Levantar el proyecto automatizado

Primero se arman las imagenes en docker tanto del api como del interceptor con los comandos (todo desde la raíz del proyecto):

- docker build -t dns-api:latest -f .\dns-api\Docker\Dockerfile .\dns-api
- docker build -t dns-interceptor:latest -f .\dns-interceptor\docker\Dockerfile .\dns-interceptor

Luego desplegar con Helm todo el proyecto:

- helm upgrade --install dns-project .\chart --namespace dns --create-namespace

docker compose down -v
docker compose up --build

*** Es posible que en LENS o al ejecutar "kubectl get pods -n dns" alguno de los 2 o ambos no aparezcan como "Running" al principio, les toma tiempo levantar.

*** Para probar el estado de la api. Es posible que haya que cambiar el puerto.

En lens / network / services / dns-api-dns-project-api --- en el port, se reemplaza por el puerto que diga curl http://localhost:32731/healthz

En caso de necesitar editar algún archivo

En buena teoria reconstruyendo las imagenes en docker con los comandos de arriba y volviendo a ejecutar "helm upgrade --install dns-project .\chart --namespace dns --create-namespace" se actualizan los pods y las imagenes, sin embargo me ocurrió un error que hacía que no funcionara si no borraba todo y lo reinstalaba así que para eliminar todo de forma segura se hace lo siguiente:

- Eliminar el realease de Helm "helm uninstall dns-project -n dns" y "kubectl delete namespace dns"
- Eliminar imagenes de docker "docker rmi -f dns-interceptor:latest" y "docker rmi -f dns-api:latest"
- una vez hecho eso y confirmando que se eliminó todo se ejecutan de nuevo los comandos de "Levantar el proyecto automatizado"

Para probar el interceptor

- Hay que levantar un pod de prueba que se comunique con el interceptor con el comando "kubectl run -it --rm --restart=Never dns-tester -n dns --image=busybox:stable -- sh"
- Hay 2 casos posibles query estándar (QR = 0 y OPCODE = 0) y query no estándar (QR != 0 y/o OPCODE != 0)

Caso query NO estándar:

- Ejecutar dentro del pod el comando "nslookup -type=SOA example.com dns-api-dns-project-interceptor"
- Debe retornar algo similar a esto:
  Server: dns-api-dns-project-interceptor
  Address: 10.96.220.74:53

Non-authoritative answer:
example.com
origin = ns.icann.org
mail addr = noc.dns.icann.org
serial = 2025082261
refresh = 7200
retry = 3600
expire = 1209600
minimum = 3600

Caso query estándar:

**Caso tipo "Single":

- Ejecutar dentro del pod el comando "nslookup single.example.com dns-api-dns-project-interceptor"
- Deberia retornar algo similar a esto:
  Server: dns-api-dns-project-interceptor
  Address: 10.96.220.74:53

Non-authoritative answer:
Name: single.example.com
Address: 198.135.127.5

**Caso tipo "Multi":

- Ejecutar dentro del pod el comando "for i in $(seq 1 6); do nslookup multi.example.com dns-api-dns-project-interceptor; echo ""; done"
- Deberia retornar algo similar a esto:

Server: dns-api-dns-project-interceptor
Address: 10.96.220.74:53

Non-authoritative answer:
Name: multi.example.com
Address: 198.135.126.10

** server can't find multi.example.com: NXDOMAIN

Server: dns-api-dns-project-interceptor
Address: 10.96.220.74:53

Non-authoritative answer:
Name: multi.example.com
Address: 198.135.127.10

** server can't find multi.example.com: NXDOMAIN

Server: dns-api-dns-project-interceptor
Address: 10.96.220.74:53

Non-authoritative answer:
Name: multi.example.com
Address: 198.135.126.10

** server can't find multi.example.com: NXDOMAIN

Server: dns-api-dns-project-interceptor
Address: 10.96.220.74:53

Non-authoritative answer:
Name: multi.example.com
Address: 198.135.127.10

** server can't find multi.example.com: NXDOMAIN

Server: dns-api-dns-project-interceptor
Address: 10.96.220.74:53

Non-authoritative answer:
Name: multi.example.com
Address: 198.135.126.10

** server can't find multi.example.com: NXDOMAIN

Server: dns-api-dns-project-interceptor
Address: 10.96.220.74:53

Non-authoritative answer:
Name: multi.example.com
Address: 198.135.127.10

**Caso tipo "Weight":

- Ejecutar dentro del pod el comando "for i in $(seq 1 20); do nslookup weight.example.com dns-api-dns-project-interceptor | grep Address; done"
- Deberia retornar algo similar a esto:

Address: 10.96.220.74:53
Address: 198.135.167.20
Address: 10.96.220.74:53
Address: 198.135.167.20

Address: 10.96.220.74:53
Address: 198.135.167.20
Address: 10.96.220.74:53
Address: 198.136.186.20
Address: 10.96.220.74:53
Address: 198.136.186.20
Address: 10.96.220.74:53
Address: 198.135.167.20
Address: 10.96.220.74:53
Address: 198.135.167.20
Address: 10.96.220.74:53
Address: 198.135.167.20
Address: 10.96.220.74:53
Address: 198.135.167.20
Address: 10.96.220.74:53
Address: 198.135.167.20
Address: 10.96.220.74:53
Address: 198.136.186.20
Address: 10.96.220.74:53
Address: 198.135.167.20
Address: 10.96.220.74:53
Address: 198.136.186.20
Address: 10.96.220.74:53
Address: 198.136.186.20
Address: 10.96.220.74:53
Address: 198.136.186.20
Address: 10.96.220.74:53
Address: 198.136.186.20
Address: 10.96.220.74:53
Address: 198.136.186.20
Address: 10.96.220.74:53
Address: 198.135.167.20
Address: 10.96.220.74:53
Address: 198.135.167.20
Address: 10.96.220.74:53
Address: 198.135.167.20

**Caso tipo "roundtrip":

- Ejecutar dentro del pod el comando "for i in $(seq 1 5); do nslookup roundtrip.example.com dns-api-dns-project-interceptor; echo ""; done"
- Deberia retornar algo similar a esto:

Server: dns-api-dns-project-interceptor
Address: 10.96.220.74:53

Non-authoritative answer:
Name: roundtrip.example.com
Address: 198.137.87.70

** server can't find roundtrip.example.com: NXDOMAIN

Server: dns-api-dns-project-interceptor
Address: 10.96.220.74:53

Non-authoritative answer:
Name: roundtrip.example.com
Address: 198.137.87.70

** server can't find roundtrip.example.com: NXDOMAIN

Server: dns-api-dns-project-interceptor
Address: 10.96.220.74:53

Non-authoritative answer:
Name: roundtrip.example.com
Address: 198.137.87.70

** server can't find roundtrip.example.com: NXDOMAIN

Server: dns-api-dns-project-interceptor
Address: 10.96.220.74:53

Non-authoritative answer:
Name: roundtrip.example.com
Address: 198.137.87.70

** server can't find roundtrip.example.com: NXDOMAIN

Server: dns-api-dns-project-interceptor
Address: 10.96.220.74:53

Non-authoritative answer:
Name: roundtrip.example.com
Address: 198.137.87.70

** server can't find roundtrip.example.com: NXDOMAIN

**Caso tipo "geo":

- Ejecutar dentro del pod el comando "nslookup geo.example.com dns-api-dns-project-interceptor"
- Deberia retornar algo similar a esto:

Server: dns-api-dns-project-interceptor
Address: 10.96.220.74:53

Non-authoritative answer:
Name: geo.example.com
Address: 198.145.67.50

** server can't find geo.example.com: NXDOMAIN

Notas importantes

- El proyecto debe estar totalmente automatizado por lo que en la carpeta chart están todos los archivos .yaml necesarios para eso, Chart.yaml y Values.yaml levantan todo, los deployments de los demás componentes y sus configmaps y demás van en la carpeta chart/templates.
- Los dockerfiles van dentro de las carpetas de cada componente, por ejemplo el del interceptor va en dns-interceptor/docker/Dockerfile (con eso los comandos para levantar futuras imagenes se mantienen similares y se ejecutan en la raíz del proyecto).
- Al probar el interceptor se toparán con un mensaje que dice algo parecido a "** server can't find geo.example.com: NXDOMAIN", eso es culpa de algo con Ipv6 que realmente no afecta al funcionamiento del código, ya que las respuestas que da son las correctas.
- Para pruebas con ip´s que dependen directamwente de la región se deben modificar valores directamente en la base de Firebase al parecer.
- Para correr comandos de prueba del interceptor usando dig@, por alguna razón se debe crear un pod de pruebas diferente con el comando "kubectl run -it --rm --restart=Never dns-tester -n dns --image=debian:stable-slim -- bash" y se bede correr este otro comando antes "apt-get update && apt-get install -y dnsutils"