kubernetes

Alberto Molina Coballes

IES Gonzalo Nazareno
CONSEJERÍA DE EDUCACIÓN

10 de febrero de 2019

La mayor parte de la información de esta presentación se ha obtenido directamente de la documentación oficial del proyecto kubernetes (kubernetes.io/docs), se presenta aquí de forma resumida y organizada para su explicación en clase.

Actualizado a la versión 1.13

Kubernetes (k8s)

Gestiona el despliegue de **aplicaciones** sobre contenedores, automatizando el despliegue, con énfasis en la escalabilidad y controlando todo el ciclo de vida.

- Despliega aplicaciones rápidamente
- Escala las aplicaciones al vuelo
- Integra cambios sin interrupciones
- Permite limitar los recursos a utilizar

`https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/`

k8s

- Portable: En nube publica, privada o híbrida
- Extensible: Módulos, *plugins*, adaptable, ...
- Autoreparable

- Proyecto comenzado por Google en 2014, ahora gestionado por Cloud Native Computing Foundation
- Permite gestionar un clúster de nodos en los que desplegar aplicaciones sobre contenedores
- Kubernetes es el término griego para timonel

k8s

- Primera versión: 7 junio 2014
- Última versión: 3 diciembre 2018 (1.13)
- Desarrollado en Go
- Licencia Apache 2.0
- `https://github.com/kubernetes/kubernetes`

Nodos

- ~~minion~~ Componente de un clúster de k8s
- Puede ser una máquina virtual o física
- Tiene los servicios para ejecutar pods
- Es gestionado por los componentes **master**
- Cada nodo tiene:
  - Direcciones: Hostname, IP externa, IP interna
  - Condición: Campo que describe el estado (listo, sin red, sin disco, ...)
  - Capacidad: Describe los recursos disponibles
  - Info: Información general (kernel, sw instalado, versiones)
- Nodo controlador es el que contiene componentes para controlar otros nodos. Se suele denominar **master**[1]

[1] `https://github.com/kubernetes/website/issues/6525`

Componentes de k8s. Master

kube-apiserver Gestiona la API de k8s

etcd Almacén clave-valor que guarda la configuración del clúster

kube-scheduler Selecciona el nodo donde ejecutar los pods

kube-controller-manager Ejecuta los **controladores**

cloud-controller-manager Ejecuta los controladores que interactúan con el proveedor de nube:

- nodos
- enrutamiento
- balanceadores
- volúmenes

Complementos (*Addons*

Pods y servicios que proporcionan funcionalidad al clúster

Cluster DNS Proporciona registros DNS para los servicios de k8s

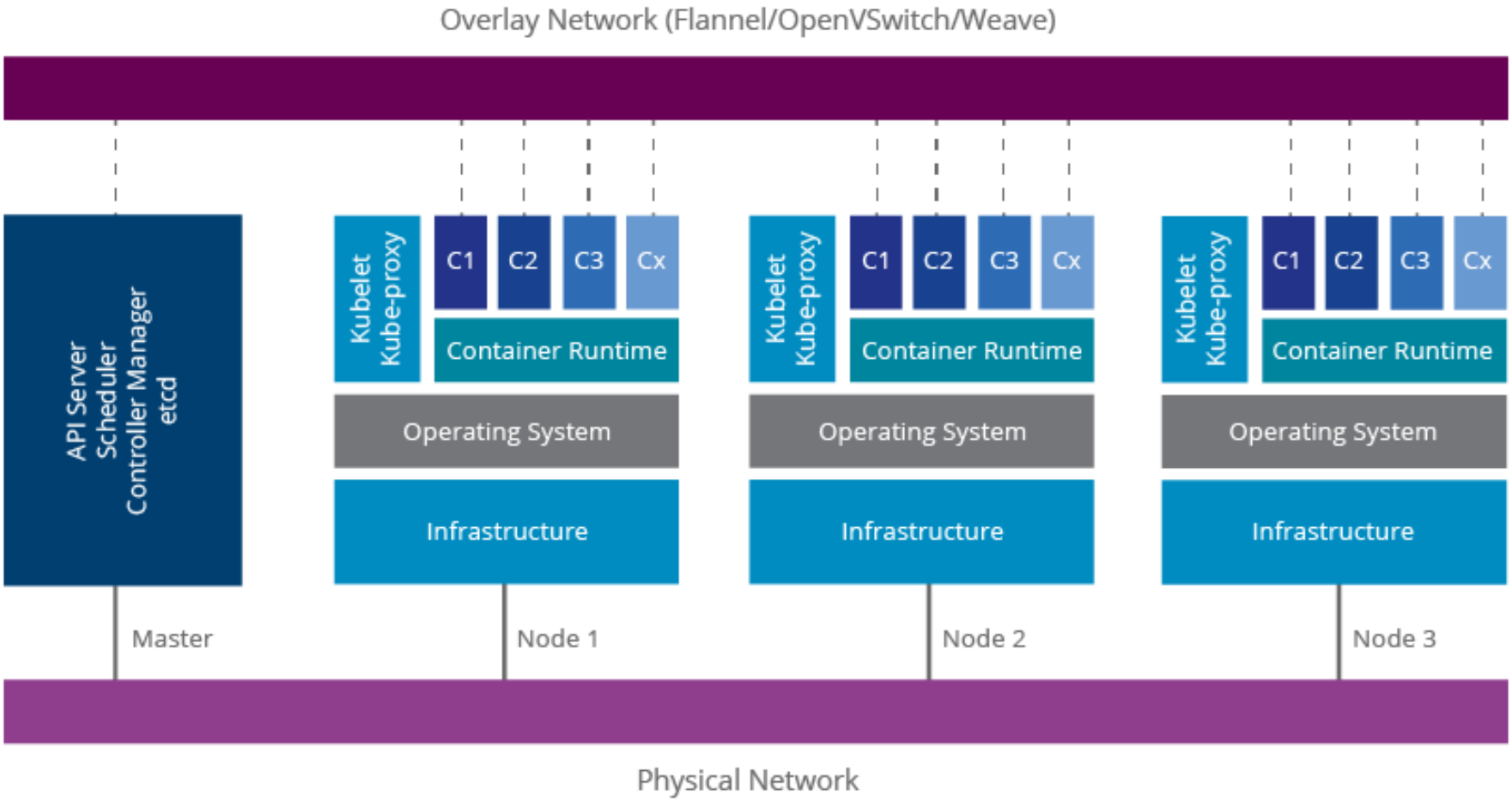
Web UI Interfaz web para el manejo de k8s

Container Resource Monitoring Recoge métricas de forma centralizada

Cluster-level Logging Almacena los logs de los contenedores

Componentes de k8s. Nodos

kubelet	Vigila los pods asignados a su nodo
kube-proxy	Permite la conexión a través de la red
docker/rkt/runc	Ejecuta los contenedores
supervisord	Monitoriza y controla kubelet y docker
fluentd	Proporciona logs de contenedores a cluster-level logging


Overlay Network (Flannel/OpenVSwitch/Weave)
API Server
Scheduler
Controller Manager
etcd
Kubelet
Kube-proxy
C1
C2
C3
Cx
Container Runtime
Operating System
Infrastructure
Master
Node 1
Node 2
Node 3
Physical Network

Objetos de k8s

- Entidades persistentes utilizadas para representar el estado del clúster
- Describen las aplicaciones ejecutándose, recursos disponibles y políticas para las aplicaciones
- k8s tratará de hacer que todos los objetos existan. Representan el **estado deseado**
- Los objetos se crean, modifican o borran a través de la API
- De cada objeto tiene dos campos el `spec` y el `status`

https://kubernetes.io/docs/concepts/overview/working-with-objects/kubernetes-objects/

Objetos de k8s. Ejemplo

- Habitualmente se describen en un fichero yaml
- Ejemplo

```
apiVersion: apps/v1 # for versions before 1.9.0 use apps/v1beta2
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.7.9
        ports:
        - containerPort: 80
```

Objetos de k8s. Nombres

- Todos los objetos tienen que tener un nombre único (entre objetos del mismo tipo) y un UUID
- Los nombres pueden incluir letras minúsculas, números, guiones y puntos
- k8s genera automáticamente un UUID a cada objeto

Objetos de k8s. Espacios de nombres

- k8s soporta múltiples clústeres virtuales que se denominan espacios de nombres (*namespaces*)
- Cada espacio de nombres puede gestionar sus propios objetos
- Hay tres espacios de nombres predefinidos:

```
kubectl get namespaces
NAME          STATUS    AGE
default       Active    3d
kube-public   Active    3d
kube-system   Active    3d
```

- La mayoría de los recursos de k8s pertenecen a un espacio de nombres (pods, despliegues, servicios, ...), pero no otros como los nodos o los volúmenes persistentes.

Objetos de k8s. Etiquetas, selectores y anotaciones

- Las etiquetas (*labels*) son pares clave/valor de uso libre asociados a los objetos
- Las etiquetas se utilizan para caracterizar y relacionar objetos
- Mediante los selectores se pueden obtener determinados objetos:
  - *Equality-based requirement*: `environment = production`
  - *Set-based requirement*: `environment in (production, qa)`

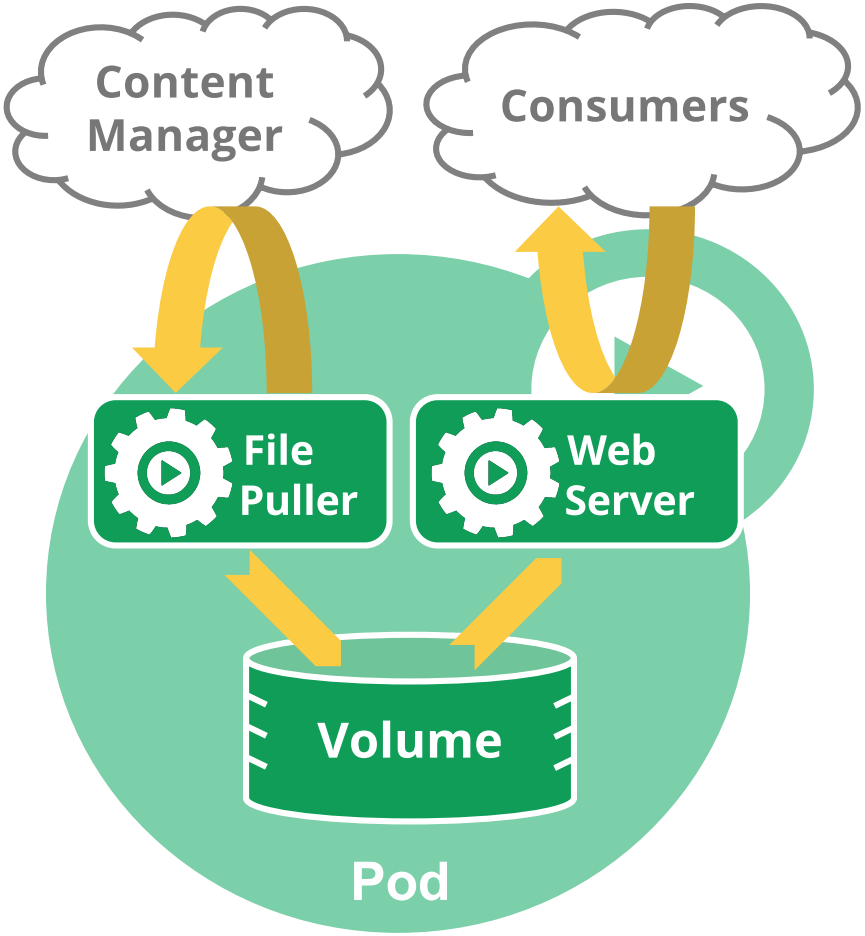
```
kubectl get pods -l environment=production,tier=frontend
...
kubectl get pods -l 'environment in (production),tier in (frontend)'
```

- También se pueden utilizar las anotaciones (*annotations*) para cualquier otra información en formato clave:valor
- Las anotaciones se utilizan para seleccionar objetos concretos

https://kubernetes.io/docs/concepts/overview/working-with-objects/labels/

Utilizando k8s

Pods



- *Pod*: Vaina/envoltura
- Unidad mínima de k8s
- Grupo de uno o más contenedores
- Comparten red y almacenamiento
- Normalmente no se lanzan directamente ⇒ Son los **controladores** los que los gestionan
- Son efímeros y se ejecutan en un nodo

`https://kubernetes.io/docs/concepts/workloads/pods/pod-overview/`

Controladores

- Vigilan el estado del clúster a través de la API y realizan los cambios necesarios para conseguir el estado deseado
- Son manejados por kube-controller-manager
- Tipos de controladores
  - ReplicaSet
  - ReplicationController
  - Deployments
  - StatefulSets
  - DaemonSet
  - Garbage Collection
  - TTL Controller for Finished Resources
  - Jobs - Run to Completion
  - CronJob

https://kubernetes.io/docs/concepts/workloads/controllers/replicaset/

Servicios (*Services*)

- Los pods son efímeros
- Abstracción que define un conjunto de Pods y la política para acceder a ellos
- Normalmente se indican los pods gestionados a través de un **selector**
- Para aplicaciones nativas de k8s, se ofrece un "endpoint" que se actualiza con los cambios en pods o servicios. Para el resto de aplicaciones se utiliza una IP virtual que redirecciona a los pods
- Diferentes tipos:
  - Proxy-mode: userspace
  - Proxy-mode: iptables
  - Proxy-mode: ipvs

`https://kubernetes.io/docs/concepts/services-networking/service/`

Servicios

Publicar servicios

- `ClusterIP`: (por defecto)
- `NodePort`: Expone el puerto en cada nodo
- `LoadBalancer`: Utiliza un proveedor del cloud
- `ExternalName`: Devuelve un CNAME y requiere kube-dns

kubectl

kubectl

- Herramienta principal para utilizar la API de k8s
- Imperativo/declarativo[2]

Management technique	Operates on	Recommended environment	Learning curve
Imperative commands	Live objects	Development projects	Lowest
Imperative object configuration	Individual files	Production projects	Moderate
Declarative object configuration	Directories of files	Production projects	Highest

[2] Sebastien Goasguen - Imperative/Declarative and a Few ‘kubectl‘ tricks

`kubectl`. Instrucciones imperativas

Uso limitado, pero adecuado para empezar.

Crear objetos

- `run`: Crea un nuevo despliegue
- `expose`: Crea un nuevo servicio para balancear el tráfico
- `autoscale`: Crea un nuevo objeto `autoscaler` para autoescalar un controlador (p.ej. un despliegue)

También puede utilizarse `create` en el que se pueden especificar más parámetros.

```
kubectl run webserver --image=nginx  --replicas=1 --port=80
...
kubectl expose deployment my-nginx --type=NodePort
```

`kubectl`. Instrucciones imperativas

Actualizar objetos

- `scale`: Escalar horizontalmente un controlador
- `annotate`: Añadir o quitar anotaciones
- `label`: Añadir o quitar etiquetas

También puede utilizarse `set` en el que se pueden especificar más parámetros.

```
kubectl scale deploy webserver --replicas=4
```

`kubectl`. Instrucciones imperativas

Eliminar objetos: `delete`

```
kubectl delete deploy nginx
```

Ver objetos:

- `get`: Información sencilla
- `describe`: Información detallada
- `logs`:

```
kubectl get pods
...
kubectl describe deploy nginx
```

`kubectl`. Uso imperativo con ficheros

Con el parámetro `--dry-run` podemos obtener ficheros de configuración a partir de instrucciones imperativas.

- Crear

```
kubectl create -f <filename|url>
```

- Modificar

```
kubectl replace -f <filename|url>
```

- Eliminar

```
kubectl delete -f <filename|url>
```

El principal problema de este enfoque es las modificaciones que se realicen en objetos que no estén definidas en los ficheros.

`kubectl`. Uso declarativo

La configuración declarativa necesita un conocimiento profundo de k8s y sale del ámbito de este curso.

De forma genérica se utiliza la siguiente instrucción para crear o modificar objetos:

```
kubectl apply -f <directory>/
```

k8s se encarga de comprobar en cada caso las diferencias entre el objeto que se está ejecutando y la definición en el fichero y aplica los cambios.

Enlaces recomendados

- `http://whatis.techtarget.com/definition/stateless-app`