Guía de Kubernetes Básico en Minikube

Prerrequisitos

1. Minikube

- Descarga el instalador de Minikube desde la página oficial de Minikube.

2. Kubectl

- Visita la documentación oficial de kubectl para seguir las instrucciones específicas de tu sistema operativo.

3. Verificación de instalación

```
# minikube versión
# kubectl version –client
```

Configuración

4. Iniciar minikube

```
# minikube start
```

Crear un Pod

6. Archivo pod.yaml

Para crear un pod, primero debes definir un archivo YAML que especifique la configuración del pod. A continuación se muestra un ejemplo básico de un archivo pod.yaml:

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
  labels:
    app: nginx
spec:
  containers:
  - name: nginx
    image: nginx:latest
    ports:
    - containerPort: 80
```

apiVersion: Indica la versión de la API de Kubernetes que se está utilizando, en este caso v1 es la versión estable para la creación de pods.

kind: Especifica el tipo de recurso que se está definiendo; en este caso, es un Pod.

metadata: Proporciona información sobre el pod, como su nombre (name) y etiquetas (labels). Las etiquetas son útiles para organizar y seleccionar recursos dentro del clúster.

spec: Define la especificación del pod. Aquí es donde se detalla cómo debe comportarse el pod y qué contenedores debe ejecutar.

containers: Es una lista de contenedores que se ejecutarán en el pod. En este ejemplo, hay un contenedor llamado nginx, que utiliza la imagen nginx:latest.

ports: Especifica los puertos expuestos por el contenedor. En este caso, el puerto 80 es el puerto HTTP predeterminado para el contenedor Nginx.

7. Crear el Pod

Una vez que hayas creado el archivo pod.yaml, puedes crear el pod ejecutando el siguiente comando en tu terminal:

```
# kubectl apply -f pod.yaml

# kubectl get pods
```

Exponer el Pod con un Service

Para que el pod que has creado sea accesible desde el exterior, necesitas exponerlo a través de un Service. Un Service en Kubernetes actúa como un punto de acceso para tus pods, permitiendo la comunicación entre ellos y el exterior de manera estable y balanceada.

9. Archivo service.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: nginx-service

spec:
  type: NodePort
  selector:
    app: nginx
  ports:
    - port: 80
      targetPort: 80
```

10. Aplicar el servicio

Una vez que has creado el archivo service.yaml, puedes aplicar la configuración del servicio ejecutando el siguiente comando en tu terminal:

```
# kubectl apply -f service.yaml

# kubectl get services
```

11. Acceder al servicio a través de Minikube

Para acceder al servicio expuesto, puedes utilizar el comando minikube service, que abrirá automáticamente el navegador en la dirección correcta. Ejecuta el siguiente comando:

```
# minikube service nginx-service
```

Esto abrirá el servicio en tu navegador, permitiéndote interactuar con la aplicación que corre en el pod a través de la dirección IP y el puerto asignado. Si prefieres acceder manualmente, puedes utilizar la dirección IP de Minikube y el puerto 30000 que configuraste en el service.yaml. Para obtener la dirección IP, ejecuta:

```
# minikube ip
```

Con esta configuración, tu pod ahora está accesible a través del servicio que has creado, lo que facilita la gestión y escalabilidad de tu aplicación.

Crear un Deployment

Crear un Deployment en Kubernetes es una forma eficaz de gestionar la implementación y el escalado de aplicaciones. Un Deployment proporciona una manera declarativa de actualizar los pods asociados, asegurando que el número deseado de réplicas esté en ejecución en todo momento.

12. Archivo deployment.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:latest
        ports:
        - containerPort: 80
```

Una vez que hayas creado el archivo deployment.yaml, puedes crear el Deployment ejecutando el siguiente comando en tu terminal:

```
# kubectl apply -f deployment.yaml

# kubectl get deployments

# kubectl get pods
```

```
# kubectl get service
```

Trabajando con Namespaces

En Kubernetes, un namespace es una forma de organizar los recursos dentro de un clúster. Permite dividir un clúster en múltiples entornos virtuales, lo que es especialmente útil para gestionar recursos de manera más eficiente en escenarios donde coexisten diferentes equipos o proyectos. Cada namespace actúa como un entorno aislado, lo que significa que los nombres de los recursos pueden ser reutilizados en diferentes namespaces sin conflictos.

13. Creación de un Namespace

Para crear un nuevo namespace en Kubernetes, puedes utilizar el siguiente comando kubectl:

```
# kubectl create namespace my-app

# kubectl get namespaces
```

Este comando te mostrará una lista de todos los namespaces disponibles en tu clúster, incluyendo el recién creado.

Aplicar un Pod en un Namespace

Una vez que tienes tu namespace, puedes desplegar un pod dentro de él. A continuación, se presenta un ejemplo de cómo crear un archivo YAML para un pod dentro del namespace mi-namespace. Crea un archivo llamado pod-en-namespace.yaml con el siguiente contenido:

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
  namespace: my-app
  labels:
    app: nginx
spec:
  containers:
  - name: nginx
    image: nginx:latest
    ports:
    - containerPort: 80


# kubectl apply -f pod-en-namespace.yaml

# kubectl get pods -n mi-namespace
```

Limpieza

```
# kubectl delete -f pod-en-namespace.yaml

# kubectl delete -f deployment.yaml

# kubectl delete -f pod.yaml

# kubectl delete -f service.yaml
```

```
# kubectl delete namespace my-app
```

Detener Minikube

```
# minikube stop
```

Este comando detiene la máquina virtual de Minikube y libera la memoria y otros recursos que se estaban utilizando. Si ya no planeas usar Minikube, también puedes eliminarlo completamente con:

```
# minikube delete
```

Este comando eliminará la máquina virtual y todos los datos asociados, asegurando que tu entorno esté completamente limpio.