Samuel Parra y Miguel Caro Sistemas Digitales III

PUNTO 3 DEL TALLER

1. ¿Qué es Kubernetes?

Kubernetes es una plataforma de orquestación de contenedores que automatiza el despliegue, el escalado y la gestión de aplicaciones contenerizadas. Maneja pods, deployments y servicios para ejecutar aplicaciones de forma distribuida y confiable.

2. Características de kubernetes

- **Descubrimiento de servicios y balanceo de carga**

Kubernetes puede exponer contenedores con un nombre DNS propio o una IP, y distribuir la carga si hay mucho tráfico.

- **Orquestación de almacenamiento**

Permite montar sistemas de almacenamiento (local, en la nube, etc.) para contenedores, de modo que puedan usar volúmenes persistentes.

- **Despliegues automáticos y rollbacks**

Puedes declarar el estado que quieres (cuántas réplicas, qué versión) y Kubernetes lo mantiene; además permite actualizar y revertir cambios.

- **Empaque automático (bin packing)**

Le dices a Kubernetes cuántos recursos necesita cada contenedor (CPU, RAM), y él decide en qué nodo correrlos para optimizar el uso de recursos.
Kubernetes

- **Auto-recuperación (self-healing)**

Si un contenedor falla, Kubernetes lo reinicia; puede matar contenedores que no respondan a chequeos de salud, y no dirige tráfico a pods que no estén listos.
Kubernetes

- **Gestión de secretos y configuración**

Kubernetes permite manejar datos sensibles (como contraseñas, tokens) y configuraciones sin tener que reconstruir la imagen del contenedor.
Kubernetes

- **Ejecución en batch**

Además de aplicaciones de servicio, Kubernetes soporta cargas de trabajo tipo batch, como tareas de CI/CD, reiniciando contenedores si es necesario.
Kubernetes

- **Autoescalado horizontal**

Puedes escalar las aplicaciones (más o menos réplicas) manualmente, por UI o automáticamente según métricas (por ejemplo CPU).
Kubernetes

- **IPv4/IPv6 dual-stack**

Kubernetes soporta asignación de direcciones IPv4 y IPv6 para pods y services.

- **Extensibilidad**

Puedes extender Kubernetes con nuevos controladores, plugins y APIs sin tener que modificar el núcleo.

- **Arquitectura del plano de control y nodos**

El plano de control (control plane) coordina todo: lo forman componentes como kube-apiserver, controller-manager y scheduler.

Los nodos de trabajo corren los contenedores (pods) y tienen agentes como kubelet y kube-proxy.

- **Objetos declarativos**

En Kubernetes, defines "objetos" (como Deployment, Pod, Service) mediante archivos YAML, que representan el estado deseado. Kubernetes se encarga de alcanzar y mantener ese estado.

3. Conceptos Clave de Kubernetes

- Cluster: conjunto de nodos que ejecutan aplicaciones.

- Pod: unidad mínima, contiene uno o varios contenedores.

- Deployment: define el estado deseado de los pods (réplicas, actualizaciones).

- Service: expone los pods dentro o fuera del cluster.

- ConfigMap y Secret: manejo de configuración y datos sensibles.

- PV/PVC: almacenamiento persistente.

- Ingress: entrada HTTP/HTTPS al cluster.

4. Creación de contenedores Docker

Docker crea las imágenes y contenedores. Kubernetes no ejecuta imágenes directamente, sino que las usa dentro de pods. El flujo es:

Código → Dockerfile → Imagen → Push al registry → Kubernetes Deployment →Pods ejecutando la imagen.

Los pasos para la creación de un contenedor en Docker y desplegarlo con kubernetes son:

A. Crear la aplicación de ejemplo (Python + Flask)

Archivo: app.py

```
from flask import Flask

app = Flask(__name__)
```

```
def home():
  return "Hola desde Docker + Kubernetes!"

  app.run(host="0.0.0.0", port=5000)
```

B. Crear el Dockerfile

```
FROM python:3.10-slim
WORKDIR /app
COPY app.py .  RUN
pip install flask
EXPOSE 5000
CMD ["python", "app.py"]
```

C. Construir la imagen Docker

Comando dentro del directorio del proyecto:

```
docker build -t usuario/mi_app:1.0 .
```

D. Probar localmente la imagen

```
docker run -p 5000:5000 usuario/mi_app:1.0
```

E. Subir la imagen a Docker Hub

```
docker login
docker push usuario/mi_app:1.0
```

5. Desplegar en Kubernetes (Resumen)

- Crear un archivo YAML con un Deployment y un Service que apunten a la imagen subida.
- Aplicar con:

kubectl apply -f archivo.yaml

REFERENCIAS

[1] S. Susnjara e I. Smalley, “¿Qué es Kubernetes?”, *IBM Think*, 2025. Tomado de: https://www.ibm.com/es-es/topics/kubernetes
[2] “What is Kubernetes?”, *Kubernetes Documentation*. Tomado de: https://kubernetes.io/es/docs/concepts/overview/what-is-kubernetes/
[3] “Kubernetes”, *Cloud Native Computing Foundation Glossary*. Tomado de: https://glossary.cncf.io/es/kubernetes/
[4] “¿Qué es la orquestación de contenedores?”, *IBM Think*. Tomado de: https://www.ibm.com/es-es/think/topics/container-orchestration
[5] “What Are Docker Image Layers?”, How-To Geek. Tomado de: https://www.howtogeek.com/devops/what-are-docker-image-layers/