Práctica 2: Despliegue de aplicaciones usando Kubernetes/Docker

Partiendo de la práctica 1, se pide desplegar las aplicaciones remotas a lo largo de un cluster. En la práctica anterior se deberían haber generado al menos los siguientes ejecutables (los alumnos podrían haber adaptado el número de ejecutables):

- Dos programas servidor: Uno con la clase FileManager/imp/stub y otro con la clase MultMatrix/imp/stub. Estos servidores recibían peticiones de ejecución de métodos desde los programas cliente a través de mensajes de red.
- Dos programas cliente: Uno con el main del programa FileManager, y otro con el main del programa MultMatrix. Cada uno de ellos realizan llamadas a procedimientos remotos a través de las interfaces implementadas, comunicándose con los programas servidores usados.

En la siguiente práctica se pide implementar un despliegue de las aplicaciones implementadas en la práctica 1, usando una red de ordenadores configurados en instancias/pods de kubernetes. El alumno deberá realizar una configuración de al menos 2 ordenadores en los que habrá un ordenador Master (contro plane) y uno o más nodos esclavos. En los esclavos estarán las aplicaciones tipo "servidor" implementadas en la práctica 1, instaladas en contenedores Docker y virtualizadas en despliegues de Kubernetes. Las aplicaciones "cliente" se ejecutarán desde el lado del usuario, en el propio ordenador del alumno . Puede estar todo en el mismo ordenador, o usar una red de ordenadores real para comunicar los nodos comentados anteriormente. En concreto, la práctica cubrirá los siguientes puntos:

- Creación de imágenes Docker
- Replicación de la imagen Docker a lo largo de un cluster
- Uso de Kubernetes para gestionar las replicaciones y balanceo de carga

Objetivos:

El alumno deberá desarrollar al menos una imagen docker con los programas "servidor" desarrollados en la práctica 1. Esta imagen deberá estar configurada con las librerías y archivos, puertos de red, etc... necesarios para poder albergar estos ejecutables y permitir su conexión con los clientes.

El cluster virtual contendrá al menos los siguientes nodos:

- Nodo/máquina master
- Un nodo/máquina esclavo: Al menos dos pods de kubernetes ejecutando cada uno uno de los servidores desarrollados en la práctica 1 (un pod para multMatrix y un pod para FileManager)
- Opcionalmente: En caso de haber implementado broker de objetos, éste se encontrará en un pod más dentro del sistema. Los programas cliente se conectarán a los pods con los servidores a través de este programa

En la máquina máster se tendrá los ejecutables “cliente” de la práctica 1. En los nodos esclavo se desplegarán las imágenes docker con los programas servidor. Se tendrá al menos dos nodos ejecutándose, uno para servir las operaciones de “FileManager”, y el segundo para servir las operaciones de “MultMatrix”.

El despliegue de aplicaciones deberá cubrir los siguientes objetivos:

- Al ejecutar uno de los clientes (el de fileManager o MultMatrix), este cliente se conectará al nodo que contenga su servidor de métodos remotos. En este caso, el alumno puede dejar grabada la IP/Puerto dentro del mismo programa cliente, para que se conecte automáticamente a la máquina deseada, o usar algún archivo de configuración.
- Los nodos esclavo estarán levantados antes de ejecutar los clientes. Cada uno de los nodos tendrá al menos un pod, uno con el servidor de FileManager y otro con el servidor de MultMatrix.
- Los clientes deberán funcionar de una forma similar a lo entregado en la práctica 1, esta vez conectándose a los nodos virtualizados de kubernetes.

Obligatorio: 5 puntos

Despliegue de las aplicaciones “servidor” usando Docker/kubernetes. Se deja cierta libertad de configuración, siempre que haya dos pods distintos dentro de una red de kubernetes. Adicionalmente, en caso de haber usado un broker de objetos, éste estará también en un pod del sistema a disposición del resto de programas.

Los clientes se ejecutarán desde la máquina “máster” de kubernetes. Se conectarán a los servicios exportados en kubernetes a través de alguno de los puertos configurados como servicios.

Opcional: hasta 4 puntos

Configuración con balanceo de carga, al menos dos nodos físicos esclavos, y más de 2 máquinas virtuales/pods por despliegue. Al menos dos máquinas virtuales/pods por aplicación servidor (se podrían repetir si se quiere ahorrar recursos, en ese caso debería haber al menos dos pods por “servidor de aplicación”). En este caso, los requisitos son los siguientes

1 punto:

Réplicas de la clase multMatrix. En este caso, se hará un despliegue de docker que permita ese balanceo de carga entre varios pods del mismo tipo. Se debe demostrar que se redirigen las peticiones a distintas máquinas (en la defensa, se deberá mostrar el contenido del pod con los archivos resultado usando matrices random)

3 puntos:

Réplicas de la clase FileManager. Dado que FileManager guarda archivos, y estos archivos deberían estar disponibles para todos los clientes, se necesita un directorio compartido entre todos los pods. Los archivos guardados en un pod deberían estar disponibles para todos los pods que estén dentro de la red que sirve FileManager. Se deja al alumno encontrar una solución óptima, pero probablemente facilite el uso de directorios compartidos entre todos los pods, usando sshfs, nfs o sistemas de volúmenes compartidos. Se valorará lo siguiente:

2 puntos:

- Usar un único nodo físico que contenga todos los pods del despliegue de fileManager. Se puede usar el driver hostpath usado en clase.

+1 punto (es decir,3 puntos en total):

- Configurar la carpeta usando volúmenes en NFS u otro gestor de directorios compartidos en red. Se puede consultar la siguiente documentación:
  - Crear un servidor de NFS:

    https://www.digitalocean.com/community/tutorials/how-to-set-up-an-nfs-mount-on-ubuntu-20-04-es

  - Uso de volúmenes compartidos en kubernetes (sección NFS):

    https://kubernetes.io/docs/concepts/storage/volumes/

Opcional: 1 punto

Uso de varios nodos de tipo máquina EC2. En este caso, lo mínimo que se pide son dos ordenadores “esclavo” dentro de la red de kubernetes. Uno de ellos ejecutará el despliegue de multMatrix y otro FileManager. Adicionalmente, en caso de usar broker de objetos, se puede añadir una tercera máquina esclavo para ello con el despliegue indicado.

Entrega:

Las prácticas se pueden realizar por grupos de hasta dos personas. **Ambos alumnos** deben realizar la entrega del código desarrollado a través de blackboard, y realizar una defensa oral de la misma. Ambos recibirán la misma nota, y ambos deben estar preparados para responder cualquier pregunta sobre la práctica. En caso de que uno de ellos no sea capaz de responder, o sus errores en respuestas sean demasiado graves, se puede evaluar la práctica como suspensa para ambos. En resumen, los dos integrantes del grupo deben ser capaces de demostrar que han participado en el desarrollo.

Las prácticas se deben entregar antes del día 9 **de Diciembre (23:59 del día 8)**, fecha en la que el profesor dará los horarios de defensa para los grupos que hayan entregado. En caso de querer presentarlas antes, se podrá realizar la defensa en un horario concretado entre el profesor y los alumnos. Se deberá entregar lo siguiente:

- Fichero ZIP con los archivos generados/editados por los alumnos. Cada uno de los ficheros entregados DEBERÁ contener el nombre de los integrantes del grupo escrito en comentarios de código al inicio de cada archivo. Se deja a elección del alumno la estructura de directorios que contendrá su entrega.
- Nombre del fichero (OBLIGATORIO): PR2SISDIS_Alumno1_Apellido1_Alumno2_Apellido2.zip

Cualquier archivo entregado que no se adecúe a estas prescripciones podrá ser ignorado. Por favor, no entreguéis archivos con otros compresores o nomenclaturas.