Laboratorio de sistemas operativos y redes

Deployando una aplicación en Kubernetes

Agustín García Smith, Inés Sosa

15 de Diciembre, 2017

Indice

1. Introducción
2. Que es kubernetes
3. Historia
4. Componentes
5. Instalando Kubernetes
6. Corriendo la aplicación
7. Bibliografía

Introducción

El objetivo de este trabajo es mostrar como deployar y escalar una aplicación utilizando Kubernetes como contenedor de software. Para este ejemplo utilizaremos una imagen de Pharo Smalltalk como servidor que nos proveerá de una ruta predefinida para mostrar el funcionamiento del deploy.

¿Que es Kubernetes?

Es un sistema de código libre para la automatización del despliegue, ajuste de escala y manejo de aplicaciones en contenedores. que fue originalmente diseñado por Google y donado a la Cloud Native Computing Foundation (parte de la Linux Foundation).

Historia

Kubernetes (“timonel” o “piloto” en griego) fue fundado por Joe Beda, Brendan Burns and Craig McLuckie, a quienes se le unieron rápidamente otros ingenieros de Google incluyendo a Brian Grant y Tim Hockin, y fue anunciado por Google a mediados de 2014. Su diseño estuvo fuertemente influenciado por el sistema Borg de Google y muchos de los principales contribuyentes al proyecto trabajaron antes en Borg. El nombre en clave original para Kubernetes dentro de Google era “Project Seven”, una referencia a un persona de Star Trek que es un más “amigable” Borg. Los siete radios en la rueda del logo de Kubernetes es una referencia al nombre en clave.

Google se asoció con la Linux Foundation para formar la Cloud Native Computing Foundation (CNCF) y ofreció Kubernetes como una tecnología semilla.

Rancher Labs incluye una distribución Kubernetes en su plataforma de mejoramiento de contenedores Rancher. También está siendo utilizada por Red Hat para su producto OpenShift, CoreOS para su producto Tectonic, e IBM para su producto IBM Spectrum Conductor for Containers.

Kubernetes define un conjunto de bloques de construcción (primitivas) que conjuntamente proveen los mecanismos para el despliegue, mantenimiento y escalado de aplicaciones. Los componentes que forman Kubernetes están diseñados para estar débilmente acoplados pero a la vez ser extensibles para que puedan soportar una gran variedad de flujos de trabajo. La extensibilidad es provista en gran parte por la API de Kubernetes, que es utilizada por componentes internos así como extensiones y contenedores ejecutando sobre Kubernetes.

Componentes

Pods

La unidad básica de planificación en Kubernetes es llamada un “pod”. Este agrega un nivel de abstracción más elevado a los componentes en contenedores. Un pod consta de uno o más contenedores en los que se garantiza su ubicación en el mismo equipo anfitrión y pueden compartir recursos. Cada pod en Kubernetes es asignado a una única dirección IP (dentro del clúster) que permite a las aplicaciones utilizar puertos sin riesgos de conflictos. Un pod puede definir un volumen, como puede ser un directorio de disco local o un disco de red, y exponerlo a los contenedores dentro del pod. Los pods pueden ser administrados manualmente a través de la API de Kubernetes, o su administración puede ser delegada a un controlador.

Etiquetas y selectores

Kubernetes permite a los clientes (usuarios o componentes internos) vincular pares clave-valor llamados etiquetas a cualquier objeto API en el sistema, como pods o nodos. Correspondientemente, selectores de etiquetas son consultas contra las etiquetas que resuelven a los objetos que las satisfacen.

Las etiquetas y los selectores son el mecanismo principal de agrupamiento en Kubernetes, y son utilizados para determinar los componentes sobre los cuales aplica una operación.

Por ejemplo, si un pod de una aplicación tiene la etiqueta para un nivel del sistema (“front-end”, “back-end”, por ejemplo) y un release_track (“canary”, “production”, por ejemplo), entonces una operación sobre todos los nodos “back-end” y “canary” podría utilizar un selector de etiquetas como el siguiente:

nivel=back-end AND release_track=canary

Controladores

Un controlador es un bucle de reconciliación que lleva al estado real del clúster hacia el estado deseado. Hace esto mediante la administración de un conjunto de pods. Un tipo de controlador es un "Replication Controller", que se encarga de la replicación y escala mediante la ejecución de un número especificado de copias de un pod a través de un clúster. También se encarga de crear pods de reemplazo si un nodo subyacente falla. Otros controladores que forma parte del sistema central de Kubernetes incluye al "DaemonSet Controller" para la ejecución de exactamente un pod en cada máquina (o algún subconjunto de máquinas), y un "Job Controller" para ejecutar pods que ejecutan hasta su finalización, por ejemplo como parte de un trabajo

batch. El conjunto de pods que un controlador administra está determinado por los selectores de etiquetas que forman parte de la definición del controlador.

Servicios

Un servicio Kubernetes es un conjunto de pods que trabajan en conjunto, como una capa de una aplicación multicapas. El conjunto de pods que constituyen un servicio está definido por el selector de etiquetas. Kubernetes provee de un servicio de descubrimiento y enrutamiento de pedidos mediante la asignación de una dirección IP estable y un nombre DNS al servicio, y balancea la carga de tráfico en un estilo round-robin hacia las conexiones de red de las direcciones IP entre los pods que verifican el selector (incluso cuando fallas causan que los pods se muevan de máquina en máquina). Por defecto un servicio es expuesto dentro de un cluster (por ejemplo, pods de un back end pueden ser agrupados en un servicio, con las peticiones de los pods de front end siendo balanceadas entre ellos), pero un servicio también puede ser expuesto hacia afuera del clúster.

Instalación

A continuación se detallara como instalar kubernetes. Se recomienda tener una maquina virtual por cualquier conflicto en la instalación. Las instrucciones están basadas en una instalación sobre Linux. Como primer paso se tiene que instalar kubernetes se tiene que instalar Kubectl, este sirve para controlar el cluster manager de Kubernetes, mediante el siguiente comando en una terminal:

```
curl -LO https://storage.googleapis.com/kubernetes-release/release/
$(curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt)/bin/linux/amd64/kubectl
```

- A continuación le daremos permisos de ejecución de la siguiente manera:

```
chmod +x ./kubectl
```

- Por último movemos el directorio

```
sudo mv ./kubectl /usr/local/bin/kubectl
```

Por otro lado, tenemos que tener instalado docker que va a ser el contenedor donde tenemos nuestra aplicación configurada. Para este ejemplo tenemos un docker que contiene una imagen de Pharo Smalltalk como servidos que al acceder a él nos trae un mensaje preconfigurado.

Una vez ejecutadas las anteriores líneas ya estamos en condiciones de ejecutar la instancia de kubernetes. simplemente vamos a una terminal y ejecutamos

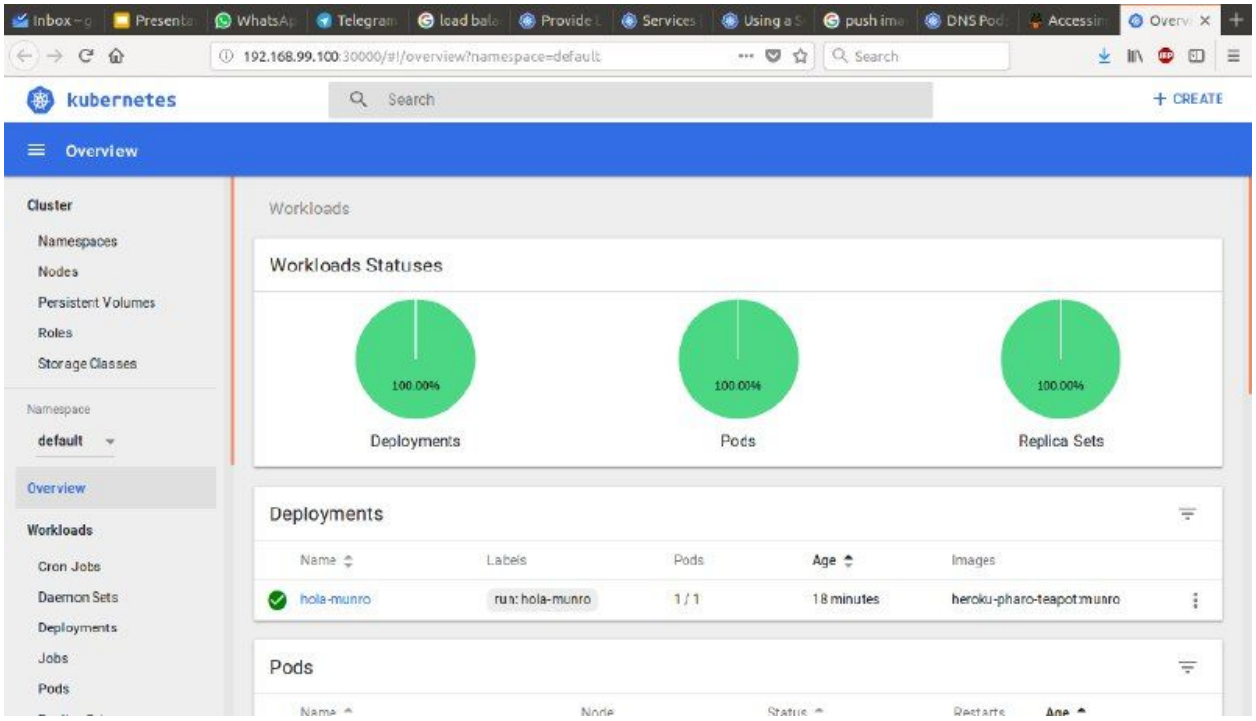
```
~ $ minikube start
```

Este comando ya inicia la instancia de minikube.

Kubernetes cuenta con un dashboard que nos facilita el control y la visualización de los pods. para acceder a este ejecutamos:

```
~ $ minikube dashboard
```

Esto nos abrira en el navegador por defecto el siguiente dashboard



Una vez que comprobamos que minukube funciona correctamente procederemos a conectar el cliente local de docker que viene integrado con minikube a la instancia local de docker que tenemos en nuestra VM

```
~ $ eval $(minikube docker-env)
```

Una vez terminado esto, procederemos a ir a docker a buildear la imagen que procederemos a deployar.

```
~ $ docker image build -t heroku-pharo-teapot:munro .
```

Ya buildeada la imagen de docker crearemos el deployment correspondiente en kubernetes indicando la imagen que debe deployar y el puerto que tiene que exponer

```
~ $ kubectl run hola-munro --image=heroku-pharo-teapot:munro --port=5000
```

Después, se debe exponer el servicio de kubernetes para poder ser accedido

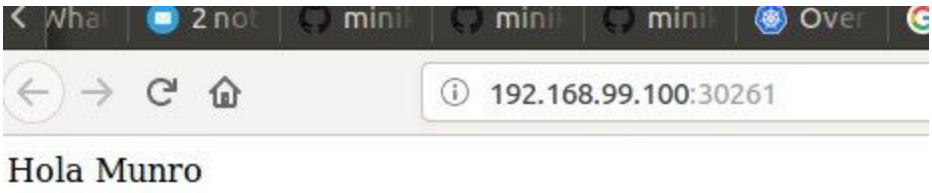
```
~ $ kubectl expose deployment hola-munro --type=NodePort
```

Corriendo la aplicación

ya configurado y expuesto el servicio de kubernetes procederemos a comprobar que la instalación se concreto correctamente

```
~ $ minikube service hola-munro
```

Este ultimo comando levantara en el navegador por defecto el servicio de kubernetes en el puerto que especificamos, en nuestro caso el puerto 5000, y nos mostrara la respuesta del servidor que nos confirma que el servicio está activo y funcionando


192.168.99.100:30261
Hola Munro

Bibliografía

- https://kubernetes.io/docs/tasks/tools/install-kubectl/
- https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/
- https://github.com/kubernetes/minikube