National College of Ireland

Configuration Manual

MSc Research Project
Msc in Cybersecurity

Kamal Kishore
Student ID: x22146270

School of Computing
National College of Ireland

Supervisor: Prof. Imran Khan

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Kamal Kishore
Student ID:	X22146270
Programme:	Msc in Cybersecurity **Year:** 2023
Module:	MSc Research Practicum/Internship part 2
Lecturer:	Prof. Imran Khan
Submission Due Date:	12th August 2024
Project Title:	Config manual for Securing Postfix Mail servers using Fail2ban and containerization
Word Count:	3815 **Page Count:** 27

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Kamal Kishore
Date:	11 Aug. 24

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Kamal Kishore
Student ID: x22146270

1 Introduction

This research project will look at the efficacy of Fail2Ban and containerization in improving the security of Postfix email servers. It was designed as part of the Msc in Cybersecurity research project at the National College of Ireland. The technique is followed in virtual environment and is divided into many steps, including building environment, security design and execution, effectiveness evaluation, and outcomes analysis.
Basic walkthrough of this manual is as followed:

1. Prerequisites
2. Installation and Configuring MariaDB
3. Installation of core application like podman, python and pip
4. Building 1st container for DMARC
5. Building 2nd container for DKIM
6. Building 3rd container for DOVECOT
7. Building 4th container for POSTFIX
8. Configuring fail2ban on container's host machine
9. Evaluation
10. Outcome analysis

2 Prerequisites

To proceed with the setup, two virtual machines are required with minimum version of operating system = Rocky Linux 9.2 and one virtual machine with Kali Linux operating system is needed. Resources allocated for the machines are as followed:

1. **Storage**: 30 GB
2. **RAM**: 2 GB
3. **Processors**: 2
4. **Family of Processor** (*optional*): Intel(R) Core (TM) i7-4810MQ CPU @ 2.80GHz

Roles of the machines are as followed:

1. *Database machine*: This will be a "virtual machine 1" (VM 1) where MariaDB is configured as database.
2. *Mail Server*: This will be a "virtual machine 2" (VM 2) where containers and all mail services will be configured inside the containers.
3. *Attacker's machine*: This will be a "virtual machine 3" (VM 3) based on Kali Linux that will act as attacker and will try to perform brute-force attack.

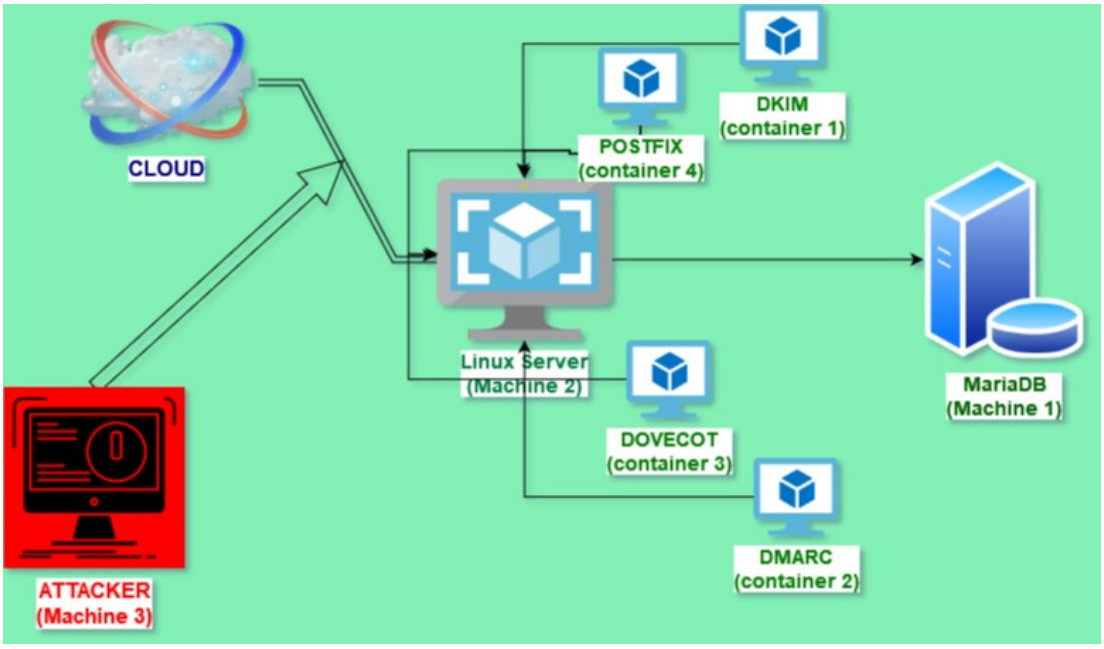

FIGURE 1: Demonstrating architecture for the research.

Virtual machines can be created using any hypervisor. For this research, Virtualbox 6.1 is chosen for building virtual machines. Here are the steps mentioned for creating virtual machines:

Click on “New” button, as highlighted.

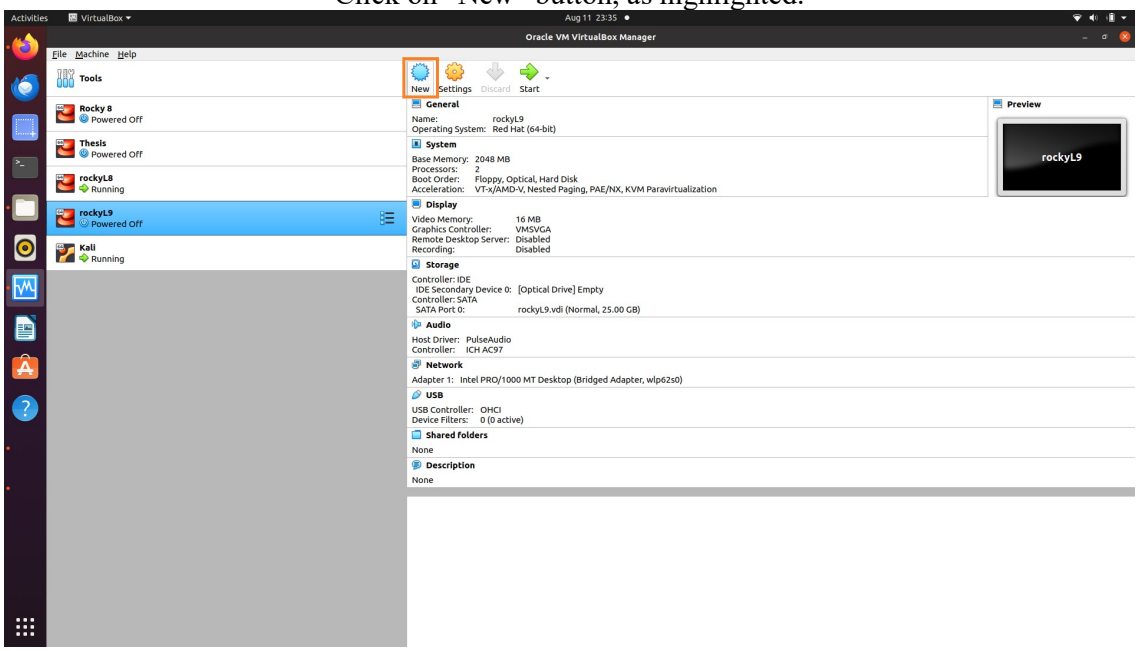

FIGURE 2: how to start building VM.

1. Give the desired name – “Rocky 9 ver2” then Press Next button.

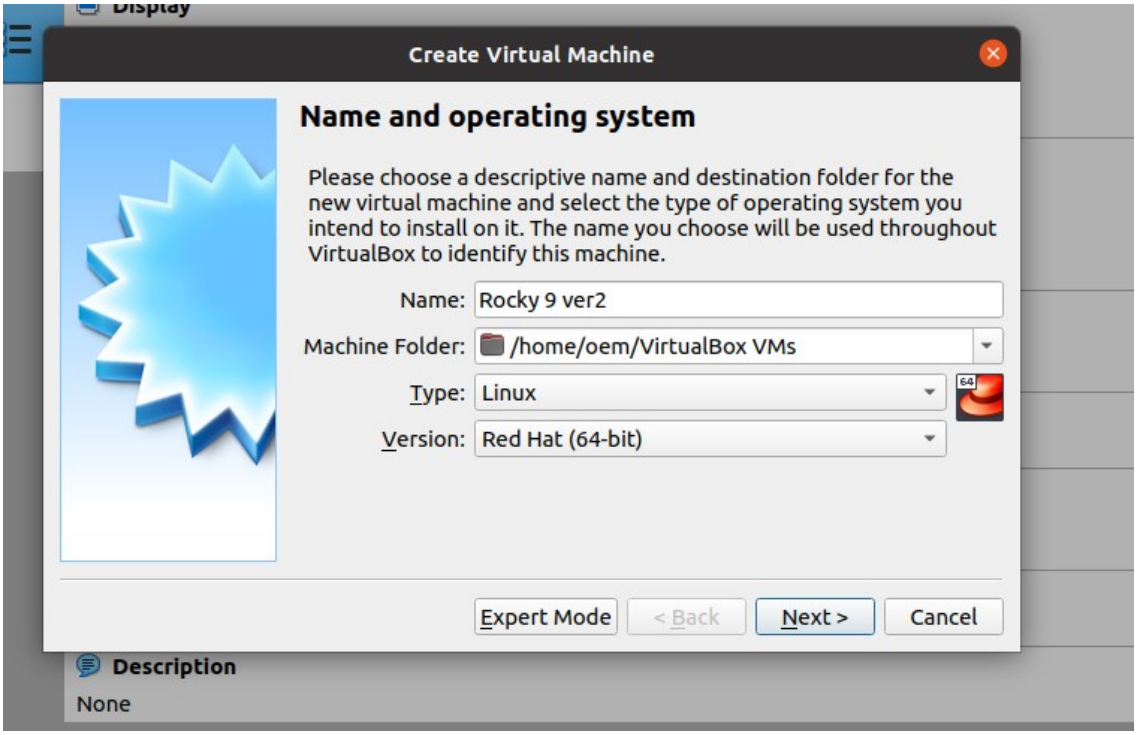

FIGURE 3: Desired name, Type and Version chosen for the building VM.

2. Give the desired Memory. For this research, 2 GB is alotted.

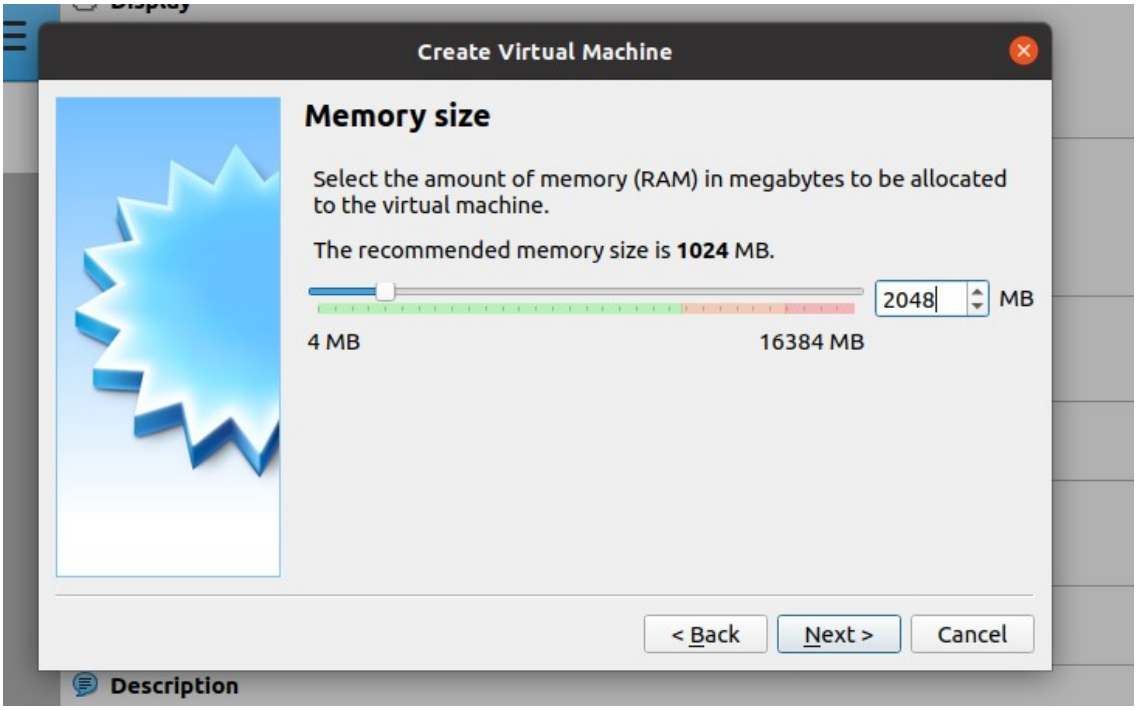

FIGURE 4: Demonstrating amount of RAM allocated to VM.

3. Create a new virtual disk.

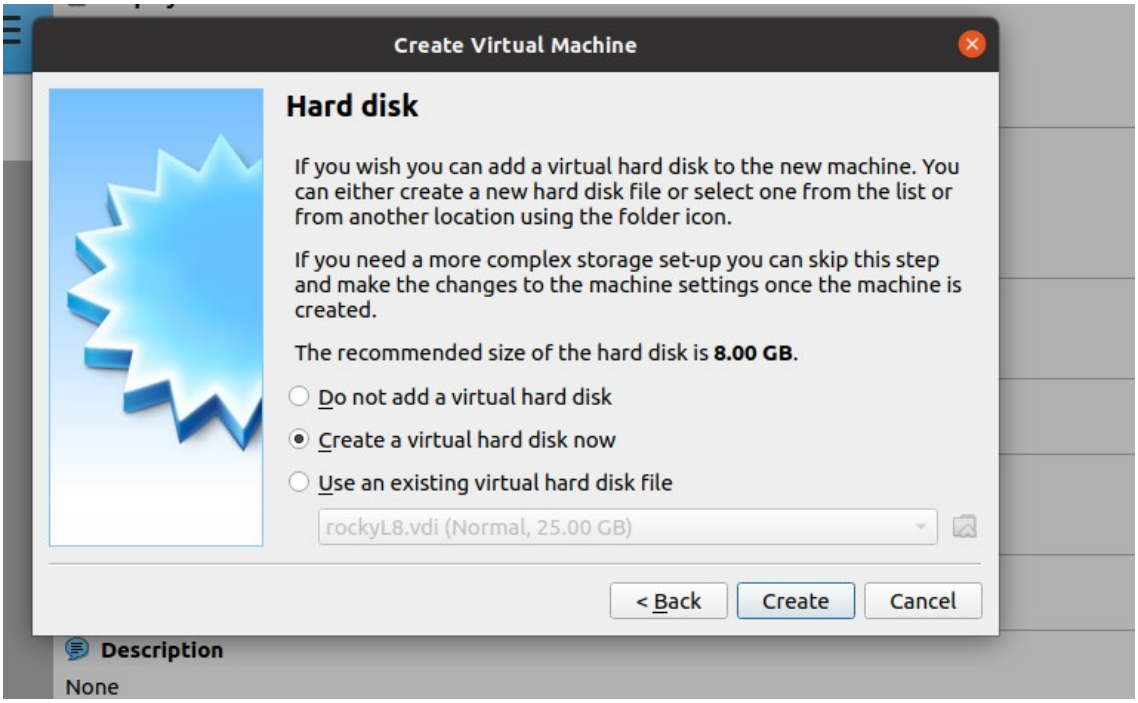

FIGURE 5: Demonstrating creation of new virtual HDD.

4. Continue pressing "Next" button till size of "File location and size" is asked. Mention 25.00 GB, as demonstrated. Then click "Create".

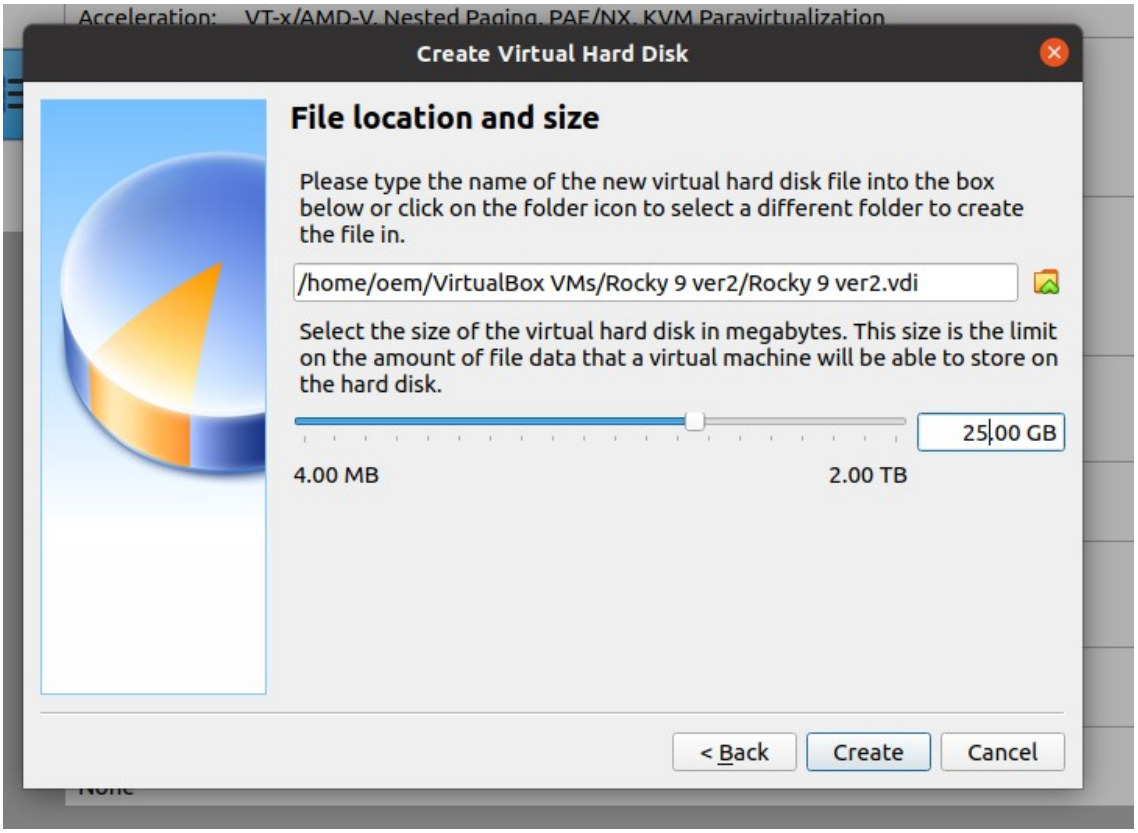

FIGURE 6: Demonstrating size of HDD allocated to VM.

5. Now, goto settings and assign 2 processors.

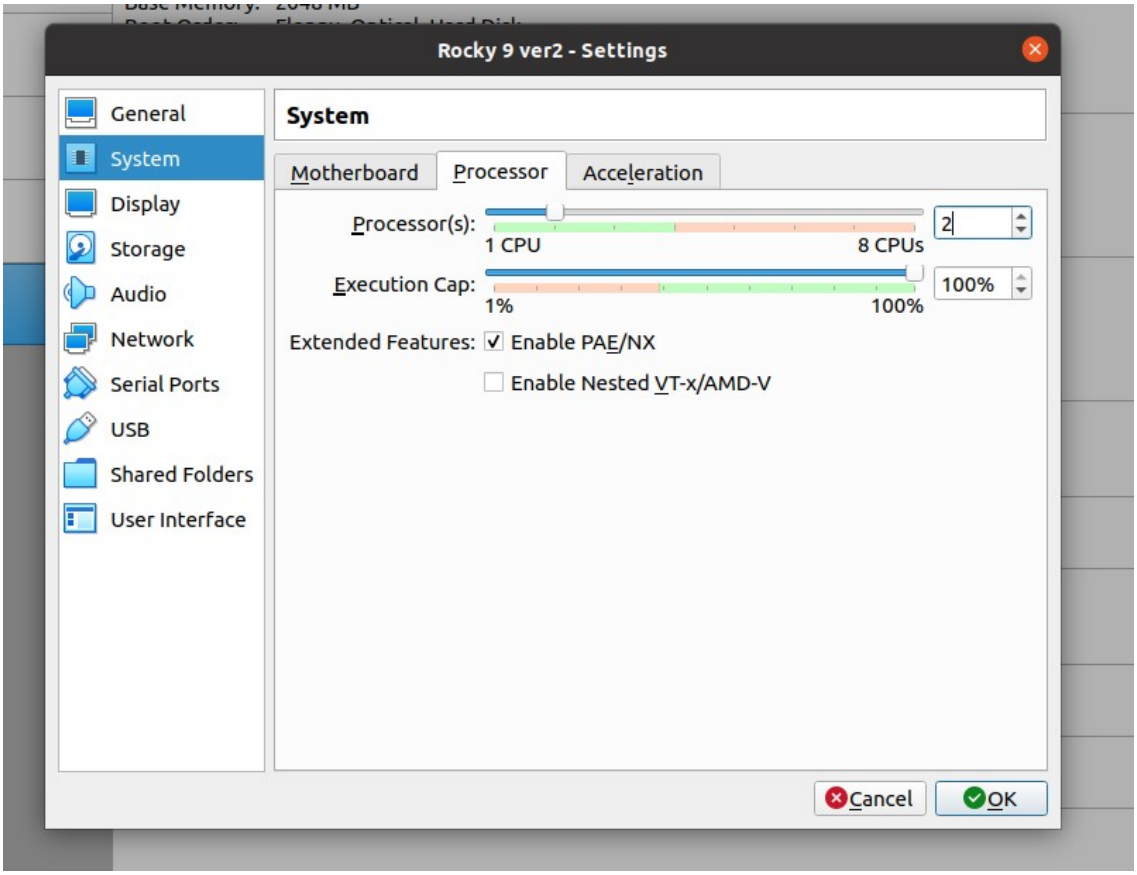

FIGURE 7: Allocating number of processors for VM.

6. Then, change the Network adapter to Bridge.

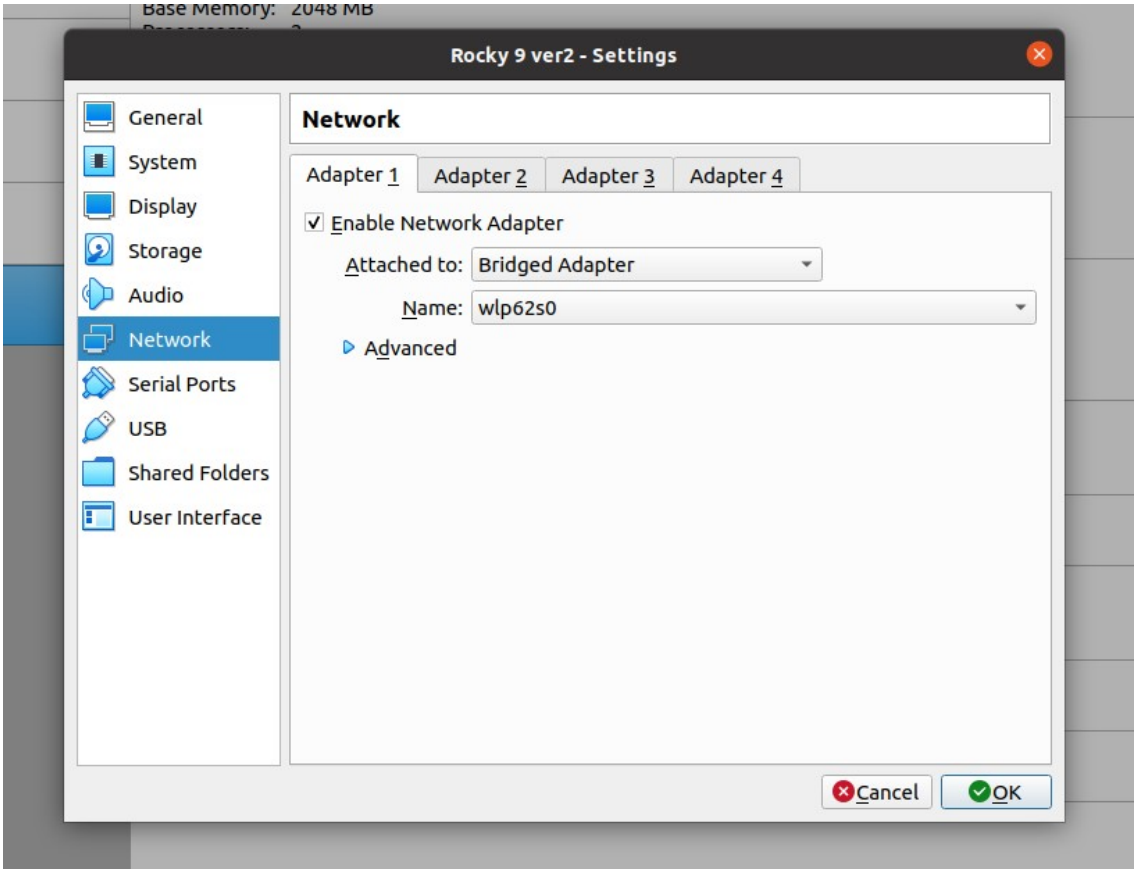

FIGURE 8: Assigning Bridged adapter to VM

7. Now, need to define the ISO image.

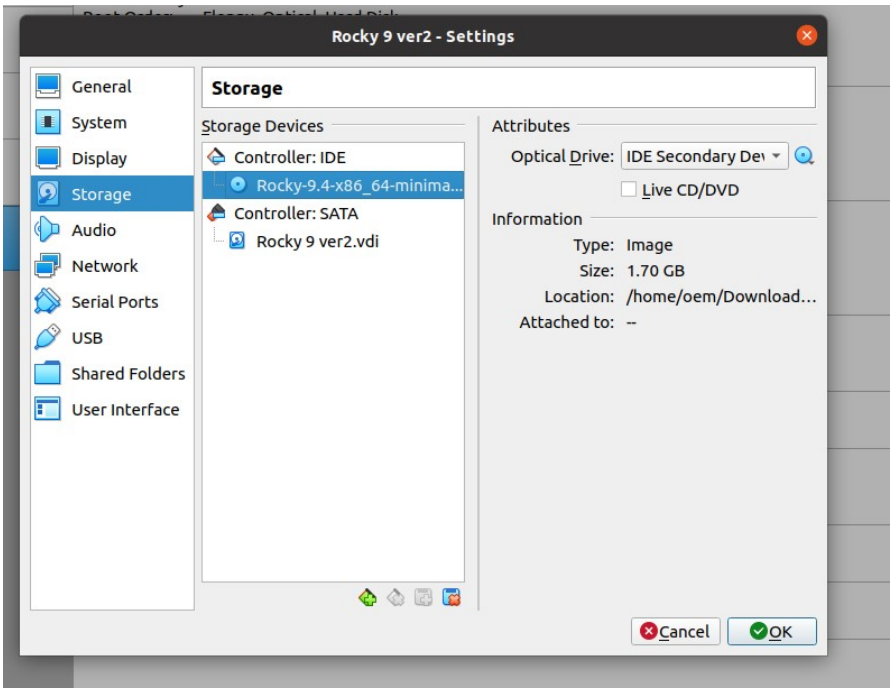

FIGURE 9: Assigning ISO for VM.

8. Finally, boot the image.

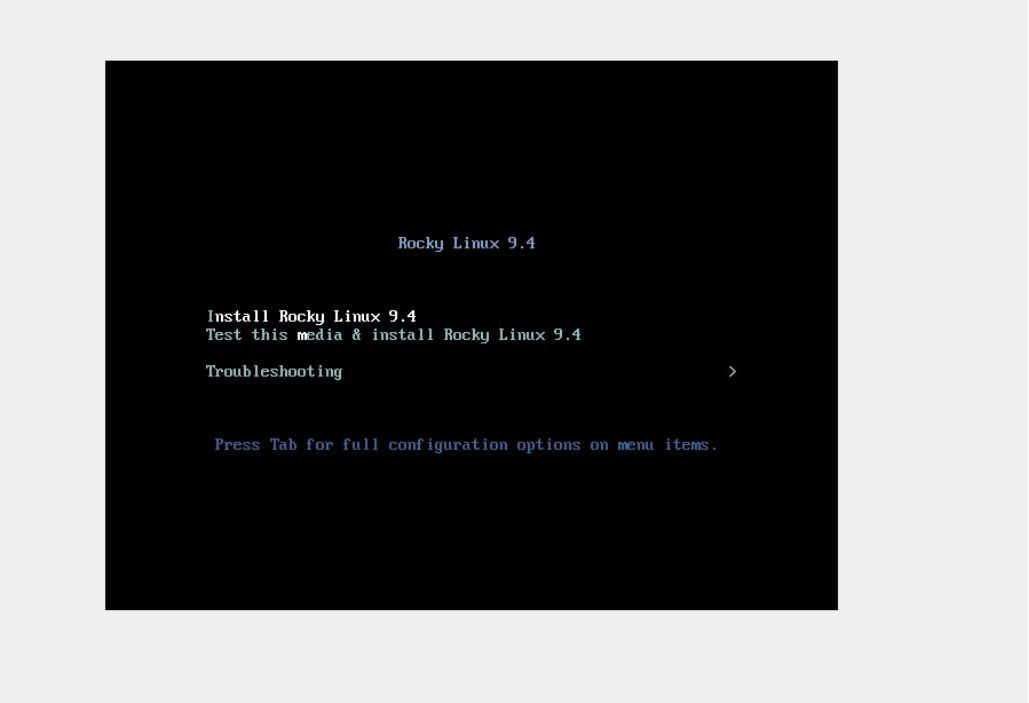

FIGURE 1: Mounted ISO booted for installing Rocky Linux.

9. Select the HDD, define user credentials as per demonstrated installation configs.

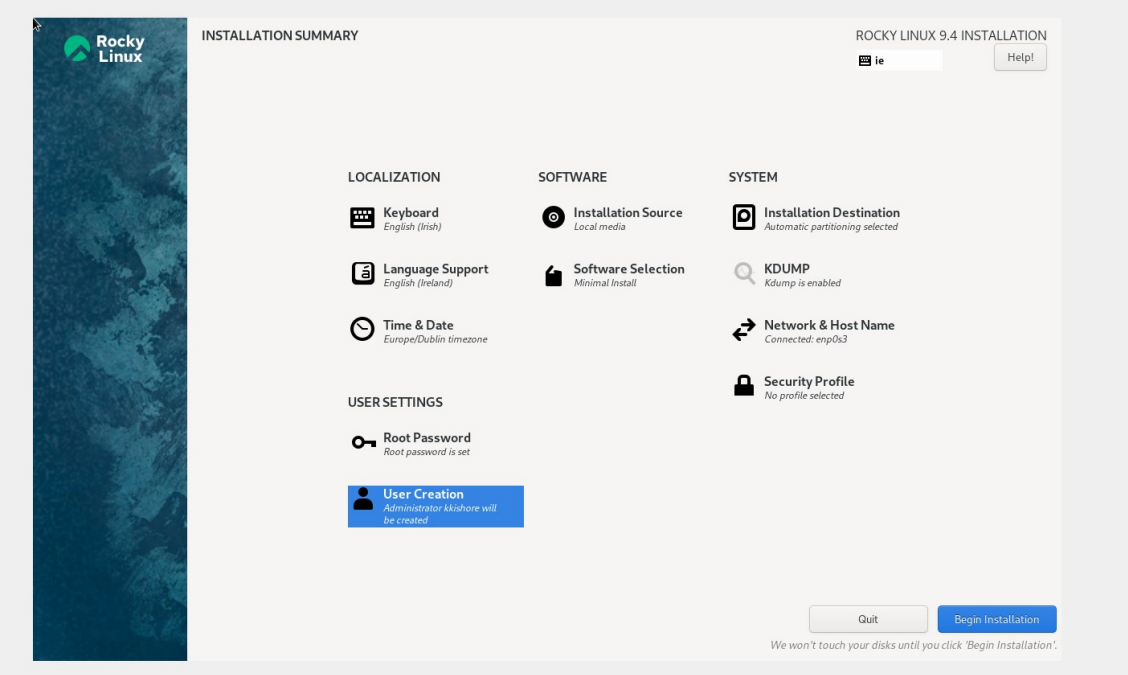

FIGURE 10: Installation Config

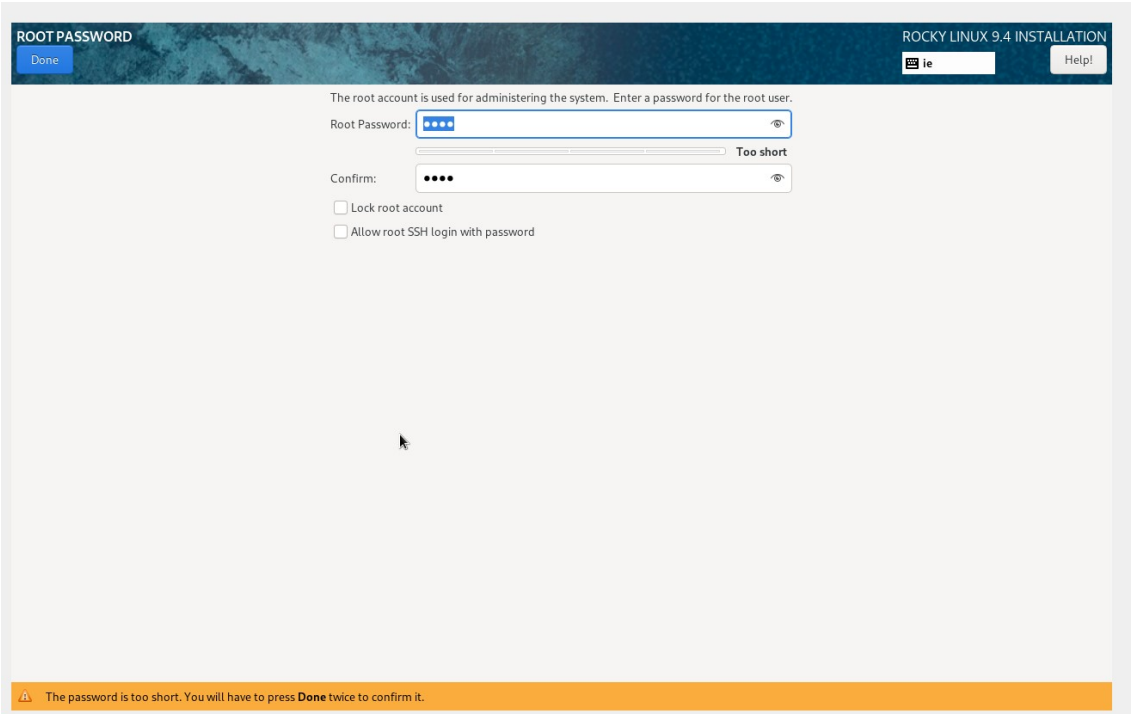

FIGURE 11: Setting Root password

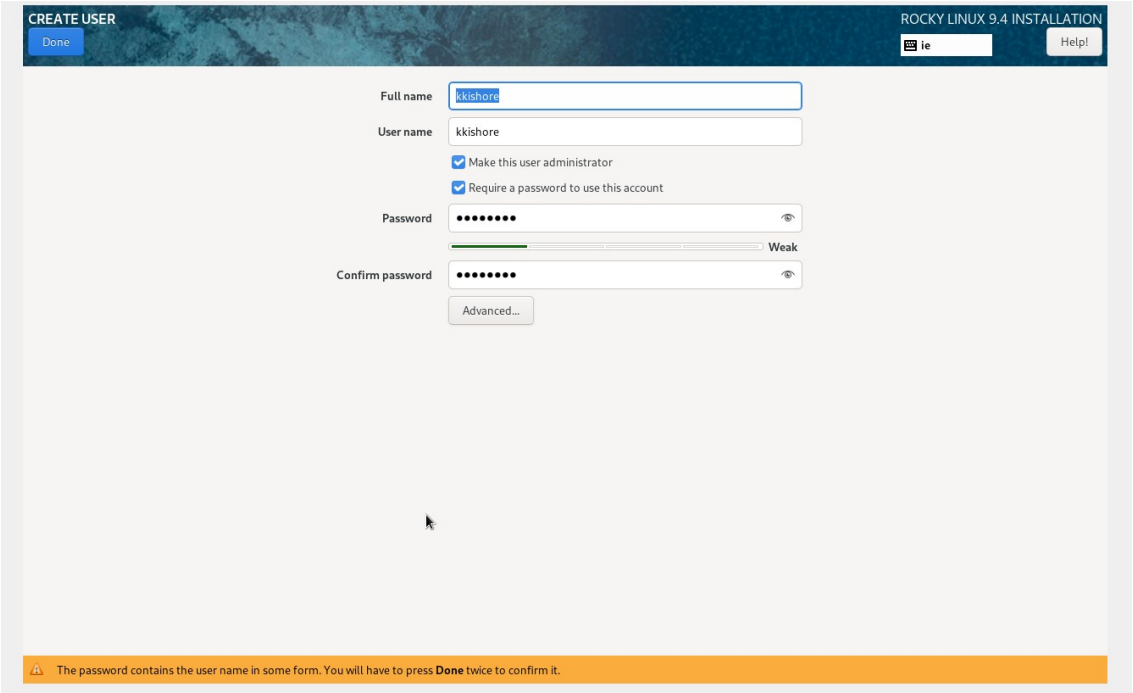

FIGURE 12: Configuring user and Setting password

10. Finally, installation started. Press “Reboot” after completion:

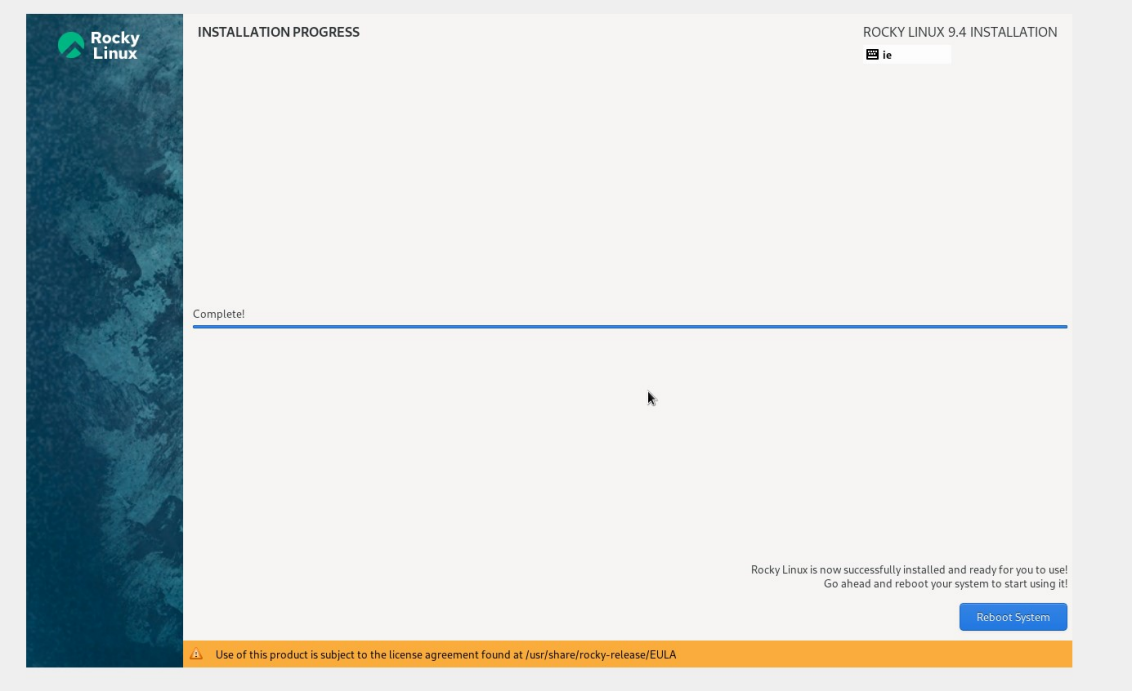

FIGURE 13: Installation completed and Reboot needed.

3 Installing and configuring MariaDB

Firstly, referring to a book published by Tonello in 2022, MariaDB Database server will be built in VM1 (Tonello, 2022). As this machine belongs to Rocky Linux 9.2, prior to installation ensure IP address is static IP is configured. In this research 192.168.1.16 is assigned by DHCP to this server. Now, installation will be done through executing command "*dnf install -y mariadb mariadb-server*". Further, configuration steps are mentioned below:

1. Start the service and make service persistent: *sudo systemctl start mariadb; sudo systemctl enable --now mariadb*
2. Allow the service to allow connections at firewall: *sudo firewall-cmd --add-service=mysql*
3. Reload firewall: *sudo firewall-cmd --runtime-to-permanent*
4. Proceed for configuring initial settings for MariaDB: *sudo mysql_secure_installation*

Switch to unix_socket authentication [Y/n] n
Change the root password? [Y/n] Y
New password: *(root@123)*
Re-enter new password: *(root@123)*
Password updated successfully!
Reloading privilege tables..
... Success!
Remove anonymous users? [Y/n] *y*
... Success!
Disallow root login remotely? [Y/n] *y*
... Success!
Remove test database and access to it? [Y/n] *y*
- Dropping test database...
... Success!

- Removing privileges on test database...
... Success!

Reload privilege tables now? [Y/n] *y*
... Success!
Thanks for using MariaDB!

5. Login to MariaDB database server via password created in previous step: *sudo mysql -u root -p*
6. Now create a new database: *CREATE DATABASE mailserver;*
7. Create the MySQL user and give the new user access to the database: *GRANT ALL PRIVILEGES ON mailserver.* TO 'mailuser'@'%' IDENTIFIED BY 'password';*
8. To activate the modification, flush MySQL privileges*: FLUSH PRIVILEGES;*
9. Change the database: use mailserver;
10. Now, create table for the users; *CREATE TABLE `virtual_users` (`id` int(11) NOT NULL AUTO_INCREMENT, `password` varchar(106) NOT NULL, `email` varchar(100) NOT NULL, PRIMARY KEY (`id`), UNIQUE KEY `email` (`email`)) ENGINE=InnoDB DEFAULT CHARSET=utf8;*
11. Finally, add data to the table: *INSERT INTO virtual_users (password, email) VALUES (ENCRYPT('passw1234', CONCAT('6', SUBSTRING(SHA(RAND()), -16))), 'user6');*

```
[root@rocky9 ~]# sudo mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 11
Server version: 10.5.22-MariaDB MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> CREATE DATABASE mailserver;
Query OK, 1 row affected (0.001 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON mailserver.* TO 'mailuser'@'%' IDENTIFIED BY 'password';
Query OK, 0 rows affected (0.011 sec)

MariaDB [(none)]>  FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.000 sec)

MariaDB [(none)]> use mailserver;
Database changed
MariaDB [mailserver]> CREATE TABLE `virtual_users` (     `id` int(11) NOT NULL AUTO_INCREMENT,     `password` varchar(106) NOT NULL,     `email` var
char(100) NOT NULL,     PRIMARY KEY (`id`),     UNIQUE KEY `email` (`email`) ) ENGINE=InnoDB DEFAULT CHARSET=utf8;
Query OK, 0 rows affected (0.048 sec)

MariaDB [mailserver]> INSERT INTO virtual_users (password, email) VALUES (ENCRYPT('passw1234', CONCAT('$6$', SUBSTRING(SHA(RAND()), -16))), 'user1')
;
Query OK, 1 row affected (0.018 sec)

MariaDB [mailserver]> exit
Bye
```

FIGURE 14: Database and User Creation in MariaDB for postfix.

VERIFYING CONFIGURATION

1. Again, login into mysql server: *sudo mysql -u root -p*
2. Check the contents of the virtual_users table: *SELECT * FROM mailserver.virtual_users;*

Check that the output includes the email addresses you added to the virutal_users database. Your hashed passwords will seem longer than what is shown below:

FIGURE 15: Validating user creation into database

4 Installing and configuring containers

Referring to official documentation on Podman from Chauhan and Morvan (2024), VM 2 will be configured with container utility and then multiple containers will be built on it through podman-compose.yml utility. Prior to proceeding configuration part ensure that static IP is assigned to the machine. In this research, IP assigned by DHCP to the machine is 192.168.1.18.

BUILDING CONTAINERS

Steps for building containers are mentioned below:

STEP 1: Update the machine: *dnf update -y*
STEP 2: Install the podman container via command: *dnf install podman -y*
STEP 3: Start the podman service: *systemctl start podman*
STEP 4: Install the python3 and pip few software together: *dnf install python3 python3-pip podman-compose vim epel-release -y*
STEP 5: Install podman compose utility: *pip3 install podman-compose*
 # Install Docker Compose CLI plugin
mkdir -p ~/.docker/cli-plugins
curl -SL https://github.com/docker/compose/releases/download/v2.18.1/docker-compose-linux-x86_64 -o ~/.docker/cli-plugins/docker-compose
chmod +x ~/.docker/cli-plugins/docker-compose

STEP 6: Switch to root user: sudo -i
STEP 7: Now, create a new folder by name “smtpserver”: *mkdir smtpserver*
STEP 8: Change directory to “smtpserver” and create a new file with name docker-compose.yml: *cd smtpserver; vim docker-compose.yml*
STEP 9: Copy and paste below contents in docker-compose.yml:

```
#version: '3.8'

networks:
  mailnet:
    driver: bridge
    ipam:
      config:
        - subnet: 172.18.0.0/16

services:
  dmarc:
    build:
```

```
    context: ./dmarc
  container_name: dmarc
  volumes:
    - /opt/container/log/maillog:/var/log/maillog
  networks:
    mailnet:
      ipv4_address: 172.18.0.5

dkim:
  build:
    context: ./dkim
  container_name: dkim
  volumes:
    - /opt/container/log/maillog:/var/log/maillog
  networks:
    mailnet:
      ipv4_address: 172.18.0.4

dovecot:
  build:
    context: ./dovecot
  container_name: dovecot
  ports:
    - "143:143"
  volumes:
    - /opt/container/log/maillog:/var/log/maillog
    - /opt/container/private/auth:/var/spool/postfix/private/auth
    - /opt/container/home:/home
  networks:
    mailnet:
      ipv4_address: 172.18.0.3

postfix:
  build:
    context: ./postfix
  container_name: postfix
  ports:
    - "25:25"
    - "587:587"
  volumes:
    - /opt/container/log/maillog:/var/log/maillog
    - /opt/container/home:/home
    - /opt/container/private/auth:/var/spool/postfix/private/auth
  networks:
    mailnet:
      ipv4_address: 172.18.0.2
```

STEP 10: Create some important directories to be shared among containers: *mkdir /opt/container/{log,home,private} -p*

STEP 11: Create some important files into newly created directories: *touch /opt/container/log/maillog /opt/container/private/auth*
STEP 12: Assign full permission to the container folder: *chmod 777 /opt/container -R*
STEP 13: Disable the selinux: *sed -i -e "7s/permissive/disabled/g" /etc/selinux/config; setenforce 0; reboot*

5 Building 1st container for DMARC

On same VM 2, DMARC will be configured via referring to information shared by Xiao, (2021). Below are the required steps to be followed:

STEP 1: Switch to root user: *sudo -i*
STEP 2: Create and new directory by name dmarc and change directory to it: *mkdir ~/smtpserver/dmarc; cd ~/smtpserver/dmarc*
STEP 3: Create a file name opendmarc.conf and copy paste below contents: *vim opendmarc.conf*

##

UserID opendmarc:mail
Socket inet:54321@172.18.0.5
SoftwareHeader true
SPFIgnoreResults true
SPFSelfValidate true
Syslog true
UMask 007
RejectFailures false
SyslogFacility mail
TrustedAuthservIDs mail.domainname.com
IgnoreMailFrom mail.domainname.com,domainname.com
FailureReportsSentBy postmaster@domainname.com
FailureReportsBcc postmaster@domainname.com
FailureReports false
AutoRestart true
HistoryFile /var/log/opendmarc.log

STEP 4: Create a Dockerfile and copy paste below contents: *vim Dockerfile*

FROM rockylinux/rockylinux:9

RUN dnf update -y ; dnf install -y rsyslog net-tools epel-release redhat-rpm-config 'dnf-command(config-manager)' ; dnf config-manager --set-enabled crb; dnf install -y opendmarc

COPY opendmarc.conf /etc/opendmarc.conf
*RUN chown root:root -R /etc/opendmarc**
RUN sed -i "s/domainname.com/sample1.in/g" /etc/opendmarc.conf
RUN touch /var/log/opendmarc.log; chmod 777 /var/log/opendmarc.log

CMD ["/bin/sh", "-c", "/usr/sbin/rsyslogd && /usr/sbin/opendmarc -f -c /etc/opendmarc.conf"]

6 Building 2nd container for DKIM

EasyDMARC (2023) will help in configuring DKIM service inside the container.
STEP 1: Create and new directory by name dkim and change directory to it: *mkdir ~/smtpserver/dkim; cd ~/smtpserver/dkim*
STEP 2: Create a file name opendkim.conf and copy paste below contents: *vim opendkim.conf*

```
##############################################################################
# Statistics    /var/spool/opendkim/stats.dat
SendReports     yes
# ReportAddress "Example.com Postmaster" <postmaster@example.com>
SoftwareHeader  yes
# Domain        example.com
Selector        default
MinimumKeyBits  1024
KeyFile /etc/opendkim/keys/default.private
# PeerList      X.X.X.X
OversignHeaders From
# QueryCache    yes

##############################################################################
##################
##  Additional Config
#

AutoRestart             Yes
AutoRestartRate         10/1h
UMask                   002
Syslog                  yes
SyslogSuccess           Yes
LogWhy                  Yes

Canonicalization        relaxed/simple

ExternalIgnoreList      refile:/etc/opendkim/TrustedHosts
InternalHosts           refile:/etc/opendkim/TrustedHosts
KeyTable                refile:/etc/opendkim/KeyTable
SigningTable            refile:/etc/opendkim/SigningTable

Mode                    sv
PidFile                 /var/run/opendkim/opendkim.pid
SignatureAlgorithm      rsa-sha256

UserID                  opendkim:opendkim

Socket                  inet:12301@172.18.0.4
```

STEP 3: Create opendkim directory and 3 files – KeyTable, SigningTable and TrustedHosts inside it*: mkdir opendkim; touch opendkim/{KeyTable,SigningTable,TrustedHosts}*
STEP 4: Copy and paste below content in KeyTable: *vim opendkim/KeyTable*

dkim._domainkey.domainname.com
domainname.com:dkim:/etc/opendkim/keys/domainname.com/dkim.private

---

STEP 5: Copy and paste below contents in SigningTable: *vim opendkim/SigningTable*
**@domainname.com dkim._domainkey.domainname.com*

STEP 6: Copy and paste below contents in TrustedHosts: *vim opendkim/TrustedHosts*

127.0.0.1
*.domainname.com

STEP 7: Create a folder by name of "keys": *mkdir keys*
STEP 8: Create a Dockerfile and copy paste below contents: *vim Dockerfile*

FROM rockylinux/rockylinux:9

RUN dnf update -y ; dnf install -y vim rsyslog net-tools epel-release redhat-rpm-config
RUN dnf install 'dnf-command(config-manager)' -y; dnf config-manager --set-enabled crb; dnf install -y opendkim opendkim-tools

COPY opendkim.conf /etc/opendkim.conf
COPY opendkim/ /etc/opendkim/
COPY keys/ /etc/opendkim/keys/
RUN sed -i "s/domainname.com/sample1.in/g" /etc/opendkim/TrustedHosts
RUN sed -i "s/domainname.com/sample1.in/g" /etc/opendkim/KeyTable
RUN sed -i "s/domainname.com/sample1.in/g" /etc/opendkim/SigningTable
RUN opendkim-genkey -s dkim -d sample1.in; mkdir /etc/opendkim/keys/sample1.in; mv dkim. /etc/opendkim/keys/sample1.in/*
*RUN chown opendkim /etc/opendkim/keys/sample1.in/dkim**
RUN cat /etc/opendkim/keys/sample1.in/dkim.txt

CMD ["/bin/sh", "-c", "/usr/sbin/rsyslogd && /usr/sbin/opendkim -f -x /etc/opendkim.conf"]

---

7 Building 3rd container for Dovecot

Linode (2018) has provided information about configuring dovecot service on host machine but in this research that information will be used to configure the service inside container. Steps are given below:

STEP 1: Create a directory by name dovecot and change the path to it: mkdir *~/smtpserver/dovecot; cd ~/smtpserver/dovecot*
STEP 2: Create a file by name dovecot.conf and paste below contents: *vim dovecot.conf*

Dovecot configuration file

 # If you're in a hurry, see http://wiki2.dovecot.org/QuickConfiguration

 # "doveconf -n" command gives a clean output of the changed settings. Use it
 # instead of copy&pasting files when posting to the Dovecot mailing list.

 # '#' character and everything after it is treated as comments. Extra spaces
 # and tabs are ignored. If you want to use either of these explicitly, put the
 # value inside quotes, eg.: key = "# char and trailing whitespace "
# Protocols want to be serving.
protocols = imap pop3 lmtp
listen = *
dict {
}
!include conf.d/.conf*
!include_try local.conf

STEP 3: Create a directory name "conf.d" and change path to it: *mkdir conf.d; cd conf.d*
STEP 4: Create a file by name "10-auth.conf" and paste below contents; *vim 10-auth.conf*

disable_plaintext_auth = no
auth_mechanisms = plain login
!include auth-sql.conf.ext

STEP 5: Create a file by name "10-mail.conf" and paste below contents; *vim 10-mail.conf*

mail_location = maildir:~/Maildir
namespace inbox {
* inbox = yes*
}
first_valid_uid = 1000
mbox_write_locks = fcntl

STEP 6: Create a file by name "10-logging.conf" and paste below contents: *vim 10-logging.conf*

log_path = /var/log/maillog
auth_debug_passwords = yes
plugin {
}

STEP 7: Create a file by name "10-master.conf" and paste below contents: *vim 10-master.conf*

service imap-login {
* inet_listener imap {*
* }*
* inet_listener imaps {*
* }*
}

service pop3-login {

```
  inet_listener pop3 {
  }
  inet_listener pop3s {
  }
}

service lmtp {
  unix_listener lmtp {
  }

}

service imap {
}

service pop3 {
}

service auth {
  unix_listener auth-userdb {
  }

  inet_listener {
    port = 24
  }
}

service auth-worker {
}

service dict {
  unix_listener dict {
  }
}
```

STEP 8: Create a file by name “dovecot-sql.conf.ext” and paste below contents (Point to note here is host parameter. It will depend on IP of database server as discussed in building MariaDB server) : *vim ~/smtpserver/dovecot/dovecot-sql.conf.ext*

```
driver = mysql
connect = host=192.168.1.16 dbname=mailserver user=mailuser password=password
default_pass_scheme = SHA512-CRYPT

password_query = SELECT email as user, password FROM virtual_users WHERE
email='%u';
user_query = SELECT email as user, 0 as uid, 0 as gid, concat('/home/',
SUBSTRING_INDEX(email, '@', 1), '/Maildir') as home, concat('/home/',
SUBSTRING_INDEX(email, '@', 1), '/Maildir') as mail FROM virtual_users WHERE
email='%u';
```

STEP 9: Create a Dockerfile and copy paste below contents: cd ~/smtpserver/dovecot; *vim Dockerfile*

```
FROM rockylinux/rockylinux:9

RUN yum update -y ; yum install -y vim rsyslog net-tools dovecot dovecot-mysql

COPY dovecot.conf /etc/dovecot/dovecot.conf
COPY dovecot-sql.conf.ext /etc/dovecot/dovecot-sql.conf.ext
COPY conf.d/ /etc/dovecot/conf.d/
RUN chown -R root:root /etc/dovecot*

EXPOSE 143

RUN sed -i "8s/required/no/" /etc/dovecot/conf.d/10-ssl.conf
RUN sed -i "14s/^/#/; 15s/^/#/" /etc/dovecot/conf.d/10-ssl.conf

CMD ["/bin/sh", "-c", "/usr/sbin/rsyslogd && /usr/sbin/dovecot -F "]
```

8 Building 4th container for Postfix

Linode (2018) has provided information about configuring postfix service on host machine but in this research that information will be used to configure the service inside container. Steps are as followed:

STEP 1: Create a directory by name postfix and change the path to it: *mkdir ~/smtpserver/postfix; cd ~/smtpserver/postfix*
STEP 2: Create a file by name main.cf and paste below contents: *vim main.cf*

```
compatibility_level = 2
myhostname = mail.sample1.in # desired host name. Usually "mail." keyword is prefix to the name.
mydomain = sample1.in #desired domain name
unknown_local_recipient_reject_code = 550
queue_directory = /var/spool/postfix
command_directory = /usr/sbin
daemon_directory = /usr/libexec/postfix
data_directory = /var/lib/postfix
mail_owner = postfix
myorigin = $mydomain # desired name to send mails. Usually mydomain variable is used.
inet_interfaces = all
inet_protocols = ipv4
unknown_local_recipient_reject_code = 550
alias_maps = hash:/etc/aliases
alias_database = hash:/etc/aliases
debug_peer_level = 2
debugger_command =
        PATH=/bin:/usr/bin:/usr/local/bin:/usr/X11R6/bin
        ddd $daemon_directory/$process_name $process_id & sleep 5
sendmail_path = /usr/sbin/sendmail.postfix
```

newaliases_path = /usr/bin/newaliases.postfix
mailq_path = /usr/bin/mailq.postfix
setgid_group = postdrop
html_directory = no
manpage_directory = /usr/share/man
sample_directory = /usr/share/doc/postfix-2.10.1/samples
readme_directory = /usr/share/doc/postfix-2.10.1/README_FILES
home_mailbox = Maildir/

mynetworks = 10.0.100.0/24, 10.0.101.0/24, 127.0.0.0/8 #your network range
smtpd_banner = $myhostname ESMTP $mail_name
mydestination = $myhostname, localhost.$mydomain, localhost, $mydomain

#dovecot
smtpd_sasl_type = dovecot
smtpd_sasl_path = inet:172.18.0.3:24
smtpd_sasl_auth_enable = yes
smtpd_sasl_security_options = noanonymous
broken_sasl_auth_clients = yes
#smtpd_use_tls = yes
#smtpd_tls_cert_file = /etc/pki/tls/certs/CertFile.crt
#smtpd_tls_key_file = /etc/pki/tls/certs/KeyFile.key
#smtp_tls_security_level = may

#RESTRICTION 1
smtpd_sender_restrictions = permit_mynetworks, permit_sasl_authenticated,
reject_unknown_sender_domain
smtpd_recipient_restrictions = check_sender_access hash:/etc/postfix/sender_access,
permit_mynetworks, permit_sasl_authenticated, reject_sender_login_mismatch,
reject_unauth_destination
smtpd_client_restrictions = permit_mynetworks, permit_sasl_authenticated, reject_rbl_client
zen.spamhaus.org, reject_rbl_client sbl.spamhaus.org, reject_rbl_client xbl.spamhaus.org,
reject_rbl_client dnsbl.sorbs.net, reject_rbl_client b.barracudacentral.org, permit

#RESTRICTION 2
queue_run_delay=1000s
minimal_backoff_time=2000s
bounce_queue_lifetime=15m
maximal_queue_lifetime=15m
message_size_limit=20480000

#Common for OPENDKIM & OPENDMARC
milter_protocol = 2
milter_default_action = accept
smtpd_milters = inet:172.18.0.4:12301, inet:172.18.0.5:54321
non_smtpd_milters = inet:172.18.0.4:12301, inet:172.18.0.4:54321, unix:private/opendkim,
unix:private/opendmarc

maillog_file = /var/log/maillog
virtual_transport = lmtp:unix:private/dovecot-lmtp

virtual_mailbox_maps = mysql:/etc/postfix/mysql-virtual-mailbox-maps.cf
virtual_alias_maps = mysql:/etc/postfix/mysql-virtual-alias-maps.cf

---

STEP 3: Paste below content in new file = /etc/postfix/mysql-virtual-mailbox-maps.cf

user=mailuser
password=password
hosts=192.168.1.16
dbname=mailserver
query= SELECT 1 FROM virtual_users WHERE email='%s'

STEP 4: Paste below content in new file = /etc/postfix/mysql-virtual-alias-maps.cf

user = mailuser
password = password
hosts = 192.168.1.16
dbname = mailserver
query = SELECT email FROM virtual_users WHERE email='%s'

STEP 5: Create file "master.cf" and paste below contents: *vim master.cf*

```
smtp      inet  n       -       n       -       -       smtpd
submission inet n       -       n       -       -       smtpd
smtps     inet  n       -       n       -       -       smtpd
  -o smtpd_tls_wrappermode=yes
pickup    unix  n       -       n       60      1       pickup
cleanup   unix  n       -       n       -       0       cleanup
  -o header_checks=pcre:/etc/postfix/header_checks
qmgr      unix  n       -       n       300     1       qmgr
#qmgr     unix  n       -       n       300     1       oqmgr
tlsmgr    unix  -       -       n       1000?   1       tlsmgr
rewrite   unix  -       -       n       -       -       trivial-rewrite
bounce    unix  -       -       n       -       0       bounce
defer     unix  -       -       n       -       0       bounce
trace     unix  -       -       n       -       0       bounce
verify    unix  -       -       n       -       1       verify
flush     unix  n       -       n       1000?   0       flush
proxymap  unix  -       -       n       -       -       proxymap
proxywrite unix -       -       n       -       1       proxymap
smtp      unix  -       -       n       -       -       smtp
relay     unix  -       -       n       -       -       smtp
        -o syslog_name=postfix/$service_name
showq     unix  n       -       n       -       -       showq
error     unix  -       -       n       -       -       error
retry     unix  -       -       n       -       -       error
discard   unix  -       -       n       -       -       discard
local     unix  -       n       n       -       -       local
virtual   unix  -       n       n       -       -       virtual
lmtp      unix  -       -       n       -       -       lmtp
anvil     unix  -       -       n       -       1       anvil
scache    unix  -       -       n       -       1       scache
```

postlog unix-dgram n - n - 1 postlogd

STEP 6: Create a Dockerfile and copy paste below contents: *vim Dockerfile*

FROM rockylinux/rockylinux:9

RUN yum update -y && yum install -y vim rsyslog postfix postfix-pcre

COPY main.cf /etc/postfix/main.cf
COPY master.cf /etc/postfix/master.cf
*RUN chown root:root /etc/postfix/ma**

WORKDIR /var/spool/postfix

EXPOSE 25 587

CMD ["/bin/sh", "-c", "/usr/sbin/rsyslogd && /usr/sbin/postfix start-fg"]

STEP 7: Time to build podman containers for all 4 services described above: systemctl start podman; cd ~/smtpserver/ ; *podman compose build ; podman compose up -d*

9 Configuring Fail2ban on host machine

Report shared by Ichioui (2021) has helped in achieving the main objective this research. The phenomena demonstrated in that report gave direction to create solution to the problem of blocking the ports exposed through container to host machine. Here is the details steps:

STEP 1: Execute command to update the machine: *yum update -y*
STEP 2: Execute command to install fail2ban: *yum install -y fail2ban*
STEP 3: Create jail.local to describe jails for services and paste below contents: *vim /etc/fail2ban/jail.local*

```
[DEFAULT]
banaction = iptables-multiport
findtime  = 604800

[sshd]
enabled   = true
port      = ssh
filter    = sshd
logpath   = %(sshd_log)s
backend   = %(sshd_backend)s
maxretry  = 2
bantime   = -1

[dovecot]
enabled = true
port  = 143
filter = dovecot
action = dockeraction
logpath = /opt/container/log/maillog
backend = auto
```

maxretry = 2
findtime = 600
bantime = -1

[postfix]
enabled = true
port = 25
filter = postfix
action = dockeraction_postfix
logpath = /opt/container/log/maillog
backend = auto
maxretry = 2
findtime = 600
bantime = -1

[postfix-sasl]
enabled = true
port = 587
filter = postfix
action = dockeraction_postfixsasl
logpath = /opt/container/log/maillog
backend = auto
maxretry = 2
findtime = 600
bantime = -1

STEP 4: Create a file "postfix.conf" in filter.d: *vim /etc/fail2ban/filter.d/postfix.conf*
STEP 5: Paste below contents:
[Definition]
*failregex = ^.*postfix/.*\[\d+\]: \S+\[<HOST>\]: SASL (LOGIN|PLAIN) authentication failed$*
*^.*postfix/smtpd\[\d+\]: warning: \S+\[<HOST>\]: SASL (LOGIN|PLAIN) authentication failed: .**
*^.*postfix/smtpd\[\d+\]: warning: \S+\[<HOST>\]: SASL authentication failed: .**
*^.*postfix/smtpd\[\d+\]: lost connection after AUTH from \S+\[<HOST>\]*
*^.*postfix/smtpd\[\d+\]: disconnect from \S+\[<HOST>\] \(auth failed\)*
*^.*postfix/smtpd\[\d+\]: NOQUEUE: reject: RCPT from \S+\[<HOST>\]: 550 5\.1\.1 <.*>: Recipient address rejected: User unknown in local recipient table$*

mdre-normal =

ignoreregex =

STEP 6: Create file "dovecot.conf" in filter.d: *vim /etc/fail2ban/filter.d/dovecot.conf*
[INCLUDES]
before = common.conf

[Definition]
*failregex = ^.*imap-login: Info: Disconnected: Connection closed \(auth failed, \d+ attempts in \d+ secs\): user=<[^>]*>, method=\w+, rip=<HOST>, lip=.*$*

^pam\(\S+,<HOST>(?:,\S)?\): pam_authenticate\(\) failed: (?:User not known to the underlying authentication module: \d+ Time\(s\)|Authentication failure \([Pp]assword mismatch\?\)|Permission denied)\s*$*

^[a-z\-]{3,15}\(\S,<HOST>(?:,\S*)?\): (?:[Uu]nknown user|[Ii]nvalid credentials|[Pp]assword mismatch)*

mdre-normal =

ignoreregex =

STEP 7: Define actions for handling container port 143. For this create a file in action directory and paste below contents: *vim /etc/fail2ban/action.d/dockeraction.conf*

```
[Definition]
actionstart = iptables -N f2b-npm-dovecot
        iptables -A f2b-npm-dovecot -j RETURN
#         iptables -I FORWARD -p tcp -m multiport --dports 0:65535 -j f2b-npm-dovecot
        iptables -I FORWARD -p tcp -m multiport --dport 143 -j f2b-npm-dovecot

actionstop = iptables -D FORWARD -p tcp -m multiport --dport 143 -j f2b-npm-dovecot
        iptables -F f2b-npm-dovecot
        iptables -X f2b-npm-dovecot

actioncheck = iptables -n -L FORWARD | grep -q 'f2b-npm-dovecot[ \t]'

actionban = iptables -I f2b-npm-dovecot -p tcp --dport 143 -s <ip>  -j DROP

actionunban = iptables -D f2b-npm-dovecot -p tcp --dport 143 -s <ip>  -j DROP
```

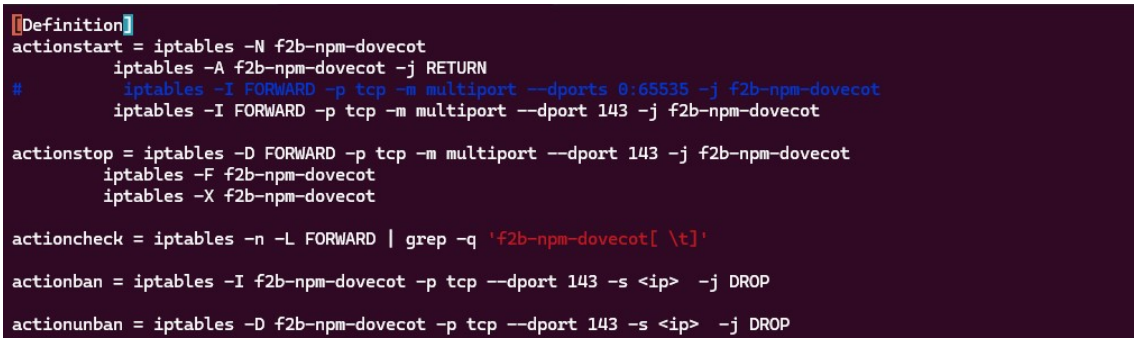

FIGURE 16: Rule defined in actions for banning IP on port 143.

STEP 8: Define actions for handling container port 25. For this create a file in action directory and paste below contents: *vim /etc/fail2ban/action.d/dockeraction_postfix.conf*

```
[Definition]
actionstart = iptables -N f2b-npm-postfix
        iptables -A f2b-npm-postfix -j RETURN
#         iptables -I FORWARD -p tcp -m multiport --dports 0:65535 -j f2b-npm-postfix
        iptables -I FORWARD -p tcp -m multiport --dport 25 -j f2b-npm-postfix

actionstop = iptables -D FORWARD -p tcp -m multiport --dport 25 -j f2b-npm-postfix
        iptables -F f2b-npm-postfix
```

iptables -X f2b-npm-postfix

actioncheck = iptables -n -L FORWARD | grep -q 'f2b-npm-postfix[\t]'

actionban = iptables -I f2b-npm-postfix -p tcp --dport 25 -s <ip> -j DROP

actionunban = iptables -D f2b-npm-postfix -p tcp --dport 25 -s <ip> -j DROP

```
[Definition]
actionstart = iptables -N f2b-npm-postfix
         iptables -A f2b-npm-postfix -j RETURN
#         iptables -I FORWARD -p tcp -m multiport --dports 0:65535 -j f2b-npm-postfix
         iptables -I FORWARD -p tcp -m multiport --dport 25 -j f2b-npm-postfix

actionstop = iptables -D FORWARD -p tcp -m multiport --dport 25 -j f2b-npm-postfix
         iptables -F f2b-npm-postfix
         iptables -X f2b-npm-postfix

actioncheck = iptables -n -L FORWARD | grep -q 'f2b-npm-postfix[ \t]'

actionban = iptables -I f2b-npm-postfix -p tcp --dport 25 -s <ip>  -j DROP

actionunban = iptables -D f2b-npm-postfix -p tcp --dport 25 -s <ip>  -j DROP
```

FIGURE 17: Rule defined in actions for banning IP on port 25.

STEP 9: Define actions for handling container port 587. For this create a file in action directory and paste below contents: *vim /etc/fail2ban/action.d/dockeraction_postfixsasl.conf*
[Definition]
actionstart = iptables -N f2b-npm-postfix-sasl
iptables -A f2b-npm-postfix-sasl -j RETURN
 # iptables -I FORWARD -p tcp -m multiport --dports 0:65535 -j f2b-npm-postfix
iptables -I FORWARD -p tcp -m multiport --dport 587 -j f2b-npm-postfix-sasl

actionstop = iptables -D FORWARD -p tcp -m multiport --dport 587 -j f2b-npm-postfix-sasl
iptables -F f2b-npm-postfix-sasl
iptables -X f2b-npm-postfix-sasl

actioncheck = iptables -n -L FORWARD | grep -q 'f2b-npm-postfix-sasl[\t]'

actionban = iptables -I f2b-npm-postfix-sasl -p tcp --dport 587 -s <ip> -j DROP

actionunban = iptables -D f2b-npm-postfix-sasl -p tcp --dport 587 -s <ip> -j DROP

```
[Definition]
actionstart = iptables -N f2b-npm-postfix-sasl
         iptables -A f2b-npm-postfix-sasl -j RETURN
#         iptables -I FORWARD -p tcp -m multiport --dports 0:65535 -j f2b-npm-postfix
         iptables -I FORWARD -p tcp -m multiport --dport 587 -j f2b-npm-postfix-sasl

actionstop = iptables -D FORWARD -p tcp -m multiport --dport 587 -j f2b-npm-postfix-sasl
         iptables -F f2b-npm-postfix-sasl
         iptables -X f2b-npm-postfix-sasl

actioncheck = iptables -n -L FORWARD | grep -q 'f2b-npm-postfix-sasl[ \t]'

actionban = iptables -I f2b-npm-postfix-sasl -p tcp --dport 587 -s <ip>  -j DROP

actionunban = iptables -D f2b-npm-postfix-sasl -p tcp --dport 587 -s <ip>  -j DROP
```

FIGURE 18: Rule defined in actions for banning IP on port 587.

STEP 10: Restart fail2ban service to take config in action: *systemctl restart fail2ban*

STEP 11: Wait for 2 minutes to get all configuration activated, then execute command: fail2ban-client status

10 Evaluation

STEP 1*:* Verify ban status: fail2ban-client status postfix; fail2ban-client status dovecot
STEP 2: Now, attack is made in two ways: using telnet and using Hydra tool. Both ways are used to login twice with wrong credential at port 25 and 143. Then verified if IP is getting ban:

fail2ban-client status postfix

```
[root@tech ~]# fail2ban-client status postfix
Status for the jail: postfix
|- Filter
|  |- Currently failed: 0
|  |- Total failed:     2
|  `- File list:        /opt/container/log/maillog
`- Actions
   |- Currently banned: 1
   |- Total banned:     2
   `- Banned IP list:   192.168.1.15
```

FIGURE 19: Before brute-force attack through telnet.

```
Trying 192.168.1.18...
Connected to 192.168.1.18.
Escape character is '^]'.
220 mail.sample1.in ESMTP Postfix
ehlo mail
250-mail.sample1.in
250-PIPELINING
250-SIZE 20480000
250-VRFY
250-ETRN
250-AUTH PLAIN LOGIN
250-AUTH=PLAIN LOGIN
250-ENHANCEDSTATUSCODES
250-8BITMIME
250-DSN
250-SMTPUTF8
250 CHUNKING
^]
telnet> close
Connection closed.
```

FIGURE 20: Trying password attack twice through telnet utility.

telnet 192.168.1.18 143

➔ *a1 login test echo*

➔ *quit*

fail2ban-client status postfix

```
[root@tech ~]# fail2ban-client status postfix
Status for the jail: postfix
|- Filter
|  |- Currently failed: 0
|  |- Total failed:     4
|  `- File list:        /opt/container/log/maillog
`- Actions
   |- Currently banned: 2
   |- Total banned:     3
   `- Banned IP list:   192.168.1.15 192.168.1.2
```

FIGURE 21: Before brute-force attack through Hydra tool.

```
Status for the jail: dovecot
|- Filter
|  |- Currently failed: 0
|  |- Total failed:     0
|  `- File list:        /opt/container/log/maillog
`- Actions
   |- Currently banned: 0
   |- Total banned:     0
   `- Banned IP list:
```

FIGURE 22: Before brute-force attack through Hydra tool.

```
┌──(kkishore㉿tech)-[~]
└─$ hydra -l kkishore -p kkish123 -u -e sr -s 143  192.168.1.18 imap
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these ***** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-08-12 03:46:53
[INFO] several providers have implemented cracking protection, check with a small wordlist first - and stay legal!
[DATA] max 3 tasks per 1 server, overall 3 tasks, 3 login tries (l:1/p:3), ~1 try per task
[DATA] attacking imap://192.168.1.18:143/
1 of 1 target completed, 0 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-08-12 03:47:05

┌──(kkishore㉿tech)-[~]
└─$ hydra -l kkishore -p kkish123 -u -e sr -s 143  192.168.1.18 imap
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these ***** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-08-12 03:47:24
[INFO] several providers have implemented cracking protection, check with a small wordlist first - and stay legal!
[DATA] max 3 tasks per 1 server, overall 3 tasks, 3 login tries (l:1/p:3), ~1 try per task
[DATA] attacking imap://192.168.1.18:143/
1 of 1 target completed, 0 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-08-12 03:47:27

┌──(kkishore㉿tech)-[~]
```

FIGURE 23: Performing brute-force attack (IP = 192.168.1.21) through Hydra tool on port 143.

```
Status for the jail: dovecot
|- Filter
|  |- Currently failed: 0
|  |- Total failed:     6
|  `- File list:        /opt/container/log/maillog
`- Actions
   |- Currently banned: 1
   |- Total banned:     1
   `- Banned IP list:   192.168.1.21
```

FIGURE 24: IP of Kali Linux got banned while performing brute-force attack through Hydra tool on port 143.

STEP 3: Similarly, it will ban for port 25 and 587.

11 Outcome analysis

Integrating Fail2Ban, centralized authentication, and containerization led to major improvements in email server security and administration. Fail2Ban showed a significant decrease in successful brute-force attacks when compared to regular IP blacklisting. MariaDB's centralized authentication simplified user administration while also improving security. Containerization using Podman enabled isolation, scalability, and eased deployment.

While these combined techniques significantly reduced security concerns, difficulties such as improving container performance, guaranteeing smooth scaling, and adjusting to emerging threats still exist. Future research should focus on these areas, including the implementation of robust SSL/TLS encryption, in order to improve email server resilience.

References

Chauhan, N. and Morvan, A. L. (2024) *Podman.* Available at: https://docs.rockylinux.org/guides/containers/podman_guide/ [Accessed 11 August 2024].

EasyDMARC (2023) *How to configure dkim (opendkim) with postfix*. Available at: https://easydmarc.com/blog/how-to-configure-dkim-opendkim-with-postfix/ [Accessed 12 August 2024].

Ichioui, Y. (2021) *Abusing fail2ban misconfiguration to escalate privileges on linux*. Available at: https://youssef-ichioui.medium.com/abusing-fail2ban-misconfiguration-to-escalate-privileges-on-linux-826ad0cdafb7 [Accessed 12 August 2024].

Linode (2018) *Email with Postfix, Dovecot and mariadb on centos 7*. Available at: https://www.linode.com/docs/guides/email-with-postfix-dovecot-and-mariadb-on-centos-7/ [Accessed 11 August 2024].

Tonello, J. S. (2022) *Practical Linux DevOps: Building a Linux lab for modern software development.* Available at: https://learning.oreilly.com/library/view/practical-linux-devops/9781484283189/ [Accessed 10 August 2024].

Xiao, G. (2021) *Set up OpenDMARC with Postfix on centos/RHEL to block email spoofing*. Available at: https://www.linuxbabe.com/redhat/opendmarc-postfix-centos-rhel [Accessed 12 August 2024].