Kubernetes

Computación Tolerante a Fallas

Murillo Cortes, Jeanette

Lunes 28 / 03 / 22

Kubernetes

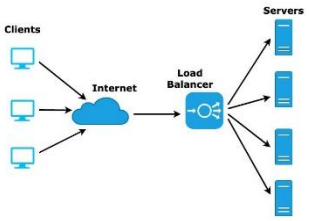
k8s o kube

Computación Tolerante a Fallas
Sección D06
Horario: Lunes – Miérc. 11:00am – 1:00pm
Dr. Michel Emmanuel López Franco

01 **Ingress**

02 **Load Balancer**

03 **Rancher**



¿Qué es Kubernetes?

k8s o kube. Es un sistema de código abierto de orquestación de contenedores para implementar y administrar contenedores a gran escala, permitiendo el despliegue, escalamiento y la administración automática de aplicaciones, trabajos y servicios.

Servicios

Ofrece servicios, algunos y los más comunes de ellos son como:

- Service Discovery
- Load balancing
- Self-Healing
- Horizontal Scaling
- Secretos/Configuración
- Batch processing
- Storage Orchestration

Datos interesantes

Los kubernetes usan dockers, además de que k8s se maneja mediante el uso de objetos con archivos. yml, donde tu defines el estado de los objetos y Kubernetes intenta mantenerlos de esa forma.

Las maneras de comunicarse con Kubernetes es mediante herramientas de terceros (Dashboard y terraform), HTTP/API REST y CLI. Actualmente es respaldado por Cloud Native Computing Foundation (CNCF).

Importancia

Kubernetes permite mantener contenedores con autoescalado,

01. ¿Qué es Ingress?

Ingress permite acceder a pods desde fuera de los clústers.

Es un pod que redirige las solicitudes a los servicios dentro del clúster, pero accediendo a este pod mediante Kubernetes con un servicio de LoadBalancer.

02. ¿Qué es LoadBalancer?

Un balanceador de carga es el que distribuye el tráfico de red entre servidores. Utilizarlo permite una mejor infraestructura en un sitio web de alta demanda.

Es útil porque cuando un sitio web tiene alta demanda, puede llegar al limite de su capacidad, por lo que un balanceador de carga le permite que se distribuya este trabajo para que no se caiga el sitio web y tenga mayor rendimiento en el tráfico de datos, dando servicio a todas las peticiones que puedan, para proporcionar al usuario una mejor experiencia.

03. ¿Qué es Rancher?

Rancher es un software que permite gestionar la capacidad de múltiples clústeres de Kubernetes desde una sola interfaz.

Es útil porque permite la monitorización automática, además que es sencillo de usar, aprender, de código abierto, y gratuito. Además, resuelve desafíos operativos y de seguridad, al ser una de las distribuciones de Kubernetes por y para equipos que adaptan contenedores.

Kubernetes

PREINSTALACIONES

Para el desarrollo de esta práctica, se hizo uso de:

- Docker, el cuál el archivo de instalación se encuentra en 'Instalar Docker.txt'
- Kubectl

DESARROLLO DE LA PRÁCTICA

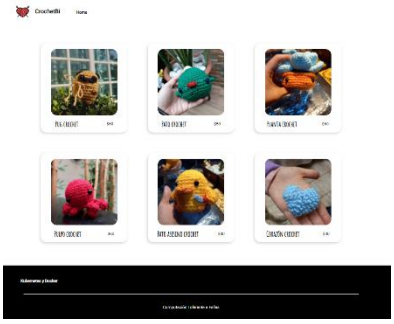
El desarrollo del uso de Kubernetes de esta práctica es aplicado al Sitio Web Crochet-Bi, el cuál es un sistema pequeño de ventas de figuras tejidas creado en una aplicación de React.Js, como se muestra en la siguiente imagen.

```
PS C:\Users\patoa\Documents\Escuela\Kubernetes-docker-minikube> npx create-react-app crochet-bi

Creating a new React app in C:\Users\patoa\Documents\Escuela\Kubernetes-docker-minikube\crochet-bi.

Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts with cra-template...

[########.........] / idealTree:@babel/core: sill fetch manifest klona@^2.0.4
```

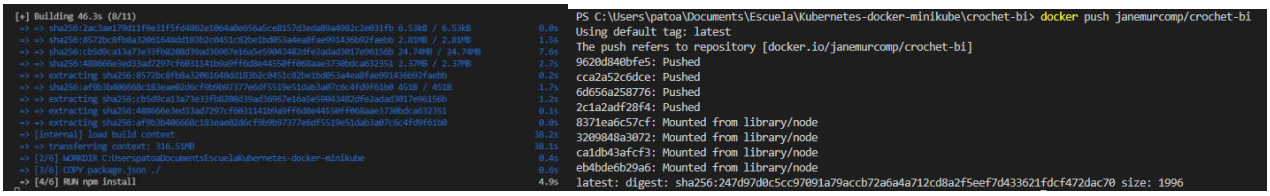


Una vez que se tuvo creada e implementada la aplicación, a partir de ahí, primeramente se agregó un archivo '`.dockerignore`' donde se agregó que se iban a ignorar los modules de Node modules.

Además, en un archivo Dockerfile se añadió en la imagen de Docker:

- FROM: Especificación de una imagen padre donde normalmente se escoge con las dependencias y librerías que se ocuparán para el trabajo. Para este caso, se usó una imagen de Node con Alpine3.
- WORKDIR: Especifica el directorio donde se va a crear, y si no existe, se crea.
- COPY. . : Copea todos los archivos que hay dentro del directorio
- RUN: Corre el código
- CMD: Pasa mediante argumentos el comando que va a correr.

Luego en la línea de comandos, para la construcción de la imagen del Docker, se escribió el comando '`docker build -t janemurcomp/crochet-bi .`' para crear la imagen de una vez accediendo a una cuenta de Docker. Se hico push, escribiendo el comando '`docker push janemurcomp/crochet-bi`', como se muestran en las siguientes dos imágenes.



Una vez creada la imagen de Docker, se implementó el archivo .yaml, el cuál contiene:

- Uso de apps/v1beta2 para versiones anteriores a 1.9.0
- Tipo de objeto a crear, en este caso de deployment y de servicio
- Metadatos, como nombres y etiquetas del deployment
- Número de pods a ejecutar

En las siguientes líneas de código se muestra dónde se encuentran las especificaciones anteriores.

```yaml
apiVersion: app/v1
kind: Deployment
metadata:
  name: crochet-bi-depl
```

Versión
Tipo de objeto
Metadatos de nombres o etiquetas del deployment

```yaml
spec:
  selector:
    matchLabels:
      app: react-app
  replicas: 3
  template:
```

Descripción del estado del objeto

Pods

```yaml
    metadata:
      labels:
        app: react-app
    spec:
      containers:
        - name: react-app
          image: janemurcomp/crochet-bi
...
apiVersion: v1
kind: Service
metadata:
  name: crochet-bi-srv
spec:
  type: NodePort
  selectors:
    app: react-app
  ports:
    -name: croch-bi
    port: 3000
    targetPort: 3000
```

Imagen de Docker

Versión
Tipo de objeto
Metadatos de nombre

Descripción del estado del objeto

Puertos

Una vez definida la configuración del archivo, para aplicar y correr dicho archivo, se hizo uso del comando `'kubectl apply -f crochet-bi-depl.yml'` para el deployment.

```
PS C:\Users\patoa\Documents\Escuela\Kubernetes-docker-minikube\yaml> kubectl apply -f crochet-bi-depl.yaml
deployment.apps/crochet-bi-depl created
```

De manera que mediante `'kubectl get pods'` se pueden ver los pods y `'kubectl get services'` para ver los servicios que configuramos.

```
PS C:\Users\patoa\Documents\Escuela\Kubernetes-docker-minikube\yaml> kubectl get pods
NAME                              READY   STATUS             RESTARTS      AGE
crochet-bi-depl-f59b67dc8-4crm8   0/1     CrashLoopBackOff   4 (69s ago)   2m53s
crochet-bi-depl-f59b67dc8-79wd9   0/1     CrashLoopBackOff   4 (79s ago)   2m53s
crochet-bi-depl-f59b67dc8-qkpfz   0/1     CrashLoopBackOff   4 (77s ago)   2m53s
PS C:\Users\patoa\Documents\Escuela\Kubernetes-docker-minikube\yaml> kubectl get services
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP   24m
PS C:\Users\patoa\Documents\Escuela\Kubernetes-docker-minikube\yaml> kubectl get pods
```

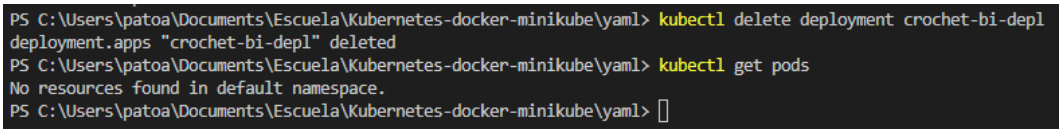
Otro aspecto que es posible mencionar es que cuando se ejecutan los pods, estos pueden tener distintos estados:

- Pending: es el estado en el que inicia un pod.
- Running: es el estado que toma si al menos uno de sus contenedores se inicia correctamente.
- CrashLoopBackOff: es el estado en el que un pod falla y se vuelve a ejecutar, pero entra en un ciclo.
- Error: es el estado en el que un pod termina en un fallo.

Además, existen otros comandos que informan acerca de los pods de los Kubernetes:

- `kubectl cluster-info`: informa acerca de en donde está corriendo
- `kubectl namespaces`: informa y lisaa acerca de los nombres actuales dentro de un clúster.

Incluso, para eliminar un pod, se puede utilizar el comando `kubectl get deployment` para ver cuáles deployments tenemos actuales, y el comando `kubectl delete deployment` *`nombre`*-deployment para borrarlo. De manera que para confirmarlo, se puede poner nuevamente el comando `kubectl get pods` como se muestra en la siguiente imagen.



CONCLUSIÓN

Además de ser una herramienta bastante actual y que proporciona la posibilidad de desplegar sistemas distribuidos, orquestar almacenamiento y mejorar el rendimiento de los contenedores mediante el equilibrio de carga, el uso de estas tecnologías nos permite implementar una mayor robustes en servidores tolerantes a fallos ya que cada vez que falla un pod, este se vuelve a crear, auto reiniciándose, proporcionando a los usuarios menores inconvenientes y fallos al realizar múltiples acciones en donde se está corriendo el clúster de kubernetes, por lo que la realización de esta práctica nos permite conocer cómo administrarlo para automatizar y reducir costos a nuestros servicios.

BIBLIOGRAFÍA

- *"Qué es un balanceador de carga y cómo mejora el rendimiento de la web"* (redeszone.net) . Obtenido del URL: https://www.redeszone.net/tutoriales/servidores/balanceador-carga-load-balancer-que-es-funcionamiento/
- *"Todo lo que necesita saber sobre Rancher: gestión de Kubernetes para empresas - SiXe Ingeniería*". Obtenido del URL: https://sixe.es/noticias/suse-rancher-kubernetes-toda-la-informacion
- *"Una breve introducción a Kubernetes Ingress de forma gráfica - programador clic"*. Obtenido del URL: (programmerclick.com) https://programmerclick.com/article/28441815020/
- *"Entender los Objetos de Kubernetes"* (Mayo, 2020). Kubernetes. Obtenido del URL: https://kubernetes.io/es/docs/concepts/overview/working-with-objects/kubernetes-objects/
- Wissem Khrarib (Junio, 2021). *"Deploy a react app in K8s"*. Obtenido del URL: https://www.youtube.com/watch?v=JL6RTbtS9p0&list=PLeOcwkjDsoExvynzi_TvC69gnXyzZ9M--&index=2&t=212s
- Urlichs, Anais (Diciembre, 2020). *"Kubernetes CrashLoopBackOff: Day 7 of #100DaysOfKubernetes"*. Obtenido del URL: https://www.youtube.com/watch?v=-yplje ZrSgc