Assignment

Refer to https://www.digitalocean.com/community/tutorials/how-to-install-and-configure-postfix-as-a-send-only-smtp-server-on-ubuntu-20-04 to learn how to deploy Postfix.

For system configuration , use **<your vm hostname>.bc.local** as the system mail name. You can SKIP **Step 2** completely, just do Step 1 and move straight to Step 3. It. Just. Works.

Send yourself email, use your student email address.

If it doesn't work, let's do something about it!

Once it works, take a screencap of your success from your Outlook!

Prerequisites

Setting up and maintaining your own mail server is complicated and time-consuming. For most users, it's more practical to instead rely on a paid mail service. If you're considering running your own mail server, we encourage you to review this article on why you may not want to do so.

If you're sure you want to follow this guide to install and configure Postfix, then you must first have the following:

One Ubuntu 20.04 server set up with the Initial Server Setup with Ubuntu 20.04, including creating a sudo non-root user.

A fully registered domain name. This tutorial will use your_domain throughout. You can purchase a domain name on Namecheap, get one for free on Freenom, or use the domain registrar of your choice.

An A DNS record with your_domain pointing to your server's public IP address. You can follow this introduction to DigitalOcean DNS for details on how to add them.

Note: Your server's hostname and your Droplet's name must match your_domain, because DigitalOcean automatically sets PTR records for the Droplet's IP address according to its name.

You can verify the server's hostname by typing hostname at the command prompt. The output should match the name you gave the Droplet when it was being created.

Step 1 — Installing Postfix

In this step, you'll install Postfix. The fastest way is to install the mailutils package, which bundles Postfix with a few supplementary programs that you'll use to test sending email.

First, update the package database:

```
sudo apt update
```

Then, install Postfix by running the following command:

```
sudo apt install mailutils
```

Near the end of the installation process, you will be presented with the Postfix configuration window:

Select Internet Site from the menu, then press TAB to select <Ok>, then ENTER

The default option is Internet Site. That's the recommended option for your use case, so press TAB, and then ENTER. If you only see the description text, press TAB to select OK, then ENTER.

If it does not show up automatically, run the following command to start it:

```
sudo dpkg-reconfigure postfix
```

After that, you'll get another configuration prompt regarding the System mail name:

Enter your domain name, then press TAB to select <Ok>, ENTER

The System mail name must be the same as the name you assigned to your server when you were creating it. When you've finished, press TAB, followed by ENTER.

You have now installed Postfix and are ready to start configuring it.

Step 2 — Configuring Postfix

In this step, you'll configure Postfix to send and receive emails only from the server on which it is running—that is, from localhost.

For that to happen, you need to configure Postfix to listen only on the loopback interface, the virtual network interface that the server uses to communicate internally. To make the changes, you'll need to edit the main Postfix configuration file called main.cf, stored under etc/postfix.

Open it for editing using your favorite text editor:

sudo nano /etc/postfix/main.cf

Find the following lines:

/etc/postfix/main.cf

. . .

mailbox_size_limit = 0

recipient_delimiter = +

inet_interfaces = all

. . .

Set the value of the inet_interfaces setting to loopback-only:

/etc/postfix/main.cf

. . .

mailbox_size_limit = 0

recipient_delimiter = +

inet_interfaces = loopback-only

. . .

Another directive you'll need to modify is mydestination, which specifies the list of domains that are delivered via the local_transport mail delivery transport. By default, the values are similar to these:

/etc/postfix/main.cf

. . .

mydestination = $myhostname, your_domain, localhost.com, , localhost

. . .

Change the line to look like this:

/etc/postfix/main.cf

. . .

mydestination = localhost.$mydomain, localhost, $myhostname

. . .

If your domain is actually a subdomain and you want the email messages to look as if they were sent from the main domain, you can add the following line to the end of main.cf:

/etc/postfix/main.cf

...

masquerade_domains = your_main_domain

The optional masquerade_domains setting specifies the domains for which the subdomain will be stripped off in the email address.

When you are done, save and close the file.

Note: If you're hosting multiple domains on a single server, the other domains can also be passed to Postfix using the mydestination directive.

Then, restart Postfix by running the following command:

```
sudo systemctl restart postfix
```

You've configured Postfix to only send emails from your server. You'll now test it by sending an example message to an email address.

Step 3 — Testing the SMTP Server

In this step, you'll test whether Postfix can send emails to an external email account using the mail command, which is part of the mailutils package that you installed in the first step.

To send a test email, run the following command:

```
echo "This is the body of the email" | mail -s "This is the subject line"
your_email_address
```

You can change the body and the subject of the email to your liking. Remember to replace your_email_address with a valid email address that you can access.

Now, check the email address to which you sent this message. You should see the message in your inbox. If it's not there, check your spam folder. At this point, all emails you send are unencrypted, which makes service providers think it's likely spam. You'll set up encryption later, in step 5.

If you receive an error from the mail command, or you haven't received a message after a prolonged period of time, check that the Postfix configuration you edited is valid and that your server's name and hostname are set to your domain.

Note that with this configuration, the address in the From field for the test emails you send will be in the form of your_user_name@your_domain, where your_user_name is the username of the server user you ran the command as.

You have now sent an email from your server and verified that it's successfully received. In the next step, you'll set up email forwarding for root.

Step 4 — Forwarding System Mail

In this step, you'll set up email forwarding for user root, so that system-generated messages sent to it on your server get forwarded to an external email address.

The /etc/aliases file contains a list of alternate names for email recipients. Open it for editing:

sudo nano /etc/aliases

In its default state, it looks like this:

/etc/aliases

See man 5 aliases for format

postmaster: root

The only directive present specifies that system-generated emails are sent to root.

Add the following line to the end of the file:

/etc/aliases

...

root: your_email_address

With this line, you specify that emails sent to root end up being forwarded to an email address. Remember to replace your_email_address with your personal email address. When you are done, save and close the file.

For the change to take effect, run the following command:

sudo newaliases

Running newaliases will build up a database of aliases that the mail command uses, which are taken from the config file you just edited.

Test that sending emails to root works by running:

echo "This is the body of the email" | mail -s "This is the subject line" root

You should receive the email at your email address. If it's not there, check your spam folder.

In this step, you set up forwarding system-generated messages to your email address. You'll now enable message encryption, so that all emails your server sends are immune to tampering in transit and will be viewed as more legitimate.

Step 5 — Enabling SMTP Encryption

You'll now enable SMTP encryption by requesting a free TLS certificate from Let's Encrypt for your domain (using Certbot) and configuring Postfix to use it when sending messages.

Ubuntu includes Certbot in their default package repositories, so you can install it by running the following command:

sudo apt install certbot

When asked for confirmation, type Y and press ENTER.

As part of the initial server setup in the prerequisites, you installed ufw, the uncomplicated firewall. You'll need to configure it to allow the HTTP port 80, so that domain verification can be completed. Run the following command to enable it:

```
sudo ufw allow 80
```

The output will look like this:

```
Output
Rule added
Rule added (v6)
```

Now that the port is open, run Certbot to get a certificate:

```
sudo certbot certonly --standalone --rsa-key-size 4096 --agree-tos --preferred-challenges http -d your_domain
```

This command orders Certbot to issue certificates with an RSA key size of 4096 bits, to run a temporary standalone web server (--standalone) for verification, and to check via port 80 (--preferred-challenges http). Remember to replace your_domain with your domain before running the command, and enter your email address when prompted.

The output will be similar to this:

```
Output
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Plugins selected: Authenticator standalone, Installer None
Obtaining a new certificate
Performing the following challenges:
http-01 challenge for `your_domain`
Waiting for verification...
Cleaning up challenges
```

IMPORTANT NOTES:

- Congratulations! Your certificate and chain have been saved at:

/etc/letsencrypt/live/your_domain/fullchain.pem

Your key file has been saved at:

/etc/letsencrypt/live/your_domain/privkey.pem

Your cert will expire on 2020-07-11. To obtain a new or tweaked

version of this certificate in the future, simply run certbot

again. To non-interactively renew *all* of your certificates, run

"certbot renew"

- If you like Certbot, please consider supporting our work by:

Donating to ISRG / Let's Encrypt: https://letsencrypt.org/donate

Donating to EFF: https://eff.org/donate-le

As written in the notes, your certificate and private key file were saved under /etc/letsencrypt/live/your_domain.

Now that you have your certificate, open main.cf for editing:

sudo nano /etc/postfix/main.cf

Find the following section:

/etc/postfix/main.cf

TLS parameters

smtpd_tls_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem

smtpd_tls_key_file=/etc/ssl/private/ssl-cert-snakeoil.key

smtpd_tls_security_level=may

smtp_tls_CApath=/etc/ssl/certs

smtp_tls_security_level=may

smtp_tls_session_cache_database = btree:${data_directory}/smtp_scache

Modify it to look like this, replacing your_domain with your domain where necessary. This will update your TLS settings for Postfix:

/etc/postfix/main.cf

TLS parameters

smtpd_tls_cert_file=/etc/letsencrypt/live/your_domain/fullchain.pem

smtpd_tls_key_file=/etc/letsencrypt/live/your_domain/privkey.pem

smtpd_tls_security_level=may

smtp_tls_CApath=/etc/ssl/certs

smtp_tls_security_level=may

smtp_tls_session_cache_database = btree:${data_directory}/smtp_scache

Once you're done, save and close the file.

Apply the changes by restarting Postfix:

sudo systemctl restart postfix

Now, try sending an email again:

echo "This is the body of an encrypted email" | mail -s "This is the subject line" your_email_address

Then, check the email address you provided. It's possible that you'll see the message in your inbox immediately, because email providers are much more likely to mark unencrypted messages as spam.

You can check the technical info about the email message in your client to see that the message is indeed encrypted.

Solution