

UNIVERSIDAD MAYOR, REAL Y PONTIFICIA DE SAN FRANCISCO XAVIER DE CHUQUISACA

FACULTAD DE CIENCIAS Y TECNOLOGÍA

INGENIERÍA EN TELECOMUNICACIONES



TRABAJO FINAL

UNIVERSITARIOS: Mogro Castillo Shanon Nicole

Rojas Pareja Fabián

DOCENTE: Ing. Quispe Ortega Lucio Marcelo

MATERIA: Trabajando en la Nube

TÍTULO: Implementación de servidor de correo electrónico en la nube

Sucre - Bolivia
2025

1. Introducción

Los servicios establecidos en la nube ofrecen grandes ventajas al momento de querer establecer una topología simplificada y de recursos variables, ese es el caso de la implementación de un servidor de correo electrónico en la nube, ofreciendo seguridad y simplicidad al cliente que lo necesite. Cabe destacar que para la realización de este proyecto solamente se lo hará localmente, ya que, se presentaron distintas dificultades al momento de contratar los servicios en la nube disponibles en el mercado.

Desarrollo

Para la realización de este proyecto primero vamos a definir que tipos de servicios utilizaremos:

- Servicio DNS: Este servidor albergará el dominio de nuestro correo electrónico.
- Servicio Email: Mediante este servidor podremos realizar el envío y recepción de mensajes con Postfix (MTA) y Dovecot (MDA).
- Servicio de Base de Datos: Se necesitará de una base de datos para almacenar los clientes del servicio Email.

Las IP de red será la 192.168.16.0/24

Servidor DNS	192.168.16.10
Servidor Email	192.168.16.11
Servidor MySQL	192.168.16.12
GATEWAY	192.168.16.47

Previamente a cualquier instalación de paquetes en los servidores Linux estableceremos las IP's estáticas dentro, a través de la modificación del archivo de configuración **50-cloud-init.yaml**, que se encuentra en la dirección **/etc/netplan**. Modificamos cada uno de los servidores con las ip's respectivas.

2. Servicio DNS

Para configurar el servicio DNS, primero debemos instalar los paquetes necesarios para el mismo, en este caso se escogió el paquete bind9

```
sudo apt install bind9 bind9utils bind9-doc -y
```

Una vez realizada la instalación del paquete, debemos dirigirnos a los archivos de configuración y modificarlos para el dominio que escogimos, en este caso el dominio es neo.domus.

```
GNU nano 7.2                                named.conf.local
//
// Do any local configuration here
//
// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";
zone "neo.domus" {
    type master;
    file "/etc/bind/zones/db.neo.domus";
}
```

Captura 1.

Después de realizada la modificación, creamos el **db.neo.domus** dentro del directorio **/etc/bind/zones** y editamos de la siguiente manera.

```
GNU nano 7.2                                db.neo.domus *
$TTL      604800
@         IN      SOA      ns1.neo.domus.  admin.neo.domus. (
                        2025062101      ; Serial
                        604800           ; Refresh
                        86400            ; Retry
                        2419200          ; Expire
                        604800 )         ; Negative Cache TTL

@         IN      NS       ns1.neo.domus.
ns1       IN      A        192.168.16.10
mail      IN      A        192.168.16.11
@         IN      MX       10 mail.neo.domus.
```

Captura 2.

De esta forma establecemos las direcciones del dominio y del servidor email, además del registro MX para que el dominio pueda recibir correos electrónicos

Con esto tendríamos realizado nuestro servidor DNS, aunque previamente debemos indicar al firewall de nuestro servidor, permita la comunicación a través del puerto 53.

```
admin43@serdns:~$ sudo ufw allow 53
Skipping adding existing rule
Skipping adding existing rule (v6)
admin43@serdns:~$
```

Captura 3.

Podemos verificar desde otros clientes a través del comando dig:

```
Dig @192.168.16.10 mail.neo.domus
```

3. Servicio Email

Para el servidor de correo electrónico debemos instalar los siguientes paquetes:

```
sudo apt install postfix dovecot-core dovecot-imapd dovecot-lmtpd
dovecot-mysql mailutils -y
```

Cuando se proceda la instalación de postfix nos pedirá que tipo de servidor queremos y el nombre del sistema

Tipo: Sitio de Internet

Nombre de sistema de correo: neo.domus

Luego creamos los usuarios para el sistema de buzones virtuales utilizando los siguientes comandos:

```
sudo groupadd -g 5000 vmmail
sudo useradd -g vmmail -u 5000 vmmail -d /var/mail/vhosts -m
```

Nos vamos al directorio **/etc/postfix** para modificar el archivo de configuración main.cf donde corregiremos o aumentaremos varias de las líneas como veremos a continuación:

```
myhostname = mail.neo.domus
mydomain = neo.domus
myorigin = /etc/mailname
inet_interfaces = all
inet_protocols = ipv4
mydestination = localhost
virtual_mailbox_base = /var/mail/vhosts
virtual_mailbox_domains = proxy:mysql:/etc/postfix/mysql-virtual-
domains.cf
virtual_mailbox_maps = proxy:mysql:/etc/postfix/mysql-virtual-
mailboxes.cf
virtual_alias_maps = proxy:mysql:/etc/postfix/mysql-virtual-aliases.cf
virtual_transport = lmtp:unix:private/dovecot-lmtp
smtpd_tls_cert_file = /etc/ssl/certs/ssl-cert-snakeoil.pem
smtpd_tls_key_file = /etc/ssl/private/ssl-cert-snakeoil.key
smtpd_use_tls = yes
smtpd_tls_auth_only = yes
smtpd_sasl_type = dovecot
smtpd_sasl_path = private/auth
smtpd_sasl_auth_enable = yes
smtpd_recipient_restrictions =
    permit_sasl_authenticated,
    permit_mynetworks,
    reject_unauth_destination
```

Donde definimos la autenticación de los usuarios los protocolos, las interfaces, los buzones virtuales, el nombre del dominio y del servidor de correo electrónico, luego con el comando:

```
echo "neo.domus" | sudo tee /etc/mailname
```

Este comando nos permite mostrar texto en la terminal y, al mismo tiempo, guardarlo en uno o más archivos

```
GNU nano 7.2 main.cf
smtpd_banner = $myhostname ESMTP $mail_name (Ubuntu)
biff = no

# appending .domain is the MUA's job.
append_dot_mydomain = no

# Uncomment the next line to generate "delayed mail" warnings
#delay_warning_time = 4h

readme_directory = no

# See http://www.postfix.org/COMPATIBILITY_README.html -- default to 3.6 on
# fresh installs.
compatibility_level = 3.6

# 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

smtpd_tls_CApath=/etc/ssl/certs
smtpd_tls_security_level=may
smtpd_tls_session_cache_database = btree:${data_directory}/smtp_scache

smtpd_relay_restrictions = permit_mynetworks permit_sasl_authenticated defer_unauth_destination
myhostname = mail.neo.domus
mydomain = neo.domus
alias_maps = hash:/etc/aliases
alias_database = hash:/etc/aliases
myorigin = /etc/mailname
mydestination = localhost
virtual_mailbox_base = /var/mail/vhosts
virtual_mailbox_domains = proxy:mysql:/etc/postfix/mysql-virtual-domains.cf
virtual_mailbox_maps = proxy:mysql:/etc/postfix/mysql-virtual-mailboxes.cf
virtual_alias_maps = proxy:mysql:/etc/postfix/mysql-virtual-aliases.cf
virtual_transport = lmtp:unix:private/dovecot-lmtp
smtpd_use_tls = yes
smtpd_tls_auth_only = yes
smtpd_sasl_type = dovecot
smtpd_sasl_path = private/auth
smtpd_sasl_auth_enable = yes
smtpd_recipient_restrictions =
    permit_sasl_authenticated,
```

Captura 5.

Ahora debemos crear los archivos de conexión para con la base de datos MySQL, a través del directorio **/etc/postfix** creamos el archivo **mysql-virtual-domains.cf** y lo editamos de la siguiente forma:

```
GNU nano 7.2 mysql-virtual-domains.cf
user = correo
password = 10349849
hosts = 192.168.16.12
dbname = correodb
query = SELECT 1 FROM dominios WHERE nombre='%s'
```

Captura 6.

También configuramos otro archivo `mysql-virtual-mailboxes.cf` para los buzones

```
GNU nano 7.2                                mysql-virtual-mailboxes.cf
user = correo
password = 10349849
hosts = 192.168.16.12
dbname = correodb
query = SELECT 1 FROM usuarios WHERE correo='%s'
```

Captura 6.

Y otro para el manejo de alias dentro del correo electrónico **mysql-virtual-aliases.cf**:

```
GNU nano 7.2                                mysql-virtual-aliases.cf *
user = correo
password = 10349849
hosts = 192.168.16.12
dbname = correodb
query = SELECT destino FROM alias WHERE fuente='%s'
```

Captura 7.

Posteriormente otorgamos permisos a estos archivos y cambiar la propiedad de los mismos

```
sudo chmod 640 /etc/postfix/mysql-*.cf
```

```
sudo chown root:postfix /etc/postfix/mysql-*.cf
```

Procedemos a realizar la configuración de Dovecot en la dirección **/etc/dovecot/conf.d** en el archivo de configuración **10-mail.conf**:

```
GNU nano 7.2                                10-mail.conf *
#
# Mailbox locations and namespaces
#

Location for users' mailboxes. The default is empty, which means that Dovecot
tries to find the mailboxes automatically. This won't work if the user
doesn't yet have any mail, so you should explicitly tell Dovecot the full
location.

If you're using mbox, giving a path to the INBOX file (eg. /var/mail/%u)
isn't enough. You'll also need to tell Dovecot where the other mailboxes are
kept. This is called the "root mail directory", and it must be the first
path given in the mail_location setting.

There are a few special variables you can use, eg.:

%u - username
%n - user part in user@domain, same as %u if there's no domain
%d - domain part in user@domain, empty if there's no domain
%h - home directory

See doc/wiki/Variables.txt for full list. Some examples:

mail_location = maildir:~/Maildir
mail_location = mbox:~/mail:INBOX=/var/mail/%u
mail_location = mbox:/var/mail/%d/%n/INBOX=/var/indexes/%d/%n/INBOX

<doc/wiki/MailLocation.txt>

mail_location = maildir:/var/mail/vhosts/%d/%n
```

También modificamos el archivo 10-auth.conf para la autenticación de usuarios:

```
GNU nano 7.2                                10-auth.conf *
##
## Authentication processes
##
# Disable LOGIN command and all other plaintext authentications unless
# SSL/TLS is used (LOGINDISABLED capability). Note that if the remote IP
# matches the local IP (ie. you're connecting from the same computer), the
# connection is considered secure and plaintext authentication is allowed.
# See also ssl=required setting.
disable_plaintext_auth = yes
```

Captura 7.

```
Space separated list of wanted authentication mechanisms:
  plain login digest-md5 cram-md5 ntlm rpa apop anonymous gssapi otp
  gss-spnego
NOTE: See also disable_plaintext_auth setting.
auth_mechanisms = plain
```

Captura 8.

Modificamos el archivo de configuración **auth-sql.conf.ext** de forma que esté relacionado con el servidor de base de dato MySQL:

```
GNU nano 7.2                                auth-sql.conf.ext *
# Authentication for SQL users. Included from 10-auth.conf.
#
# <doc/wiki/AuthDatabase.SQL.txt>
passdb {
  driver = sql

  # Path for SQL configuration file, see example-config/dovecot-sql.conf.ext
  args = /etc/dovecot/dovecot-sql.conf.ext
}

# "prefetch" user database means that the passdb already provided the
# needed information and there's no need to do a separate userdb lookup.
# <doc/wiki/UserDatabase.Prefetch.txt>
#userdb {
#  driver = prefetch
#}

userdb {
  driver = static
  args = uid=vmail gid=vmail home=/var/mail/vhosts/%d/%n
}

# If you don't have any user-specific settings, you can avoid the user_query
# by using userdb static instead of userdb sql, for example:
# <doc/wiki/UserDatabase.Static.txt>
#userdb {
#  driver = static
#  args = uid=vmail gid=vmail home=/var/vmail/%u
#}
```

Captura 9.

Modificamos el archivo **/etc/dovecot/dovecot-sql.conf.ext** de forma que los usuarios que se encuentran en la base de datos se relacione con el servidor de correo.

```
# Examples:
# connect = host=192.168.1.1 dbname=users
# connect = host=sql.example.com dbname=virtual user=virtual password=blarg
# connect = /etc/dovecot/authdb.sqlite
#
connect = host=192.168.16.12 dbname=correodb user=correo password=10349849
```

Captura 10.

```
GNU nano 7.2 dovecot-sql.conf.ext
# Examples:
# connect = host=192.168.1.1 dbname=users
# connect = host=sql.example.com dbname=virtual user=virtual password=blarg
# connect = /etc/dovecot/authdb.sqlite
#
connect = host=192.168.16.12 dbname=correodb user=correo password=10349849
# Default password scheme.
#
# List of supported schemes is in
# http://wiki2.dovecot.org/Authentication/PasswordSchemes
#
default_pass_scheme = SHA512-CRYPT
```

Captura 11.

```
# Example:
# password_query = SELECT userid AS user, pw AS password \
# FROM users WHERE userid = '%u' AND active = 'Y'
#
password_query = SELECT correo AS user, clave AS password FROM usuarios WHERE correo='%u'
# SELECT username, domain, password \
# FROM users WHERE username = '%n' AND domain = '%d'
```

Captura 12.

Luego nos dirigimos a la dirección **/etc/dovecot/conf.d** para modificar el archivo de configuración 10-master.conf con sus permisos (mode) respectivos :

```
service lmtp {
    unix_listener /var/spool/postfix/private/dovecot-lmtp {
        mode = 0600
        user = postfix
        group = postfix
    }
}
```

Captura 13.

0600: Solo el propietario tiene permiso de escribir y leer el archivo

```
# To give the caller full permissions to lookup all users, set the mode to
# something else than 0666 and Dovecot lets the kernel enforce the
# permissions (e.g. 0777 allows everyone full permissions).
unix_listener /var/spool/postfix/private/auth {
    mode = 0660
    user = postfix
    group = postfix
}
```

Captura 14.

0660: el propietario y el grupo pueden leer y escribir en el archivo

Luego reiniciamos los servicios de postfix y dovecot con el siguiente comando.

```
sudo systemctl restart postfix dovecot
sudo systemctl enable postfix dovecot
```

Y habilitamos los puertos necesarios para que se comuniquen los servidores

```
sudo ufw allow 25,587,143,993/tcp
```

El puerto 25 y 587 es para el protocolo SMTP, el último admite cifrado en SSL/TLS
(Security Socket Layer y Transport Layer Security)

El puerto 143 y 993 son los puertos por los cuales funciona el protocolo IMAP,
siendo el último el que ofrece más seguridad (SSL/TLS).

4. Servicio de base de datos MySQL

Como primer paso, instalamos el paquete de MySQL dentro de nuestro servidor
con el comando

```
sudo apt install mysql-server -y
```

Luego de la instalación de todos los paquetes podemos asegurar algunas configuraciones de esta base de datos.

sudo mysql_secure_installation

```
admin43@servemaildb:/etc/netplan$ sudo mysql_secure_installation
Securing the MySQL server deployment.

Connecting to MySQL using a blank password.

VALIDATE PASSWORD COMPONENT can be used to test passwords
and improve security. It checks the strength of password
and allows the users to set only those passwords which are
secure enough. Would you like to setup VALIDATE PASSWORD component?

Press y|Y for Yes, any other key for No: y

There are three levels of password validation policy:

LOW      Length >= 8
MEDIUM  Length >= 8, numeric, mixed case, and special characters
STRONG Length >= 8, numeric, mixed case, special characters and dictionary          file

Please enter 0 = LOW, 1 = MEDIUM and 2 = STRONG: 0

Skipping password set for root as authentication with auth_socket is used by default.
If you would like to use password authentication instead, this can be done with the "ALTER_USER" command.
See https://dev.mysql.com/doc/refman/8.0/en/alter-user.html#alter-user-password-management for more information.

By default, a MySQL installation has an anonymous user,
allowing anyone to log into MySQL without having to have
a user account created for them. This is intended only for
testing, and to make the installation go a bit smoother.
You should remove them before moving into a production
environment.

Remove anonymous users? (Press y|Y for Yes, any other key for No) : y
Success.

Normally, root should only be allowed to connect from
'localhost'. This ensures that someone cannot guess at
the root password from the network.

Disallow root login remotely? (Press y|Y for Yes, any other key for No) : y
Success.

By default, MySQL comes with a database named 'test' that
anyone can access. This is also intended only for testing,
and should be removed before moving into a production
environment.

Remove test database and access to it? (Press y|Y for Yes, any other key for No) : _
```

Captura 15.

Dentro de esa configuración nos dará las siguientes opciones de seguridad para nuestro servidor de base de datos

- Establecer contraseña para root: sí
- Eliminar usuarios anónimos: sí
- Deshabilitar acceso remoto de root: sí
- Eliminar base de datos de prueba: sí
- Recargar privilegios: sí

Luego nos dirigimos a la dirección **/etc/mysql/mysql.conf.d** y modificamos el archivo de configuración **mysqld.cnf**

```
# If MySQL is running as a replication slave, this should be
# changed. Ref https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_tmpdir
# tmpdir                = /tmp
#
# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
bind-address            = 192.168.16.12
mysqlx-bind-address     = 127.0.0.1
```

Captura 16.

La modificamos con la ip de nuestro servidor de base de datos, para que este pueda identificarse con su IP respectiva

Ahora procedemos a crear la base de datos y el usuario de conexión. Para eso debemos ingresar a MySQL con el siguiente comando

```
sudo mysql -u root -p
```

Ya dentro de la base de datos, creamos una base de datos con el siguiente comando:

```
CREATE DATABASE correodb;
```

Y el usuario para la base de datos desde el cual se puede administrar remotamente, además de los privilegios que este tendrá dentro de la base de datos

```
mysql> CREATE USER 'correo'@'192.168.16.11' IDENTIFIED BY 'Stroke69#';
Query OK, 0 rows affected (0,05 sec)

mysql> GRANT ALL PRIVILEGES ON correodb.* TO 'correo'@'192.168.16.11';
Query OK, 0 rows affected (0,03 sec)

mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0,03 sec)
```

Captura 17.

Teniendo ya el usuario creado, ingresamos a la base de datos que creamos para general las tablas donde almacenaremos los usuarios para el correo electrónico

```
mysql> USE correodb
Database changed
mysql> CREATE TABLE dominios (id INT AUTO_INCREMENT PRIMARY KEY, nombre VARCHAR(255) NOT NULL UNIQUE);
Query OK, 0 rows affected (0,15 sec)

mysql> CREATE TABLE usuarios (id INT AUTO_INCREMENT PRIMARY KEY, correo VARCHAR(255) NOT NULL UNIQUE, clave VARCHAR(255) NOT NULL, dominio_id INT, FOREIGN KEY (dominio_id) REFERENCES dominios (id) ON DELETE CASCADE);
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'NULL U
NIQUE, clave VARCHAR(255) NOT NULL, dominio_id INT, FOREIGN KEY (dominio_id' at line 1
mysql> CREATE TABLE usuarios (id INT AUTO_INCREMENT PRIMARY KEY, correo VARCHAR(255) NOT NULL UNIQUE, clave VARCHAR(255) NOT NULL, dominio_id INT, FOREIGN KEY (dominio_id) REFERENCES dominios (id) ON DELETE CASCADE);
Query OK, 0 rows affected (0,29 sec)

mysql> CREATE TABLE alias (id INT AUTO_INCREMENT PRIMARY KEY, fuente VARCHAR(255) NOT NULL, destino VARCHAR(255) NOT NULL);
Query OK, 0 rows affected (0,14 sec)

mysql> _
```

Captura 18..

Establecemos tablas para los dominios, los usuarios y los alias, para mantener un orden dentro de nuestra base de datos.

Insertamos nuestro usuario virtual, desde el cual podremos enviar o recibir mensajes e incluimos una clave de cifrado, así como también el dominio de correo electrónico

```
mysql> INSERT INTO dominios (nombre) VALUES ('neo.domus');
Query OK, 1 row affected (0,04 sec)
```

Captura 19.

Para insertar a usuario debemos generar un código desde el servidor de correo electrónico con el siguiente comando

doveadm pw -s SHA512-CRYPT

```
mysql> INSERT INTO usuarios (correo, clave, dominio_id) VALUES ('usuario1@neodomus', '{SHA512-CRYPT}$6$Kvpygmw/zTJAosL3$K46sJ.L7Rh6/Hx7mnDAFdqdkXpXsSics1W/I0cx1KyyS4324f7JNbr7nGRtaFR2f9RbeYUOhQcpJn4yocQqT.', 1);
Query OK, 1 row affected (0,02 sec)
```

Captura 20.

Ya teniendo establecidas estas configuraciones podemos proceder a realizar las pruebas necesarias para la conexión entre el servidor de correo electrónico y la base de datos, podemos verificar con el mismo con el siguiente comando

mysql -h 192.168.16.12 -u correo -p correodb

Conclusiones

Como se pudo apreciar en el presente documento, la configuración de este tipo de servidores requieren amplios conocimientos en el manejo de sistemas Linux, además de comprender el funcionamiento básico de redes informáticas, sin embargo, un detalle muy importante que lamentablemente no pudo realizarse dentro del mismo, fue la realización de pruebas locales exitosas de envío y recepción de mensajes entre los clientes, siendo un paso fundamental e importante para saber el funcionamiento adecuado del sistema más allá de conseguir una conexión exitosa entre todos los servicios.