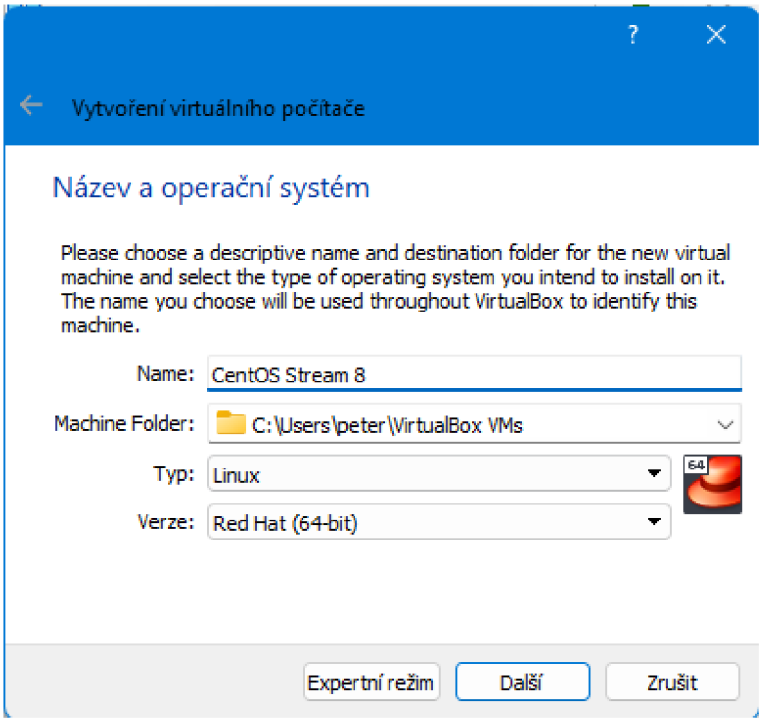

Figure C.20: Step 3

4. Use 1024 MB of ram (can be used more if you want)

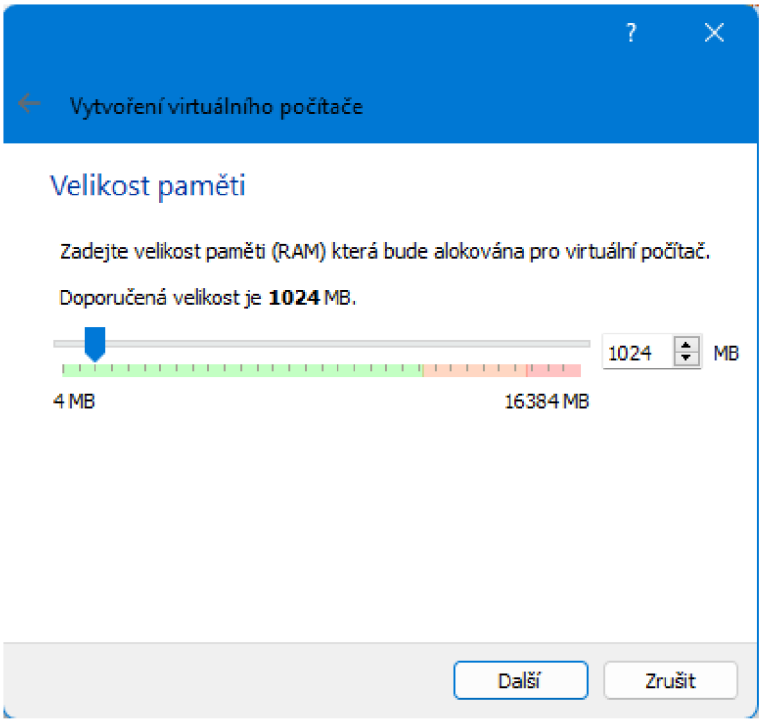

Figure C.21: Step 4

5. Create now Virtual hard disk (second option)

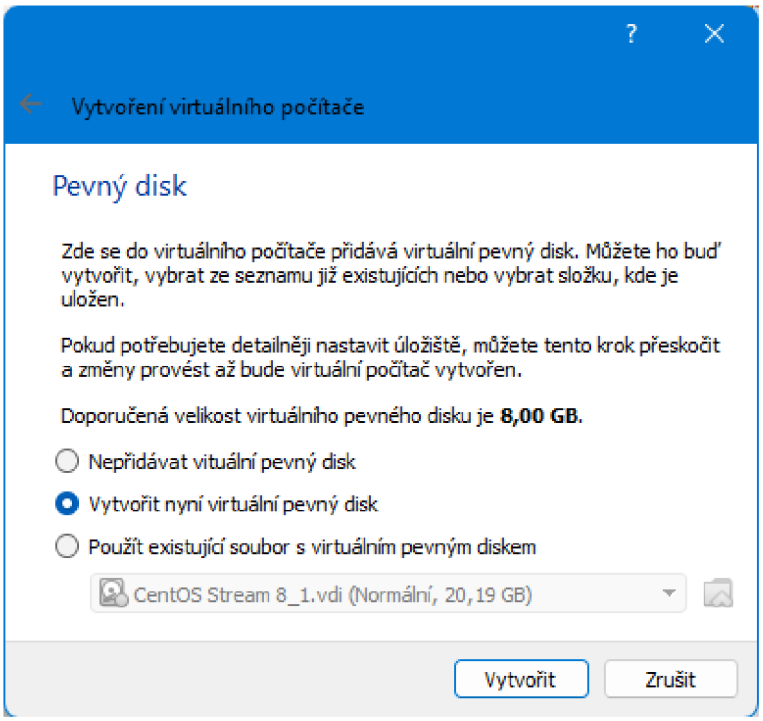

Figure C.22: Step 5

6. Select the first option, VDI (VirtualBox Disk Image)

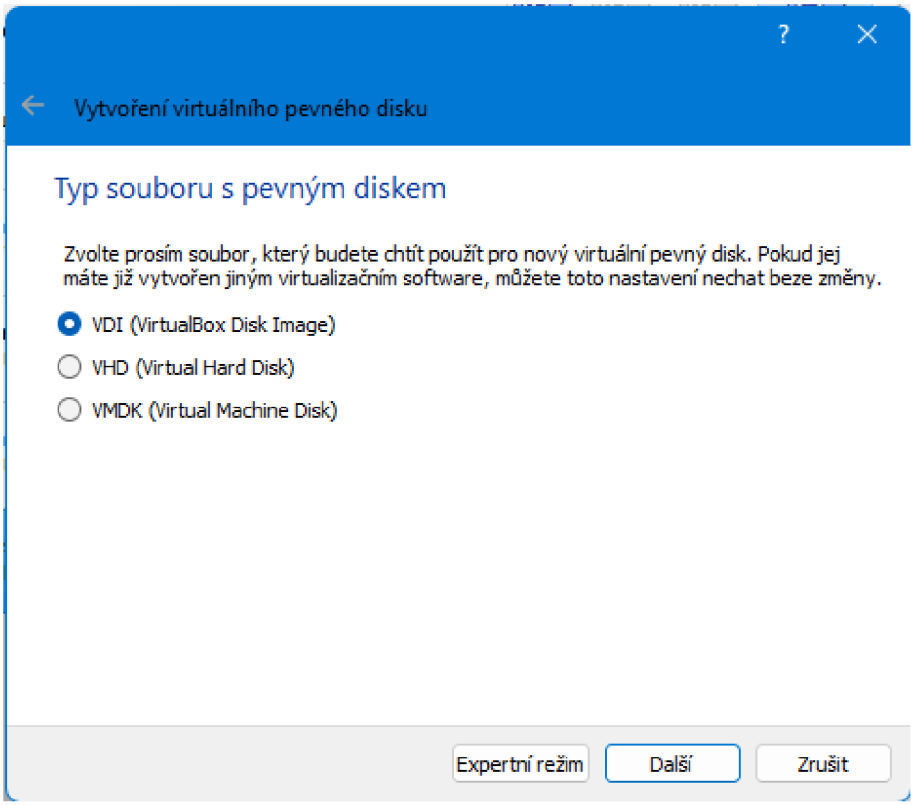

Figure C.23: Step 6

7. Use the second option “fixed size”

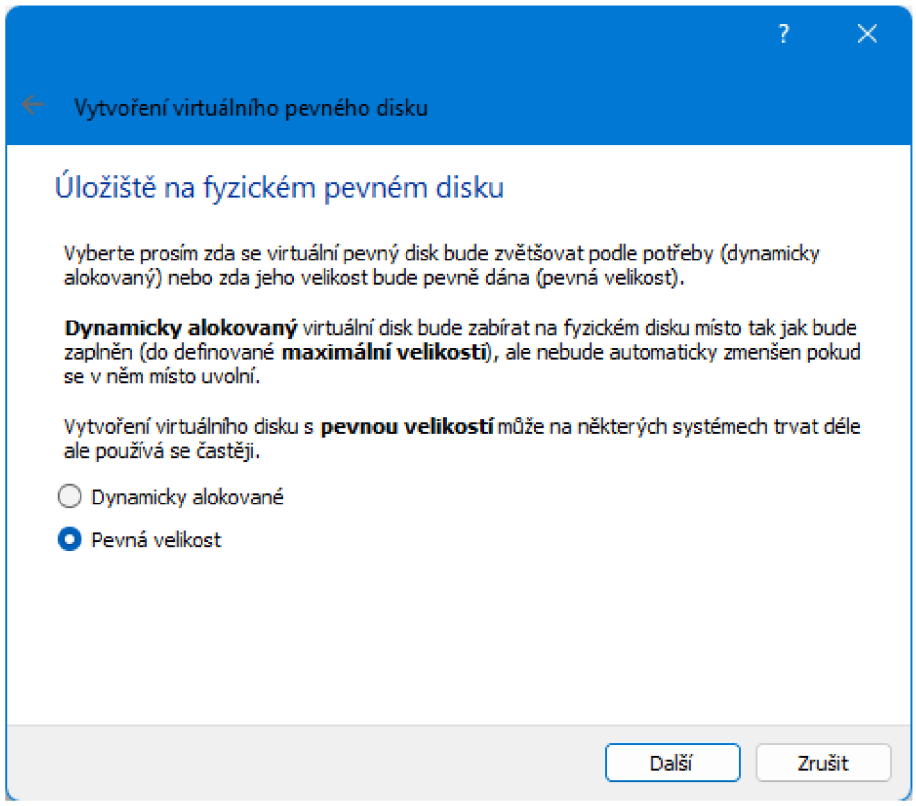

Figure C.24: Step 7

8. Set fixed size of the VDI, recommended is 8 GB, but I used 20 GB, then click “Create”

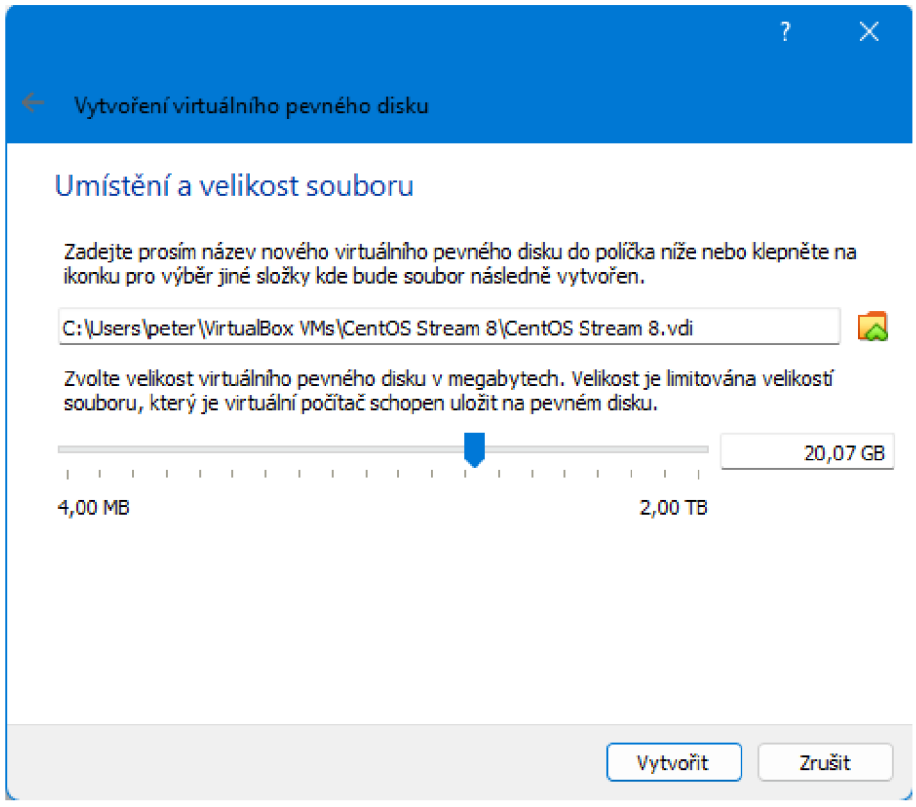

Figure C.25: Step 8

9. In VirtualBox right click at your new system, then click at “settings” and “storage”

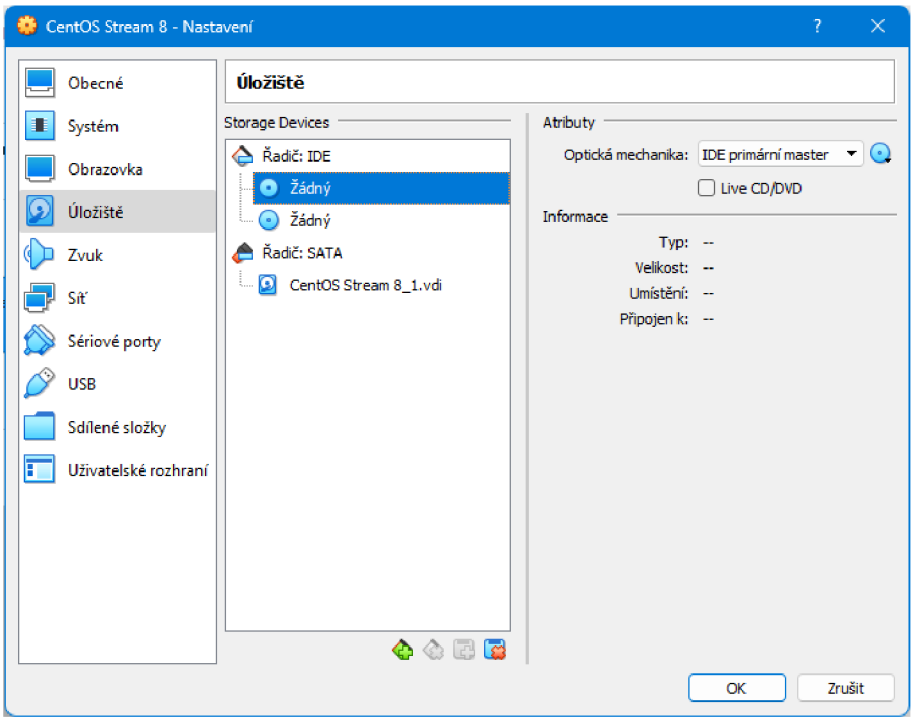

Figure C.26: Step 9

10. In storage devices click at “empty” and then in the right corner near “IDE primary master” click at the blue disc.

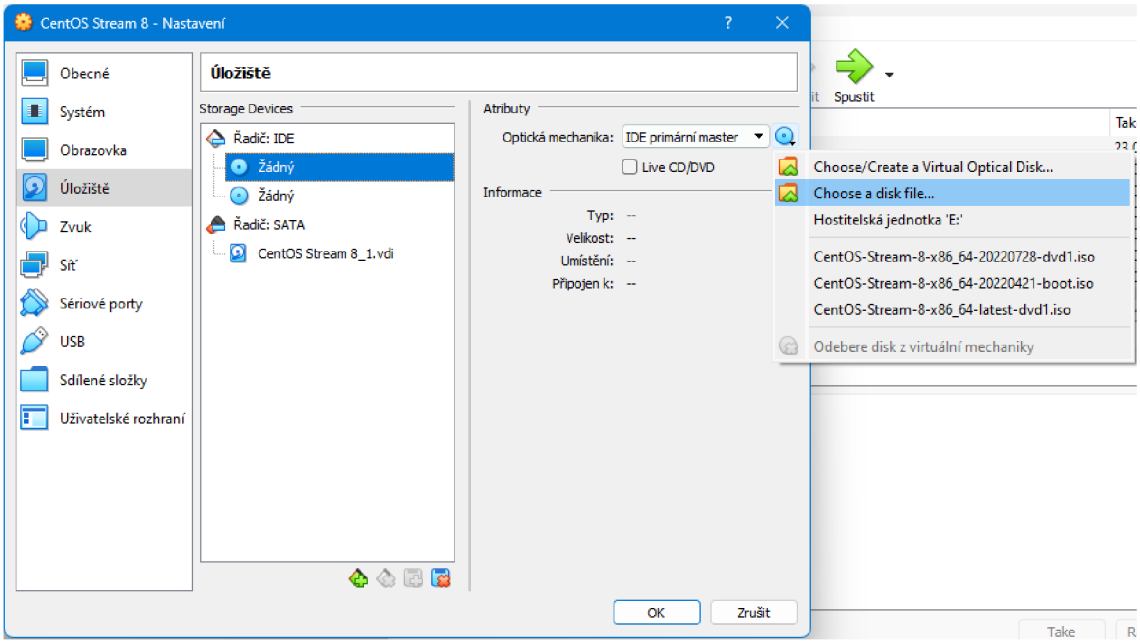

Figure C.27: Step 10

11. Use option “Choose a disk file” and then select your downloaded .iso file and click “open”

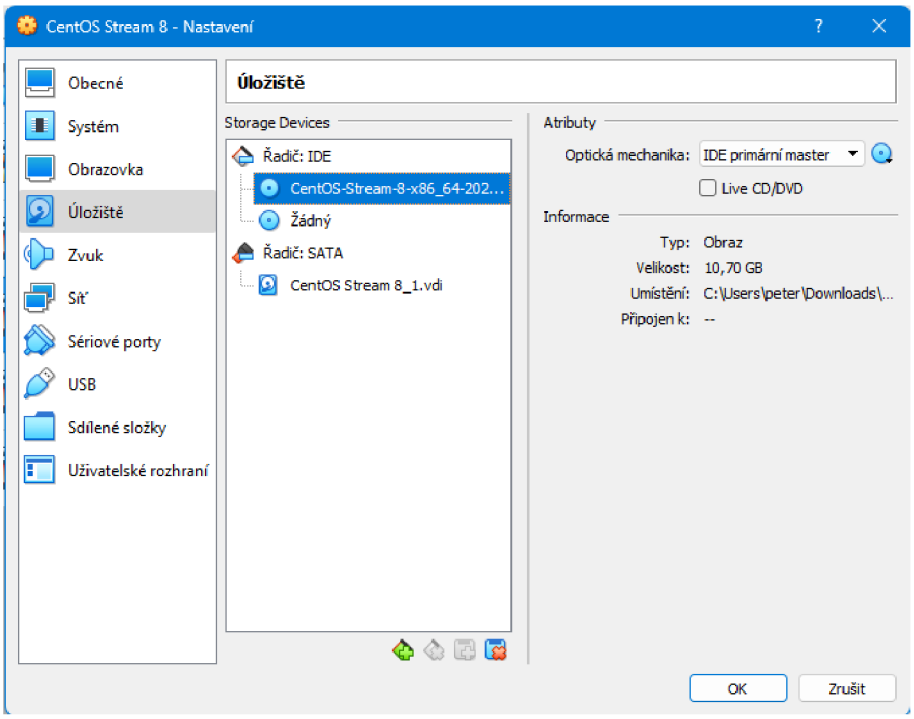

Figure C.28: Step 11

12. Click “OK” and in the menu click at the green arrow “Start”

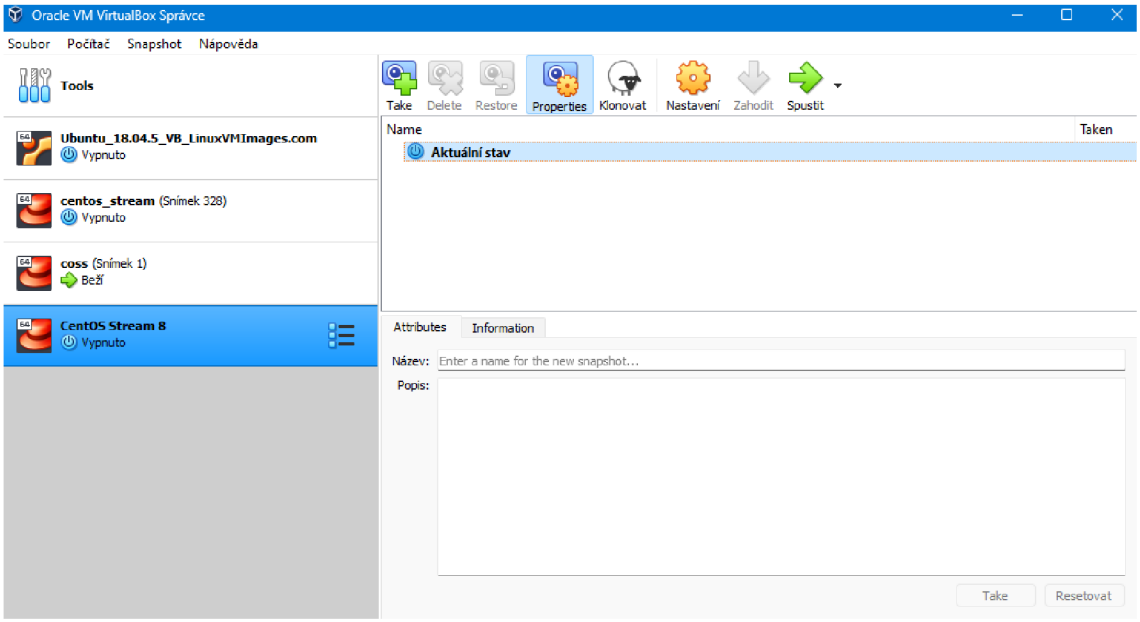

Figure C.29: Step 12

13. Choose your boot disk and click “start”

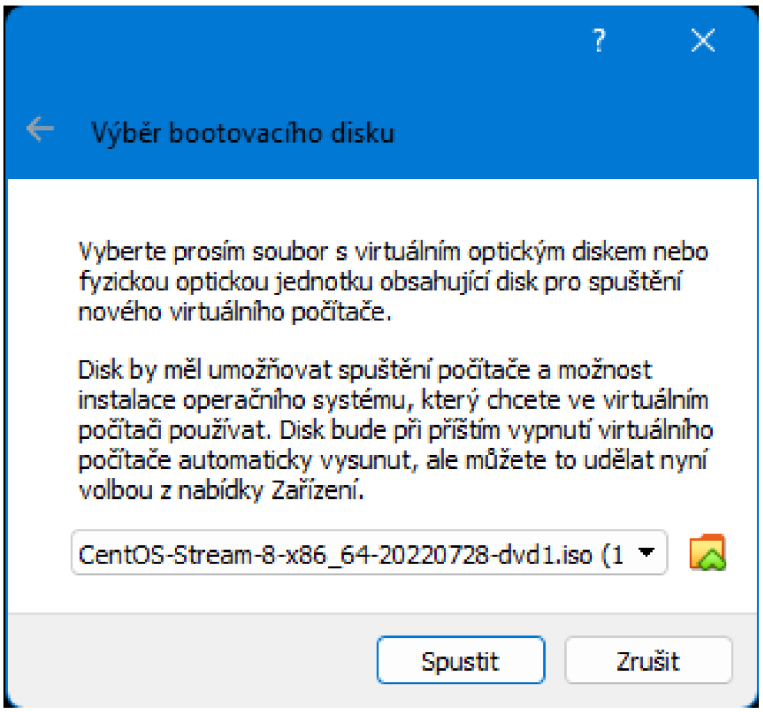

Figure C.30: Step 13

14. Choose first option

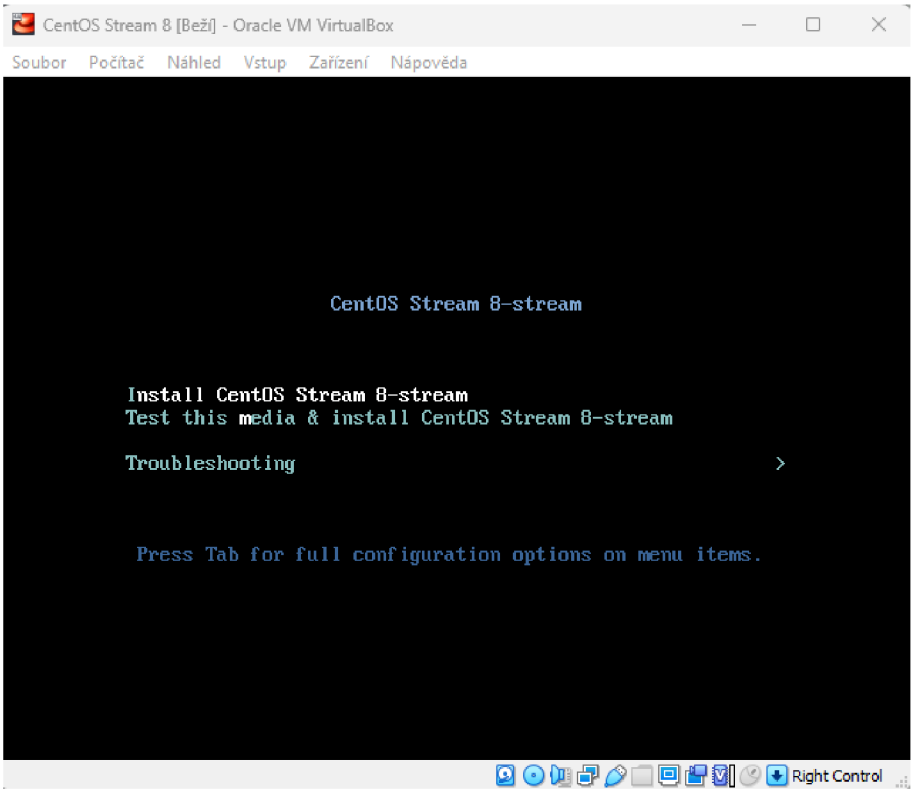

Figure C.31: Step 14

15. Select English (US) language and click continue

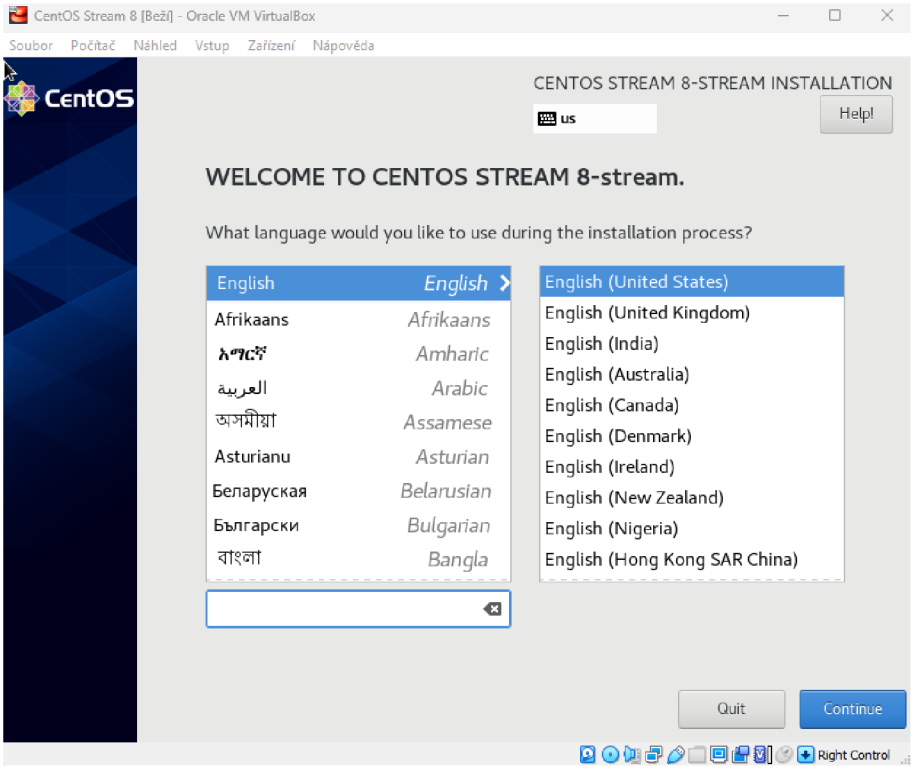

Figure C.32: Step 15

16. Select installation destination and click at “done”, make sure your disk has a check mark

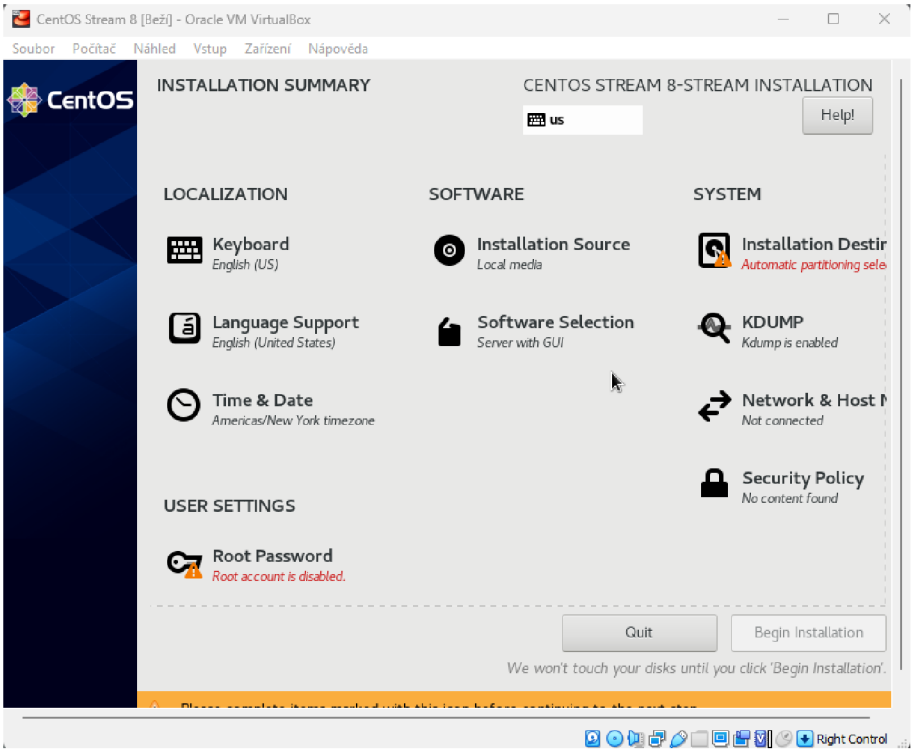

Figure C.33: Step 16 A

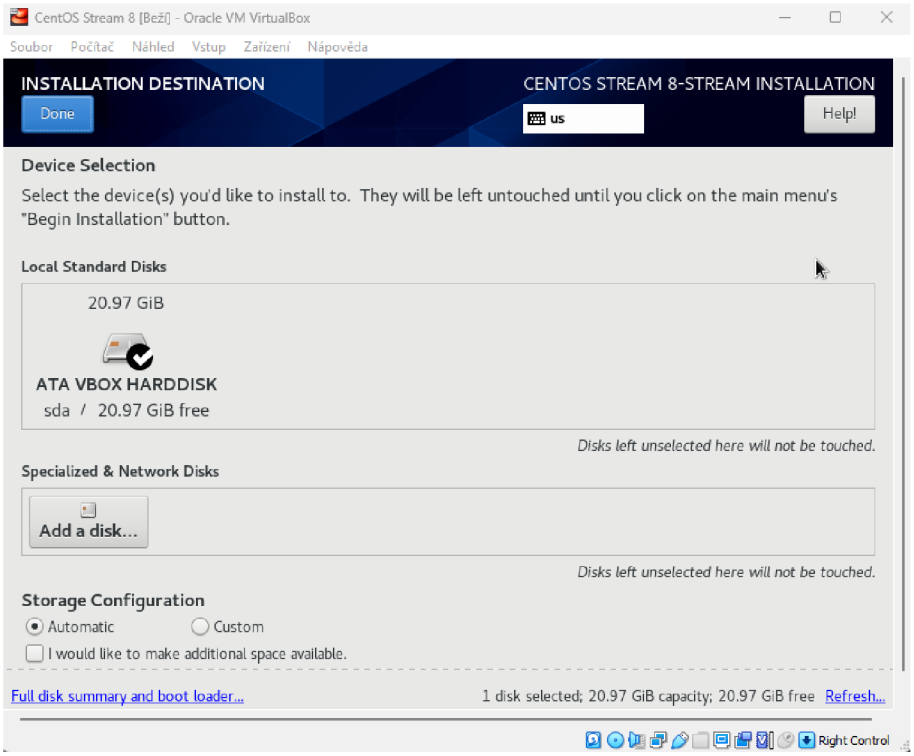

Figure C.34: Step 16 B

17. Open “Network and Host Name” and activate network connection, then click “done”

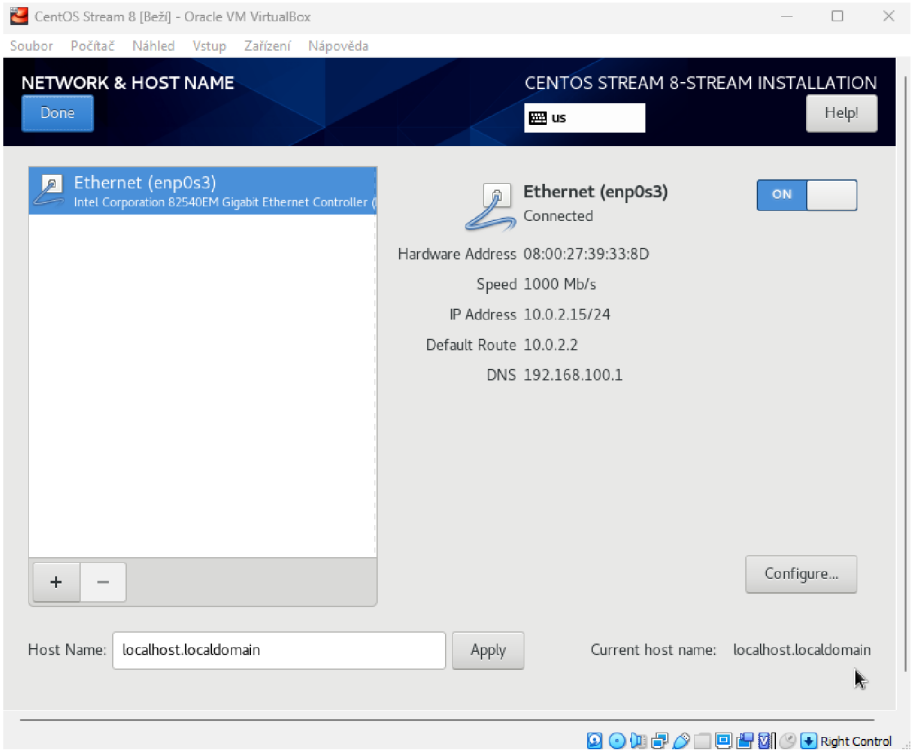

Figure C.35: Step 17

18. Click at “Software selection” and select “Minimal Install”, click at “done”

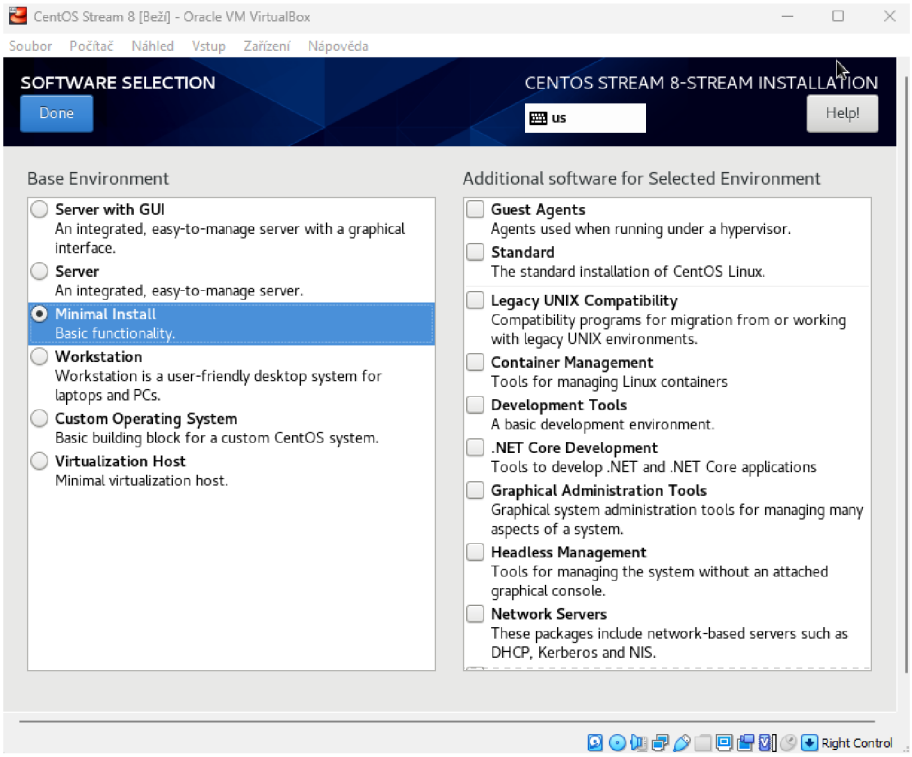

Figure C.36: Step 18

19. Click at “Root Password” and set password, then click at “done”

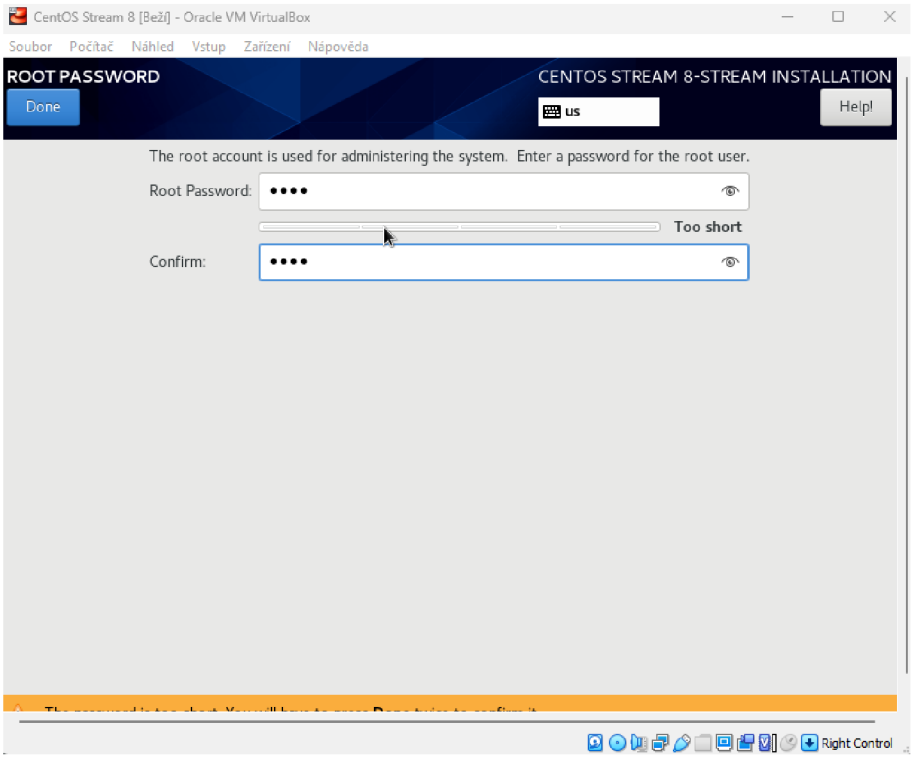

Figure C.37: Step 19

20. Click at “Begin Installation”

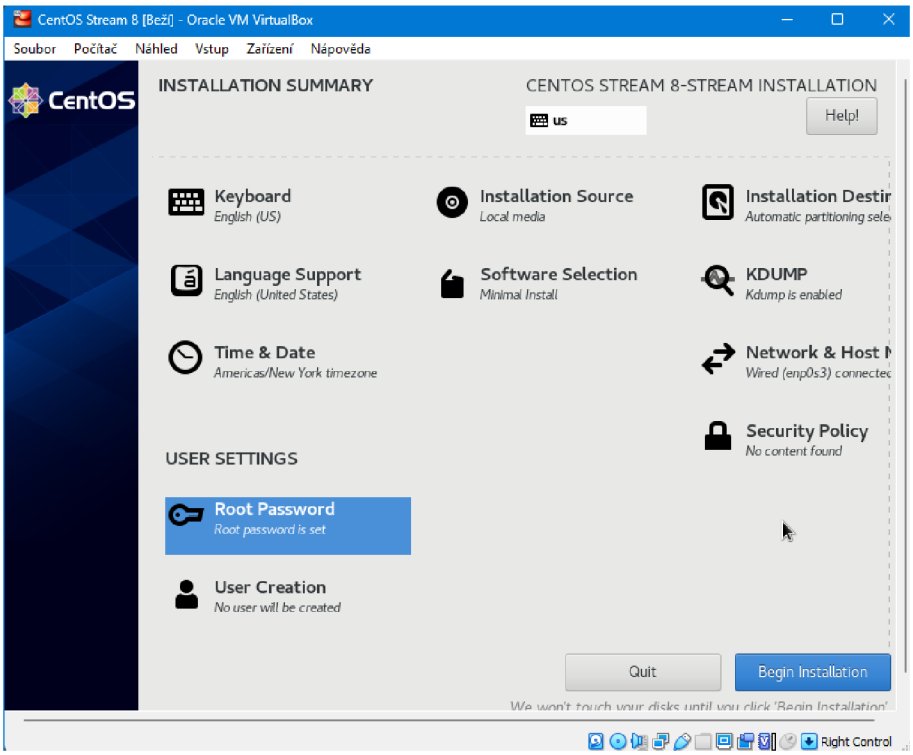

Figure C.38: Step 20

21. Reboot System

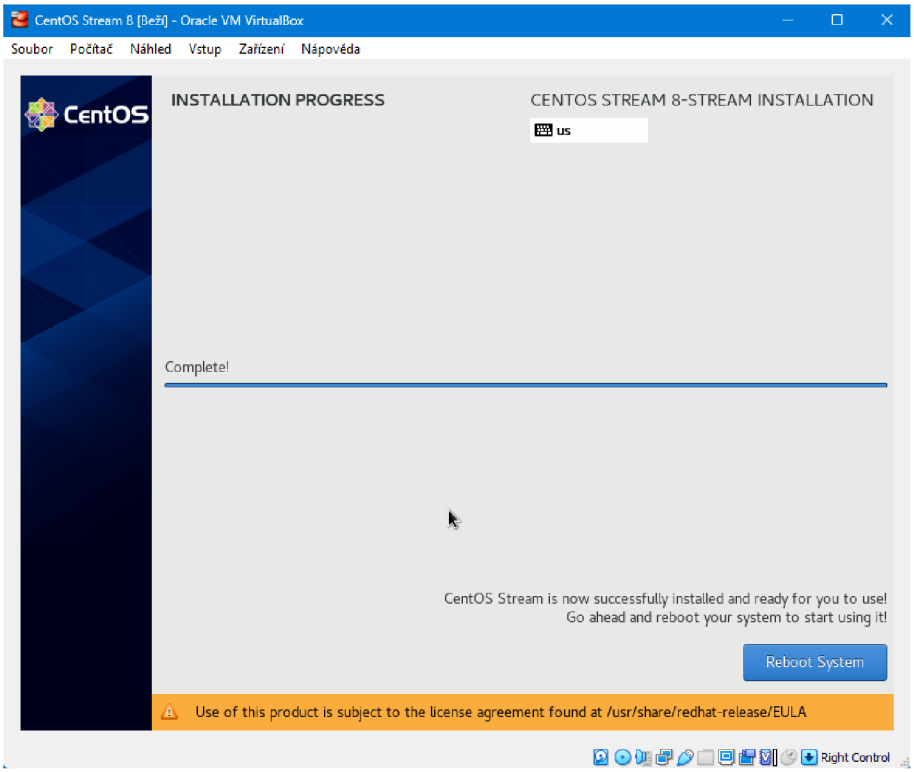

Figure C.39: Step 21

22. Select the first option

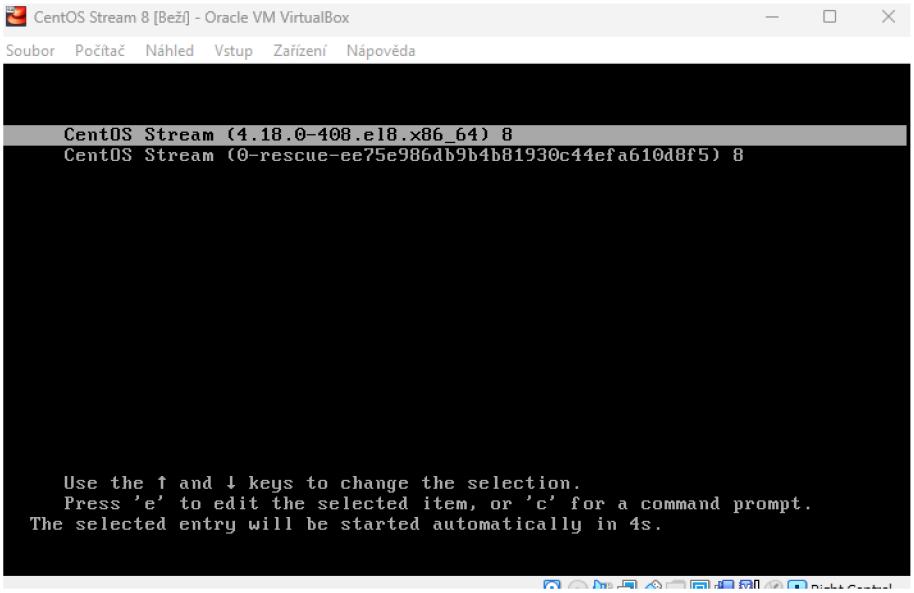

Figure C.40: Step 22

23. login

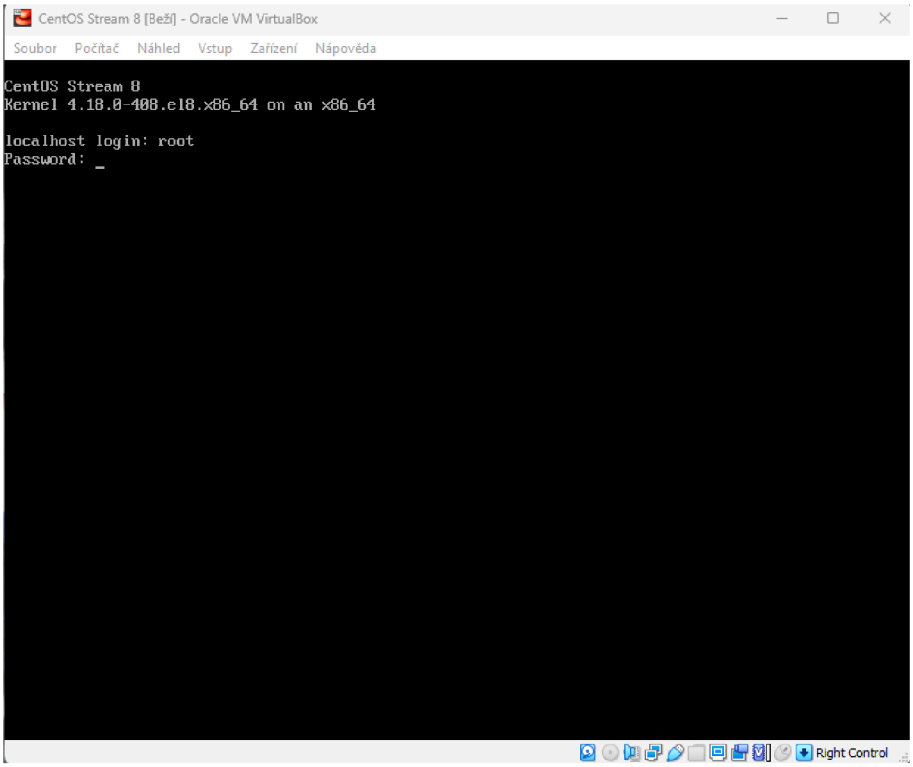

Figure C.41: Step 23