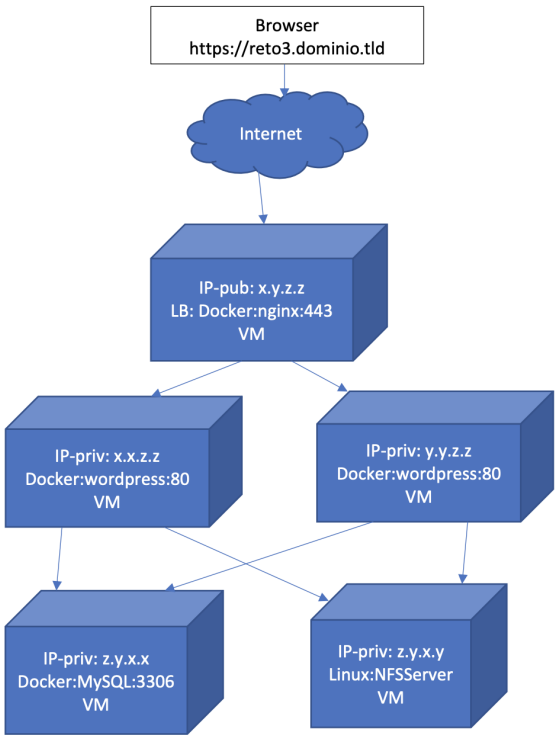
Guía para reto 3: Tópicos especiales en telemática

By: Sebastián Guerra

Para el reto 3 nos piden instanciar un CMS con redundancia en dos instancias, soportando los datos en una base de datos y los archivos de gestión en un sistema NFS. Finalmente, montar un reverse proxy con NGINX que actúa como puerta de entrada a tu VPC. A continuación, una imagen de la arquitectura:



Ahora bien, en vez de trabajar con MySQL lo haremos con Postgres como base de datos y con Drupal como CMS. Dejo bajo tu responsabilidad la configuración de los security groups con base en los puertos correspondientes (el de Postgres es el 5432 y el de NFS 2049) y la descarga de Docker y Docker Compose que está en este repositorio sobre cada instancia.
Una friendly reminder es que si tienes problemas con permisos y/o claves uses el comando sudo o le des permisos al directorio con el comando chmod.

Configuración de la base de datos

Partiremos con esta configuración. Para empezar, te comparto el docker-compose.yml que usarás en la instancia de la DB. Lo puedes abrir con `nano docker-composer.yml` y pegas:

```
version: '3.8'
services:
  db:
    image: postgres:14.1-alpine
    restart: always
    environment:
      - POSTGRES_USER=postgres
      - POSTGRES_PASSWORD=postgres
    ports:
      - '5432:5432'
    volumes:
      - db:/var/lib/postgresql/data
volumes:
  db:
    driver: local
```

Tras esto, lo guardas y lo corres con:

```
docker-compose up -d
```

Esto correrá una imagen liviana de Postgres sobre la que vamos a inicializar la base de datos como tal, sobre la cual accederemos a su contenedor. Para poder acceder requieres saber el nombre o el ID con el que se creó con el comando `docker ps`:

```
docker exec -it ID_DE_TU_CONTENEDOR /bin/bash
```

Dentro de la máquina empiezas a correr los siguientes comandos, pues son unas configuraciones necesarias para instanciar la base de datos como tal, te dejo en libertad el nombre de la DB.

```
apk update
apk add postgresql-contrib
psql -U postgres -d mi_basedatos -c "CREATE EXTENSION IF NOT EXISTS pg_trgm;"
```

Una vez que la base de datos este creada, con `exit` sales de la ejecución del bash en el contenedor y anota la IP privada de esta máquina para más tarde.

Configuración del NFS

Para esta parte es seguir los pasos del siguiente tutorial. Aquí te dejo algunos comentarios sobre consideraciones de la lectura de este tutorial (leer PRIMERO).

- Concéntrate primero en los comandos con fondo azul oscuro, que son sobre los que se va construyendo el servidor
- En esta guía se trabaja con dos carpetas, una para propósitos generales (example 1) y otra con permisos de administrador (example 2). Solo sigue los pasos sobre el segundo porque es el único que vas a necesitar.
- La parte del firewall no es necesaria, y puede ser causante de que te rompa la conexión SSH si no la declaras, para este proyecto si se realizó añadiendo el puerto 22 al ufw.

Luego de que termines de configurar el servidor, puedes cambiarte a una de las dos máquinas donde se va a alejar el contenedor de Drupal, a partir de aquí se continua con la configuración del cliente que son los comandos con fondo azul claro. Recuerda que solo hace falta trabajar con el de permisos de administrador, así que solo usa el comando respectivo.

Como última recomendación, realizar el paso 7 es importante para que durante el boot de cada máquina, este proceso se automatice y no toque hacer mount periódicamente.

Extracción de recursos de Drupal

Dado que ya tenemos un sistema de archivos listo para guardar, en la misma instancia donde hiciste el cliente empezaremos a guardar todos los archivos de configuración necesarios para Drupal. Esto lo vamos a alojar en el NFS Server para que, a pesar de que los contenedores mueran, la metadata del servicio se mantenga disponible y permita construir mirrors del CMS (es decir, configurar más instancias con contenedores distancias pero con la misma información para mejorar su tolerancia a fallos).

Para esto empezaremos a construir el contenedor más básico de Drupal. A continuación de te dejo el *docker-composer.yml* que usarás:

```
version: '3'
services:
  drupal:
    image: drupal:latest
    ports:
      - "80:80"
```

Una vez hecho esto, guardas y corres con:

```
docker-compose up -d
```

Una vez hecho esto, el contenedor va a estar creado y, por lo tanto, todo el contenido de la imagen de Drupal ya va a estar cargada.

El siguiente paso es continuar con la creación del sitio. Esto lo harás accediendo desde el navegador web con la ip pública de tu VM (donde tienes el contenedor de Drupal). A lo largo de este formulario te pedirán la IP privada de tu base de datos y algunas configuraciones adicionales, gestiona y espera que se cree el sitio de manera exitosa.

Luego de esto, lo que haremos es migrar esa información hacía nuestro NFS Server. Lo haremos creando una copia sobre la ruta donde se aleja todos estos datos por dentro del contenedor, y exportándola en la ruta que definiste en el NFS Client. Y al estar este ya configurado, lo que pasará es que toda esta data irá a parar a nuestro servidor.

```
cd /nfs/home
docker cp TU_CONTENEDOR_DRUPAL:/opt/drupal ./
```

La intención del primer comando es que te ubiques en la ruta donde hiciste el mount, osea, el punto donde está el cliente NFS. Luego, haciendo referencia al nombre o al ID de tu contenedor, se copiará todos los archivos bajo esa ruta dentro del contenedor sobre la carpeta en la que estás parado en tu VM. Este comando tomará bastante tiempo así que te alcanzas a preparar un cafecito mientras se va exportando.

Una vez termine esto, podemos pasar a vincular todo esto en una segunda instancia de Drupal

Vincular una segunda máquina de Drupal

Para esta parte corresponde con todo el procedimiento inicial de nuestra segunda máquina en Drupal. Es decir, tener instalado Docker y demás herramientas. Para continuar con esta parte, debes de volver a configurar el NFS Client y asociarlo con el servidor. Aquí cobra relevancia haber configurado de forma correcta las IPs privadas de las instancias de Drupal para que te acepte la conexión sin tener que modificar el servidor. Si no lo hiciste, agrega la nueva regla y reinicia el servidor.

Una vez ya lo tengas listo, podemos pasar a una v2.0 de nuestro *docker-compose.yml*, que va a utilizar la estrategia de bind mounts para poder complementar la información del contenedor con base en lo que se le defina en tu máquina. Sin embargo, esa información en realidad no está en tu máquina actual, sino que está en el servidor NFS!

Aquí dejo el archivo docker-compose.yml:

```
version: '3.8'

services:

  drupal:
    image: drupal:latest
    ports:
      - 80:80
    volumes:
      - /nfs/home/drupal/web:/var/www/html
    restart: always
```

Lo que está con amarillo es lo nuevo sobre la versión anterior. Lo que va a hacer esto es:

- La ruta que está antes de los dos puntos va a ser el mount del cliente NFS, y la que está después es la ruta interna del contenedor
- Esto establece que todo lo que esté en la primera ruta, la debe usar el contenedor en la segunda ruta.
- Por lo tanto, estamos estableciendo una referencia directa desde el contenedor al servidor NFS por transitividad
- Lo último es una flag adicional que actúa como watchdog del contenedor: si muere por cualquier razón, créame otro con lo mismo.

Con esto hecho y guardado, crea el contenedor con:

```
docker-compose up -d
```

Ahora ve al navegador web y coloca la IP pública de la máquina donde acabamos de hacer este proceso (la segunda de Drupal). En este momento, deberías ver exactamente la misma página web que creamos al principio!

Ahora todo esto que hicimos lo debes de hacer nuevamente en la primera instancia de Drupal para que ambos queden con un bind mount hacía el servidor NFS. No hace falta recrear el cliente NFS, solamente tumbar el contenedor actual, usar este nuevo docker-compose y levantar:

Para dar de baja el primer contenedor puedes usar:

```
docker-compose down
```

Una vez tengas todo, debes de hacer algunas pruebas como login, creación y modificación de posts. Ve alternando entre instancias y los cambios que hagas se deben ver reflejados en la otra.

Configuración de NGINX

Teniendo dos instancias con la misma información, nos queda configurar un reverse proxy que redirija el tráfico hacía estas dos máquinas.

Para hacer esto, crearas un archivo de configuración de NGINX bajo el nombre de *nginx.conf*. Dentro de él, le pondrás lo siguiente:

```
worker_processes auto;
error_log /var/log/nginx/error.log;
pid /run/nginx.pid;

events {
  worker_connections  1024;
}
http {

upstream backend {
        server my.dummy.ip.1; #Aquí va la IP privada donde tienes el container 1
        server my.dummy.ip.2; #Aquí va la IP privada donde tienes el container 2
    }

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

        server_name _;

        location / {
            proxy_pass http://backend;
            proxy_redirect off;
            proxy_set_header Host $host;
            proxy_set_header X-Real-IP $remote_addr;
            proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
            proxy_set_header X-Forwarded-Host $host;
            proxy_set_header X-Forwarded-Server $host;
            proxy_set_header X-Forwarded-Proto $scheme;
        }
    }
}
```

Con esto hecho, proseguiremos con la instanciación de un contenedor de NGINX. Recuerda que para este paso debes tener Docker y sus herramientas instaladas. Aquí te dejo el *docker-compose.yml*

```
version: '3.1'
services:
  nginx:
    container_name: nginx
    image: nginx
    volumes:
    - /home/ubuntu/nginx.conf:/etc/nginx/nginx.conf:ro
    ports:
    - 80:80
```

En este archivo estás definiendo nuevamente un bind mount, en el que dices que el contenedor debe de utilizar el archivo que creaste previamente como configuración del reverse proxy.

Una vez hecho esto, le haces up a tu contenedor y te vas a AWS para crear una IP elástica. Esta IP la vas a enlazar a la máquina donde creaste el contenedor de NGINX.

Como última prueba, lo que debes de hacer es tomar esa IP elástica y ponerla en tu navegador web. El resultado esperado es que puedas ver de forma satisfactoria tu página de Drupal, internamente el contenedor ya está haciendo forwarding del tráfico hacía tus instancias de Drupal según él lo considere.

FELICITACIONES, LOGRASTE HACER EL RETO 3

Recursos adicionales

- Si configuras un DNS, esta herramienta te ayuda a revisar la propagación.
- Si quieres configurar un certificado SSL para tráfico HTTPS, aquí está el repositorio.
- Aquí está más claro si quieres crear un firewall en tu servidor NFS.