How To Set Up a Postfix Email Server with Dovecot: Dynamic Maildirs and LMTP

Authored by: ***ASPHostServer Administrator*** *[asphostserver@gmail.com]*
Saved From: http://faq.asphosthelpdesk.com/article.php?id=231

Preface

This tutorial is based on How To Set Up a Postfix E-Mail Server with Dovecot and picks up where the first part ended.

Please go through that tutorial first.

In this article, we will divorce mailboxes from system accounts using dovecot"s LMTP server as delivery mechanism, as well as use postgresql to keep user records.

No more mail will be delivered to the standard linux mailboxes.

Like the first guide, this tutorial is based on Debian 7 wheezy, Postfix 2.9, and dovecot 2.1 (+ Postgresql 9.1).

Packages

Install postgresql:

```
# aptitude install postgresql postfix-pgsql
```

dovecot in version 2.1 should already come with pgsql enabled. If you"re on a system where dovecot is modularised, run

```
# aptitude install dovecot-lmtpd dovecot-pgsql
```

to install the needed modules.

Postgres Database Setup

Adjust this for your needs if you already have a postgres setup running! But from a fresh postgres install, let"s set up authentication so that we can give dovecot access to the database. Add the following to `/etc/postgresql/9.1/main/pg_ident.conf`:

```
mailmap         dovecot                 mailreader
mailmap         postfix                 mailreader
mailmap         root                    mailreader
```

And the following to `/etc/postgresql/9.1/main/pg_hba.conf` (**Warning: Make sure to add it right after the `Put your actual configuration here` comment block! Otherwise one of the default entries might catch first and the databse authentication will fail.)**

```
local       mail    all     peer map=mailmap
```

Then reload postgresql (`service postgresql reload`). Now set up the database:

```
# sudo -u postgres psql
postgres=# CREATE USER mailreader;
postgres=# REVOKE CREATE ON SCHEMA public FROM PUBLIC;
postgres=# REVOKE USAGE ON SCHEMA public FROM PUBLIC;
postgres=# GRANT CREATE ON SCHEMA public TO postgres;
postgres=# GRANT USAGE ON SCHEMA public TO postgres;
postgres=# CREATE DATABASE mail WITH OWNER mailreader;
postgres=# \q
# sudo psql -U mailreader -d mail
postgres=# \c mail
mail=# CREATE TABLE aliases (
    alias text NOT NULL,
    email text NOT NULL
);
mail=# CREATE TABLE users (
    email text NOT NULL,
    password text NOT NULL,
    maildir text NOT NULL,
    created timestamp with time zone DEFAULT now()
);
mail=# ALTER TABLE aliases OWNER TO mailreader;
mail=# ALTER TABLE users OWNER TO mailreader;
mail=# \q
```

You can then add virtual mailboxes like this, starting from a rootshell:

```
# doveadm pw -s sha512 -r 100
Enter new password: ...
Retype new password: ...
{SHA512}.....................................................==
# psql -U mailreader -d mail
mail=# INSERT INTO users (
    email,
    password,
    maildir
) VALUES (
    'foo@yourdomain.tld',
    '{SHA512}.....................................................==',
    'foo/'
);
```

Administration Interface (optional)

If you don"t want to use the command line interface to maintain the mail database, you can set up an administration interface. Let"s first add a database user that is only allowed to edit the mail databse. Go back to `/etc/postgresql/9.1/main/pg_hba.conf` and add this line just under the peer authentication line you added earlier:

```
host pgadmin          mail            127.0.0.1/32            md5
```


PDF Generated by PHPKB Knowledge Base Script

This will allow local socket connections on port 5432. (The default port for postgres)

Add the database user:

```
# sudo -u postgres psql
postgres=# CREATE USER pgadmin WITH PASSWORD 'new password';
postgres=# \q
```

Give the user the privileges to edit the mail database:

```
# sudo psql -U mailreader -d mail
mail=> GRANT SELECT, UPDATE, INSERT, DELETE ON users TO pgadmin;
mail=> GRANT SELECT, UPDATE, INSERT, DELETE ON aliases TO pgadmin;
mail=> \q
```

You can now use an administration interface like pgAdmin that is capable of using SSH tunneling to directly connect to the database, or you could set up something like phpPgAdmin.

Dovecot Setup

We need to connect dovecot to the database and set up the LMTP server. Set up a new user (dovecot will refuse to handle mail without a system user set up for it) and directory for maildirs first: (you could use **/var/mail**, but that traditionally uses the mbox format, while we"re going to be using the superior maildir format).

```
# adduser --system --no-create-home --uid 500 --group --disabled-password
--disabled-login --gecos 'dovecot virtual mail user' vmail
# mkdir /home/mailboxes
# chown vmail:vmail /home/mailboxes
# chmod 700 /home/mailboxes
```

Now save the following configuration as `/etc/dovecot/dovecot-sql.conf`:

```
driver = pgsql
connect = host=/var/run/postgresql/ dbname=mail user=mailreader
default_pass_scheme = SHA512
password_query = SELECT email as user, password FROM users WHERE email = '%u'
user_query = SELECT email as user, 'maildir:/home/mailboxes/maildir/'||maildir
as mail, '/home/mailboxes/home/'||maildir as home, 500 as uid, 500 as gid FROM
users WHERE email = '%u'
```

Make sure it is owned by root and chmodded 600.

Now open `/etc/dovecot/dovecot.conf` and edit the `passdb` and `userdb` settings to look like this:

```
userdb {
  driver = prefetch
}
passdb {
  args = /etc/dovecot/dovecot-sql.conf
  driver = sql
}
```


PDF Generated by PHPKB Knowledge Base Script

Change the protocols stanza to

```
protocols = imap lmtp
```

and add the lmtp service socket and some lmtp protocol settings:

```
service lmtp {
    unix_listener /var/spool/postfix/private/dovecot-lmtp {
    group = postfix
    mode = 0600
    user = postfix
    }
}
protocol lmtp {
    postmaster_address=postmaster@yourdomain.com
hostname=mail.yourdomain.com}
```

The mail_location stanza is now superfluous and can be removed.

Postfix

We now need to tell postfix to deliver mail directly to dovecot. Open `/etc/postfix/main.cf` and add

```
mailbox_transport = lmtp:unix:private/dovecot-lmtp
```

to the end. Now we need to set the database config for postfix.

Create the file `/etc/postfix/pgsql-aliases.cf` and enter:

```
user=mailreader
dbname=mail
table=aliases
select_field=alias
where_field=email
hosts=unix:/var/run/postgresql
```

Then create the file `/etc/postfix/pgsql-boxes.cf` and enter:

```
user=mailreader
dbname=mail
table=users
select_field=email
where_field=email
hosts=unix:/var/run/postgresql/
```

Now amend the alias_maps line in main.cf to read

```
alias_maps = hash:/etc/aliases proxy:pgsql:/etc/postfix/pgsql-aliases.cf
```

and the local_recipient_maps line to read

```
local_recipient_maps = proxy:pgsql:/etc/postfix/pgsql-boxes.cf $alias_maps
```


PDF Generated by PHPKB Knowledge Base Script

In whole, your main.cf should look similar to this:

```
myhostname = mail.mydomain.com
myorigin = mydomain.com
mydestination = mydomain.com, mail.mydomain.com, localhost,
localhost.localdomain
relayhost =
mynetworks = 127.0.0.0/8 [::ffff:127.0.0.0]/104 [::1]/128
mailbox_size_limit = 0
recipient_delimiter = +
inet_interfaces = all
alias_maps = hash:/etc/aliases proxy:pgsql:/etc/postfix/pgsql-aliases.cf
local_recipient_maps = proxy:pgsql:/etc/postfix/pgsql-boxes.cf $alias_maps
mailbox_transport = lmtp:unix:private/dovecot-lmtp
smtpd_tls_cert_file=/etc/ssl/certs/mailcert.pem
smtpd_tls_key_file=/etc/ssl/private/mail.key
smtpd_use_tls=yes
smtpd_tls_session_cache_database = btree:${data_directory}/smtpd_scache
smtp_tls_session_cache_database = btree:${data_directory}/smtp_scache
smtpd_tls_security_level=may
smtpd_tls_protocols = !SSLv2, !SSLv3
```

Finishing Up

Now simply reload:

```
# postfix reload

# service dovecot restart
```

And you"re set! Test your setup as you did after the first article, and make sure mail to `postmaster@yourdomain.com` finds its way into an attended mailbox!

PDF Generated by PHPKB Knowledge Base Script