INSTITUTE FOR ADVANCED COMPUTING
AND
SOFTWARE DEVELOPMENT
AKURDI, PUNE

DOCUMENTATION ON

"Secure E-mail Server Deployment With Malware Protection"

PG-DITISS March-2023

SUBMITTED BY:

GROUP NO: 06

KHAN WASEEM (233414)

ONKAR RASKAR (233428)

MR. KARTIK AWARI
PROJECT GUIDE

MR. ROHIT PURANIK
CENTRE CO-ORDINATOR

ABSTRACT

The project, "Secure Email Server Deployment with Malware Protection," tackles the critical concern of email communication security. By implementing an advanced email server with robust encryption and proactive malware detection mechanisms, such as real-time scanning and heuristic analysis, the project aims to bolster protection against threats lurking within email attachments and links. Through a comprehensive analysis of existing email server architectures and security protocols, the project tailors a deployment strategy that seamlessly integrates these security measures. By leveraging a combination of open-source and commercial security tools, the email server's malware protection capabilities are fortified. The projected outcome is an email server infrastructure that significantly curtails malware risks, thereby promoting a more secure communication environment. Through this initiative, the project contributes to the broader goal of enhancing digital security amidst evolving cyber threats.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

Sr. No.	Abbreviation	Full-Form
1.	SPF	Sender Policy Framework
2.	DKIM	Domain Keys Identified Mail
3.	DMARC	Domain-Based Message Authentication Reporting & Conformance
4.	SHA	Secure Hash Algorithm
5.	VPC	Virtual Private Cloud
6.	VM	Virtual Machine

LIST OF FIGURES

1. INTRODUCTION

In today's digital world, emails are a big part of how we communicate. But these emails can be vulnerable to bad things like viruses and other types of harmful software. That's where the project "Secure Email Server Deployment with Malware Protection" comes in.

Imagine if we could make email safer by building a special kind of email server. This server would use strong locks to keep the information inside emails safe and private. It would also have smart tools that can check emails in real-time to find any bad stuff hiding in attachments or links.

To make this happen, the project looks at how email servers work now and figures out ways to make them better. It combines different tools to make the email server really good at stopping harmful things from getting in.

The goal is simple: to create an email system that's like a fortified castle against viruses and other online dangers. This way, when we send or receive emails, we can be more confident that our information and personal data are safe from harm. By doing this, the project wants to help make the digital world a safer place for everyone.

1.1 Problem Statement

In today's digital world, sending emails is common, but these emails can carry viruses and other harmful things that can damage our computers and steal our information. Many email systems don't have strong protection against these dangers. This project aims to solve this problem by creating a special email system for organizations that keeps out viruses and harmful stuff, making our emails safer and our information more secure

2. LITERATURE SURVEY

Before starting our project to make emails safer from viruses and harmful things, we looked at what other people have already done in this area. We read about different ways people have tried to protect emails and the technology they used. Some researchers suggested using strong locks to keep emails safe, while others talked about using smart tools to scan emails and find bad things.

We also learned about different types of viruses and how they can sneak into emails. Some viruses pretend to be harmless files, but when you open them, they can cause a lot of trouble. By reading these studies, we got a better idea of what works and what doesn't when it comes to making emails secure.

This helped us plan our project better because we could see what has already been tried and what new ideas, we could bring in to make emails even safer. By looking at what other researchers have done, we can make sure our project is unique and effective in keeping our emails protected.

3. METHODOLOGY

3.1 SYSTEM ARCHITECTURE

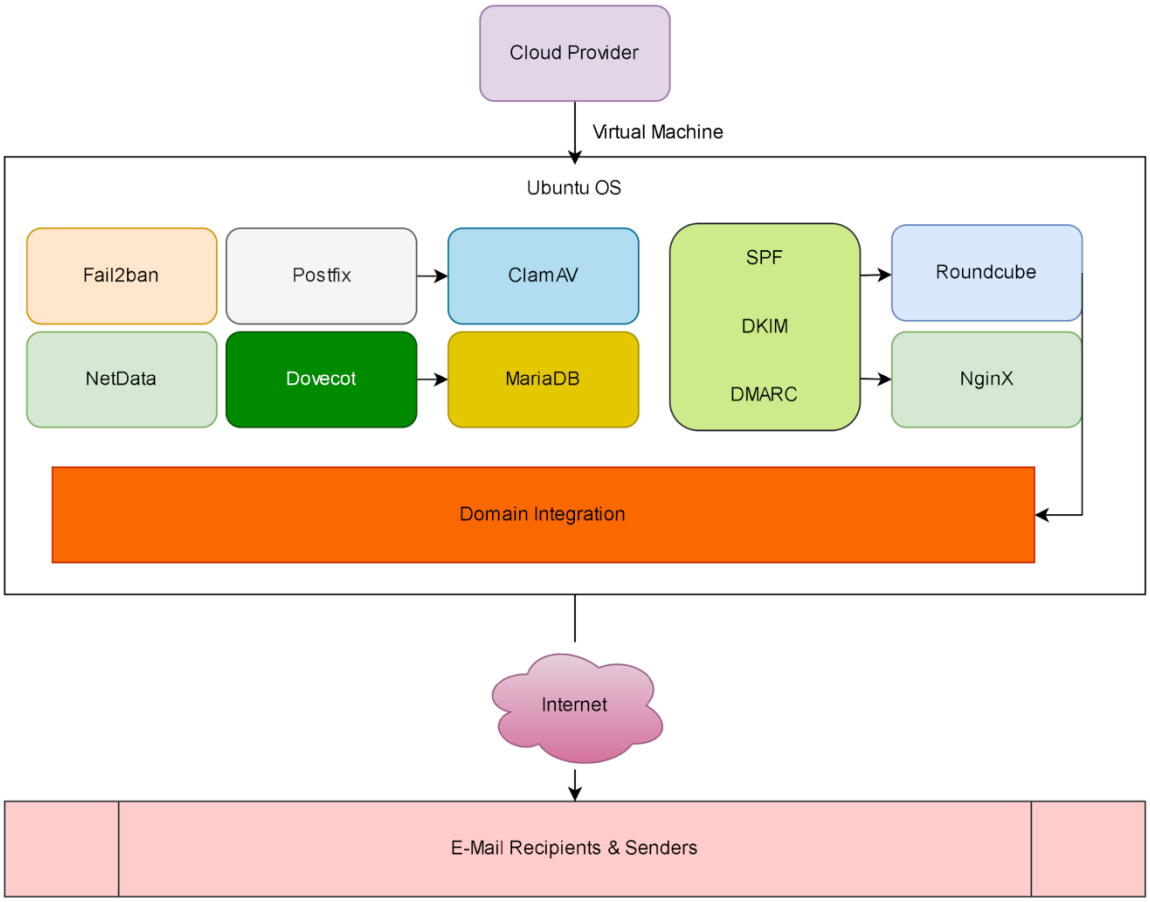


Figure 1: Architecture Diagram

- At the heart of our project lies a comprehensive architecture designed to fortify email communication. This diagram showcases how various components collaborate to create a secure and seamless email ecosystem.
- The architecture entails a secure email server setup within a virtual machine, running Ubuntu OS. Fail2ban guards against unauthorized access, while Netdata monitors system health. Postfix and Dovecot manage email sending and receipt, supported by SPF and DKIM for authentication. ClamAV scans emails for viruses, with MariaDB storing email data securely.
- Roundcube offers a user-friendly web interface for email access, while Nginx ensures secure web communication. Domain Integration ensures smooth domain linkage. The Cloud Provider hosts the virtual machine, connecting to the Internet for email exchange between recipients and senders.
- In summary, the architecture employs a secure, multi-layered approach to facilitate safe email communication, integrating encryption, authentication, malware detection, and user-friendly access, all within a robust virtualized environment.

3.2 SOFTWARE REQUIREMENTS

The software requirements for this project outline the essential features and functionalities necessary to create a robust and secure email server system.

a. **Operating system:** Ubuntu v.22.04.3 LTS
b. **Webmail:** Roundcube
c. **Webserver:** Nginx
d. **Admin Panel:** iRedMail
e. **Database:** MariaDB
f. **Monitoring:** NetData

3.3 HARDWARE REQUIREMENTS

a. **CPU:** 1 Core
b. **RAM:** 2 GB
c. **Storage:** 15

4. WORKING

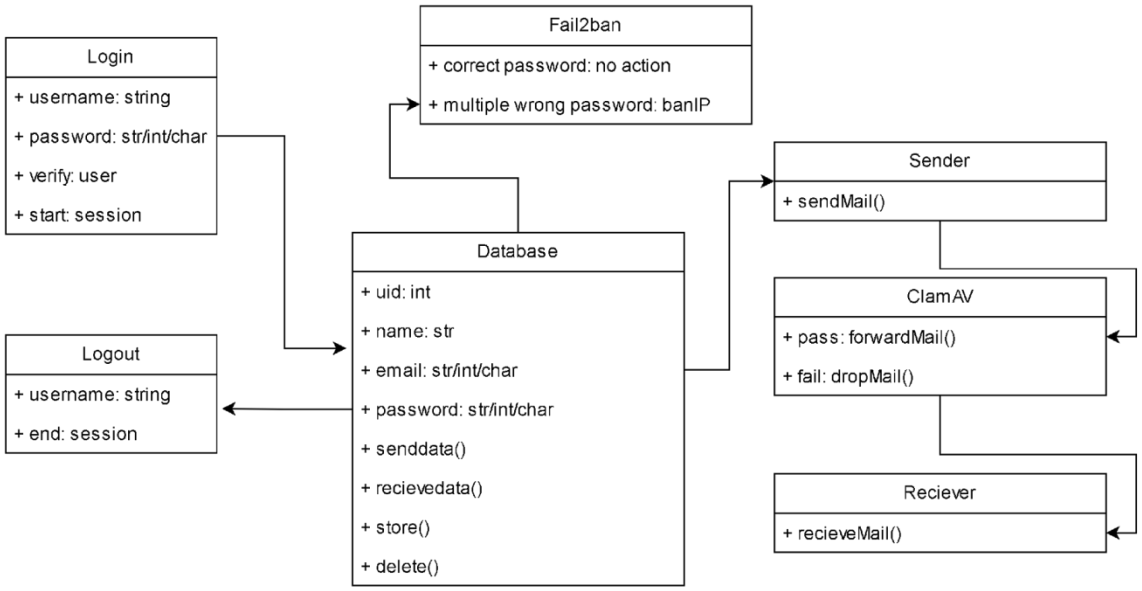


Figure 2: Working of Secure E-mail Server

- This project's work revolves around establishing a secure and efficient email server system that safeguards communication while ensuring ease of use. When a user sends an email, the system's Postfix component manages the email's sending process, while Dovecot organizes the received emails. These components collaborate with SPF and DKIM, verifying the email's authenticity and protecting against potential phishing attacks.

- The emails undergo a thorough security check through ClamAV, which scans for viruses and malicious content, preventing harmful attachments from reaching the recipient. The MariaDB database maintains essential email information, ensuring reliable data storage and retrieval.

- For user interaction, Roundcube offers a user-friendly web interface, enabling easy email access and management. Nginx ensures secure web communication, guarding against cyber threats.

- The architecture integrates Fail2ban, a vigilant security guard that identifies and blocks suspicious activities. Netdata acts as a watchful observer, monitoring system health and performance.

- Overall, this project streamlines email communication: it secures transmission through encryption, validates authenticity via SPF and DKIM, employs ClamAV for malware detection, and offers a user-friendly interface with Roundcube. This comprehensive approach addresses various facets of email security, ensuring reliable communication within a protected environment

5. FLOWCHART

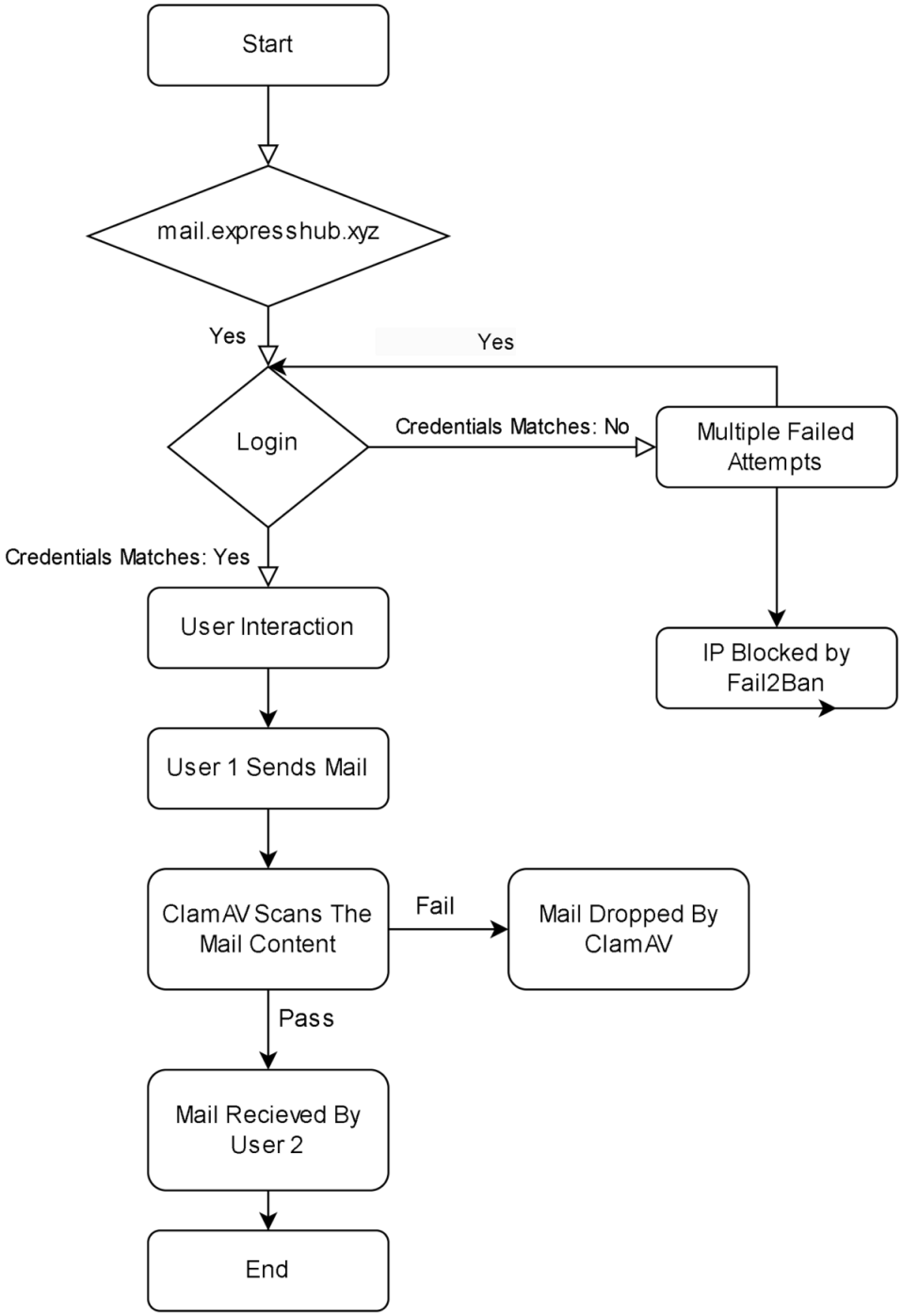
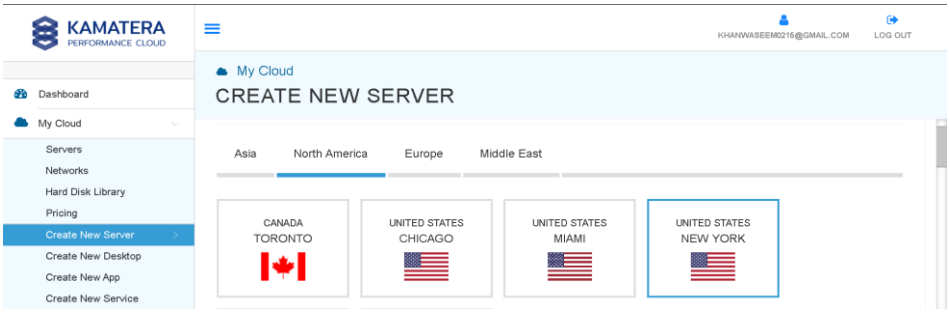


Figure 3: Flowchart Of E-mail Server

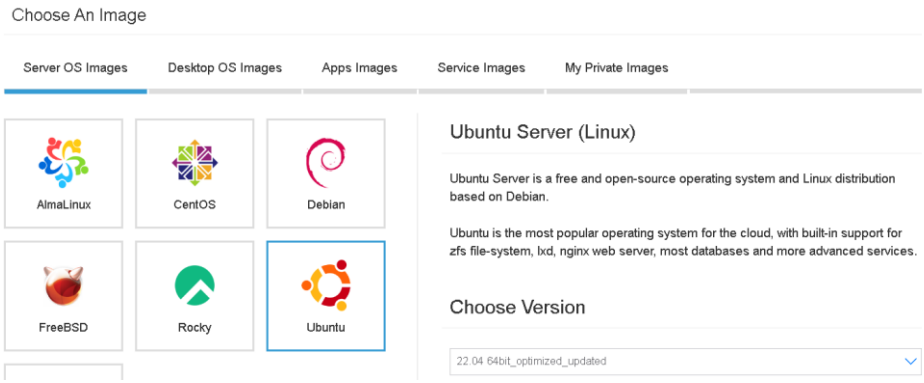
6. IMPLEMENTATION

The following are the steps for: **Server Setup on "Kamatera Cloud Platform":**

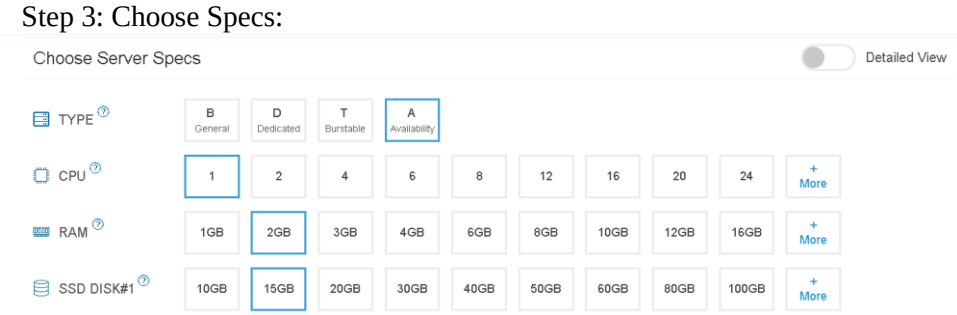
Step 1: Choose Zone:



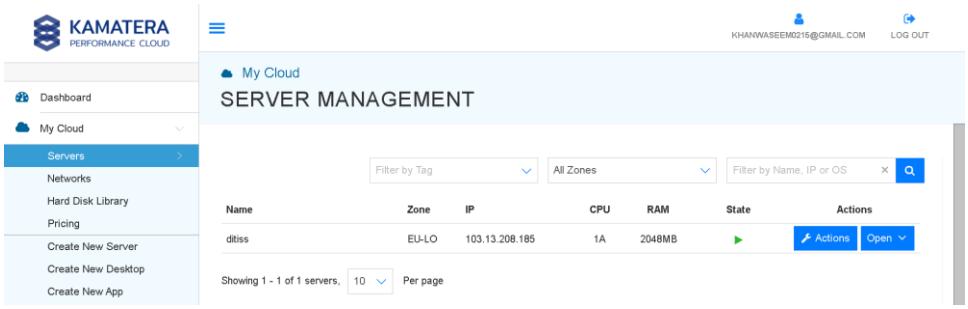
Step 2: Choosing Image:



Step 3: Choose Specs:



Step 4: Server is now Ready:

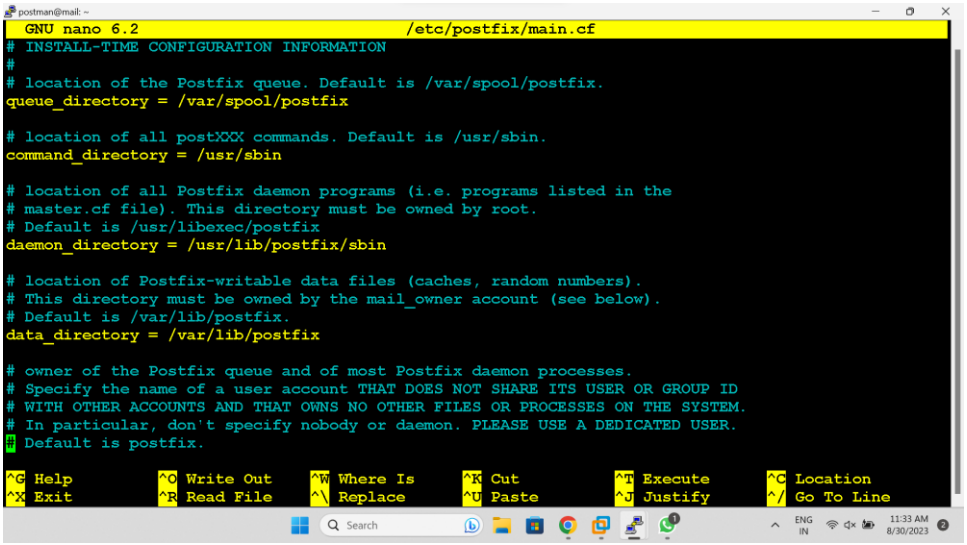


Steps for: **Setting Up Postfix:**

Step 1: Update Ubuntu Repo then installing Postfix:

```
sudo apt-get update

sudo apt-get install postfix
```

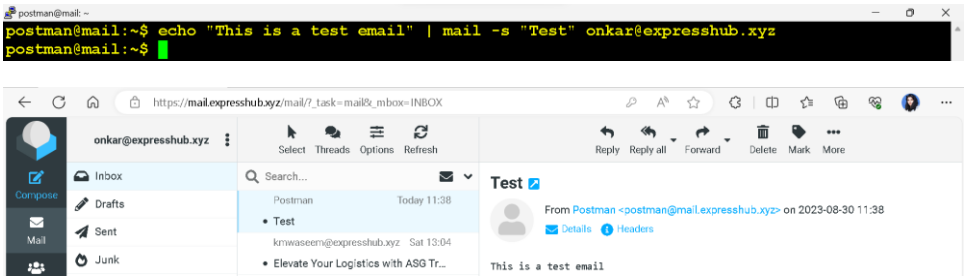
Step 2: Configuring the Postfix:



Step 3: Restarting the Postfix:

```
sudo systemctl start postfix

sudo systemctl enable postfix
```

Step 4: Testing the Postfix:

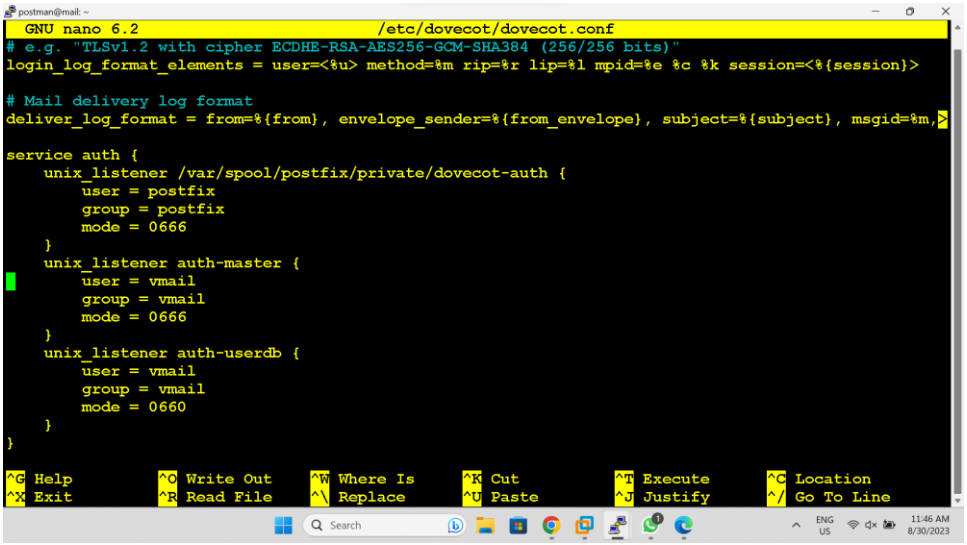


Steps for: **Setting Up Dovecot:**

Step 1: Update Ubuntu Repo then installing Dovecot:

```
sudo apt-get update

sudo apt-get install dovecot-imapd dovecot-pop3d
```

Step 2: Configuring the Dovecot:

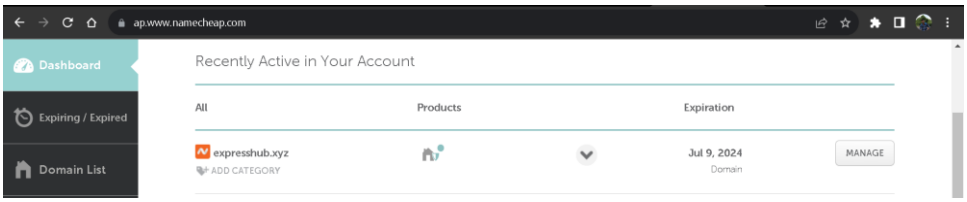


Step 3: Restarting the Dovecot:

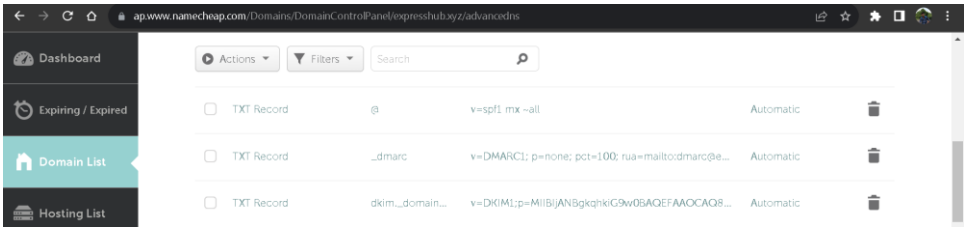
```
sudo systemctl start dovecot
sudo systemctl enable dovecot
```

Steps for **Integrating SPF and DKIM:**

Step 1: Log in to domain's DNS management panel:



Step 2: Set Up SPF (Sender Policy Framework), DKIM (DomainKeys Identified Mail) and DMARC (Domain-based Message Authentication, Reporting and Conformance):



Steps for: **Installing Let's Encrypt TLS Certificate**

Step 1: Obtaining the Certificate:

```
sudo apt install certbot python3-certbot-nginx -y

sudo certbot certonly --webroot --agree-tos --email
postman@expresshub.xyz -d mail.expresshub.xyz -w
/var/www/html/
```

Step 2: Installing the certificate in NginX:

```
postman@mail: ~
  GNU nano 6.2                    /etc/nginx/templates/ssl.tmpl

# Greatly improve the performance of keep-alive connections over SSL.
# With this enabled, client is not necessary to do a full SSL-handshake for
# every request, thus saving time and cpu-resources.
ssl_session_cache shared:SSL:10m;

# To use your own ssl cert (e.g. "Let's Encrypt"), please create symbol link to
# ssl cert/key used below, so that we can manage this config file with Ansible.
#
# For example:
#
# rm -f /etc/ssl/private/iRedMail.key
# rm -f /etc/ssl/certs/iRedMail.crt
# ln -s /etc/letsencrypt/live/<domain>/privkey.pem /etc/ssl/private/iRedMail.key
# ln -s /etc/letsencrypt/live/<domain>/fullchain.pem /etc/ssl/certs/iRedMail.crt
#
# To request free "Let's Encrypt" cert, please check our tutorial:
# https://docs.iredmail.org/letsencrypt.html
ssl_certificate /etc/letsencrypt/live/mail.expresshub.xyz/fullchain.pem;
ssl_certificate_key /etc/letsencrypt/live/mail.expresshub.xyz/privkey.pem;
#ssl_certificate /etc/ssl/certs/iRedMail.crt;
#ssl_certificate_key /etc/ssl/private/iRedMail.key;
```

Step 3: Installing the certificate in Postfix and Dovecot:

Postfix:

```
postman@mail: ~
  GNU nano 6.2                    /etc/postfix/main.cf
#
smtpd_tls_key_file = /etc/letsencrypt/live/mail.expresshub.xyz/privkey.pem
smtpd_tls_cert_file = /etc/letsencrypt/live/mail.expresshub.xyz/cert.pem
smtpd_tls_CAfile = /etc/letsencrypt/live/mail.expresshub.xyz/chain.pem
```

Dovecot:

```
postman@mail: ~
  GNU nano 6.2                    /etc/dovecot/dovecot.conf
verbose_ssl = no
#ssl_ca = </path/to/ca
ssl_cert = </etc/letsencrypt/live/mail.expresshub.xyz/fullchain.pem
ssl_key = </etc/letsencrypt/live/mail.expresshub.xyz/privkey.pem
```

Step 4: Reloading NginX, Postfix and Dovecot:

```
sudo systemctl reload nginx

sudo systemctl reload postfix

sudo systemctl reload dovecot
```

Steps for: **Installing and Setup of Fail2Ban:**

Step 1: Update Repo and Install Fail2ban:

```
sudo apt-get update

sudo apt-get install fail2ban
```

Step 2: Configuring the Fail2ban:

```
postman@mail: ~
  GNU nano 6.2                    /etc/fail2ban/jail.local
[DEFAULT]
# time is in seconds. 3600 = 1 hour, 86400 = 24 hours (1 day)
findtime    = 3600
bantime     = 3600
maxretry    = 5
ignoreip    = 127.0.0.1 127.0.0.0/8 10.0.0.0/8 172.16.0.0/12 192.168.0.0/16
```

Step 3: Restarting the Fail2ban:

```
sudo systemctl enable fail2ban

sudo systemctl start fail2ban
```

Steps for: **Installing and Setup of MariaDB:**

Step 1: Update Repo and Install MariDB:

```
sudo apt update

sudo apt install mariadb-server
```

Step 2: All Databases of MySQL:

```
MariaDB [(none)]> SHOW databases
    -> ;
+--------------------+
| Database           |
+--------------------+
| amavisd            |
| fail2ban           |
| information_schema |
| iredadmin          |
| iredapd            |
| mysql              |
| opendmarc          |
| performance_schema |
| roundcubemail      |
| sys                |
| vmail              |
+--------------------+
```

Step 3: Tables of database vmail in MySQL:

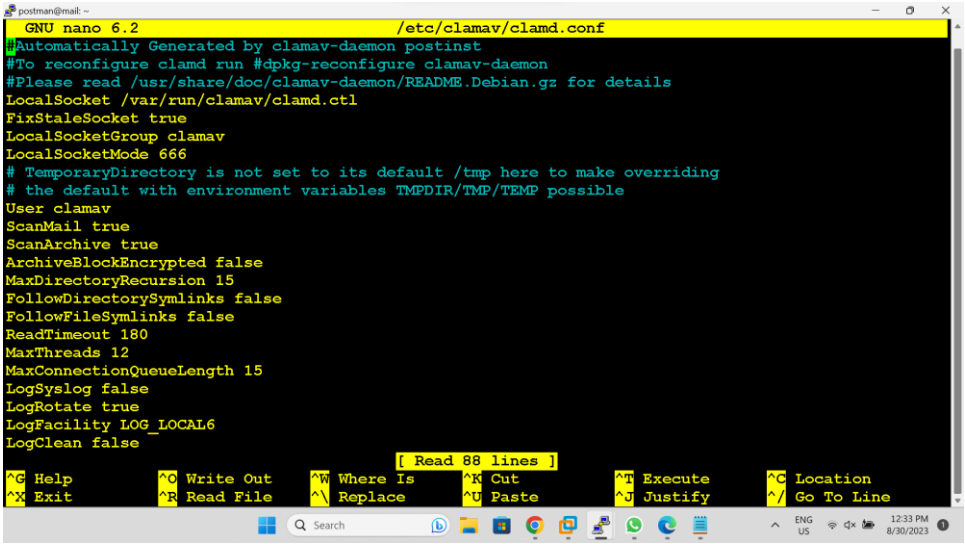
```
postman@mail: ~
+----------------------+
| Tables_in_vmail      |
+----------------------+
| admin                |
| alias                |
| alias_domain         |
| anyone_shares        |
| deleted_mailboxes    |
| domain               |
| domain_admins        |
| forwardings          |
| last_login           |
| mailbox              |
| maillist_owners      |
| maillists            |
| moderators           |
| recipient_bcc_domain |
| recipient_bcc_user   |
| sender_bcc_domain    |
| sender_bcc_user      |
| sender_relayhost     |
| share_folder         |
| used_quota           |
+----------------------+
20 rows in set (0.000 sec)
```

Steps for: **Installing and Setup of ClamAV:**

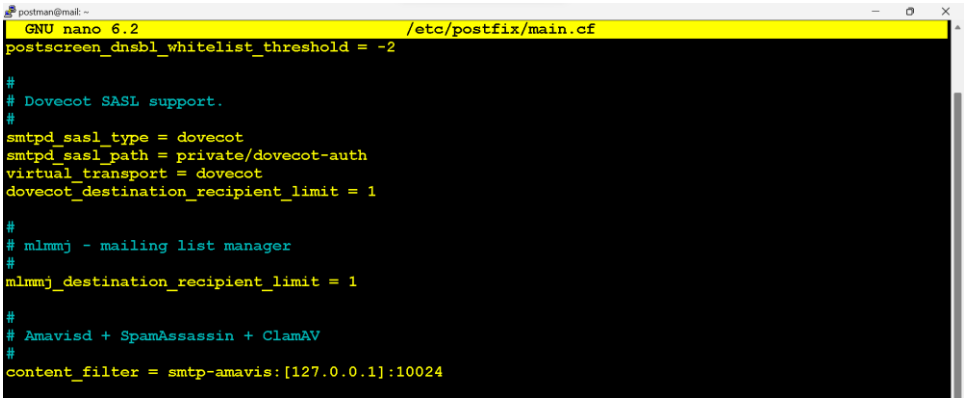
Step 1: Update Repo and Install ClamAV:

```
sudo apt-get update
sudo apt-get install clamav clamav-daemon clamav-freshclam
```

Step 2: Configuring the ClamAV:

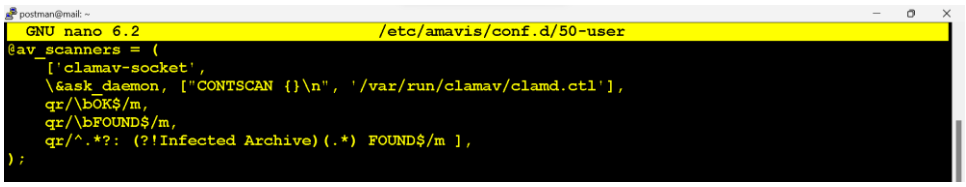


Step 3: Integrating with Postfix:



Step 4: Install and configure Amavis – a content filter for Postfix:

```
sudo apt-get install amavisd-new
```



Step 5: Restart the services:

```
sudo service clamav-daemon restart

sudo service amavis restart

sudo service postfix restart
```

Steps for: **Installing and Setup of Roundcube:**

Step 1: Install Apache/Nginx, PHP, and Dovecot:

```
sudo apt update

sudo apt install apache2

sudo apt install php libapache2-mod-php php-mysql php-curl
php-json php-mbstring php-intl php-xml php-zip
```

Step 2: Install and configure Roundcube:

```
# Download Roundcube
wget
https://github.com/roundcube/roundcubemail/releases/download/1.5.4
/roundcubemail-1.5.4-complete.tar.gz

# Extract and move Roundcube
tar -xzvf roundcubemail-1.5.4-complete.tar.gz
sudo mv roundcubemail-1.5.4 /var/www/html/roundcube

# Rename the sample config
sudo mv /var/www/html/roundcube/config/config.inc.php.sample
/var/www/html/roundcube/config/config.inc.php

# Edit the Roundcube configuration
sudo nano /var/www/html/roundcube/config/config.inc.php
```

Step 3: Restart the services:

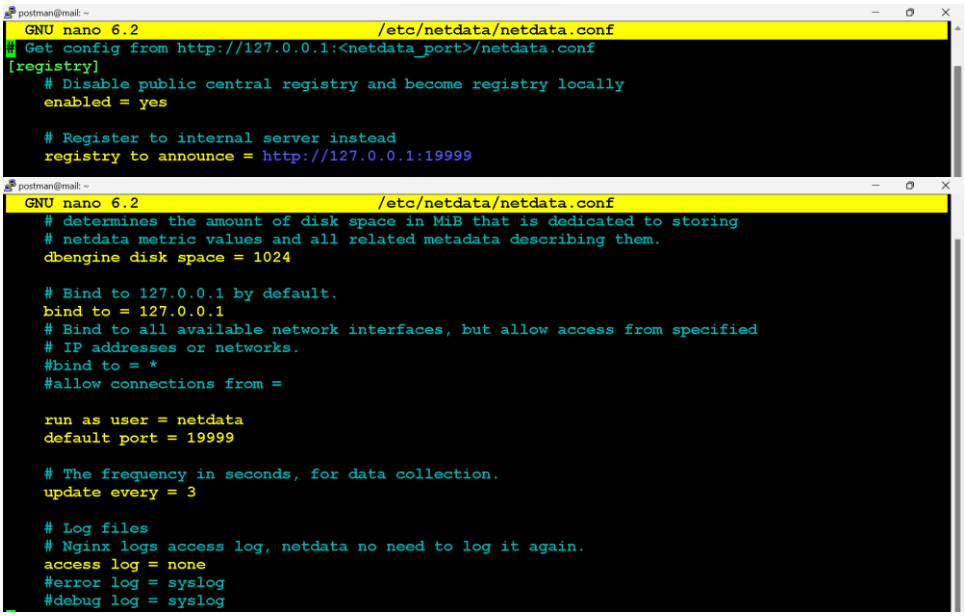
```
# Restart Apache
sudo systemctl restart apache2

# Restart Dovecot
sudo systemctl restart dovecot
```

Steps for: **Installing and Setup of NetData:**

Step 1: Update the system repo and install netdata:

```
sudo apt-get update
sudo apt-get install netdata
```

Step 2: Configuring the netdata:



Step 2: Restart the netdata:

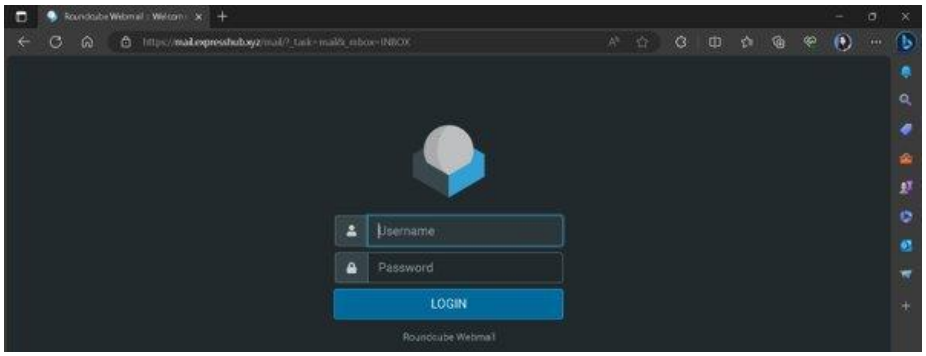
```
sudo systemctl restart netdata
```

Security and Firewall:

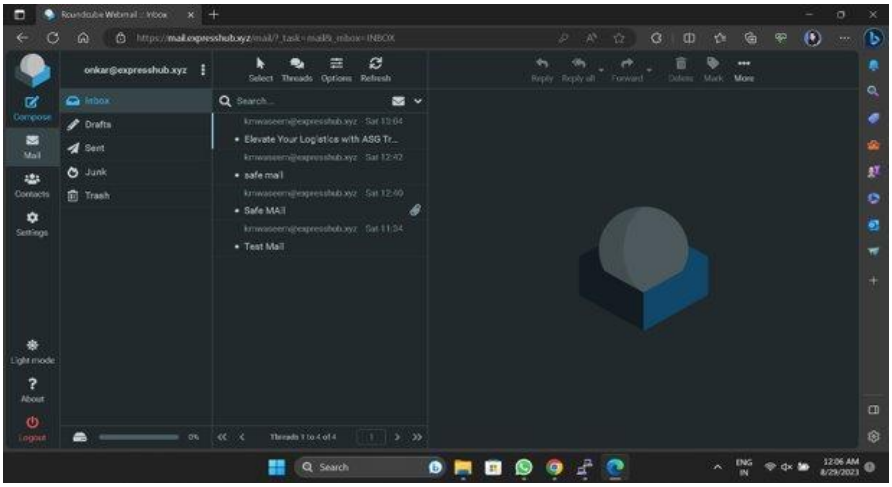
```
postman@mail: ~
  GNU nano 6.2                    firewall.sh *
#!/bin/bash

# Flush existing rules and set default policies
iptables -F
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT ACCEPT

# Allow established and related connections
iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT

# Allow localhost
iptables -A INPUT -i lo -j ACCEPT

# Allow SSH access
iptables -A INPUT -p tcp --dport 22 -j ACCEPT

# Allow DNS queries
iptables -A INPUT -p udp --dport 53 -j ACCEPT
iptables -A INPUT -p tcp --dport 53 -j ACCEPT

# Allow SMTP and Submission
iptables -A INPUT -p tcp --dport 25 -j ACCEPT
iptables -A INPUT -p tcp --dport 587 -j ACCEPT

# Allow IMAP
iptables -A INPUT -p tcp --dport 143 -j ACCEPT
iptables -A INPUT -p tcp --dport 993 -j ACCEPT

# Allow HTTP and HTTPS
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -A INPUT -p tcp --dport 443 -j ACCEPT

# Allow MariaDB
iptables -A INPUT -p tcp --dport 3306 -j ACCEPT

# Allow Netdata Monitoring
iptables -A INPUT -p tcp --dport 19999 -j ACCEPT

# Allow outbound traffic
iptables -A OUTPUT -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT

# Save the rules
service iptables save
service iptables restart

^G Help      ^O Write Out  ^W Where Is   ^K Cut      ^T Execute   ^C Location
^X Exit      ^R Read File  ^\ Replace    ^U Paste    ^J Justify   ^/ Go To Line
```

Steps for: **Testing:**

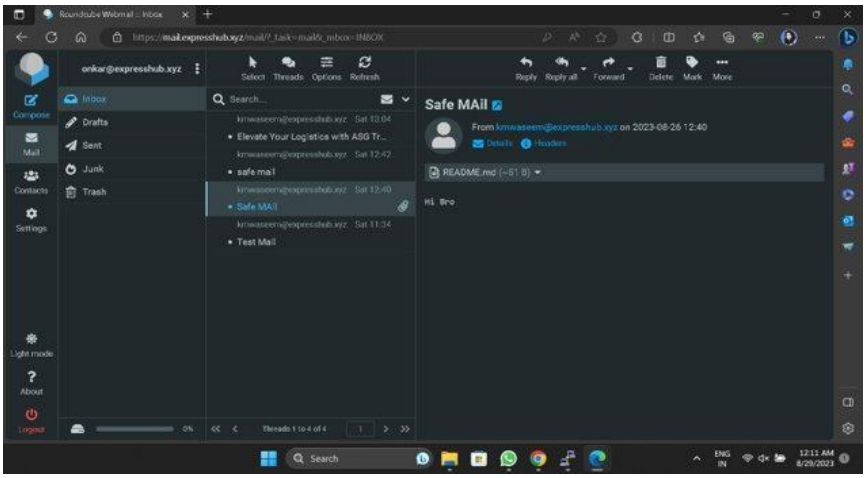
Step 1: Web Panel



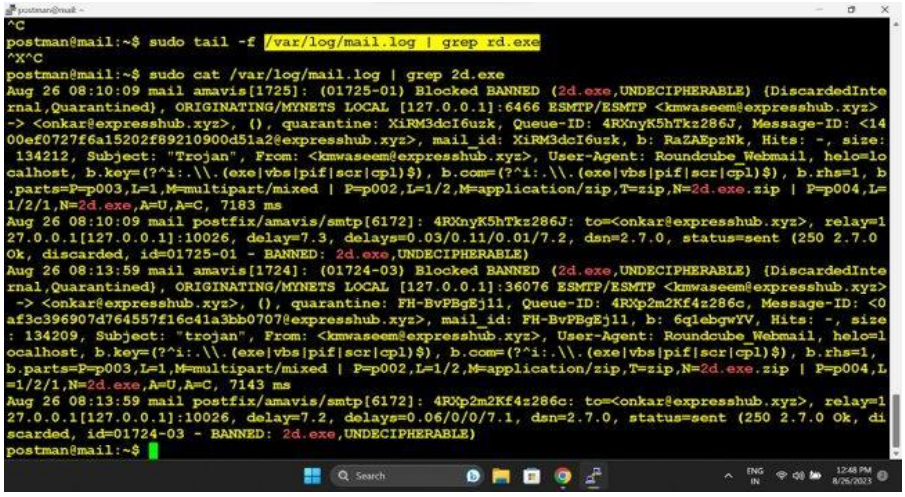
Step 2: User Logged-in:



Step 3: Safe Mail Received:



Step 4: Malicious Mail Dropped: Mail Log



7. APPLICATIONS

The versatile applications of this project extend across various sectors, each benefiting from enhanced email security and streamlined communication:

1. IT Sector:
    a. Establishes secure communication channels for IT professionals.
    b. Mitigates cybersecurity risks in email exchanges and attachments.
    c. Ensures confidential data sharing among IT teams and stakeholders.
2. Personal Email Security:
    a. Protects personal emails from viruses and malicious content.
    b. Ensures safe communication with friends and family.
    c. Enhances privacy by authenticating emails and safeguarding attachments.
3. Business Communication Enhancement:
    a. Provides secure email platform for internal and external communication.
    b. Guards against cyber threats, ensuring sensitive business information remains confidential.
    c. Enables employees to interact with clients and partners in a safe digital environment.
4. Educational Institutions:
    a. Offers a secure communication channel for students and faculty.
    b. Prevents cyberattacks and malware spread within the institution's network.
    c. Helps educational institutions maintain data integrity and student privacy.
5. Healthcare Facilities:
    a. Facilitates secure communication among healthcare professionals and patients.
    b. Safeguards sensitive medical information and ensures HIPAA compliance.
    c. Prevents potential data breaches that could compromise patient confidentiality.
6. Government Agencies:
    a. Provides a fortified email infrastructure for official correspondence.
    b. Enhances data security for sensitive government communications.
    c. It helps prevent cyber espionage and other digital threats targeted at government entities.

8. ADVANTAGES & DISADVANTAGES

This project has many good things:

1. **Strong Protection:** It keeps emails safe from bad computer stuff.
2. **No Harmful Stuff:** It finds and stops bad things in emails.
3. **Real Emails Only:** It makes sure emails are from real people, not fake ones.
4. **Easy to Use:** It has a simple website to read and send emails.
5. **Helps Everyone:** It's useful for people, businesses, schools, hospitals, and government offices.

Despite its benefits, this project has a couple of limitations:

1. **Complex Implementation:** Setting up and maintaining the secure email server can be technically challenging.
2. **Resource Intensive:** The system may require significant computing resources and regular updates for optimal performance.

9. CONCLUSION

In wrapping up, this project stands as a shield, making emails safer from bad things like viruses. It's a helpful solution for personal and business emails, schools, healthcare, and even governments. By ensuring emails are real and free from harm, this project boosts trust in digital communication. Despite some technical complexity and resource needs, the benefits of enhanced security and easy email access outweigh the challenges. In a world where cyber threats are real, this project shines as a protector, bringing a safer and more trustworthy email experience to everyone.

10. REFERENCES

1. " **Email Security for Dummies** " by *John R. Levine, Michael A. Simon, and Mark S. Ryan*. This book is part of the "For Dummies" series, known for its approachable explanations and practical insights.

2. " **Cyber Security for Beginners** " by *Raef Meeuwisse.*
   Publisher: Apress, While not exclusively focused on email security, this book offers a beginner-friendly introduction to cybersecurity concepts, including aspects relevant to email server security.

3. Reference Link (Kamatera VPC): *https://www.linuxbabe.com/linux-server/how-to-create-a-linux-vps-server-on-kamatera*

4. SPF and DKIM Setup link : https://www.linuxbabe.com/mail-server/spf-dkim-postfix-debian-server

5. DMARC Setup Link:
   https://www.linuxbabe.com/mail-server/opendmarc-postfix-debian