Koneru Lakshmaiah Education Foundation

(Deemed to be University)

DEPARTMENT OF EL&GE

Report on

Linux,Encryption, Self Hosting&Open Source Contributions

SUBMITTED BY:

ID Number	Name
2400100046	Pothineni Viswa Nikhitha

UNDER THE GUIDANCE OF

Mr.Sripath Roy Koganti

Assistant Professor

KL UNIVERSITY

Green fields, Vaddeswaram – 522 502

Guntur Dt., AP, India.

Contents

1 About the Linux Distro Used

Ubuntu 24.04.1 LTS Overview

Ubuntu 24.04.1 LTS (Long Term Support), released in 2024, is one of the most stable and widely used Linux distributions. I installed this operating system on my laptop and explored its features, performance, and usability. Ubuntu is known for its user-friendly interface, reliable updates, and strong community support, making it an excellent choice for students, developers, and professionals.

Installation Experience

The installation process of Ubuntu 24.04.1 was smooth and straightforward. The graphical installer automatically detected my hardware and provided clear options for disk partitioning, time zone selection, and user creation. However, during installation, I did not select the option to configure wireless (Wi-Fi) connectivity. Because of this, after the installation was completed, the Wi-Fi option did not appear in the system settings.

I attempted multiple times to reconfigure the wireless settings, but the Wi-Fi option still did not show up. As an alternative solution, I used USB tethering with my mobile phone, which allowed the system to connect to the internet and download necessary packages.

Performance

In terms of performance, Ubuntu 24.04.1 ran efficiently on my laptop. It uses optimized system resources and offers faster boot times compared to several other operating systems. The built-in software like LibreOffice, Firefox, and system utilities allowed me to start working immediately without the need for additional installations. The Ubuntu Software Center made it easy to install applications such as VS Code, Python, and media players.

Security and Stability

Security and stability remain major strengths of Ubuntu's LTS versions. Regular updates are provided for five years, ensuring long-term support and minimal vulnerabilities. Features like firewall configuration (UFW), secure user permissions, and automatic update notifications helped maintain a safe and dependable environment.

Development Tools Support

Ubuntu also supports strong development tools. I was able to use the terminal for package installation, file management, and running commands efficiently. Tools like the APT package manager, Snap, and Flattop enhanced software availability and flexibility.

Conclusion

Overall, using Ubuntu 24.04.1 LTS provided a reliable and smooth experience. Despite the initial Wi-Fi configuration issue, the OS performed well using USB tethering. Its stability, open-source nature, and powerful features make it a suitable operating system for academic work, programming, and general use. This experience helped me better understand Linux-based systems and improved my confidence in using open-source technologies.

Figure 1.1: Successful Ubuntu Installation

2 Encryption and GPG

I learned about encryption and how it is used to protect data. Encryption is the process of converting readable information (plaintext) into an unreadable form called ciphertext. By doing this, the data become secure and only someone with the correct decryption key can access the original information. I understood that encryption is widely used in secure communication, online transactions, file protection, and data privacy. It helps prevent unauthorized access and ensures the confidentiality and security of information stored or transmitted over networks.

I also explored GPG (GNU Privacy Guard), which is an open-source tool used in Linux for encryption and digital signatures. GPG is based on the OpenPGP standard and works using public-key cryptography. While studying this, I learned that each user has a pair of keys:

- **Public Key:** This is shared with others so that they can encrypt data that is intended for me.
- **Private Key:** This is kept secret and is used by me to decrypt data or digitally sign files.

Using GPG, I was able to understand how to encrypt files, decrypt sensitive information, verify digital signatures, and maintain data integrity. Through this, I learned why GPG is widely used by developers, cybersecurity professionals, and system administrators for secure communication and software verification. With its strong encryption capabilities and open-source nature, GPG helped me understand the importance of privacy and trust in digital environments.

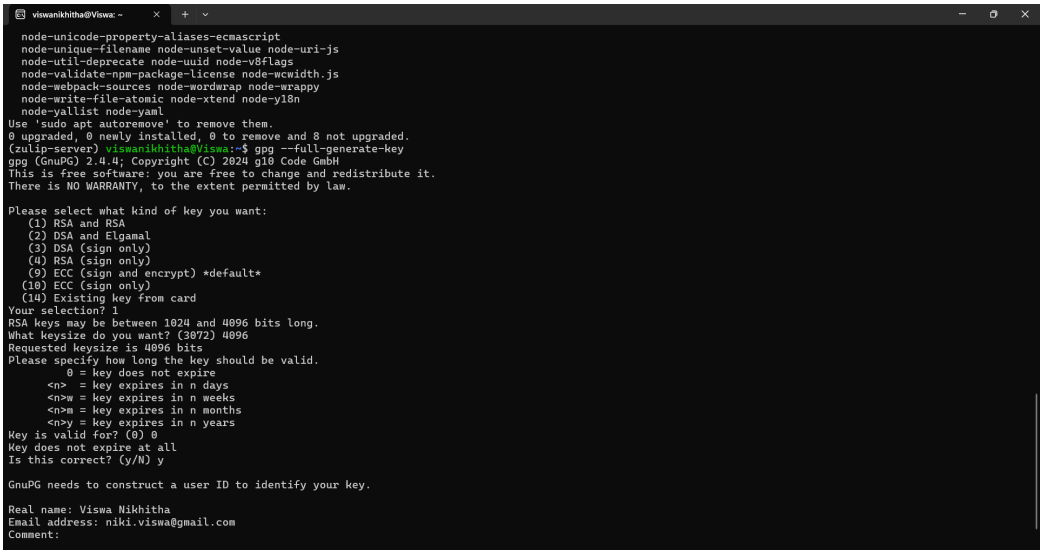

Figure 2.1: Encryption and GPG Illustration

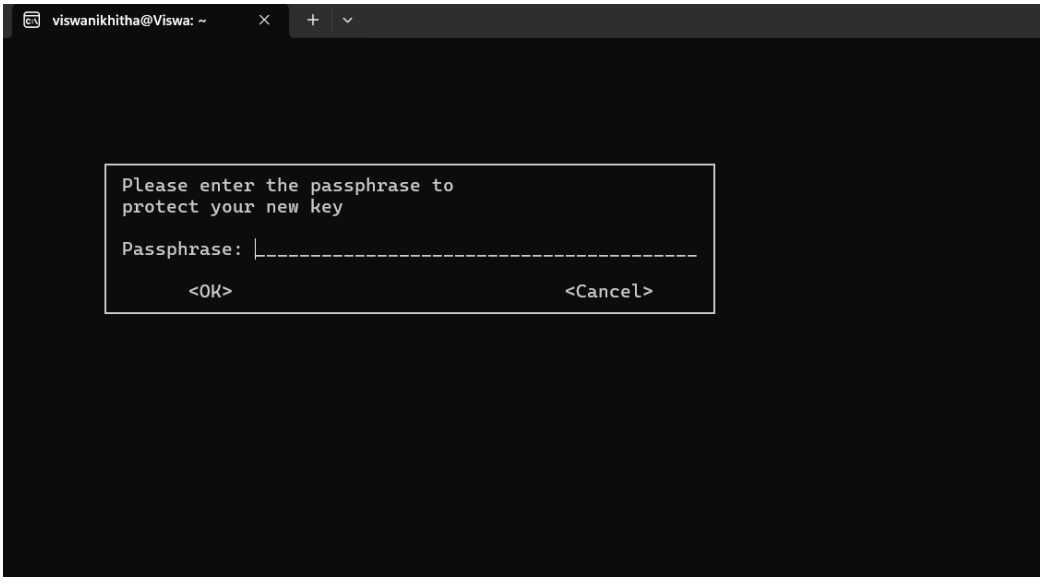

Figure 2.2: Encryption and GPG Illustration

3 Sending Encrypted Email

In this project, I learned how to send encrypted emails to ensure secure communication. Normally, emails are sent in plain text, which means anyone who intercepts them can read the content. To protect sensitive information, I used encryption so that only the intended recipient can read the message.

Encrypted email works using public-key cryptography. To send an encrypted email, I used the public key of the recipient to encrypt my message. Once encrypted, the message becomes unreadable to everyone except the recipient, who can decrypt it using their private key. This method ensures confidentiality even if the email is intercepted during transmission.

I used the GPG (GNU Privacy Guard) tool in Linux to perform this task. With GPG, I imported the recipient's public key, encrypted my email content, and then sent the encrypted message. The recipient could later decrypt it using their private key. Through this process, I understood how encryption provides security and protects the privacy of email communication.

The benefits I observed while sending encrypted email include the following:

- My message remained secure and unreadable to unauthorized users.
- Sensitive information could be shared safely.
- The communication offered privacy and data protection.
- Digital signatures could be used to verify sender identity.

Overall, this task helped me understand the importance of secure communication and how encryption plays a key role in protecting email content.

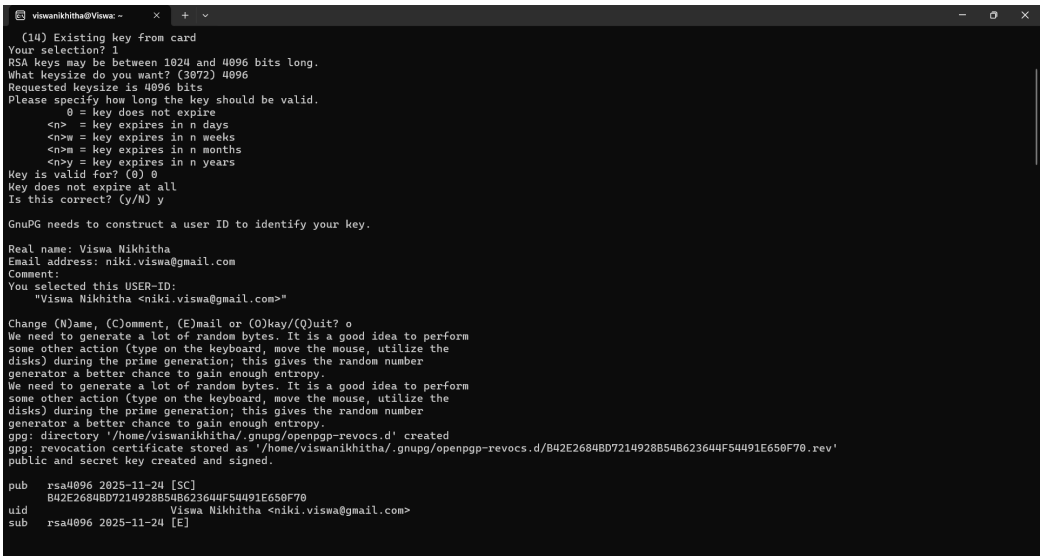

Figure 3.1: Sending Encrypted Email

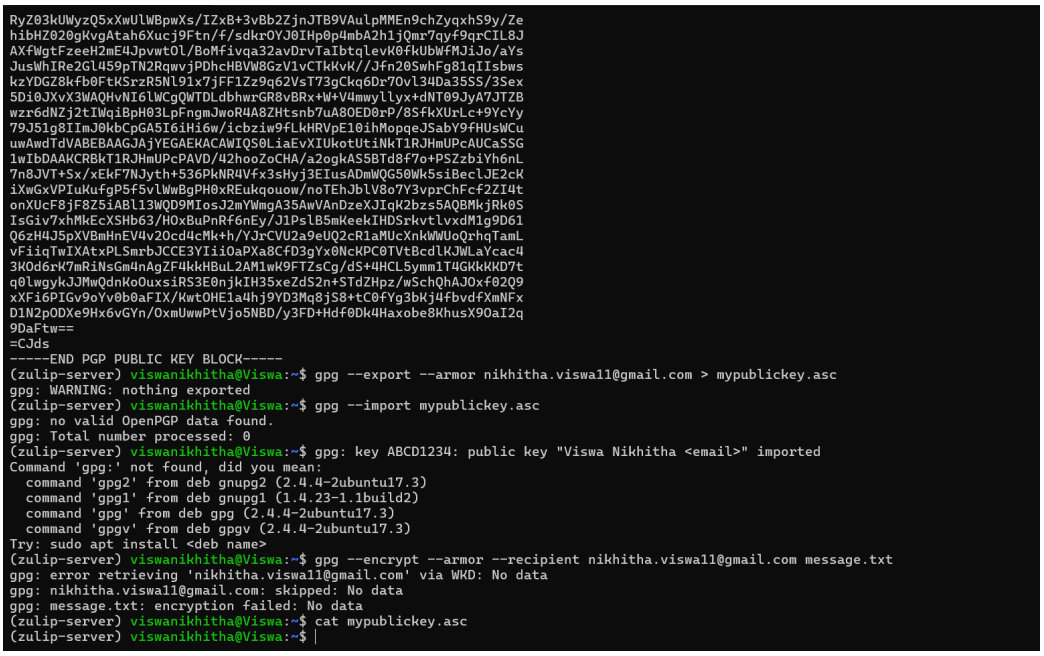

Figure 3.2: Sending Encrypted Email

4 Privacy Tools from PRISM-Break

PRISM-Break is a community-driven website that recommends free, open-source, and privacy-respecting software alternatives. These tools help users protect their personal data and avoid corporate or government surveillance. Below are five commonly recommended privacy tools from PRISM-Break.

4.1 Signal

Signal is an end-to-end encrypted messaging application that provides secure text messaging, voice calls, and video calls. It ensures that conversations are protected from third-party access, including service providers.

Figure 4.1: Signal

4.2 Tor Browser

Tor Browser is designed to protect user anonymity by routing internet traffic through a network of volunteer-operated servers. It hides the user's IP address and helps prevent tracking, fingerprinting, and censorship.

Figure 4.2: Tor Browser

4.3 ProtonMail

ProtonMail is an encrypted email service based in Switzerland. It uses strong end-to-end encryption, ensuring that only the sender and the recipient can read the email content.

Figure 4.3: ProtonMail

4.4 KeePassXC

KeePassXC is an open-source password manager that securely stores passwords in an encrypted database. It helps users maintain strong, unique passwords without remembering all of them manually.

Figure 4.4: KeePassXC

4.5 VeraCrypt

VeraCrypt is a disk encryption software used to protect sensitive files and folders. It allows users to encrypt entire drives or create secure encrypted containers for storing confidential data.

Figure 4.5: VeraCrypt

5 Open Source License Used

In my project **Daily Task Manager** uses the **MIT License**. The MIT License is one of the most widely used open-source licenses, known for its simplicity, flexibility, and permissive nature. It allows users to freely use, modify, distribute, and integrate the software into other projects, even commercial ones, while requiring only proper attribution to the original author.

Reason for Choosing the MIT License

- **Permissive usage:** The MIT License allows anyone to freely use the code without restrictions, making the project more accessible.
- **Allows modification:** Users can modify or extend the Daily Task Manager project to fit their needs, encouraging improvement and collaboration.
- **Supports open-source learning:** The license enables students, beginners, and developers to study the code and learn from it without legal limitations.
- **Lightweight and simple:** MIT License contains minimal legal wording, making it easy to include and understand.
- **Promotes wider adoption:** Since it is business-friendly and compatible with many other licenses, more developers can adopt or contribute to the project.

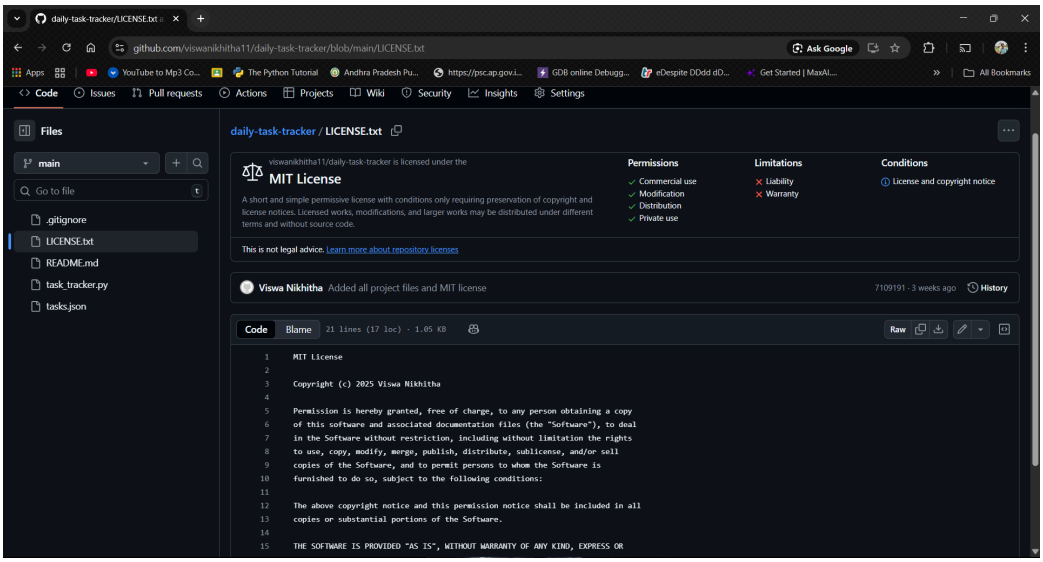

Figure 5.1: MIT License

6 Self-Hosted Server

About the Server

In this project, I self-hosted the application **Draw.io (diagrams.net)** on my system. Draw.io is an open-source diagramming tool used to create flowcharts, network diagrams, UML diagrams, organizational charts, and many other visual designs. The advantage of self-hosting Draw.io is that it works completely offline, providing enhanced privacy, high security, and full control over the diagrams created. Since no data is uploaded to the cloud, it is extremely suitable for academic work and sensitive project documentation.

Installation Steps

The steps followed to self-host Draw.io are:

1. **Install Docker** If Docker is not installed:

```
sudo apt update
sudo apt install docker.io -y
sudo systemctl enable --now docker
```

   Check installation:

```
docker --version
```

2. **Pull the official Draw.io image**

```
sudo docker pull jgraph/drawio
```

3. **Run Draw.io container** Run it on port 8080:

```
sudo docker run -d --name drawio \
    -p 8080:8080 \
    jgraph/drawio
```

4. **Run the Application Locally**

```
http://localhost:8080
```

After this setup, Draw.io is successfully self-hosted.

Localized (Translated) Document

Below is the translated version of the "About the Server" section in **Telugu**.

ఈ ప్రాజెక్ట్‌లో నేను Draw.io (diagrams.net) ను నా సిస్టమ్‌లో స్వయంగా హోస్ట్ చేసాను. Draw.io అనేది ఫ్లోచార్ట్‌లు, నెట్‌వర్క్ డయాగ్రామ్‌లు, డయాగ్రామ్‌లు మరియు ఇతర సాంకేతిక గ్రాఫ్‌లు రూపొందించడానికి ఉపయోగించే ఓపెన్ సోర్స్ టూల్.

దీనిని స్వయంగా హోస్ట్ చేయడం వలన ఇది పూర్తిగా ఆఫ్‌లైన్‌లో పనిచేస్తుంది. అందువల్ల డేటా పూర్తిగా స్వంతం అవుతుంది, సురక్షితంగా ఉంటుంది మరియు ప్రాజెక్ట్ డాక్యుమెంటేషన్‌కు చాలా అనుకూలంగా ఉంటుంది.

Poster

Figure 6.1: Self-Hosted Draw.io Poster

7 Open Source Contributions

As part of my open-source learning and participation in Hacktoberfest, I contributed to multiple GitHub repositories by solving coding problems, improving the structure of the project, and submitting pull requests (PR). Below are the details of the contributions I made, including the issues solved and the pull requests raised.

List of Pull Requests Submitted

1. **Create LongestValidParentheses.c**

   I implemented the solution for the "Longest Valid Parentheses" problem using stack-based and DP logic. The PR included clean code, comments, and sample test cases.

2. **Create MergekSortedLists.c**

   I solved the classic linked list problem "Merge K Sorted Lists". My contribution optimized the merging process using a min-heap approach.

3. **Create GroupAnagrams.c**

   I wrote a solution to group a list of strings into anagrams using hashing (frequency map as key). This PR improved the repository's algorithm collection.

4. **Create LetterCombinationsOfPhoneNumber.c**

   Implemented backtracking-based solution for generating all possible letter combinations from digit input.

5. **Create ContainerWithMostWater.c**

   Added the two-point solution optimized for the “problem of ”Container With the most” water”, improving performance and readability.

Screenshots of Contributions

Below are the screenshots of issues solved, pull requests raised, and merged status.

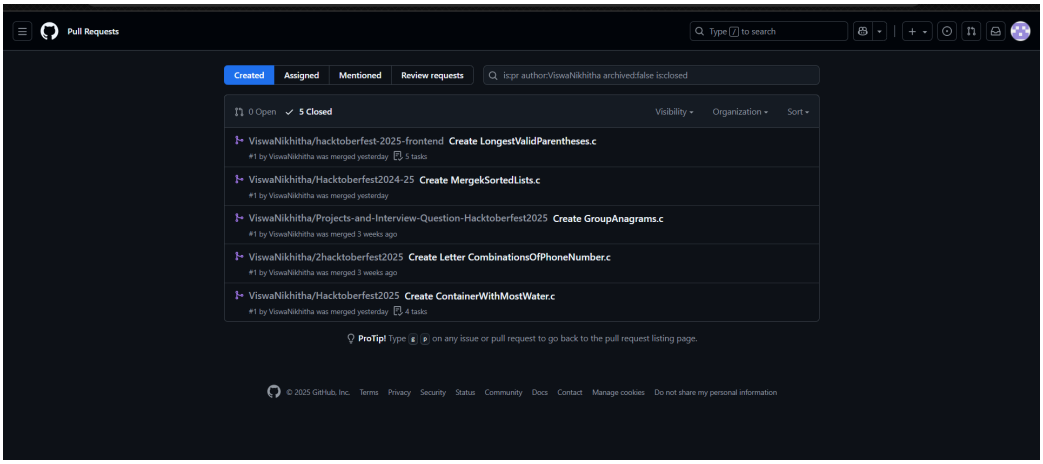

Figure 7.1: Pull Requests Submitted and Merged

8 LinkedIn Activity

As part of documenting my learning and contributions, I actively shared my progress on LinkedIn. Below are three LinkedIn posts reflecting my work on self-hosting, open-source contributions, and technical blogging.

1. LinkedIn Post – Self Hosting Project

I shared my experience of successfully self-hosting **Draw.io (diagrams.net)** on my local server.

Post Link:

{https://www.linkedin.com/posts/pothineni-viswa-nikhitha-702bb2390_drawio-diagramsnet-ubuntu-activity-7390417879915081728-60Y0?utm_source=social_share_send&utm_medium=member_desktop_web&rcm=ACoAAGBJEHIBKdJzYumIKTVJWx4oSQ_RWnj_ztY}

2. LinkedIn Post – Open Source PR Merge

In this post, I highlighted my **Hacktoberfest pull requests** that were merged successfully. I discussed the issues I solved, algorithm implementations, and how contributing to open source helped me grow technically.

Post Link:

https://www.linkedin.com/posts/pothineni-viswa-nikhitha-702bb2390_hacktoberfest-opensource-coding-activity-7398776479150530560-HaP4?utm_source=social_share_send&utm_medium=android_app&rcm=ACoAAFsK5IQBfMLSDGiCrTXVnuQvd9WrseHskFc&utm_campaign=copy_link

3. LinkedIn Post – Blog Writing

I published a technical blog explaining a topic I learned recently (e.g., Encryption, GPG, Self-Hosting, or Problem-Solving). The LinkedIn post covered the purpose of the blog and shared insights from my learning.

Post Link: https://www.linkedin.com/pulse/unlocking-open-source-world-my-linux-self-hosting-journey-nikhitha-sfffc

Conclusion

This report summarizes my exploration and hands-on experience with Linux, encryption, privacy tools, open-source licensing, server hosting, and real-world open-source contributions. Throughout this journey, I gained a deeper understanding of how Linux empowers users with flexibility, control, and transparency. Learning about encryption and GPG strengthened my awareness of digital security, helping me understand how sensitive information can be protected through secure communication practices.

Working with privacy tools and open-source licenses gave me clarity on responsible software usage, distribution rights, and ethical contributions to the community. Configuring a server and experimenting with self-hosting allowed me to apply theoretical concepts in a practical environment, improving my technical confidence and problem-solving skills.

Most importantly, contributing to open-source projects through pull requests connected me with real-world developer workflows. It helped me understand collaboration, version control, issue tracking, and the importance of writing clean, maintainable code.

Overall, this journey has enhanced my technical knowledge, improved my hands-on skills, and motivated me to continue exploring the open-source world. It has shaped me into a more confident learner, problem-solver, and contributor, preparing me for future academic and professional challenges.