Raspberry Pi as mail server

What Is an Email Server?

An email server, also called a mail server, is essentially a computer system that sends and receives emails.

What Is the Purpose of an Email Server?

At its simplest, a mail server collects and distributes emails to their intended destination. It is a hardware that acts as an electronic post office for email, which allows you to control the transfer of emails within a network through different protocols. [1]

The mail server can also encrypt the transfer of emails, so others cannot have access to personal email information. Some mail servers can also provide additional security features against cybersecurity attacks

What are the benefits of using Raspberry Pi as an email server?

A Raspberry Pi email server is a powerful project. Because the Raspberry Pi runs Linux operating systems (OSes) ranging from Debian and Ubuntu to Arch and Manjaro, possible to install compatible email server software. The Pi itself is incredibly energy efficient so it's suitable for an always-on environment. And a small footprint means that Raspberry Pi possible to find anywhere. Finally, with the Pi Zero and Zero W clocking in at $5 USD, and a mere $35 for the Raspberry Pi 4 2GB RAM model, it's cost-effective. [2]

In order to host an email Server on Raspberry Pi the next steps are to be done:

1. Installing Postfix SMTP and Maildir on Raspbery Pi
2. Validating if SMTP server is working correctly by sending the first email from Raspberry Pi using Telnet
3. Ensure protection of email server from Malicious users and spam
4. Adding SASL authentication to an email server
5. Receiving first email and Setting up IMAP encryption

Installing Postfix SMTP and Maildir on Raspbery Pi

Postfix is a free and open-source mail transfer agent (MTA) that routes and delivers electronic mail.

- Postfix makes up to 34% of reachable mail –servers on the internet
- Postfix is an open source SMTP server(Simple Mail Transfer Protocol)

Commands used in order to install Postfix and Maildir:

```
File  Edit  Tabs  Help
pi@raspberrypi:~ $ sudo chown -R pi:pi ./Maildir
pi@raspberrypi:~ $ cd Maildir/
pi@raspberrypi:~/Maildir $ ls
cur  new  tmp
pi@raspberrypi:~/Maildir $ cd ..
pi@raspberrypi:~ $ sudo chmod -R 700 ./Maildir/
pi@raspberrypi:~ $ historie
bash: historie: command not found
pi@raspberrypi:~ $ history
    1  ssh pi
    2  dir
    3  sudo apt-get update
    4  sudo apt-get install postfix
    5  cd etc
    6  dir
    7  cd &etc
    8  cd /etc
    9  cd postfix/
   10  ls
   11  service postfix status
   12  sudo nano main.cf
   13  sudo apt-get install dovecot-common dovecot-imapd
   14  sudo maildirmake
   15  sudo apt-get install dovecot-common dovecot-imapd
   16  sudo maildirmake.dovecot /etc/skel/Maildir
   17  sudo maildirmake.dovecot /etc/skel/Maildir/.Drafts
   18  sudo maildirmake.dovecot /etc/skel/Maildir/.Sent
   19  sudo maildirmake.dovecot /etc/skel/Maildir/.Spam
   20  sudo maildirmake.dovecot /etc/skel/Maildir/.Trash
   21  sudo maildirmake.dovecot /etc/skel/Maildir/.Templates
   22  sudo cp -r /etc/skel/Maildir/ /home/pi/
   23  cd ~
   24  ls
   25  cd Maildir/
   26  sudo chown -R pi:pi ./Maildir
   27  cd Maildir/
   28  ls
   29  cd ..
   30  sudo chmod -R 700 ./Maildir/
   31  historie
   32  history
pi@raspberrypi:~ $
```

1. Update the distribution
2. Launch Install Postfix command
3. In the menu popped up choose **Internet Site**, enter your domain name
4. Go to /etc directory where postfix is stored
5. Run the service postfix status command in order to make sure if the postfix is actively running
6. Edit main.cf file and add

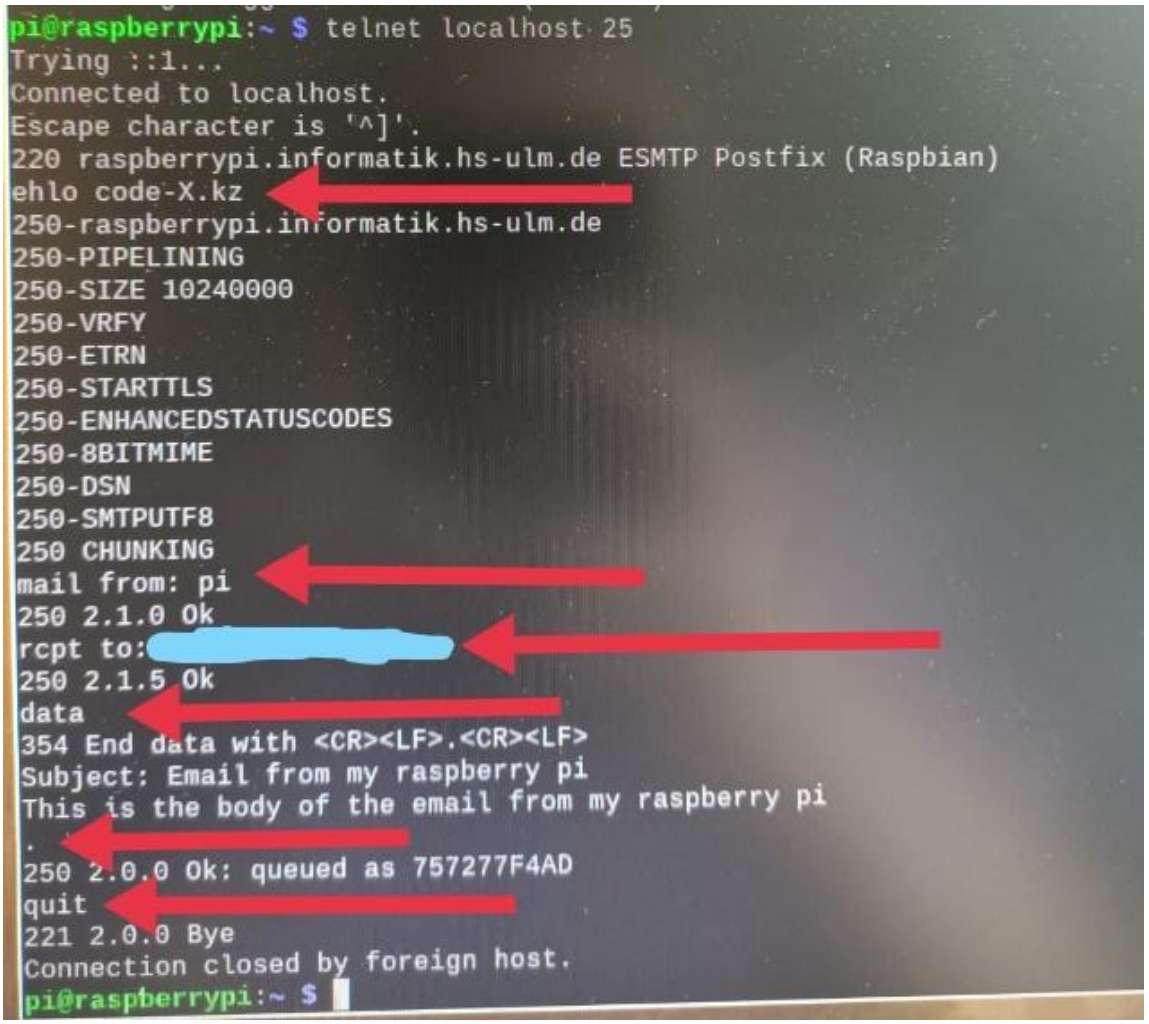
```
home_mailbox = Maildir/
mailbox_command =
```

7. In order to set up Maildir directory structure: **sudo apt-get install dovecot-common dovecot-imapd** command. Dovecot is an IMAP server, IMAP allows email clients to connect to an email server. It is basically provides an access including the security for that access to inbox, outbox, sent items and junk mail. Allows communication bvetween client and email server.
8. Create using maildirmake command Maildir, hidden Drafts, Sent, Spam, Trash, Templates

9. Copy directories to Raspberry Pi home directory
10. Provide ownership and navigate to the files
11. Change the permissions level
12. Restart Postfix

Validating if SMTP server is working correctly by sending the first email from Raspberry Pi using Telnet

1. Install Telnet- utility for sending and receiving communication traffic over TCP
2. Run **telnet localhost 25** command. SMTP server typically runs on port 25



3. Use the **EHLO** command to identify the domain name of the sending host to SMTP
4. **mail from: pi** command stating the senders name
5. **rcpt to: some@mail.com** stating the recipients email address
6. Enter **data** command in order to start data command series
7. Write the **Subject** and the **Main text** of the email
8. Put the **dot .** in order to end the content of the email
9. **Quit** command to exit Telnet interface

Ensure protection of email server from Malicious users and spam

Blocking users from maliciously using the server to send emails and setting up first level defense against incoming spam. Protection from someone trying to use this server to send an email from outside this network.

1. Use **nano** command in order to edit main.cf file and add:

```
smtpd_recipient_restrictions =
        permit_sasl_authenticated,
        permit_mynetworks,
        reject_unauth_destination
```

- Allow emails to be sent if the request is SASL authenticated.
- Second line means send if it is being sent from **mynetworks** (list of networks that are accepted by postfix, by default local networks).
- Third line means reject otherwise.

2. Imply restrictions to incoming emails called hello access restrictions, to prevent spammers getting their emails through to inbox. Use again **nano** command in order to edit main.cf file and add:

```
smtpd_helo_required = yes
smtpd_helo_restrictions =
        permit_mynetworks,
        permit_sasl_authenticated,
        reject_invalid_helo_hostname,
        reject_non_fqdn_helo_hostname,
        reject_unknown_helo_hostname,
        check_helo_access hash:/etc/postfix/helo_access
```

- Second line means send if it is being sent from mynetworks (list of networks that are accepted by postfix, by default local networks).
- Allow emails to be sent if the request is SASL authenticated.
- Reject if it does not come with hello hostname
- Reject if it does not come with fully qualified domain name
- Reject if the correct records are not detected on the DNS server
- Check for additional hello restriction to the list custom restrictions defined in the file **hello_access**, which is to be created

3. Use command **sudo /etc/postfix/helo_access** in order to create the file

```
GNU nano 3.2                                    /etc/postfix/helo_access

code-X.kz        REJECT Man, do not use my domain... --( ʊ ʊ)ᓂ--
mail.code-X.kz   REJECT Man, do not use my subdomain too... --( ʊ ʊ)ᓂ--
```

4. Map the file using command: **sudo postmap /etc/postfix/helo_access**
5. Restart postfix: **sudo service postfix restart**
6. Check the status: **sudo service postfix status**

Adding SASL authentication to an email server

By using **SASL** we are protecting from unauthenticated use of the server.

Authentication and Security Layer (SASL) is a framework for authentication and data security in Internet protocols. It decouples authentication mechanisms from application protocols, in theory allowing any authentication mechanism supported by SASL to be used in any application protocol that uses SASL. [3]

IMAP server is needed for that.

IMAP stands for **Internet Message Access Protocol**. It is a method of accessing email on a mail server; it is a mail access protocol. It means when the user wants to access a mail from the server; IMAP protocol is used. [4]

1. Use the command: **sudo nano ./conf.d/10-mail.conf** to change the mail_location

```
#
#mail_location = mbox:~/mail:INBOX=/var/mail/%u
mail_location = maildir:~/Maildir
```

2. Setting Postfix to use dovecot to handle the SASL authentication process. Use command: **sudo nano /etc/postfix/main.cf** to edit the **main.cf** file.

```
#added lines
smtpd_sasl_type = dovecot
smtpd_sasl_path = private/auth
smtpd_sasl_auth_enable = yes
```

3. Making dovecot to listening for SASL authentication requests coming from Postfix
   Use command: **sudo nano /etc/dovecot/conf.d/10-master.conf** to edit the **10-master.conf** file.

```
# Text addded
service auth {
        unix_listener /var/spool/postfix/private/auth {
                mode = 0660
                user = postfix
                group = postfix
        }
}
```

4. Enabling block of unencrypted authentication, by disabling plain text authentication.
   Use command: **sudo nano /etc/dovecot/conf.d/10-auth.conf** to edit the **10-auth.conf** file.

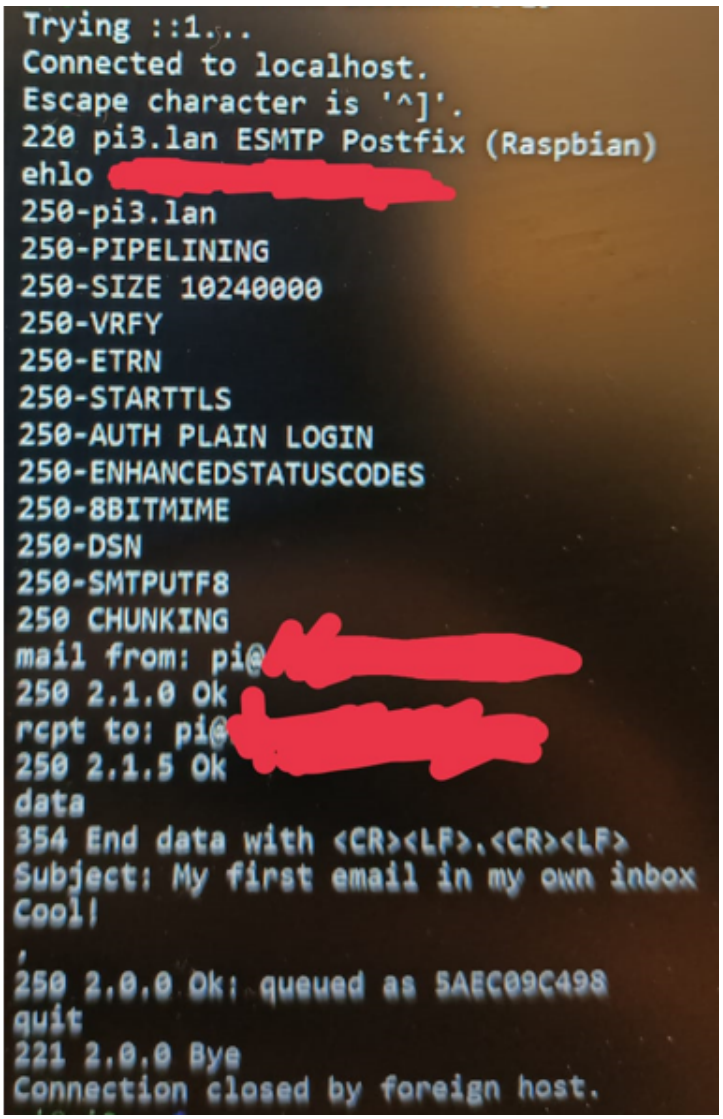
```
#Text added
auth_mechanisms = plain login
disable_plaintext_auth = yes
```

Receiving first email and Setting up IMAP encryption

Proving that the IMAP server is running correctly using Telnet tool.

Enabling **TLS** encryption on **IMAP** connection, so that communication between **Postfix email server** and **Email Client (Outlook)** can be encrypted.

1. Check the status of Dovecot by using command: **sudo service dovecot status**
2. Check the status of Postfixby using command: **sudo service postfix status**
3. Checking IMAP service using Telnet by sending an email from this email server to the same email server. Command **telnet localhost 25** is used.



4. Connect to inbox to check the email on **Port 143.**
   - typed **a "username" [password]**

```
quit
221 2.0.0 Bye
Connection closed by foreign host.
pi@raspberrypi:~ $ telnet localhost 143
Trying ::1...
Connected to localhost.
Escape character is '^]'.
* OK [CAPABILITY IMAP4rev1 SASL-IR LOGIN-REFERRALS ID ENABLE IDLE LITERAL+ STARTTLS AUTH=PLAIN AUTH=LOGIN]
 Dovecot (Raspbian) ready.
a login "pi" "[redacted]
a OK [CAPABILITY IMAP4rev1 SASL-IR LOGIN-REFERRALS ID ENABLE IDLE SORT SORT=DISPLAY THREAD=REFERENCES THRE
AD=REFS THREAD=ORDEREDSUBJECT MULTIAPPEND URL-PARTIAL CATENATE UNSELECT CHILDREN NAMESPACE UIDPLUS LIST-EX
TENDED I18NLEVEL=1 CONDSTORE QRESYNC ESEARCH ESORT SEARCHRES WITHIN CONTEXT=SEARCH LIST-STATUS BINARY MOVE
 SNIPPET=FUZZY LITERAL+ NOTIFY SPECIAL-USE] Logged in
b select "inbox"
* FLAGS (\Answered \Flagged \Deleted \Seen \Draft)
* OK [PERMANENTFLAGS (\Answered \Flagged \Deleted \Seen \Draft \*)] Flags permitted.
* 1 EXISTS
* 1 RECENT
* OK [UNSEEN 1] First unseen.
* OK [UIDVALIDITY 1654764586] UIDs valid
* OK [UIDNEXT 2] Predicted next UID
b OK [READ-WRITE] Select completed (0.095 + 0.000 + 0.094 secs).
c logout
```

- Successful login message is shown
- Entered command: **b select inbox**
- **1 EXISTS** and **1 RECENT** is shown, which means the email was sent successfully. IMAP is allowing to access the inbox.dddddd
- **c logout command** in order to logout.

5. Encrypting the IMAP connection to email server. Command: **sudo nano /etc/dovecot/conf.d/10-master.conf** in order to edit **10-master.conf** file. Set configurations.

```
# Internal user is used by unprivile
# login user, so that login processe
#default_internal_user = dovecot

service imap-login {
  inet_listener imap {
    #Uncommented that line
    port = 143
  }
}
```

6. Making SSL a required feature in SSL configuration file in dovecot.
   Command: **sudo nano /etc/dovecot/conf.d/10-ssl.conf** in order to edit **10-ssl.conf** file. Set **SSL = required**.

```
# SSL/TLS support: yes, no, required.
ssl = required
```

References

1. https://www.techtarget.com/whatis/definition/SMTP-Simple-Mail-Transfer-Protocol

   last access: 06-12-2022

2. https://www.electromaker.io/tutorial/blog/how-to-build-a-raspberry-pi-email-server

   last access: 06-12-2022

3. https://en.wikipedia.org/wiki/Simple_Authentication_and_Security_Layer

   last access: 06-12-2022

4. https://www.techtarget.com/whatis/definition/IMAP-Internet-Message-Access-Protocol#:~:text=Rahul%20Awati-,What%20is%20IMAP%20(Internet%20Message%20Access%20Protocol)%3F,on%20their%20device(s)

   last access: 06-12-2022