Escuela de ingeniería

Departamento de informática y sistemas

Proyecto N° 2

INTEGRANTES:

Juan Felipe Londoño Gaviria

Juan David Pérez Sotelo

Felipe Ríos López

Profesor: Edwin Nelson Montoya Munera

Universidad EAFIT

Escuela de ingeniería

ST0263: Tópicos especiales de telemática

Medellín

23 De abril de 2020

Contenido

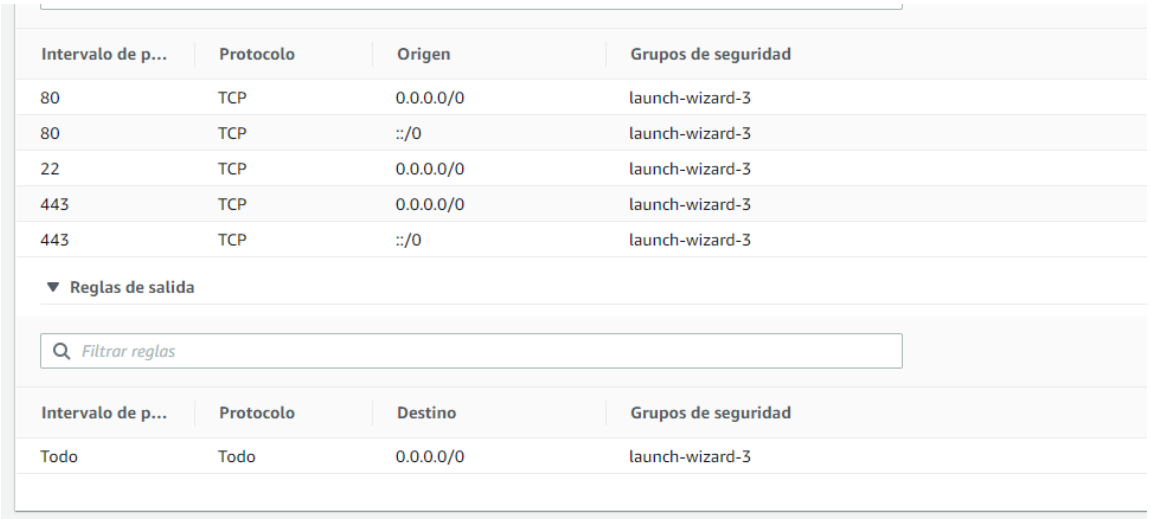
Figuras

Creación de ec2

Para la creación del ec2 utilizamos Ubuntu 18.04 lts, dejando por defecto la configuración de aws menos con los puertos

con la siguiente configuración de puertos

Figura 1



Intervalo de p...	Protocolo	Origen	Grupos de seguridad
80	TCP	0.0.0.0/0	launch-wizard-3
80	TCP	::/0	launch-wizard-3
22	TCP	0.0.0.0/0	launch-wizard-3
443	TCP	0.0.0.0/0	launch-wizard-3
443	TCP	::/0	launch-wizard-3

▼ Reglas de salida

Filtrar reglas

Intervalo de p...	Protocolo	Destino	Grupos de seguridad
Todo	Todo	0.0.0.0/0	launch-wizard-3

Para entrar a la nuestra maquina de ec2 utilizamos putty ingresando las claves suministradas por aws y la dirección para conectar

Instalación de docker

sudo apt update

sudo apt install apt-transport-https ca-certificates curl software-properties-common

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add –

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu bionic stable"

sudo apt update

sudo apt install docker-ce

Instalación de docker compose

sudo curl -L https://github.com/docker/compose/releases/download/1.21.2/docker-compose-`uname -s`-`uname -m` -o /usr/local/bin/docker-compose

sudo chmod +x /usr/local/bin/docker-compose

Configuración de wordpress

Creamos una carpeta para nuestro wordpress

```
mkdir wordpress && cd wordpress
mkdir nginx-conf
nano nginx-conf/nginx.conf
server {
    listen 80;
    listen [::]:80;

    server_name example.com www.example.com;

    index index.php index.html index.htm;

    root /var/www/html;

    location ~ /.well-known/acme-challenge {
        allow all;
        root /var/www/html;
    }

    location / {
        try_files $uri $uri/ /index.php$is_args$args;
    }
```

```
        location ~ \.php$ {
                try_files $uri =404;
                fastcgi_split_path_info ^(.+\.php)(/.+)$;
                fastcgi_pass wordpress:9000;
                fastcgi_index index.php;
                include fastcgi_params;
                fastcgi_param SCRIPT_FILENAME
$document_root$fastcgi_script_name;
                fastcgi_param PATH_INFO $fastcgi_path_info;
        }

        location ~ /\.ht {
                deny all;
        }

        location = /favicon.ico {
                log_not_found off; access_log off;
        }
        location = /robots.txt {
                log_not_found off; access_log off; allow all;
        }
        location ~* \.(css|gif|ico|jpeg|jpg|js|png)$ {
                expires max;
                log_not_found off;
        }
}
```

Cambiando example.com por nuestra dirección de freenom

Para definir variables de entorno para nuestro mysql

nano .env

```
MYSQL_ROOT_PASSWORD=your_root_password
MYSQL_USER=your_wordpress_database_user
MYSQL_PASSWORD=your_wordpress_database_password
```

nano .dockerignore

```
.env
.git
docker-compose.yml
.dockerignore
```

nano docker-compose.yml

```
version: '3'

services:
  db:
    image: mysql:8.0
    container_name: db
    restart: unless-stopped
    env_file: .env
    environment:
      - MYSQL_DATABASE=wordpress
    volumes:
```

```
    - dbdata:/var/lib/mysql
  command: '--default-authentication-plugin=mysql_native_password'
  networks:
    - app-network

wordpress:
  depends_on:
    - db
  image: wordpress:5.1.1-fpm-alpine
  container_name: wordpress
  restart: unless-stopped
  env_file: .env
  environment:
    - WORDPRESS_DB_HOST=db:3306
    - WORDPRESS_DB_USER=$MYSQL_USER
    - WORDPRESS_DB_PASSWORD=$MYSQL_PASSWORD
    - WORDPRESS_DB_NAME=wordpress
  volumes:
    - wordpress:/var/www/html
  networks:
    - app-network

webserver:
  depends_on:
    - wordpress
  image: nginx:1.15.12-alpine
```

```
    container_name: webserver
    restart: unless-stopped
    ports:
      - "80:80"
    volumes:
      - wordpress:/var/www/html
      - ./nginx-conf:/etc/nginx/conf.d
      - certbot-etc:/etc/letsencrypt
    networks:
      - app-network

  certbot:
    depends_on:
      - webserver
    image: certbot/certbot
    container_name: certbot
    volumes:
      - certbot-etc:/etc/letsencrypt
      - wordpress:/var/www/html
    command: certonly --webroot --webroot-path=/var/www/html --email
sammy@example.com --agree-tos --no-eff-email --staging -d example.com -d
www.example.com

volumes:
  certbot-etc:
  wordpress:
  dbdata:
```

```
networks:
  app-network:
    driver: bridge
```

cambiamos los examples por nuestros datos

sudo docker-compose up -d

Credenciales SSL

Sudo nano docker-compose.yml

Sudo docker-compose up --force-recreate --no-deps certbot

Modificación de la configuración del servidor web

Sudo docker-compose stop webserver

curl -sSLo nginx-conf/options-ssl-nginx.conf https://raw.githubusercontent.com/certbot/certbot/master/certbot-nginx/certbot_nginx/_internal/tls_configs/options-ssl-nginx.conf

sudo rm nginx-conf/nginx.conf

sudo nano nginx-conf/nginx.conf

```
server {
        listen 80;
        listen [::]:80;
```

```
        server_name example.com www.example.com;

        location ~ /.well-known/acme-challenge {
                allow all;
                root /var/www/html;
        }

        location / {
                rewrite ^ https://$host$request_uri? permanent;
        }
}

server {
        listen 443 ssl http2;
        listen [::]:443 ssl http2;
        server_name example.com www.example.com;

        index index.php index.html index.htm;

        root /var/www/html;

        server_tokens off;

        ssl_certificate /etc/letsencrypt/live/example.com/fullchain.pem;
        ssl_certificate_key /etc/letsencrypt/live/example.com/privkey.pem;
```

```
    include /etc/nginx/conf.d/options-ssl-nginx.conf;


    add_header X-Frame-Options "SAMEORIGIN" always;
    add_header X-XSS-Protection "1; mode=block" always;
    add_header X-Content-Type-Options "nosniff" always;
    add_header Referrer-Policy "no-referrer-when-downgrade" always;
    add_header Content-Security-Policy "default-src * data: 'unsafe-eval' 'unsafe-inline'" always;
    # add_header Strict-Transport-Security "max-age=31536000; includeSubDomains; preload" always;
    # enable strict transport security only if you understand the implications


    location / {
        try_files $uri $uri/ /index.php$is_args$args;
    }


    location ~ \.php$ {
        try_files $uri =404;
        fastcgi_split_path_info ^(.+\.php)(/.+)$;
        fastcgi_pass wordpress:9000;
        fastcgi_index index.php;
        include fastcgi_params;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        fastcgi_param PATH_INFO $fastcgi_path_info;
    }
```

```
        location ~ /\.ht {
                deny all;
        }

        location = /favicon.ico {
                log_not_found off; access_log off;
        }
        location = /robots.txt {
                log_not_found off; access_log off; allow all;
        }
        location ~* \.(css|gif|ico|jpeg|jpg|js|png)$ {
                expires max;
                log_not_found off;
        }
}
```

Sudo nano docker-compose.yml

```
...
  webserver:
    depends_on:
      - wordpress
    image: nginx:1.15.12-alpine
    container_name: webserver
    restart: unless-stopped
    ports:
      - "80:80"
```

```
    - "443:443"
  volumes:
    - wordpress:/var/www/html
    - ./nginx-conf:/etc/nginx/conf.d
    - certbot-etc:/etc/letsencrypt
  networks:
    - app-network
```

Para mas información ingresar a esta pagina [1]

Renovación de SSL

nano ssl_renew.sh

```
#!/bin/bash

COMPOSE="/usr/local/bin/docker-compose --no-ansi"
DOCKER="/usr/bin/docker"

cd /home/ubuntu/wordpress/
$COMPOSE run certbot renew && $COMPOSE kill -s SIGHUP webserver
$DOCKER system prune -af
```

```
chmod +x ssl_renew.sh
sudo crontab -e
1
Al final del texto
0 12 * * * /home/ubuntu/wordpress/ssl_renew.sh >> /var/log/cron.log 2>&1
```

Configurar cloudflare con nuestro cname de freenom

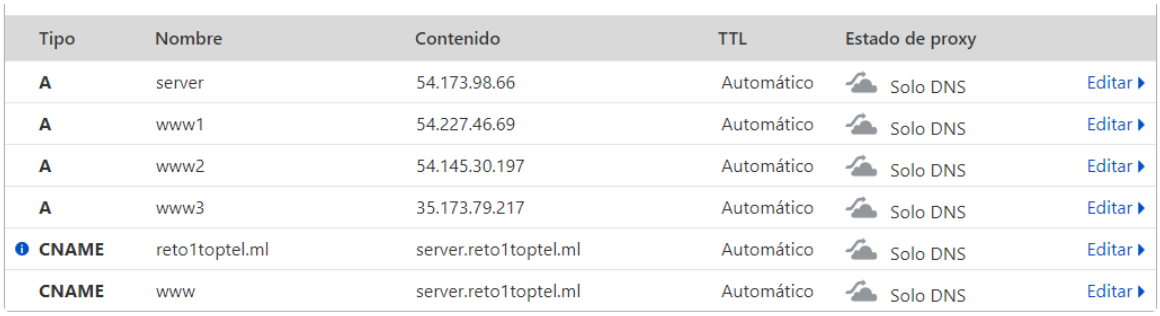
Al entrar a cloudflare y poner nuestro cname, cloudflare nos pide que eliminemos los server default de freenom y pongamos los de cloudflare quedando así:

Figura 2

Configuración de server en freenom

Figura 3

Configuración de dns en cloudFlare



Tipo	Nombre	Contenido	TTL	Estado de proxy	
A	server	54.173.98.66	Automático	Solo DNS	Editar
A	www1	54.227.46.69	Automático	Solo DNS	Editar
A	www2	54.145.30.197	Automático	Solo DNS	Editar
A	www3	35.173.79.217	Automático	Solo DNS	Editar
CNAME	reto1toptel.ml	server.reto1toptel.ml	Automático	Solo DNS	Editar
CNAME	www	server.reto1toptel.ml	Automático	Solo DNS	Editar

Bibliografía

[1] K. JUEll, “Cómo instalar WordPress con Docker Compose | DigitalOcean,” Jan. 09, 2020. https://www.digitalocean.com/community/tutorials/how-to-install-wordpress-with-docker-compose-es (accessed Apr. 17, 2021).