Sistemas distribuidos

Práctica 5: Kubernetes y Raft

Universidad de Zaragoza, curso 2022/2023

Juan Eizaguerri Serrano 816079

Jaime Domper Cerrada 816851

Índice

Diseño realizado

Se quiere poner en funcionamiento un servicio replicado de almacenamiento clave-valor basado en raft utilizando kubernetes.

Se quiere que si un nodo servidor se cae se reinicie automáticamente, y si era el líder otro nodo tome su puesto. Para esto se va a utilizar un controlador StatefulSet, que tiene incorporada tolerancia a fallos.

En primer lugar se crea un programa en golang que actuará como cliente.

Se modifica el fichero kind-with-registry.sh para lanzar un cluster de kubernetes con 3 réplicas de servidor y un cliente.

Los Docker de los servidores se crearán con una imagen scratch y ejecutarán el programa automáticamente. El cliente utilizará una imagen de Linux Alpine a la que el usuario podrá acceder para ejecutar el cliente.

El script go_statefulset.sh elimina cualquier pod que pueda estar activo y se utiliza kubectl para lanzar los pods definidos en statefulset_go.yaml, que ha sido modificado para que se lancen los servidores con las direcciones DNS del resto de réplicas como argumentos.

Por comodidad, también se ha desarrollado un script deleteS.sh que elimina los pods de servidores. Esto hace más fácil borrarlos de la máquina para que no se queden funcionando y generando logs.

Puesta en marcha

Se lanza el cluster de kubernetes

```
./kind-with-registry.sh
```

Se compila el código del cliente y del servidor utilizando el comando

```
CGO_ENABLED=0 go build -o . .
```

Una vez obtenidos los ejecutables, se mueven al directorio dockerfiles, donde se preparan los contenedores con Docker.

```
docker build . -t localhost:5001/cliente:latest
docker build . -t localhost:5001/servidor:latest
```

Se suben los contenedores al repositorio local para que sean accesibles por kubernetes.

```
docker push localhost:5001/cliente:latest
docker push localhost:5001/servidor:latest
```

Se ejecutan los scripts de bash go_statefulset.sh y go_pods.sh

```
./go_statefulset.sh
./go_pods.sh
```

Los servidores se lanzan automáticamente, habrá que entrar a la shell del pod con el cliente para lanzar el test.

```
kubectl exec c1 -ti -- sh
```

Desde dentro del pod, lanzamos el ejecutable:

```
cliente ss-0.ss-service.default.svc.cluster.local:6000
ss-1.ss-service.default.svc.cluster.local:6000
ss-2.ss-service.default.svc.cluster.local:6000
```

Podremos ver la salida del test por pantalla.

Para ver los logs de los servidores se puede ejecutar el siguiente comando

```
kubectl logs s1
```

Una vez terminado de utilizar el servicio, se pueden cerrar los servidores utilizando el script deleteS.sh

```
./deleteS.sh
```

O borrar el cluster completamente

```
kind delete cluster
```

Pruebas

Todas las pruebas se han realizado en el lab000 de la facultad.

El programa del cliente funciona a modo de test de la siguiente forma: Escribirá varios valores en el sistema clave-valor implementado con raft de los servidores, y después leerá dichos valores. Si los valores devueltos son los mismos que los introducimos previamente la salida es correcta. El sistema deberá funcionar incluso si se cae alguna réplica, y todos los nodos activos tendrán el mismo registro de operaciones.