Table of Contents

Ngin-Proxy y ACME-Companion 1
General Architecture 1

Ngin-Proxy y ACME-Companion

nginx, acme, docker, letsencrypt

Lo que se necesita

- Un host con una distro Linux y una IP a la que pueda acceder desde internet (ya sea IP pública o con puertos redirigidos en el router de casa).
- Un nombre de dominio registrado a ese host (e.g. duckdns.org los ofrece gratuitos o los propios que te regalan los servicios VPS).

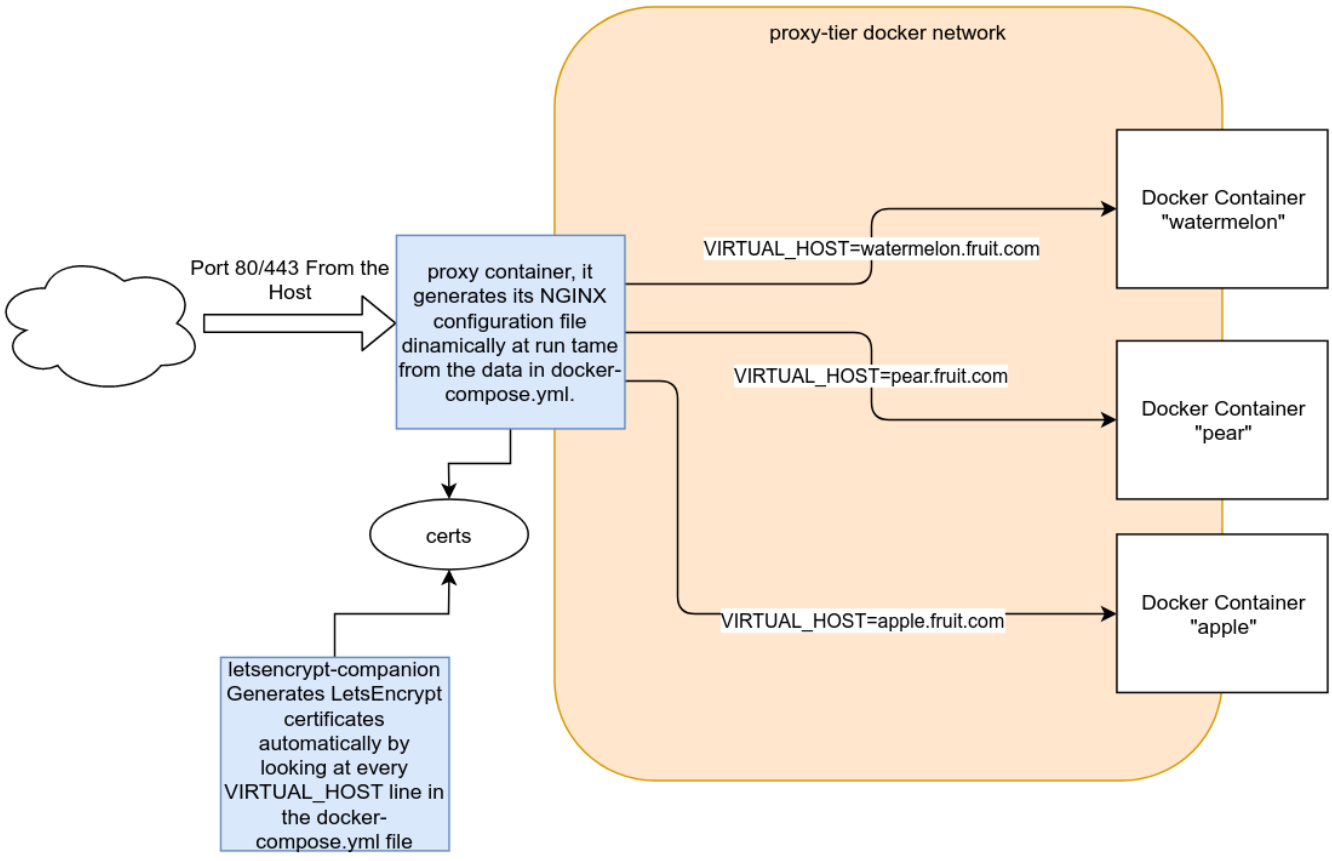
Lo que ofrece

Cualquier servicio web que desee lanzar con docker, será accesible mediante un subdominio con HTTPS con certificados válidos y sin apenas nada de trabajo extra.

Docker y Letsencrypt con los containers asistentes

	Let's Encrypt Companion	NginX proxy
——-	————————————————	———————————————-
Viejo	https://hub.docker.com/r/jrcs/letsencrypt-nginx-proxy-companion	https://hub.docker.com/r/jwilder/nginx-proxy
	https://github.com/jwilder/docker-letsencrypt-nginx-proxy-companion	
Nuevo	https://hub.docker.com/r/nginxproxy/acme-companion	https://hub.docker.com/r/nginxproxy/nginx-proxy
	https://github.com/nginx-proxy/acme-companion	https://github.com/nginx-proxy/nginx-proxy

General Architecture



Con esta configuración es suficiente para tener lanzado el Jira Software *pelado*. Sin embargo, este ejemplo está preparado para ir añadiendo más cosas.

Hay varios aspectos clave a tener en cuenta para entender este ejemplo de docker compose y poder trabajar con él más allá del ejemplo básico que sólo lanza Jira Software con LetsEncrypt.

- El punto de entrada al que está expuesto todo el ecosistema es el container `proxy` que es el único que abre puertos a la red del anfitrión. Su uso y configuración es clave.
- Muchos images docker de muchas aplicaciones llevan incrustado un nginx dentro (por algún motivo desconocido), el ejemplo más directo de ésto es BookStack que tiene un nginx dentro del container. No confundirlos entre el `proxy` que es el importante de este proyecto.
- El container `proxy` es una versión especial de nginx con una funcionalidad muy interrelacionada con el container de LetEncrypt. La clave está en que puedes añadir tantas aplicaciones al docker-compose como quieras, siempre que especifiques un `VIRTUAL_HOST` con un subdominio.
- OJO, en ningún momento necesitaremos modificar o editar a mano el archivo de configuración de `nginx.conf`. El container `proxy` generará uno dinámicamente a partir de la información contenida en `docker-compose.yml`. Podremos inspeccionar los contenidos generados si entramos en el container `proxy` y abrimos `/etc/nginx/conf.d/` y miramos los archivos de configuración.

Para añadir una nueva app con un nuevo subdominio, basta con hacer lo siguiente:

- En la máquina host, añadir una entrada por cada dominio y todos lo subdominios en `/etc/hosts` para que se resuelvan todos a `127.0.0.1`. Comprobar tambien con `dig` que lo subdominios se resuelven todos a la misma IP pública del host. E.g.

```
dig @8.8.8.8 watermelon.fruit.com
```

Donde `watermelon.fruit.com` es el dominio que queremos probar que apunta a la dirección IP pública de mi máquina. Para ello le hacemos el query a los servidores DNS de Google `8.8.8.8`.

Nota: si el servidor DNS 8.8.8.8 no responde la dirección pública de nuestro host, debemos solucionar eso lo primero antes de poder continuar.

- Debe estar en la misma red docker que el container `proxy`. En este caso debe estar en la red `proxy-tier`.

```
networks:
  - proxy-tier
```

- Añadir la configuración necesaria para que se configuren dinámicamente tanto el container `proxy` como el container letsencrypt.

```
    environment:
      - VIRTUAL_HOST=watermelon.fruit.com
      - LETSENCRYPT_HOST=watermelon.fruit.com
      - LETSENCRYPT_EMAIL=vegetables@fish.com
```

OJO, deben ser lo mismo VIRTUAL*HOST y LETSENCRYPT*HOST.

Estas líneas son importantes por que se debe configurar con ellas tanto el container LetsEncrypt como también el container `proxy` que recordemos que configurará su `nginx` dinámicamente a partir de la información contenida en el archivo `docker-compose.yml` en tiempo de ejecución.

Nótese que cada vez que añadamos un subdominio nuevo, se realizará un proceso *challenge* para generar unos certificados firmados por LetsEncrypt.

Es decir, si tenemos un dominio y varios subdominios, tendrán cada uno su certificado correspodiente, e.g.

`VIRTUAL_HOST=fruit.com` → generará un certificado LetsEncrypt
`VIRTUAL_HOST=watermelon.fruit.com` → generará un certificado LetsEncrypt distinto para este subdominio específicamente.

Nota: los certificados generados para el host `fruit.com` y `watermelon.fruit.com` serán distintos.