

Servicios de aplicación

Monitoreo y gestión de redes

Diego Mayorga

Universidad de Mendoza 2016

Requerimientos.

Desarrollar una solución completa de software que permita:

- Gestionar servicios de correo electrónico, DNS y Web.
- Acceso remoto y monitoreo a través de la aplicación.

Introducción.

El proceso de desarrollo se dividió en:

- 1- Creación de instancia en Amazon Web Services.
- 2- Comprar y configurar DNS de dominio.
- 3- Instalación y configuración de los servicios necesarios.
- 4- Implementación del software desarrollado.

Desarrollo.

Creación de instancia en AWS.

Solución se desarrolló sobre una instancia de AWS, características:

- Instancia: ec2 – t2.micro.
- OS: Linux Ubuntu 14.04.

Crear una instancia:

- Entrar en la plataforma de AWS.
- Ir a Instance y ejecutar Launch Instance.
- Ir a Key Pairs, generar una y descargarla a la PC.
- Ir a Elastic IP y generar una, asignarla a la instancia recién creada.
- Ir a Security Group y crear un grupo de seguridad nuevo o usar el creado por defecto. Aquí habilitamos en Inbound y Outbound el puerto de SSH, en source ponemos My Ip.

Con esto ya podemos entrar por ssh a nuestra máquina, para ello, primero vamos al directorio donde tenemos el archivo de llave pública y hacemos ssh -i "KeyPairGenerada.pem" [ubuntu@ip-elastica-generada](#).

Instalación y configuración de servicios necesarios.

Servidor DB MySQL

Instalamos el servidor de Base de Datos MySQL.

```
# apt-get install mysql-client mysql-server
```

Servidor DNS PowerDNS

Primero creamos la base de datos correspondiente.

```
# mysql -u root -p
# CREATE DATABASE powerdns;
# GRANT ALL ON powerdns.* TO 'dnsadmin'@'localhost' IDENTIFIED BY 'password';
# USE powerdns;
# CREATE TABLE domains (
#     id INT auto_increment,
#     name VARCHAR(255) NOT NULL,
#     master VARCHAR(128) DEFAULT NULL,
#     last_check INT DEFAULT NULL,
#     type VARCHAR(6) NOT NULL,
#     notified_serial INT DEFAULT NULL,
#     account VARCHAR(40) DEFAULT NULL,
#     primary key (id)
# );

# CREATE TABLE records (
#     id INT auto_increment,
#     domain_id INT DEFAULT NULL,
#     name VARCHAR(255) DEFAULT NULL,
#     type VARCHAR(6) DEFAULT NULL,
#     content VARCHAR(255) DEFAULT NULL,
#     ttl INT DEFAULT NULL,
```

```
#      prio INT DEFAULT NULL,
#      change_date INT DEFAULT NULL,
#      primary key(id)
#      );

# CREATE TABLE supermasters (
#      ip VARCHAR(25) NOT NULL,
#      nameserver VARCHAR(255) NOT NULL,
#      account VARCHAR(40) DEFAULT NULL
#      );
```

Instalamos el servidor de DNS (Power DNS) con soporte de MySQL.

```
# apt-get install pdns-server pdns-backend-mysql
```

Editar archivos de configuración

```
vim /etc/powerdns/pdns.conf
# allow-axfr-ips=instance-private-ip
# allow-recursion=127.0.0.1
# config-dir=/etc/powerdns
# daemon=yes
# disable-axfr=no
# guardian=yes
# launch=gmysql
# local-address=instance-private-ip
# local-port=53
# master=yes
# module-dir=/usr/lib/powerdns
# setgid=pdns
# setuid=pdns
# slave=no
# socket-dir=/var/run
# webserver-port=8081
```

```
vim /etc/powerdns/pdns.d/pdns.local
launch=gmysql
gmysql-host=127.0.0.1
gmysql-user=dnsadmin
gmysql-password=password
gmysql-dbname=powerdns
```

```
#Reiniciamos el servicio
/etc/init.d/pdns restart
```

Servidor Web Apache

```
apt-get install apache2 libapache2-mod-php5 php5 php5-common php5-curl php5-dev php5-gd php-pear php5-imap php5-mcrypt
php5-mhash php5-ming php5-mysql php5-xmlrpc gettext
```

Instalamos librerías de PERL

```
pear install DB
pear install pear/MDB2#mysql
```

Bajamos la herramienta PowerAdmin

```
cd /tmp
wget https://github.com/downloads/poweradmin/poweradmin/poweradmin-2.1.6.tgz
```

```
tar xvfz poweradmin-2.1.6.tgz
mv poweradmin-2.1.6 /var/www/poweradmin
touch /var/www/poweradmin/inc/config.inc.php
chown -R www-data:www-data /var/www/poweradmin/
```

Corremos el instalador y seguimos los pasos
http://<<IP_HOST>>/poweradmin/install

Comprar y configurar DNS de dominio.

El dominio htagro.info se compró en Godaddy.com. Para poder usar el dominio en nuestra instancia primero se debe crear el servidor DNS y luego cambiar los NS de Godaddy a los que crearemos.





Una vez terminado ingresamos a poweradmin y configuramos los parámetros necesarios para poder tener el dominio de Godaddy en nuestro servidor.

Parametros de PowerDNS:

Poweradmin

[Index](#) [Search zones and records](#) [List zones](#) [List zone templates](#) [List supermasters](#) [Add master zone](#) [Add slave zone](#) [Add supermaster](#) [Bulk registration](#) [Change password](#) [User administration](#) [Logout](#)

List zones

	Name	Type	Records	Owner
 	35.31.172.in-addr.arpa	master	6	Administrator
 	htagro.info	master	9	Administrator



















Delete zone(s)

[a complete\(r\) poweradmin v 2.1.6 - credits](#)

Poweradmin

[Index](#) [Search zones and records](#) [List zones](#) [List zone templates](#) [List supermasters](#) [Add master zone](#) [Add slave zone](#) [Add supermaster](#) [Bulk registration](#) [Change password](#) [User administration](#) [Logout](#)

Edit zone "htagro.info"

Name	Type	Content	Priority	TTL
  htagro.info	SOA	ns1.htagro.info hostmaster.htagro.info 2016020500 28800 7200 604800 86400		86400
  htagro.info	NS	ns1.htagro.info	0	600
  htagro.info	NS	ns2.htagro.info	0	600
  htagro.info	MX	mail.htagro.info	10	600
  htagro.info	A	52.24.170.144	0	600
  mail.htagro.info	A	52.24.170.144	0	600
  ns1.htagro.info	A	52.24.170.144	0	600
  ns2.htagro.info	A	52.24.170.144	0	600
  www.htagro.info	A	52.24.170.144	0	600

Comments:













Save as new template:
Template Name
Template Description

[Commit changes](#) [Reset changes](#) [Save as template](#)

Poweradmin

[Index](#) [Search zones and records](#) [List zones](#) [List zone templates](#) [List supermasters](#) [Add master zone](#) [Add slave zone](#) [Add supermaster](#) [Bulk registration](#) [Change password](#) [User administration](#) [Logout](#)

Edit zone "35.31.172.in-addr.arpa"

Name	Type	Content	Priority	TTL
  35.31.172.in-addr.arpa	SOA	ns1.htagro.info hostmaster.htagro.info 2015120705 28800 7200 604800 86400		86400
  190.35.31.172.in-addr.arpa	PTR	ns2.htagro.info	0	600
  190.35.31.172.in-addr.arpa	PTR	ns1.htagro.info	0	600
  190.35.31.172.in-addr.arpa	PTR	htagro.info	0	600
  190.35.31.172.in-addr.arpa	PTR	www.htagro.info	0	600
  190.35.31.172.in-addr.arpa	PTR	mail.htagro.info	0	600

Comments:

Save as new template:
Template Name
Template Description

[Commit changes](#) [Reset changes](#) [Save as template](#)

Configurar archivos:

vim /etc/hostname

Borrar el nombre que tenga y poner el de nuestro dominio.

vim /etc/hosts

127.0.0.1 localhost

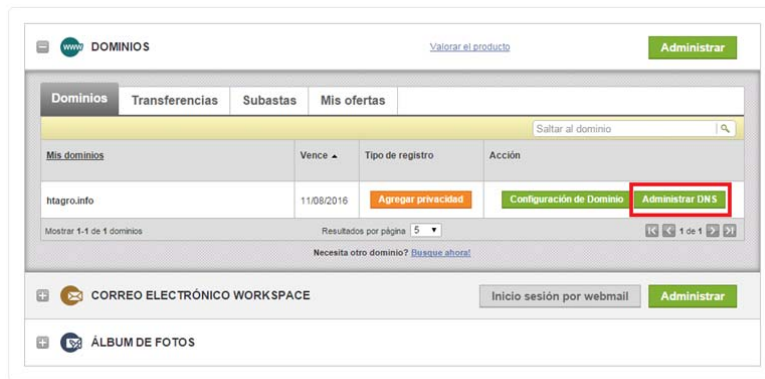
127.0.1.1 hostName

ip.elastica.de.intancia hostname.com hostname

El resto dejarlo como está

Reiniciar el servidor para que tome el nuevo nombre de host.

Ir a Godaddy al panel de administración de los DNS.



Agregar los NS que creamos en PowerDNS:

CONFIGURACIONES	ARCHIVO DE ZONA DNS	CONTACTOS
Renovación Automática	<input checked="" type="radio"/> Estándar Activado <input type="radio"/> Ampliado Desactivado Administrar	
Bloquear	<input checked="" type="radio"/> Activado Administrar	
Servidores de nombres	NS1 HTAGRO.INFO NS2 HTAGRO.INFO Actualizado 28/11/2015 Administrar	
Reenvío	<input checked="" type="radio"/> Dominio: Desactivado Administrar <input type="radio"/> Subdominio: 0 subdominios reenviados Administrar	
Premium DNS	<input checked="" type="radio"/> No propio Administrar	
Registros DS	<input type="radio"/> 0 registros DS creados Administrar	
Nombres del Host	<input checked="" type="radio"/> 3 nombres de host creados Administrar	
Transferencia de dominios	<input type="radio"/> Transferencia al exterior	

Primero agregamos los Nombres de Host:

NOMBRES DE HOST

HTAGRO.INFO

Nombres de host	Direcciones IP	
MAIL	52.24.170.144	✎ 🔄
NS1	52.24.170.144	✎ 🔄
NS2	52.24.170.144	✎ 🔄

[AÑADIR NOMBRE DE HOST](#)

[GUARDAR](#) [Cancelar](#)

Luego agregamos los Servidores de Nombre:

CONFIGURACIONES DE SERVIDORES DE NOMBRES

HTAGRO.INFO

Los servidores de nombres apuntan adonde está ubicado tu dominio.

TIPO DE CONFIGURACIÓN:

- ☒ Estándar
 Hosting, correo y dominios aparcados de Go Daddy
☐ Personalizar
 Configuraciones de servidores de nombres personalizables.

SERVIDORES DE NOMBRES:

[EDITAR SERVIDORES DE NOMBRES](#)

#	Servidor de nombres	Estado	
1	NS1 HTAGRO.INFO	Activo	🔄
2	NS2 HTAGRO.INFO	Activo	🔄

[AÑADIR SERVIDOR DE NOMBRES](#)

[GUARDAR](#) [Cancelar](#)

Esperamos que se propaguen los cambios y verificamos con el comando dig que nuestro servidor ya tenga los NS que le asignamos.

Postfix

Instalamos Postfix, seleccionando sitio de internet cuando se pregunte en el proceso

```
# apt-get install postfix postfix-mysql
```

Preparamos el directorio para el correo virtual

```
# cp /etc/aliases /etc/postfix/aliases
# postalias /etc/postfix/aliases
# mkdir /var/spool/mail/virtual
# groupadd --system virtual -g 5000
# useradd --system virtual -u 5000 -g 5000
# chown -R virtual:virtual /var/spool/mail/virtual
```

Configuración MySQL de Postfix

Configuramos el archivo con los datos para conectar con la base de datos

```
vim /etc/postfix/mysql_mailbox.cf
# user=$user
# password=$passwd
# dbname=$db
# table=users
# select_field=maildir
# where_field=id
# hosts=127.0.0.1
# additional_conditions = and enabled = 1
```

```
vim /etc/postfix/mysql_alias.cf
# user=$user
# password=$passwd
# dbname=$db
# table=aliases
# select_field=destination
# where_field=mail
# hosts=127.0.0.1
#additional_conditions = and enabled = 1
```

Como encontrar los dominios

```
vim /etc/postfix/mysql_domains.cf
# user=$user
# password=$passwd
# dbname=$db
# table=domains
# select_field=domain
# where_field=domain
# hosts=127.0.0.1
# additional_conditions = and enabled = 1
```

```
vim /etc/postfix/main.cf
myorigin = htagro.info
smtpd_banner = $myhostname ESMTP $mail_name
biff = no
append_dot_mydomain = no
readme_directory = no
# TLS parameters
smtpd_tls_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem
smtpd_tls_key_file=/etc/ssl/private/ssl-cert-snakeoil.key
smtpd_use_tls=yes
smtpd_tls_session_cache_database = btree:${data_directory}/smtpd_scache
smtp_tls_session_cache_database = btree:${data_directory}/smtp_scache
local_recipient_maps =
mydestination =
relayhost =
mynetworks = 127.0.0.0/8 [::ffff:127.0.0.0]/104 [::1]/128
mailbox_size_limit = 100000
recipient_delimiter = +
inet_interfaces = all
inet_protocols = ipv4
```

```

mynetworks_style = host
delay_warning_time = 4h
unknown_local_recipient_reject_code = 450
maximal_queue_lifetime = 7d
minimal_backoff_time = 1000s
maximal_backoff_time = 8000s
smtp_helo_timeout = 60s
smtpd_recipient_limit = 16
smtpd_soft_error_limit = 3
smtpd_hard_error_limit = 12
smtpd_helo_restrictions = permit_mynetworks, warn_if_reject reject_non_fqdn_hostname, reject_invalid_hostname, permit
smtpd_client_restrictions = reject_rbl_client sbl.spamhaus.org, reject_rbl_client blackholes.easynet.nl, permit_sasl_authenticated
smtpd_sender_restrictions = permit_sasl_authenticated, permit_mynetworks, warn_if_reject reject_non_fqdn_sender,
reject_unknown_sender_domain, reject_unauth_pipelining, permit
smtpd_recipient_restrictions = reject_unauth_pipelining, permit_mynetworks, permit_sasl_authenticated, reject_non_fqdn_recipient,
reject_unknown_recipient_domain, reject_unauth_destination, check_policy_service inet:127.0.0.1:10023, permit
smtpd_data_restrictions = reject_unauth_pipelining
smtpd_helo_required = yes
smtpd_delay_reject = yes
disable_vrfy_command = yes
alias_maps = hash:/etc/postfix/aliases
alias_database = hash:/etc/postfix/aliases
virtual_mailbox_base = /var/spool/mail/virtual
virtual_mailbox_maps = mysql:/etc/postfix/mysql_mailbox.cf
virtual_alias_maps = mysql:/etc/postfix/mysql_alias.cf
virtual_mailbox_domains = mysql:/etc/postfix/mysql_domains.cf
# SASL
smtpd_sasl_auth_enable = yes
# If your potential clients use Outlook Express or other older clients
# this needs to be set to yes
broken_sasl_auth_clients = no
smtpd_sasl_security_options = noanonymous
smtpd_sasl_local_domain =
virtual_uid_maps=static:5000
virtual_gid_maps=static:5000
# TLS parameters
smtp_tls_security_level = may
smtpd_tls_security_level = may
smtp_tls_note_starttls_offer = yes
smtpd_tls_loglevel = 1
smtpd_tls_received_header = yes
smtpd_tls_session_cache_timeout = 3600s
tls_random_source = dev:/dev/urandom
smtpd_tls_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem
smtpd_tls_key_file=/etc/ssl/private/ssl-cert-snakeoil.key
smtpd_tls_CAfile = /etc/ssl/certs/ca-certificates.crt
content_filter = amavis:[127.0.0.1]:10024

```

Removemos los permisos de lectura a los archivos que contienen credenciales MySQL:

```

# chown root:postfix /etc/postfix/mysql_*
# chmod 0640 /etc/postfix/mysql_*

```

Reiniciamos Postfix:

```
# service postfix restart
```

SMTP Seguro

SASL (Simple Authentication and Security Layer)

Instalamos SASL:

```
# apt-get install libsasl2-modules libsasl2-modules-sql libgsasl7 \libauthen-sasl-cyrus-perl sasl2-bin libpam-mysql
```

Habilitamos a Postfix acceso a los archivos SASL:

```

# adduser postfix sasl
# mkdir -p /var/spool/postfix/var/run/saslauthd

```

```
vim /etc/default/saslauthd:
# START=yes
# DESC="SASL Authentication Daemon"
# NAME="saslauthd"
# MECHANISMS="pam"
# MECH_OPTIONS=""
# THREADS=5
# OPTIONS="-r -c -m /var/spool/postfix/var/run/saslauthd"
```

Interacción de Postfix con SASL:

```
vim /etc/postfix/sasl/smtpd.conf
# pwcheck_method: saslauthd
# mech_list: plain login cram-md5 digest-md5
# log_level: 7
# allow_plaintext: true
# auxprop_plugin: sql
# sql_engine: mysql
# sql_hostnames: 127.0.0.1
# sql_user: $user
# sql_passwd: $passwd
# sql_database: $db
# sql_select: select crypt from users where id='%u@%r' and enabled = 1
```

Definimos como PAM autentica SMTP a través de MySQL:

```
vim /etc/pam.d/smtp
# auth required pam_mysql.so user=$user passwd=$passwd host=127.0.0.1
# db=postfix table=users usercolumn=id passwdcolumn=crypt crypt=1
# account sufficient pam_mysql.so user=$user passwd=$passwd host=127.0.0.1
# db=postfix table=users usercolumn=id passwdcolumn=crypt crypt=1
```

Reiniciamos los servicios:

```
# /etc/init.d/saslauthd restart
# /etc/init.d/postfix restart
```

TLS: Postfix

Creamos un certificado SSL:

```
$ cd /etc/postfix
$ openssl req -new -outform PEM -out postfix.cert -newkey rsa:2048 -nodes \-keyout postfix.key -keyform PEM -days 1825 -x509
```

```
vim /etc/postfix/master.cf :
# Postfix master process configuration file. For details on the format
# of the file, see the master(5) manual page (command: "man 5 master" or
# on-line: http://www.postfix.org/master.5.html).
#
# Do not forget to execute "postfix reload" after editing this file.
#
# =====
# service type private unpriv chroot wakeup maxproc command + args
#          (yes) (yes) (yes) (never) (100)
# =====
smtp      inet  n       -       -       -       smtpd
#smtp     inet  n       -       -       1       postscreen
#smtpd    pass  -       -       -       -       smtpd
#dnsblog  unix  -       -       -       0       dnsblog
#tlsproxy unix  -       -       -       0       tlsproxy
submission inet n       -       -       -       smtpd
#  -o syslog_name=postfix/submission
#  -o smtpd_tls_security_level=encrypt
#  -o smtpd_sasl_auth_enable=yes
#  -o smtpd_tls_auth_only=yes
#  -o smtpd_reject_unlisted_recipient=no
#  -o smtpd_client_restrictions=$mua_client_restrictions
#  -o smtpd_client_restrictions=permit_sasl_authenticated,reject_unauth_destination,reject
#  -o smtpd_sasl_security_options=noanonymous,noplaintext
#  -o smtpd_sasl_tls_security_options=noanonymous
```



```

# -o smtpd_helo_restrictions=$mua_helo_restrictions
# -o smtpd_sender_restrictions=$mua_sender_restrictions
# -o smtpd_recipient_restrictions=
# -o smtpd_relay_restrictions=permit_sasl_authenticated,reject
# -o milter_macro_daemon_name=ORIGINATING
smtps inet n - - - smtpd
# -o syslog_name=postfix/smtps
# -o smtpd_tls_wrappermode=yes
# -o smtpd_sasl_auth_enable=yes
# -o smtpd_tls_auth_only=yes
# -o smtpd_reject_unlisted_recipient=no
# -o smtpd_client_restrictions=$mua_client_restrictions
# -o smtpd_client_restrictions=permit_sasl_authenticated,permit
# -o smtpd_sasl_security_options=noanonymous,noplaintext
# -o smtpd_sasl_tls_security_options=noanonymous
# -o smtpd_helo_restrictions=$mua_helo_restrictions
# -o smtpd_sender_restrictions=$mua_sender_restrictions
# -o smtpd_recipient_restrictions=
# -o smtpd_relay_restrictions=permit_sasl_authenticated,reject
# -o milter_macro_daemon_name=ORIGINATING
#628 inet n - - - qmqpd
pickup unix n - - 60 1 pickup
# -o content_filter=
# -o receive_override_options=no_header_body_checks
cleanup unix n - - 0 cleanup
qmgr unix n - n 300 1 qmgr
#qmgr unix n - n 300 1 oqmgr
tlsmgr unix - - - 1000? 1 tlsmgr
rewrite unix - - - - trivial-rewrite
bounce unix - - - - 0 bounce
defer unix - - - - 0 bounce
trace unix - - - - 0 bounce
verify unix - - - - 1 verify
flush unix n - - 1000? 0 flush
proxymap unix - - n - - proxymap
proxywrite unix - - n - 1 proxymap
smtp unix - - - - smtp
relay unix - - - - smtp
# -o smtp_helo_timeout=5 -o smtp_connect_timeout=5
showq unix n - - - showq
error unix - - - - error
retry unix - - - - error
discard unix - - - - discard
local unix - n n - - local
virtual unix - n n - - virtual
lmtp unix - - - - lmtp
anvil unix - - - - 1 anvil
scache unix - - - - 1 scache
#
# =====
# Interfaces to non-Postfix software. Be sure to examine the manual
# pages of the non-Postfix software to find out what options it wants.
#
# Many of the following services use the Postfix pipe(8) delivery
# agent. See the pipe(8) man page for information about ${recipient}
# and other message envelope options.
# =====
#
# maildrop. See the Postfix MAILDROP_README file for details.
# Also specify in main.cf: maildrop_destination_recipient_limit=1
#
maildrop unix - n n - - pipe
flags=DRhu user=vmail argv=/usr/bin/maildrop -d ${recipient}
#
# =====
#
# Recent Cyrus versions can use the existing "lmtp" master.cf entry.
#
# Specify in cyrus.conf:
# lmtp cmd="lmtpd -a" listen="localhost:lmtp" proto=tcp4

```

```

#
# Specify in main.cf one or more of the following:
# mailbox_transport = lmtp:inet:localhost
# virtual_transport = lmtp:inet:localhost
#
# =====
#
# Cyrus 2.1.5 (Amos Gouaux)
# Also specify in main.cf: cyrus_destination_recipient_limit=1
#cyrus unix - n n - - pipe
# user=cyrus argv=/cyrus/bin/deliver -e -r ${sender} -m ${extension} ${user}
#
# =====
# Old example of delivery via Cyrus.
#
#old-cyrus unix - n n - - pipe
# flags=R user=cyrus argv=/cyrus/bin/deliver -e -m ${extension} ${user}
#
# =====
#
# See the Postfix UUCP_README file for configuration details.
#
uucp unix - n n - - pipe
 flags=Fqhu user=uucp argv=uux -r -n -z -a$sender - $nexthop!rmail ($recipient)
#
# Other external delivery methods.
#
ifmail unix - n n - - pipe
 flags=F user=ftn argv=/usr/lib/ifmail/ifmail -r $nexthop ($recipient)
bsmtp unix - n n - - pipe
 flags=Fq. user=bsmtp argv=/usr/lib/bsmtp/bsmtp -t$nexthop -f$sender $recipient
scalemail-backend unix - n n - 2 pipe
 flags=R user=scalemail argv=/usr/lib/scalemail/bin/scalemail-store ${nexthop} ${user} ${extension}
mailman unix - n n - - pipe
 flags=FR user=list argv=/usr/lib/mailman/bin/postfix-to-mailman.py
 ${nexthop} ${user}
amavis unix - - - - 2 smtp
 -o smtp_data_done_timeout=1200
 -o smtp_send_xforward_command=yes
 -o disable_dns_lookups=yes
 -o max_use=20
127.0.0.1:10025 inet n - - - - smtpd
 -o content_filter=
 -o local_recipient_maps=
 -o relay_recipient_maps=
 -o smtpd_restriction_classes=
 -o smtpd_delay_reject=no
 -o smtpd_client_restrictions=permit_mynetworks,reject
 -o smtpd_helo_restrictions=
 -o smtpd_sender_restrictions=
 -o smtpd_recipient_restrictions=permit_mynetworks,reject
 -o smtpd_data_restrictions=reject_unauth_pipelining
 -o smtpd_end_of_data_restrictions=
 -o mynetworks=127.0.0.0/8
 -o smtpd_error_sleep_time=0
 -o smtpd_soft_error_limit=1001
 -o smtpd_hard_error_limit=1000
 -o smtpd_client_connection_count_limit=0
 -o smtpd_client_connection_rate_limit=0
 -o receive_override_options=no_header_body_checks,no_unknown_recipient_checks

```

Restart Postfix:

```
# service postfix restart
```

Courier IMAP

Instalamos

```
# apt-get install courier-base courier-authdaemon courier-authlib-mysql courier-imap courier-imap-ssl courier-ssl
```

```
vim /etc/courier/imapd
ADDRESS=0
```

```
vim /etc/courier/authdaemonrc
authmodulelist="authmysql"
authmodulelistorig="authuserdb authpam authpgsql authldap authmysql
authcustom authpipe"
daemons=5
authdaemonvar=/var/run/courier/authdaemon
DEBUG_LOGIN=2
DEFAULTOPTIONS=""
LOGGEROPTS=""
```

```
vim /etc/courier/authmysqlrc
authmodulelist="authmysql"
authmodulelistorig="authuserdb authpam authpgsql authldap authmysql
authcustom authpipe"
daemons=5
authdaemonvar=/var/run/courier/authdaemon
DEBUG_LOGIN=2
DEFAULTOPTIONS=""
LOGGEROPTS=""
MYSQL_SERVER localhost
MYSQL_USERNAME $user
MYSQL_PASSWORD $passwd
MYSQL_PORT 0
MYSQL_OPT 0
MYSQL_DATABASE $db
MYSQL_USER_TABLE postfix_users
MYSQL_CRYPT_PWFIELD crypt
MYSQL_UID_FIELD uid
11
MYSQL_GID_FIELD gid
MYSQL_LOGIN_FIELD id
MYSQL_HOME_FIELD home
MYSQL_NAME_FIELD name
MYSQL_MAILDIR_FIELD concat(home,'/',maildir)
MYSQL_WHERE_CLAUSE enabled=1
Reiniciamos los servicios
# /etc/init.d/mysql restart
# /etc/init.d/postfix restart
# /etc/init.d/courier-authdaemon restart
# /etc/init.d/courier-imap restart
# /etc/init.d/courier-imap-ssl restart
```

IMAP SEGURO

TLS: Courier IMAP

Creamos el certificado SSL:

```
# cd /etc/courier
$ openssl req -x509 -newkey rsa:2048 -keyout imapd.pem \
-out imapd.pem -nodes -days 1825
Contenido de /etc/courier/imapd-ssl :
SSLPORT=993
SSLADDRESS=0.0.0.0
SSLPIDFILE=/var/run/courier/imapd-ssl.pid
SSLLOGGEROPTS="-name=imapd-ssl"
IMAPDSSLSTART=YES
IMAPDSTARTTLS=YES
IMAP_TLS_REQUIRED=1
COURIERTLS=/usr/bin/couriertls
TLS_KX_LIST=ALL
12
TLS_COMPRESSION=ALL
TLS_CERTS=X509
TLS_CERTFILE=/etc/courier/imapd.pem
TLS_TRUSTCERTS=/etc/ssl/certs
```

```
TLS_VERIFYPEER=NONE
TLS_CACHEFILE=/var/lib/courier/couriersslcache
TLS_CACHESIZE=524288
MAILDIRPATH=Maildir
```

Reiniciamos el servicio:

```
# service courier-imap-ssl restart
```

Gate One

Gate One es un emulador de terminal y cliente SSH web, que no requiere ningún plugin en el browser, es multiusuario y multisesión, permite múltiples sistemas de autenticación y para este caso en particular está embebido en el cliente, pero puede conectarse a cualquier servidor gracias a websockets. Además, permite visualizar imágenes y documentos, como por ejemplo un PDF.

Requerimientos:

Python 2.6+ o 3.2+
Tornado Framework 2.2+

Verificar que se tiene la version correspondiente de Python y Tornado.

Instalación:

- Descargamos el source desde <https://github.com/liftoff/GateOne/releases>
- tar zxvf gateone*.tar.gz
- cd gateone*
- sudo python setup.py install

La primera vez que se ejecute creará el archivo de configuración /opt/gateone/server.conf.

Modificamos el archivo de configuración con lo siguiente:

```
auth = None # Anonymous authentication
port = 10 443
disable_ssl = False
origins = "*"
url_prefix = "/"
```

Para ingresar a gateone, lo hacemos en <http://<host>:10443>

Para este caso como estamos utilizando una instancia de AWS, debemos cargar el archivo .pem a GateOne.

- Ingresamos a GateOne.
- Hacemos clic en la terminal.
- En la barra de la derecha aparecerá un icono nuevo que se llama Terminal Application Panel.
- Al final se encuentra un botón Manage Identities.
- Upload, para cargar la llave privada de AWS.

Una vez que subimos el archivo ingresamos a la terminal con:

- Host: localhost
- Port: 22
- User: nuestro usuario de AWS

Habilitar puertos necesarios.

Una vez configurado todo debemos abrir los puertos necesarios en nuestra instancia, para ello:

- Panel de control de la instancia.
- Vamos a Grupos de seguridad.
- En Inbound habilitamos los puertos:
 - o HTTP – Anywhere
 - o IMAP – Anywhere
 - o IMAPS – Anywhere
 - o SMTPS – Anywhere
 - o DNS(UPD) – Anywhere
 - o DNS(TCP) – Anywhere
 - o SMTP – Anywhere
 - o Mysql – Anywhere
 - o Cutom TCP Rule – Puerto: 10443 – My Ip
- Para Outbound colocamos los mismos puertos.

Implementación de software

El desarrollo del software se realizó en PHP/Javascript con Laravel 5.2 como framework de backend y AngularJS para el frontend. Se implementó una API REST para la comunicación entre el backend y el frontend, dotando a la aplicación flexibilidad para utilizar distintos frontends y plataformas para interactuar con la aplicación.

Backend.

Sobre Laravel 5.2 la aplicación se diagramó de la siguiente manera utilizando un patrón de diseño MVC:

Modelos:

- **Users:** Usuarios de la aplicación. pueden tener múltiples Domains.
- **Domains:** Entradas de Dominio. pueden tener múltiples Records, Mails.
- **Records :** Entradas de DNS para un dominio específico.
- **Mails :** Cuentas de correo para el dominio.

Controladores:

Se implementó un controlador para cada modelo. Esto nos permitió enrutar las peticiones a un recurso específico, delegándole la responsabilidad sobre sólo ese recurso. Al usar una arquitectura REST el enrutamiento se realizó como sigue:

```
Route::group(['prefix' => 'api'], function()
{
    Route::resource('authenticate', 'AuthenticateController', ['only' => ['index']]);
    Route::get('domains', 'DomainsController@index');
    Route::get('domains/{id}', 'DomainsController@show');
    Route::post('domains', 'DomainsController@store');
    Route::delete('domains/{id}', 'DomainsController@destroy');
    Route::patch('domains/{id}', 'DomainsController@edit');
});
```

Seguridad y Autenticación:

La aplicación ofrece autenticación basada en usuario y contraseña.

REST API Token:

Se utilizó JWT para la autenticación mediante Tokens, en cada login, la aplicación genera un token, el cual debe ser enviado como cabecera con cada request que se realice a la aplicación.

Frontend:

Para el frontend se diagramó una solución utilizando el framework AngularJS y siguiendo también un patrón de diseño MVC. El esquema de la aplicación en frontend queda de la siguiente manera:

- Scripts-Controllers: Un controlador para cada vista con los eventos a lanzar para cada recurso.
- Scripts-Directives: Directivas creadas por el usuario.
- Scripts-Filters: Filtros creados por el usuario.

Views:

Se dividieron las vistas según a que recurso pertenece:

- Dominios
- Navbar
- Records
- Zones
- Login
- Mailbox

Dentro de cada recurso se agregó la vista necesaria, listar, agregar, editar.

Consideraciones final.

Una ventaja de esquematizar la aplicación de esta manera es que nos permite manejar múltiples plataformas. Como el backend y frontend están separados, usando la misma API de backend, podemos desarrollar el frontend en cualquier lenguaje y plataforma.