Email Architecture

Overview

Email is one of the oldest still recognisable parts of the internet which is still used today. Commmon email providers such as outlook and gmail give very little options on metadata collection and privacy, so how do you create your own?

Components of Email

There are three parts of the email service:

* Mail User Agent (MUA) - Email client for sending emails, connects to email server to send emails, applications such as Outlook and Thunderbird
* Mail Transfer Agent (MTA) - Core application which transmits email between servers, applications such as Exim, sendmail, Postfix, qmail.
* Mail Delivery Agent (MDA) - Gets messages from email server (MTA) into users inboxes (MUA) using protocols such as POP or IMAP, applications such as Dovecot

There are also several other components for an email server to be setup:

* Operating system (OS) - OS with fast internet connection which does not have its email ports blocked or IP address blacklisted for spam.
* Spam filter - Keep spam out of inbox, applications such as SpamAssassin
* Virus filter - Keep viruses out of inbox, applications such as ClamAv
* Mail filters - General mail filters, applications such as Sieve
* Webmail application - Access email from the internet GUI, applications such as Roundcube
* Database - Store emails in webmail, applications such as PostgreSQL
* Webserver - Serve roundcube over the web, Nginx

DNS

For emails to be routed to the correct ip address, an MX dns record will be required. Create a MX and A dns record for the domain name which the email server should end in:

<email-domain> MX <A record domain>

<A record domain> A <IP address>

MTA - Postfix - Setup

Postfix is a very popular MTA and is very configurable and easy to use. Using Ubuntu, install Postfix:

sudo apt-get update

sudo DEBIAN\_PRIORITY=low apt-get install postfix

DEBIAN\_PRIORITY=low will give a setup GUI to answer the following:

* Internet Site - Mail is sent a received directly
* Domain name - registered domain name
* Postmaster mail recipient - emails sent to 'root@domain' and 'postmaster@domain' will get sent to this unix user account
* Other mail destinations - Other domains which the server is the final destination for
* Synchronous updates - 'no', mail is processed slower as the system will only process commands synchronously
* Mail relay - Use only localhost or spam bots will send mail to other addresses
* Mailbox size - 0 for unlimited
* Local address extension: leave default
* IP protocols - select all for compatibility

This will then give the default setup for postfix on the system.

The postfix configuration file can be found at /etc/postfix/main.cf, however configuration settings can be edited directly using the command:

sudo postconf -e '<key=value>'

MTA - Postfix - SSL Cerificates

Create signed SSL certificate for the mail server using LetEncrypt certbot (install notes separate):

sudo certbot certonly --standalone -d mail.<your-domain>

the cert will then be saved to:

/etc/letsencrypt/live/mail.<your-domain>/fullchain.pem

/etc/letsencrypt/live/mail.<your-domain>/priveky.pem

Add them to the postfix configuration using:

sudo postconf -e 'smtpd\_tls\_cert\_file = /etc/letsencrypt/live/mail.<your.domain>/fullchain.pem'

sudo postconf -e 'smtpd\_tls\_key\_file = /etc/letsencrypt/live/mail.<your.domain>/privkey.pem'

MTA - Postfix - SMTP Authentication

Add STMP Authentication with Dovecot to postfix's configuration to allow clients to identify themselves securely:

sudo postconf -e 'smtpd\_sasl\_local\_domain = mail.<your-domain>'

sudo postconf -e 'smtpd\_sasl\_auth\_enable = yes'

sudo postconf -e 'smtpd\_sasl\_security\_options = noanonymous'

sudo postconf -e 'smtpd\_sasl\_type = dovecot'

sudo postconf -e 'smtpd\_sasl\_path = private/auth'

sudo postconf -e 'smtpd\_use\_tls = yes'

sudo postconf -e 'smtpd\_tls\_auth\_only = no'

sudo postconf -e 'smtpd\_tls\_session\_cache\_timeout = 3600s'

sudo postconf -e 'smtpd\_tls\_loglevel = 1'

sudo postconf -e 'smtpd\_tls\_received\_header = yes'

sudo postconf -e 'smtpd\_tls\_security\_level = may'

sudo postconf -e 'smtpd\_recipient\_restrictions = permit\_sasl\_authenticated,permit\_mynetworks,reject\_unauth\_destination'

sudo postconf -e 'smtp\_tls\_note\_starttls\_offer = yes'

sudo postconf -e 'smtp\_tls\_security\_level = may'

sudo postconf -e 'tls\_random\_source = dev:/dev/urandom'

sudo postconf -e 'inet\_interfaces = all'

MTA - Postfix - Virtual Alias Maping

With virtual alias domains, each hosted email address can be aliased (mapped) to a local UNIX system account or remote address. Enable mapping with:

sudo postconf -e 'virtual\_alias\_domains = $mydomain'

sudo postconf -e 'virtual\_alias\_maps = hash:/etc/postfix/virtual'

Then create the mapping file:

sudo vim /etc/postfix/virtual

and mappings can be entered simply as the following on each line:

<email-address> <unix-user>

for example some standard mappings would be:

postmaster@mail.example.com root

root@mail.example.com root

info@mail.exampe.com info

Then apply the mappings:

sudo postmap /etc/postfix/virtual

sudo systemctl restart postfix

MDA - Dovecot - Install

Dovecot is a delivery agent which works very well with Postfix. Install dovecot SASL with the common package and then the dovecot plugins for IMAP and SMTP to give mail delivery protocols for clients such as Outlook and Rails:

sudo apt install dovecot-common dovecot-imapd dovecot-pop3d

Dovecots configurations files will then be located as /etc/dovcot/conf.d/.

MDA - Dovecot - Mailboxes

Add the mailbox location to postix for the logged in user in the home directory:

sudo postconf -e 'home\_mailbox = Maildir/'

Then add default mailbox setup for any new users which get added:

sudo maildirmake.dovecot /etc/skel/Maildir

sudo maildirmake.dovecot /etc/skel/Maildir/.Drafts

sudo maildirmake.dovecot /etc/skel/Maildir/.Sent

sudo maildirmake.dovecot /etc/skel/Maildir/.Trash

sudo maildirmake.dovecot /etc/skel/Maildir/.Templates

Add default mailbox to admin user:

sudo cp -r /etc/skel/Maildir /home/$USER/

sudo chown -R $USER:$USER /home/$USER/Maildir

sudo chmod -R 700 /home/$USER/Maildir

sudo adduser $USER mail

Include mailbox Maildir location in the terminal and mail profiles:

echo 'export MAIL=~/Maildir' | sudo tee -a /etc/bash.bashrc | sudo tee -a /etc/profile.d/mail.sh

Make the mail delivery setting for Dovecot match that of Postfix:

sudo vim /etc/dovecot/conf.d/10-mail.conf

mail\_location = maildir:~/Maildir

MDA - Dovecot - SASL

To enable SASL, disable plaintext auth and enable auth login in the auth.conf file:

sudo vim /etc/dovecot/conf.d/10-auth.conf

disable\_plaintext\_auth = yes

auth\_mechanisms = plain login

Add the SSL files for auth to the dovecot ssl config:

sudo vim /etc/dovecot/conf.d/10-ssl.conf

# SSL/TLS support: yes, no, required. <doc/wiki/SSL.txt>

ssl = required

...

ssl\_cert = </etc/letsencrypt/live/mail.<your-domain>/fullchain.pem

ssl\_key = </etc/letsencrypt/live/mail.<your-domain>/privkey.pem

...

# SSL protocols to use

ssl\_protocols = !SSLv3

MDA - Dovecot - POP3 and IMAP

Edit the master configuration to add the POP3 and IMAP protocols so that clients can access and send emails:

sudo vim /etc/dovecot/conf.d/10-master.conf

Uncomment and edit the following lines:

service imap-login {

inet\_listener imap {

port = 143

}

}

service pop3-login {

inet\_listener pop3 {

port = 110

}

}

service auth {

# Postfix smtp-auth

unix\_listener /var/spool/postfix/private/auth {

mode = 0660

user = postfix

group = postfix

}

Once all settings are set, check the dovecot configuration and restart the service using:

dovecot -n

sudo systemctl restart dovecot

The system should now be ready for local testing.

Local Testing

Load up telnet:

telnet localhost 25

create new message:

ehlo mail.<your-domain>

mail from: root@mail.<domain>

rcpt to: info@mail.<domain>

data

<add message>

And then send the message with:

.

The message should now in the /home/info/Maildir/new inbox and can be read using the less command.

Additional Security

Reject bad connections by requiring valid HELO and EHLO commands with a fully qualified domain name:

sudo postconf -e 'smtpd\_helo\_required = yes'

sudo postconf -e 'smtpd\_helo\_restrictions = reject\_non\_fqdn\_helo\_hostname,reject\_invalid\_helo\_hostname,reject\_unknown\_helo\_hostname'

Disable VRFY command, which allows postfix to let anyone determine if an account exists on the mailserver. Having this setting enabled allows hackers to easily target account which they know exist:

sudo postconf -e 'disable\_vrfy\_command = yes'

sudo postconf -e 'smtpd\_delay\_reject = yes'

Reject connections from made up address or those which do not exist, also add spam filters such as spamhaus:

sudo postconf -e 'smtpd\_recipient\_restrictions = permit\_sasl\_authenticated,permit\_mynetworks,reject\_unauth\_destination,reject\_invalid\_hostname,reject\_non\_fqdn\_hostname,reject\_non\_fqdn\_sender,reject\_non\_fqdn\_recipient,reject\_unknown\_sender\_domain,reject\_rbl\_client sbl.spamhaus.org,reject\_rbl\_client cbl.abuseat.org'

Redirect spam mail to dev/null using a postfix catchall:

sudo vim /etc/postfix/virtual

@mail.<your-domain> nobody

sudo vim /etc/aliases

postmaster:root

nobody: /dev/null

sudo postmap /etc/postfix/virtual

sudo postalias /etc/aliases

sudo systemctl restart postfix

Live Testing

Open up the firewall ports, on the firewall to allow outside connections to the email server:

* 25 (SMTP)
* 465 (SMTP Secure)
* 110 (POP3)
* 995 (POP3 Secure)
* 143 (IMAP)
* 993 (IMAP Secure)

Then test the server using SMTP Diagnostics online:

https://mxtoolbox.com/diagnostic.aspx

AWS and Port 25

AWS rate limits outgoing port 25 connections to stop spam. Therefore, to send mail via postfix contact amazon to remove rate limits on the port:

https://console.aws.amazon.com/support/contacts?#/rdns-limits

Alternatively use a relay node to send the mail externally. A popular relay node to use is google's gmail server:

In postfix's main configuration file:

sudo vim /etc/postfix/main.cf

# This tells Postfix to hand off all messages to Gmail, and never do direct delivery.

relayhost = [smtp.gmail.com]:587

# This enables TLS (SMTPS) certificate verification, because Gmail has a valid one.

smtp\_tls\_security\_level = verify

smtp\_tls\_CAfile = /etc/ssl/certs/ca-certificates.crt

smtp\_tls\_session\_cache\_database = btree:/var/run/smtp\_tls\_session\_cache

# This tells Postfix to provide the username/password when Gmail asks for one.

smtp\_sasl\_auth\_enable = yes

smtp\_sasl\_password\_maps = hash:/etc/postfix/sasl\_passwd

smtp\_sasl\_security\_options = noanonymous

In the postfix SASL config:

sudo vim /etc/postfix/sasl\_passwd

[smtp.gmail.com]:587 user@gmail.com:mypassword

Compile the SASL password to the database:

sudo postmap /etc/postfix/sasl\_passwd

Restart postfix:

sudo systemctl restart postfix