

***“Southern New Hampshire University”***

***Subject: “IT-600-Operating System”***

***Professor: Dr. Shauna Beaudin***

**Project on Linux From Scratch**

***Indrajeet Kadam***

## Contents

Introduction.....	1
Why Linux? .....	1
Open source: .....	1
High security:.....	1
High stability:.....	2
Ease of maintenance: .....	2
Runs on any hardware:.....	3
Free: .....	3
Ease of use: .....	4
Customization: .....	4
Education: .....	4
Support:.....	5
Related Work .....	6
Why to build the Linux from scratch? .....	6
Measuring installation time of LFS .....	8
Challenges:.....	9
Schedule.....	10
Final Report .....	11
System Preparation Report: .....	11
Checking host system's requirements:.....	12
Creating a partition: .....	12
Packages installation:.....	13
Building the Temporary System: .....	13
Calculating SBU's for Packages:.....	13
Graph For SBU .....	15
Pie Chart for Disk Space occupied by each package .....	16
Building the Final System: .....	16
Graph for SBU .....	17
Pie Chart for Disk Space occupied by each package:.....	17
Snapshots of the Built System: .....	18
Time of Installation:.....	19

Conclusion: .....	19
References .....	20

## **Introduction**

### **Why Linux?**

There are several reasons why Linux should be preferred over other software platform such as Windows and Mac. The following are some reasons to support the Linux system.

Few years ago, Linux was used mainly for servers and never considered suitable for desktops users. But its user interface and friendly use has been steadily improving over the last few years. Linux in today's world has become user-friendly enough to replace Windows on desktops. It is being used by hundreds of thousands of people across the globe effectively. Following are the different reasons for why Linux (Sathyanarayanan, 2017)?

**Open source:** The most important aspect of Linux is that its source code is available as it falls under the FOSS category (Free and Open Source Software). The developer community benefits from this as its members have the access freedom to view and modify the source code, which is not the case with other ownership software, where the use is limited. Number of countries are developing their own versions of Linux as per requirements. This will ultimately help them in developing their own Operating Systems for specialized or strategic areas like defense, communications, etc., to begin with. Hence, using FOSS tools is critical for the security of any country. Several countries including China, Russia and Cuba are developing their own OSs based on Linux. C-DAC has developed BOSS (Bharat Operating System Solutions), India's own PC operating system (Sathyanarayanan, 2017).

**High security:** As Installing and using Linux on your system is the easiest way to avoid viruses and malware. The security characteristics was kept in mind when developing Linux and it is much less vulnerable to viruses compared to Windows Operating System (Sathyanarayanan, 2017).

Programs cannot make changes to the system settings and configuration unless the user is logged in as the root (equivalent to the administrator user in Windows) user. Most of time the users do not log in as the root; hence, they cannot do much damage to the system, except to their own files and programs, since the downloaded file/malware will have limited privileges. You can browse the Internet without worrying about your system getting infected. However, users can install the antivirus software in Linux to further secure their systems (Sathyanarayanan, 2017).

The reason for this higher level of security is that since Linux is open source nature of software, where the source code is available for review. A huge number of developers all over the world have gone through the code, which means that most of the flaws have already been discovered.

**High stability:** The Linux system is very stable and is not prone to crashes. The Linux OS runs exactly as fast as it did when first installed, even after several years. Most of us must have experienced how a freshly installed Windows system runs extremely fast and the same system becomes slow after around six months to one year. Then, your only option most of the time is to reinstall the OS and all the other software (Sathyanarayanan, 2017).

The uptime for the Linux servers is very high and the availability is around 99.9%. Unlike Windows, you need not reboot a Linux server after every update or patch. Due to this, Linux has the highest number of servers running on the Internet. Some research says that 96.3 per cent of the top 1 million Web servers are running on Linux. Twenty-three out of the Top twenty-five websites run on Linux.

**Ease of maintenance:** Maintaining the Linux OS is easy, as the user can centrally update the OS and all software installed very easily. All the variants of Linux have their own central software

repository, which is used to update the system and keep it safe. They offer regular updates and the system can be updated without rebooting it. The updating can be done periodically, with just a few clicks, or users can even automate the updating process. On the other hand, updating a Windows system is not so easy. Also, in Windows, all the third-party software must be updated individually (Sathyanarayanan, 2017).

**Runs on any hardware:** All of us know that with every new release of Windows OS, a huge number of hardware systems become obsolete as their technical specifications are no longer adequate to run the latest Windows OS. Linux makes very efficient use of the system's resources. Linux installation can be customized for users and for specific hardware requirements. The installation procedure is very flexible and allows users to choose the modules they want to install. This allows them to install Linux even on old hardware, thus helping in optimal use of all the hardware resources. Linux runs on a range of hardware, right from supercomputers to watches. You can give new life to your old and slow Windows system by installing a lightweight Linux system, or even run a NAS or media streamer using a distribution of Linux (Sathyanarayanan, 2017).

**Free:** Linux is completely free, and users do not need to pay for anything. All the basic software required by a typical user and even an advanced user is available. Dozens of educational software are available under Linux. Even the equivalent of professional software for desktop publishing, photo editing, audio editing and video editing are available. Businesses can use the software free of cost and reduce their IT budgets substantially (Beekmans, 2019).

**Ease of use:** Contrary to the general belief that Linux is only for geeks, it has now become user-friendly and has a good graphical user interface (GUI). It has almost all the functionality that Windows has. The GUI has developed to the extent that most of what typical users want can be done on Linux, as easily as it is done in Windows, without knowing any commands. In case you are using some applications, which run only on Windows, you can install Wine (Windows Emulator), using which you can run those applications on a Linux system. Though the perception is that Linux is not gamer friendly, several games are now available on Linux. Even if a game is not available on Linux, you can install Play on Linux to run Windows games (Beekmans, 2019).

**Customization:** Users have tremendous flexibility in customizing the system as per their requirements. There are numerous choices for wallpapers, desktop icons and panels. There are more than half-a-dozen desktop environments to choose from, like GNOME, KDE, etc. For any task, right from the GUI interface and file managers, to DVD burners and browsers, around four to six options are available for any software. The Linux versions of most popular browsers are available. The Linux philosophy is based on using several small programs, each of which does one task very well. But these programs can be combined to write powerful programs and utilities. The Linux OS offers a command line interface with several shells to choose from. Systems administrators can enjoy the powerful command line interface and write shell scripts to automate routine maintenance and various other tasks (Sathyanarayanan, 2017).

**Education:** This is the most useful aspect for students, as they can use the software to study how it works, before modifying and extending the code to suit their needs. This will also help them to learn the internals of an OS and the software. This process will help in the development of new software and aid innovation based on local needs. Even if users are not programmers, they can

contribute to Linux by helping in documentation, translation and testing. Linux can be a fantastic educational tool for schools and colleges as free software is available to aid in teaching. Proprietary software for computation, like MATLAB, is very expensive. There are alternatives available to it like Scilab and GNU Octave. Linux software is available in many area Celestia and Stellarium for astronomy, Avogadro and Gab edit for chemistry, EMBOSS and Tree View X for biology; and ROOT, Octopus and Step for physics (Sathyanarayanan, 2017).

**Support:** There is strong community support for Linux over the Internet through various forums. Any question posed in forums will usually get a quick response as a lot of volunteers are online and working to solve the problems due to their passion for Linux. The paid support option is also available for commercial enterprises, with companies like Red Hat and Novell offering 24×7 support for critical applications and services (Sathyanarayanan, 2017).



## **Related Work**

### **Why to build the Linux from scratch?**

The important reason for to build the scratch is to help learn how a Linux system works from inside out. And how the things work together and depend on each other. This learning experience can provide the ability to customize a Linux system to suit the user's needs.

Also, the Linux From Scratch (LFS) allows the users to have more control over the system than others Linux implementation. It allows to create the compact Linux systems. When installing the regular distribution, it forces to install some programs which are never used. These programs waste the resources. Another reason of this custom build Linux system is security. By compiling from the source code, one can concern everything and apply the all the security patches that is needed. There is no need to wait to compile the system and fix the security issue. The goal is to create the complete and usable foundation-level system with the complete control and understanding of the user (Beekmans, 2019).

## **Resources Needed**

### **Hardware**

We are using the Windows Operating system for the demonstration and performing the live practical on the same. As the Linux doesn't care about the in-build hardware of the device, means what is the configuration of the physical device like its processor, RAM, storage capacity and the display. For this demonstration we have used the Dell Laptop with licensed copy of Windows 10 installed on it. With a good internet connectivity on the same.

### **Software**

As a part of software to create the virtual machine we have installed the Oracle VM VirtualBox Manager. It allows you to allocate the certain amount of CPU, RAM, Disk and other peripherals. Which allows you to import, add and create the new VMs so that you can perform the installations, modifications and all. Moreover, we also downloaded the hybrid ISO (Live DVD) of the Linux system that act as operating system here which we are developing and doing the modifications in it. As it is rotational release there is no need to worry about the version while downloading.

#### **Hybrid ISO (Live DVD)**

#### **Download**

[livedvd-x86-amd64-32ul-20160704.iso](#) or

[livedvd-x86-amd64-32ul-20160704.iso.DIGESTS](#)

**Measuring installation time of LFS**

Standard Build Units(SBUs) are a method of predicting approximately how long it will take to compile and install a package when building according to the instructions in the Linux from Scratch(LFS).The time for building the first package in LFS is measured and used that as a standard. This is taken as 1.0 SBU. Then dividing all the other times by the standard time will result in SBU time of all the other packages. For example, if the time it takes to build GCC for the first time is 3.9 SBU and it took 10 mins to compile and install the first program, Binutils, then it will be known that nearly 40 mins will be required to build the static GCC (Beekmans, 2019).

The time of installation will be measured in seconds and in the end will be converted to minutes.

We will be using time stamp command to record the time of installation for each package.

**Challenges:**

The main challenge of developing LFS is making sure that system is turned on while the packages are being installed. It is uncertain how fluent the installation will go while performing. LFS packages take different installation time depending on which machine is used for installation. Keeping the system turned on and preparing for the time of installation is the biggest challenge of this project. Also, the connectivity of the network rises some issues like the download speed of the different packages and installation of the specially the SBU's.

During the first installation attempt, the installation process was interrupted due to closing the console by mistake and the system was shut down. The files could not be retrieved and had to restart the process again after completing the installation of the temporary system. The limitations of knowledge of the Linux system made it difficult to retrieve when something went wrong.

**Schedule**

In the first week, some background research about the Linux Operating system was done. For example, what is Linux? What are the advantages of Linux Operating system? How does this operating system work? Which hardware and software will be needed for the Linux installation? How much time will the installation require? What experimental setup is needed to create for installation?

For the Linux installation procedure, "Linux from Scratch" video from YouTube, which is 22 hours in length is being referred. The steps are being followed from that video. After this, the environmental setup for the Linux installation was done. One system is being used for installation.

## Final Report

### System Preparation Report:

Figure 1 Oracle VM VirtualBox Manager

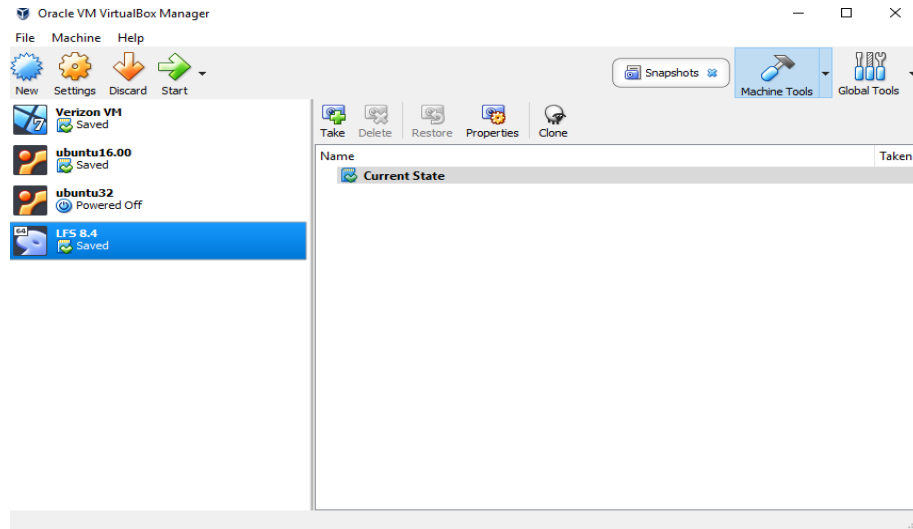
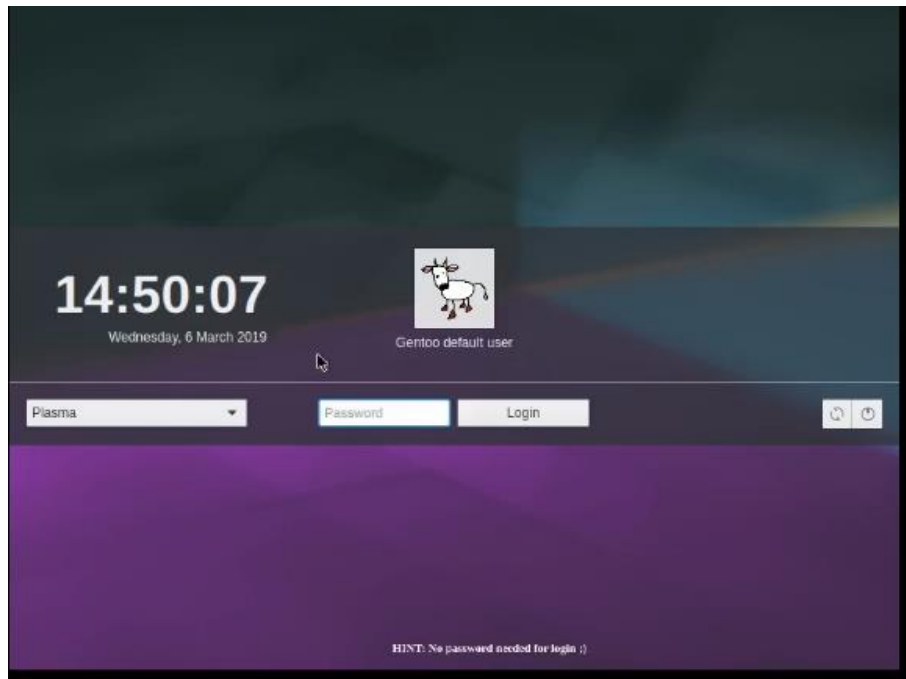


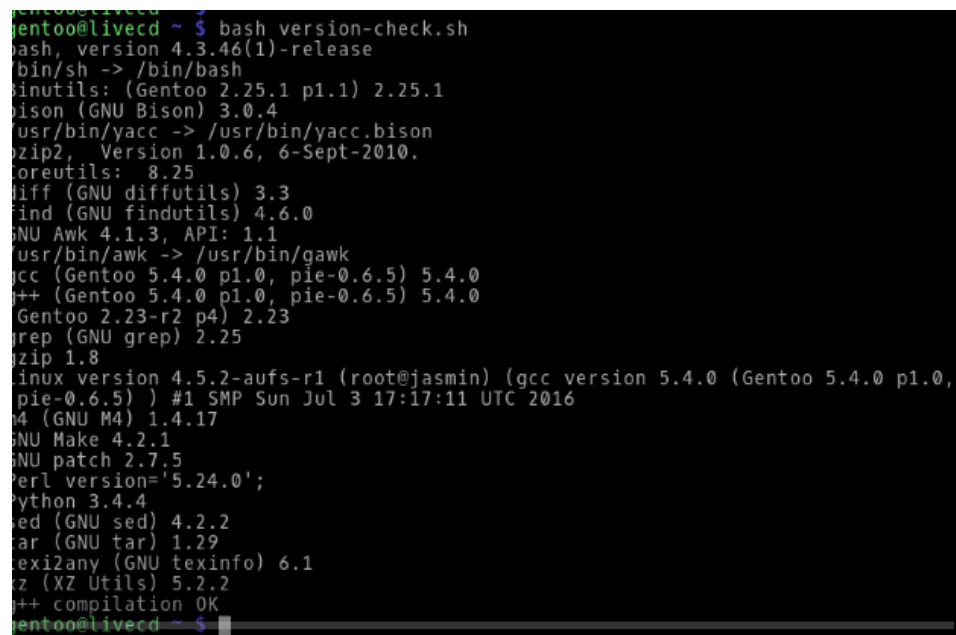
Figure 2 Host system: Gentoo Linux



### Checking host system's requirements:

For preparing the LFS installation, the system requirement for Gentoo Linux was checked using Linux command and it met all the requirements.

Figure 3 Snapshot of Testing Host System Requirement

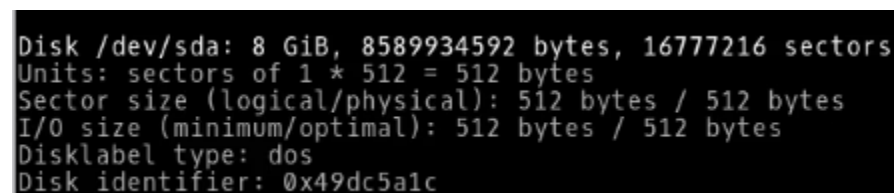


```
gentoo@livecd ~ $ bash version-check.sh
bash, version 4.3.46(1)-release
/bin/sh -> /bin/bash
binutils: (Gentoo 2.25.1 p1.1) 2.25.1
bison (GNU Bison) 3.0.4
/usr/bin/yacc -> /usr/bin/yacc.bison
bzip2, Version 1.0.6, 6-Sept-2010.
coreutils: 8.25
diff (GNU diffutils) 3.3
find (GNU findutils) 4.6.0
GNU Awk 4.1.3, API: 1.1
/usr/bin/awk -> /usr/bin/gawk
gcc (Gentoo 5.4.0 p1.0, pie-0.6.5) 5.4.0
g++ (Gentoo 5.4.0 p1.0, pie-0.6.5) 5.4.0
Gentoo 2.23-r2 p4) 2.23
grep (GNU grep) 2.25
gzip 1.8
linux version 4.5.2-aufs-r1 (root@jasmin) (gcc version 5.4.0 (Gentoo 5.4.0 p1.0,
pie-0.6.5) ) #1 SMP Sun Jul 3 17:17:11 UTC 2016
m4 (GNU M4) 1.4.17
GNU Make 4.2.1
GNU patch 2.7.5
Perl version='5.24.0';
Python 3.4.4
sed (GNU sed) 4.2.2
tar (GNU tar) 1.29
texi2any (GNU texinfo) 6.1
xz (XZ Utils) 5.2.2
g++ compilation OK
gentoo@livecd ~ $
```

### Creating a partition:

LFS is installed on a dedicated partition of the hard disk. 8 GB was used for installation with two partitions:

Figure 4 Creation of Partition for LFS



```
Disk /dev/sda: 8 GiB, 8589934592 bytes, 16777216 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x49dc5a1c
```

**Packages installation:**

LFS is built from extracting the packages which contains source code to build the LFS. Packages were installed in the source file before starting with installation.

**Building the Temporary System:**

Temporary system contains the tools to build the final system. It builds the compiler, assembler, linker and libraries that an OS is required to have. Using these tools, the final system was built as described in the later section. The following packages were configured, compiled and installed for building temporary system. The table shows the time (converted in SBU-standard built unit) and the graph for each package:

**Calculating SBU's for Packages:**

1 SBU=4 min 55 .0 sec. This time includes the sum of time for configuration, compilation and extraction of each package. Time for Binutils is considered as a standard time.

Hence 1 SBU= 4 min 55 sec= 306 secs

For the rest of the packages, time was converted into SBUs using the following math:

**Example:**

GCC pass 1 time=61.16 mins (sum of configuration, compilation and extraction time)

GCC pass 2 time=66.34 mins (sum of configuration, compilation and extraction time)

Total time = 127.5 mins=7650secs

306 secs = 1 SBU

7650 secs = 25.0 SBUs. Similarly, SBU is calculated for the rest of the packages



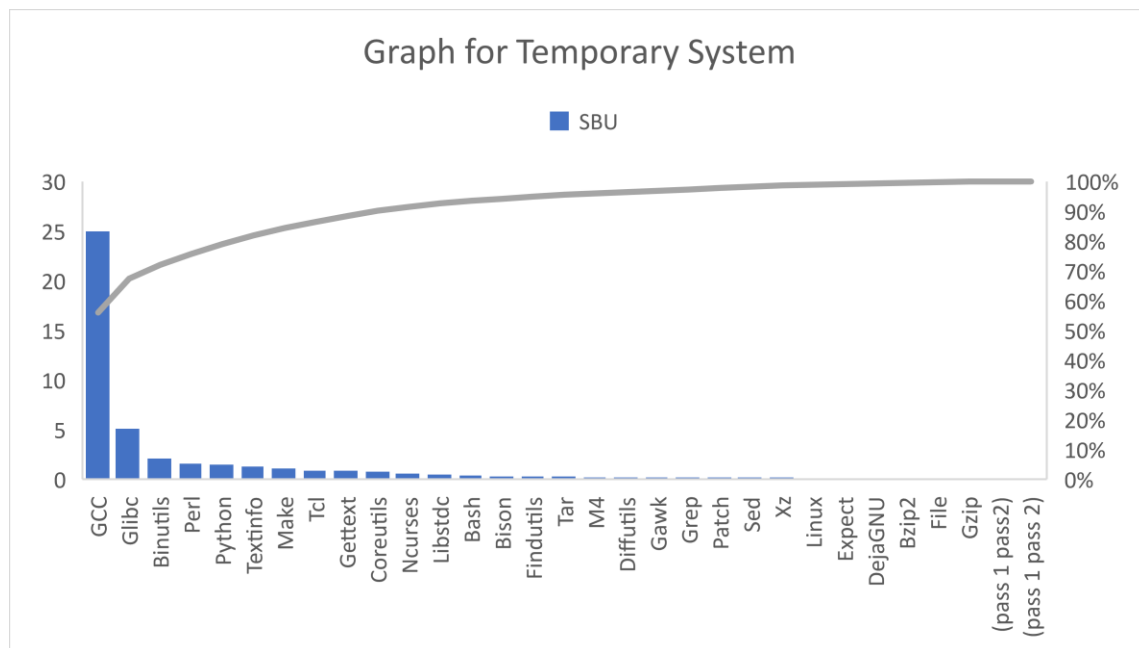
*Figure 5 SBU's for Temporary System*

<b>Packages:</b>	<b>SBU</b>	<b>Disk Space (MB)</b>
Binutils (pass 1 pass2)	2.1	1178.0
GCC (pass 1 pass 2)	25.0	6450.6
Linux	0.1	937.0
Glibc	5.1	885.0
Libstdc	0.5	803.0
Tcl	0.9	66.0
Expect	0.1	3.9
DejaGNU	0.1	3.2
M4	0.2	20.0
Ncurses	0.6	41.0
Bash	0.4	67.0
Bison	0.3	37.0
Bzip2	0.1	5.5
Coreutils	0.8	148.0
Diffutils	0.2	26.0
File	0.1	18.0
Findutils	0.3	36.0
Gawk	0.2	43.0
Gettext	0.9	173.0
Grep	0.2	24.0
Gzip	0.1	10.0
Make	1.1	13.0
Patch	0.2	12.0

Perl	1.6	275.0
Python	1.5	371.0
Sed	0.2	20.0
Tar	0.3	38.0
Textinfo	1.31	104
Xz	0.2	18

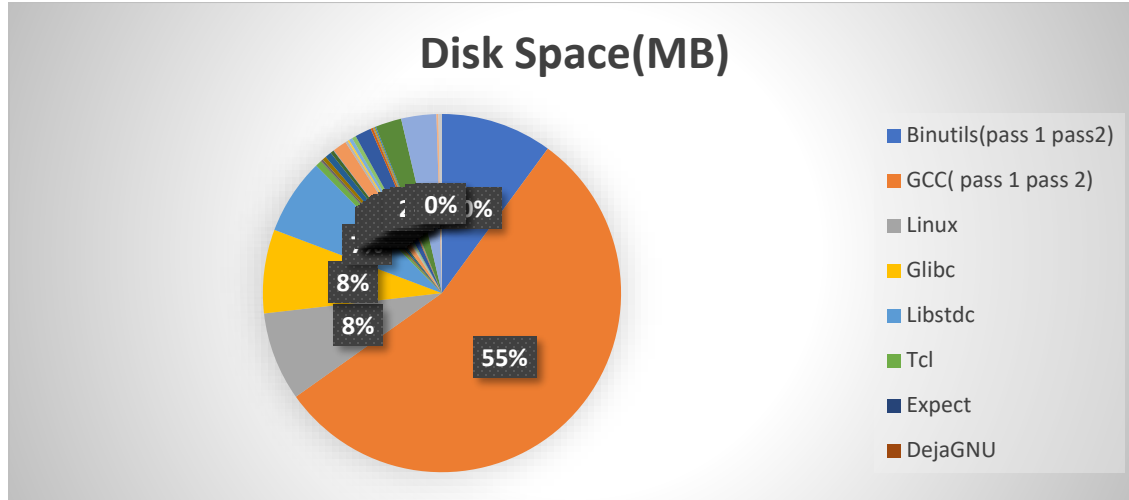
## Graph For SBU

Figure 6 GRAPH for SBU



## Pie Chart for Disk Space occupied by each package

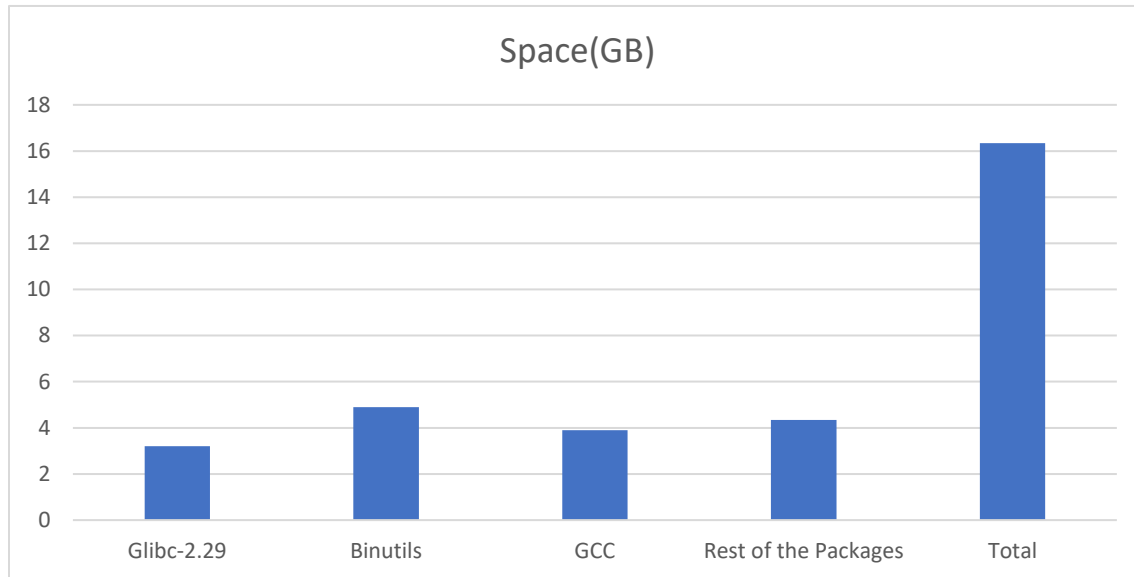
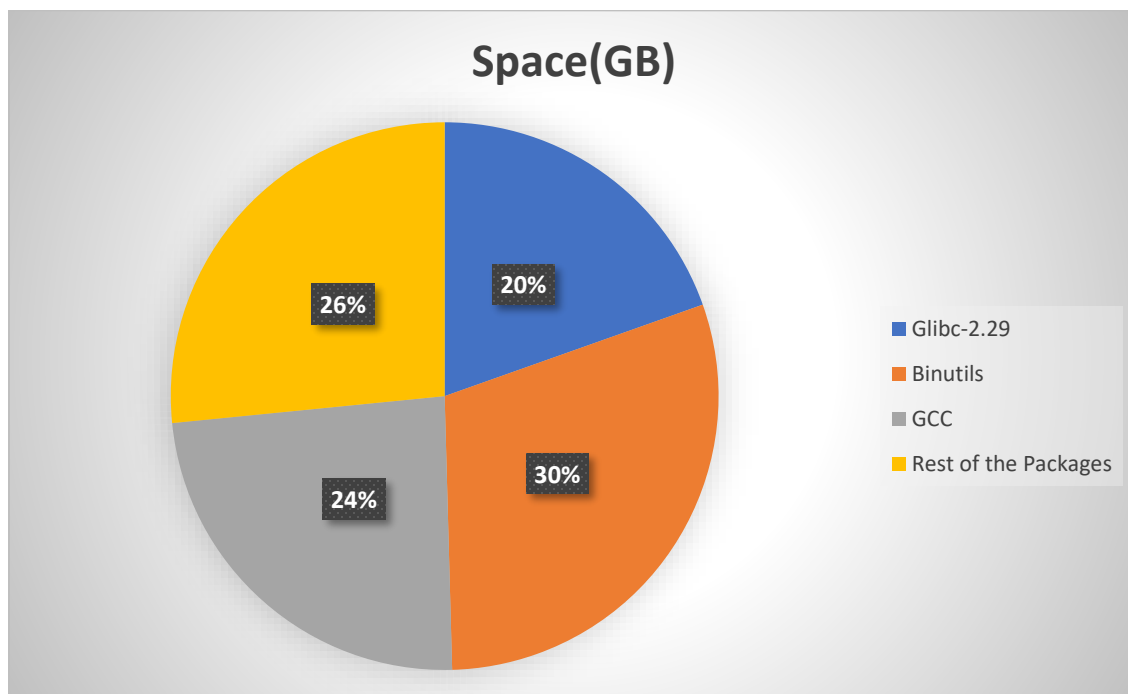
Figure 7 Disk Space occupied by each package



## Building the Final System:

Like the temporary system, packages were configured, compiled and installed for building the final system. The only difference here is that, the tools that were extracted while building the temporary system were used here.

Package	SBU	Space (GB)
Glibc-2.29	22.0	3.2
Binutils	6.9	4.9
GCC	92	3.9
Rest of the Packages	50.3	4.3426

**Graph for SBU***Figure 8 Bar Graph for SBU Final System**Figure 9 Pie Chart for SBU's of Final System*

## Snapshots of the Built System:

Figure 10 Snapshots for LFS Selection



Figure 11 Snapshot for LFS Login

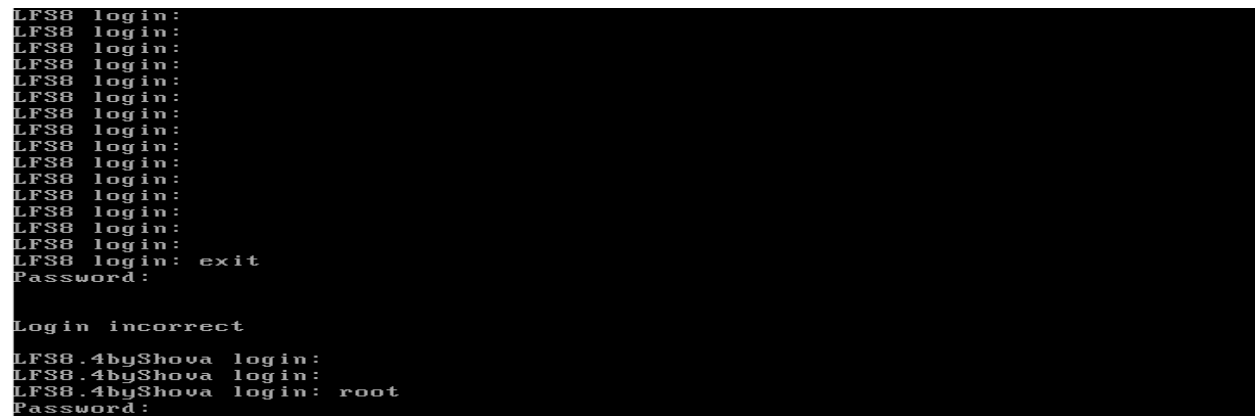
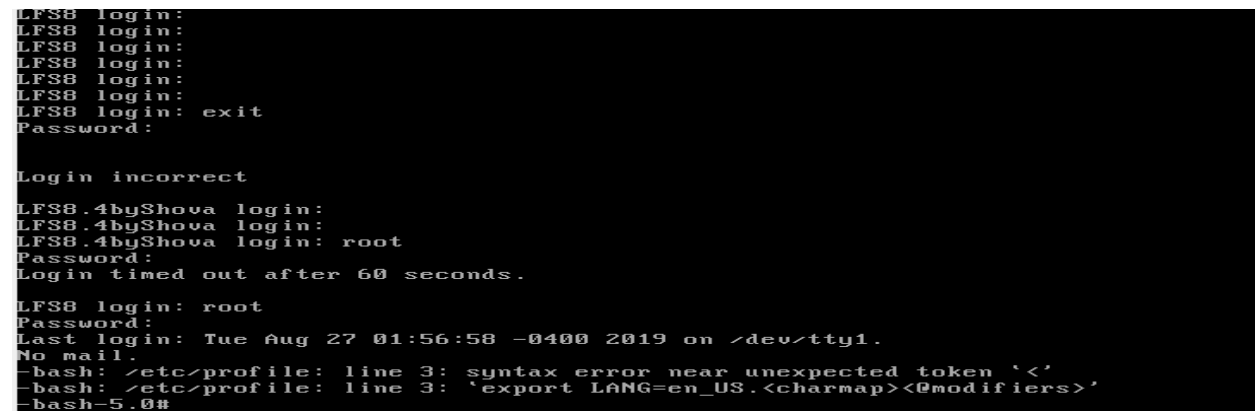


Figure 12 Snapshot for LFS Bashscript



**Time of Installation:**

The total time of installation for the project was approximately 23 hours which includes system preparation (1 hr.), building the temporary system (6 hrs.) and installing the final system from the tools built in the temporary system (17 hrs.)

**Conclusion:**

The installation process is completed successfully. The final system is a very basic Linux system which provides basic OS functions. During the installation process, there was a learning process about the basic packages, tools and the process of developing a Linux system. Each package has different roles which provides different features to the final system. Now this system could be used to learn Linux commands and even could be expanded using BLFS (Beyond Linux From Scratch). The system could be customized according to user's needs.

**Shortcomings:**

There are some errors in the system. During the installation process, two packages "meson" and "findutils" were not installed properly. Due to this reason, there are some errors in the main system. This could be solved by reinstalling the packages.

**References**

Beekmans, G. (2019). *Linux From Scratch 8.4*. Retrieved from Linux From Scratch:  
<http://www.linuxfromscratch.org/>

Sathyanarayanan, S. (2017, March 21). *Ten reasons why we should use Linux*. Retrieved from  
OpenSource: <https://opensourceforu.com/2017/03/reasons-to-use-linux/>

1 of 3 How to build Linux From Scratch (LFS) 8.4 by Kernotex - Part 1 [Video file]. (2019,  
April 3). Retrieved from <https://www.youtube.com/watch?v=fmWz9Lkd0Kw>

Oracle VM VirtualBox. (n.d.). Retrieved from <https://www.virtualbox.org/wiki/Downloads>

Hybrid ISO (Live DVD) Retrieved from <https://gentoo.osuosl.org/releases/amd64/20160704/>