

# Configuración del Proxy en Nginx

AWS

Yeray Gutiérrez Mullor

## Contenido

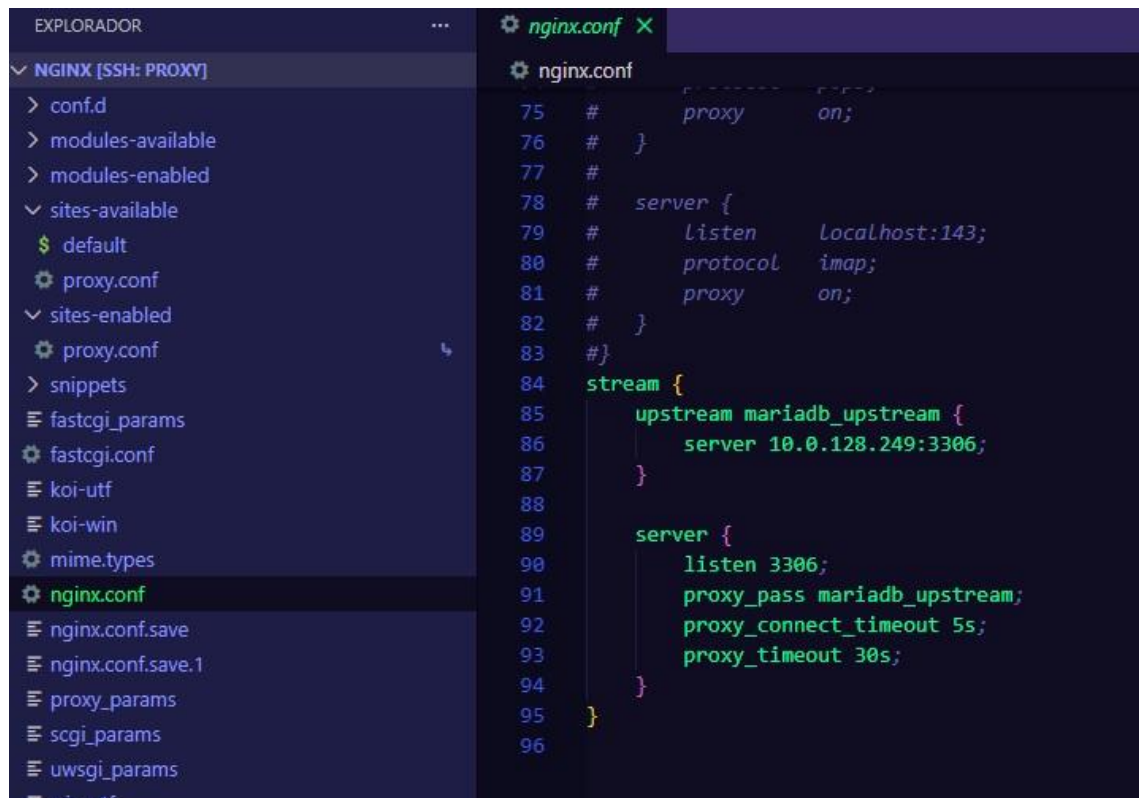
Proxy en Instancia NAT de AWS: .....	3
Configuración en Nginx:.....	3
1. Archivo nginx.conf: .....	3
2. Archivo proxy.conf: .....	3
Resultado: .....	4

## Proxy en Instancia NAT de AWS:

Para este proyecto, en AWS tenemos una VPC con 2 subredes, la privada y la pública, en la pública esta la Instancia NAT configurada anteriormente, y en la privada el servidor web; Como desde internet no se puede acceder a las instancias de la subred privada, para poder implementar y acceder a nuestro sitio web debemos implementar un Proxy que actúe de puente entre el cliente y la web, para ello configuramos un proxy en el servidor web nginx.

## Configuración en Nginx:

### 1. Archivo nginx.conf:



```
75 #       proxy      on;
76 #     }
77 #
78 #     server {
79 #         listen      localhost:143;
80 #         protocol     imap;
81 #         proxy        on;
82 #     }
83 #}
84 stream {
85     upstream mariadb_upstream {
86         server 10.0.128.249:3306;
87     }
88
89     server {
90         listen 3306;
91         proxy_pass mariadb_upstream;
92         proxy_connect_timeout 5s;
93         proxy_timeout 30s;
94     }
95 }
96
```

- Bloque upstream: Aquí se define el destino, creando un grupo llamado mariadb\_upstream que apunta a la IP privada del servidor web (10.0.128.249) en el puerto estándar de MariaDB/MySQL (3306) para poder acceder al RDS.
- Bloque server: Le dice a Nginx el puerto que tiene que escuchar.
- proxy\_pass: Redirige cualquier conexión que llegue a ese puerto directamente hacia el upstream de arriba
- Timeouts: Aquí he configurado un tiempo de espera de conexión de 5 segundos y un tiempo de inactividad de 30 segundos, para no dejar conexiones colgadas si la base de datos no responde.

### 2. Archivo proxy.conf:

En lugar de configurar un sitio web normal y corriente, se configurara de esta manera para que el “sitio” actúe como proxy:

```
1 server {
2     listen 80;
3     server_name _;
4
5     root /var/www/html;
6
7     index index.php;
8
9     location / {
10         proxy_pass http://10.0.128.249:80;
11         proxy_set_header Host $host;
12         proxy_set_header X-Real-IP $remote_addr;
13     }
14 }
15
```

- listen 80: El proxy está atento a cualquier petición web que llegue por el puerto 80
- server\_name \_: Indica que este bloque de servidor responderá a cualquier dominio o dirección IP que apunte a esta máquina.
- location / { ... }: Define que todas las peticiones (la raíz / y todo lo que cuelgue de ella) deben ser enviadas al backend.
- proxy\_pass http://10.0.128.249:80: Aquí es donde Nginx actúa como puente. Recibe la petición del usuario y la reenvía al servidor web privado por HTTP.
- Cabeceras (proxy\_set\_header):
  - o Host \$host: Pasa el nombre del sitio original al servidor backend.
  - o X-Real-IP \$remote\_addr: Le dice al backend cuál es la IP real del usuario, para que en los logs del backend no aparezca siempre la IP del proxy.

## Resultado:

Al acceder a la IP publica del proxy, se nos redirige al servidor web en la instancia privada, donde está alojado el sitio web real

