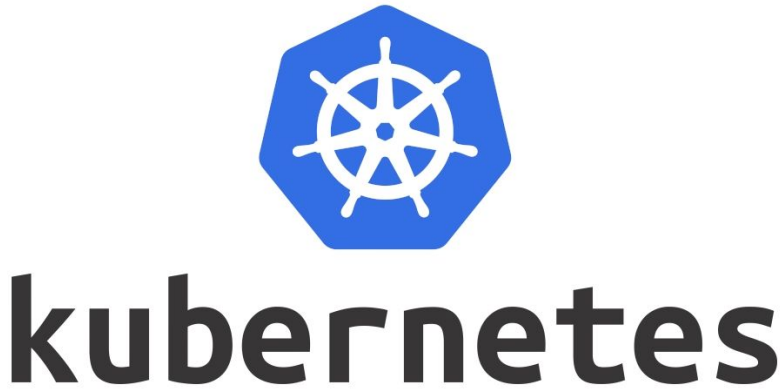
kubernetes

GeeksHubs
academy _

Introducción

Kubernetes (K8s) es una plataforma de código abierto para automatizar la implementación, el escalado y la administración de aplicaciones en contenedores.

También se le denomina k8s, por las 8 letras que hay entre la "k" y la "S", y el proyecto se inició en Julio de 2015 y fue donado posteriormente a Cloud Native Computing Foundation CNCF



GeeksHubs
academy _

Introducción

Es un proyecto **OpenSource** creado por Google, el cual lo utiliza en la mayoría de sus proyectos como son Gmail, Maps, Drive, etc.

Además expone de una forma programática para que podamos acceder a la gestión de nuestro cluster. Como orquestados permite crear un cluster de nodos que implementa determinadas funciones sobre los contenedores:

- Permiten alta disponibilidad.
- Es tolerante a fallos.
- Permite escalar cuando se queda corto de recursos.
- Trabaja de forma eficiente.
- Nos permite realizar cambios y operaciones en caliente.

GeeksHubs academy

Introducción

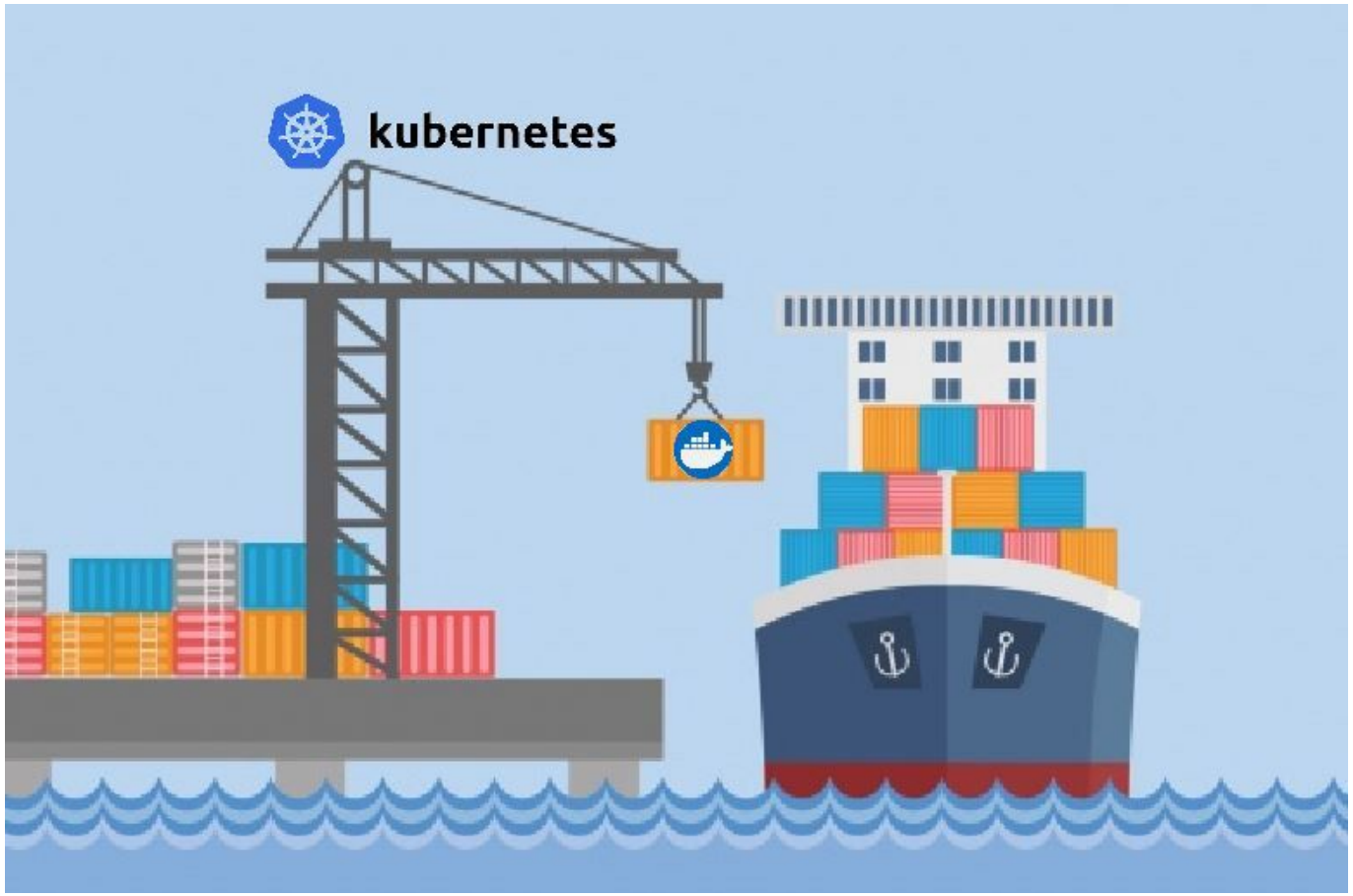
Existen otros orquestadores:

- Swarm mode overview
- Marathon: A container orchestration platform for Mesos and DC/OS
- Nomad by HashiCorp
- shipyard project

SABER MAS !

GeeksHubs
academy _

Comprendiendo el fundamento Kubernetes



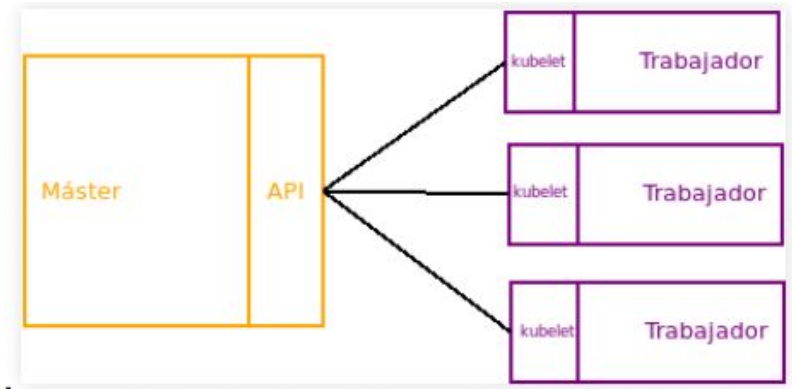
GeeksHubs academy _

Conceptos: Cluster

Un clúster es un conjunto de máquinas, donde cada máquina tiene un rol que se dividen en dos:

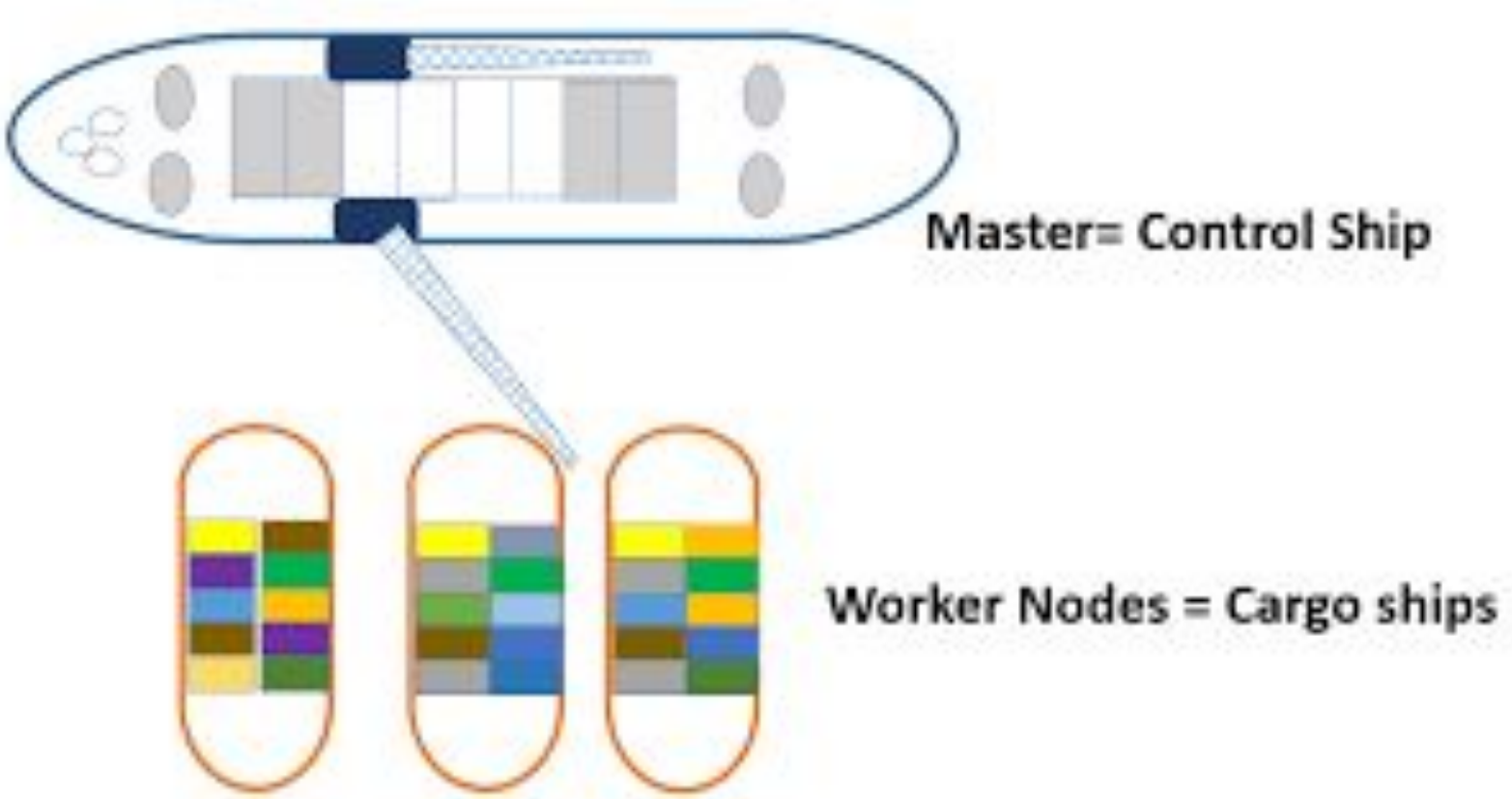
Máster: sólo hay uno en el cluster y es el encargado de repartir la carga de trabajo, entre los demás nodos, mantener el estado deseable de trabajo, escalar las aplicaciones o actualizarlas.

Workers: los demás nodos que nos Máster. Son los que ejecutan las aplicaciones por medio de contenedores:



GeeksHubs
academy _

Conceptos: Cluster

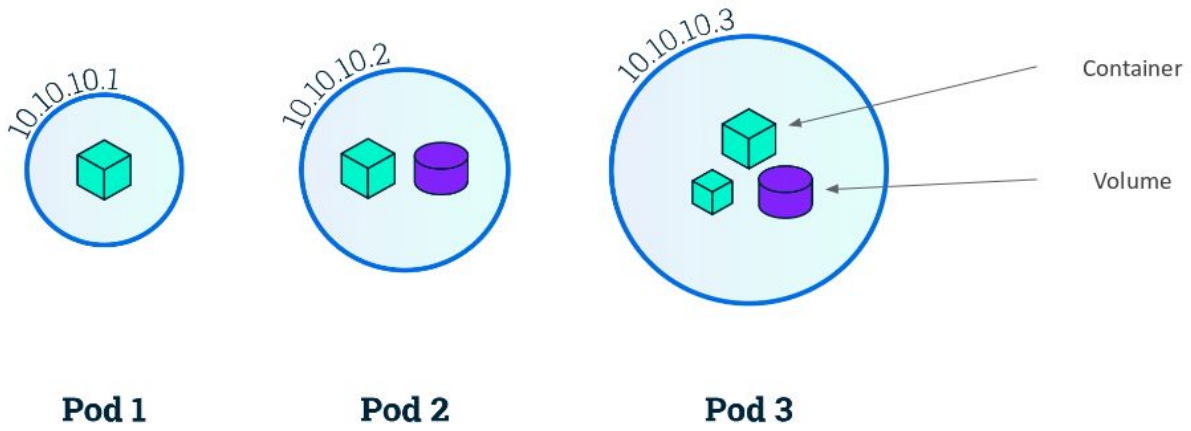


GeeksHubs academy

Conceptos: Pod

Es la unidad mínima de Kubernetes. Representa en ejecución en el clúster. Un Pod contiene uno o varios contenedores (por ejemplo Docker), y administrar los recursos de la aplicación, dirección IP y distintas opciones que determina cómo funcionan los contenedores.

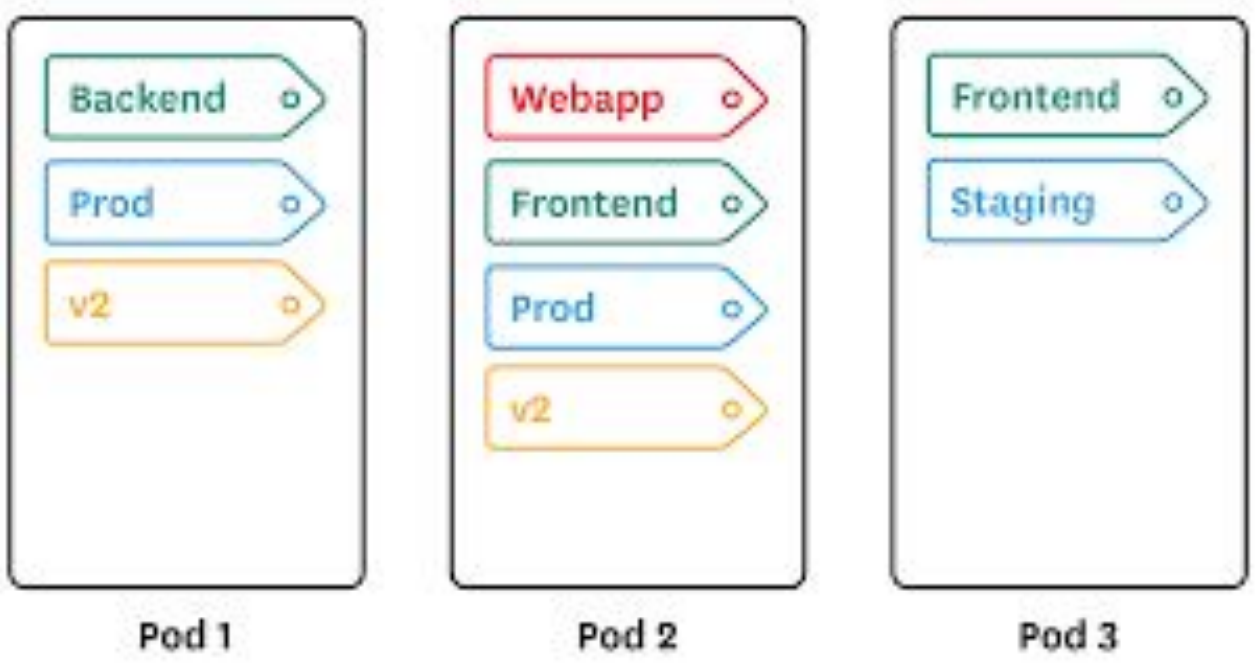
https://kubernetes.io/es/docs/concepts/workloads/pods/pod/



GeeksHubs academy _

Conceptos: Labels y selectors

Son pares de claves y valores, las cuales se puede aplicar a pods, services, etc. y con ellos podemos identificarlos para poder gestionarlos.



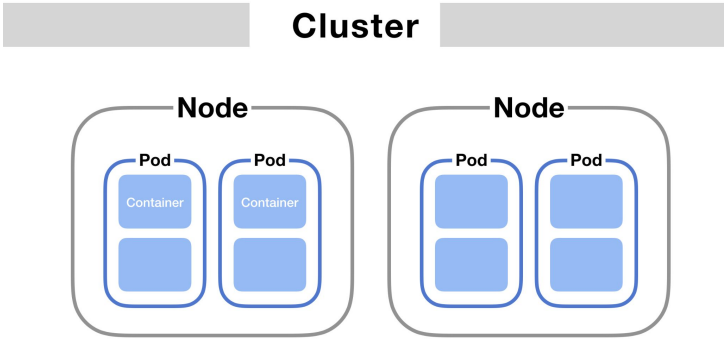
GeeksHubs academy _

Conceptos: Node

Un nodo **es una máquina de trabajo en Kubernetes**, previamente conocida como minion. Un nodo puede ser una máquina virtual o física, dependiendo del tipo de clúster. Cada nodo está gestionado por el componente máster y contiene los servicios necesarios para ejecutar pods.

Los servicios en un nodo incluyen el container runtime, kubelet y el kube-proxy. Accede a la sección The Kubernetes Node en el documento de diseño de arquitectura para más detalle.

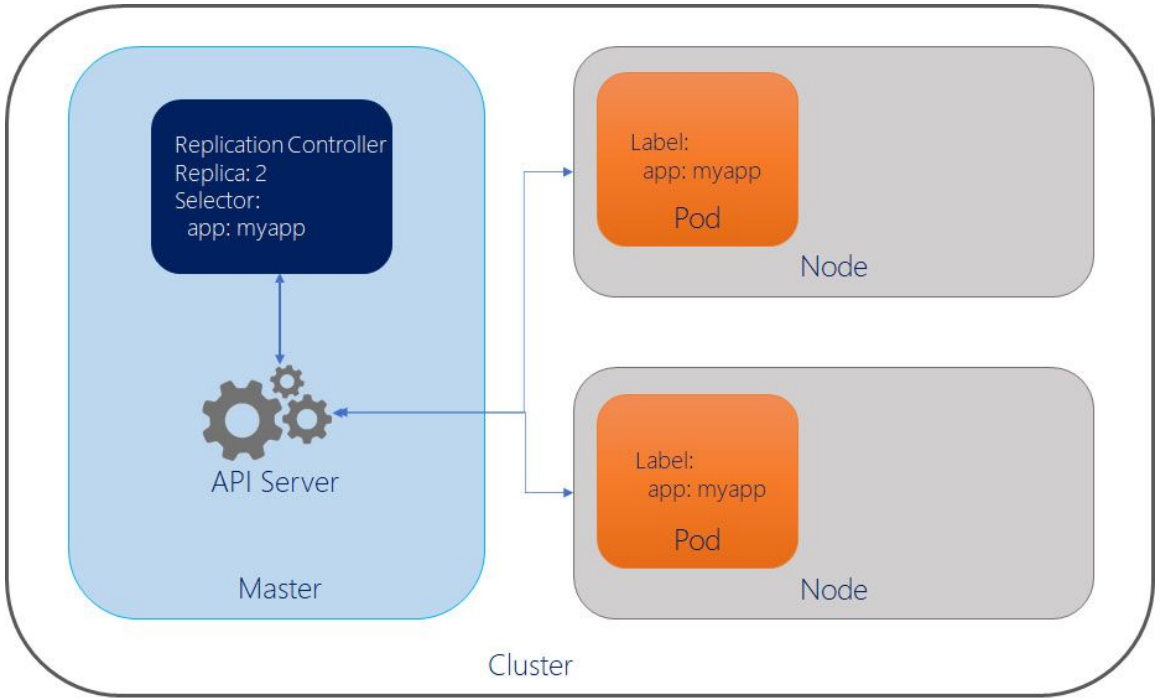
[Controlador de Nodos]



GeeksHubs academy _

Conceptos: Replication Controller.

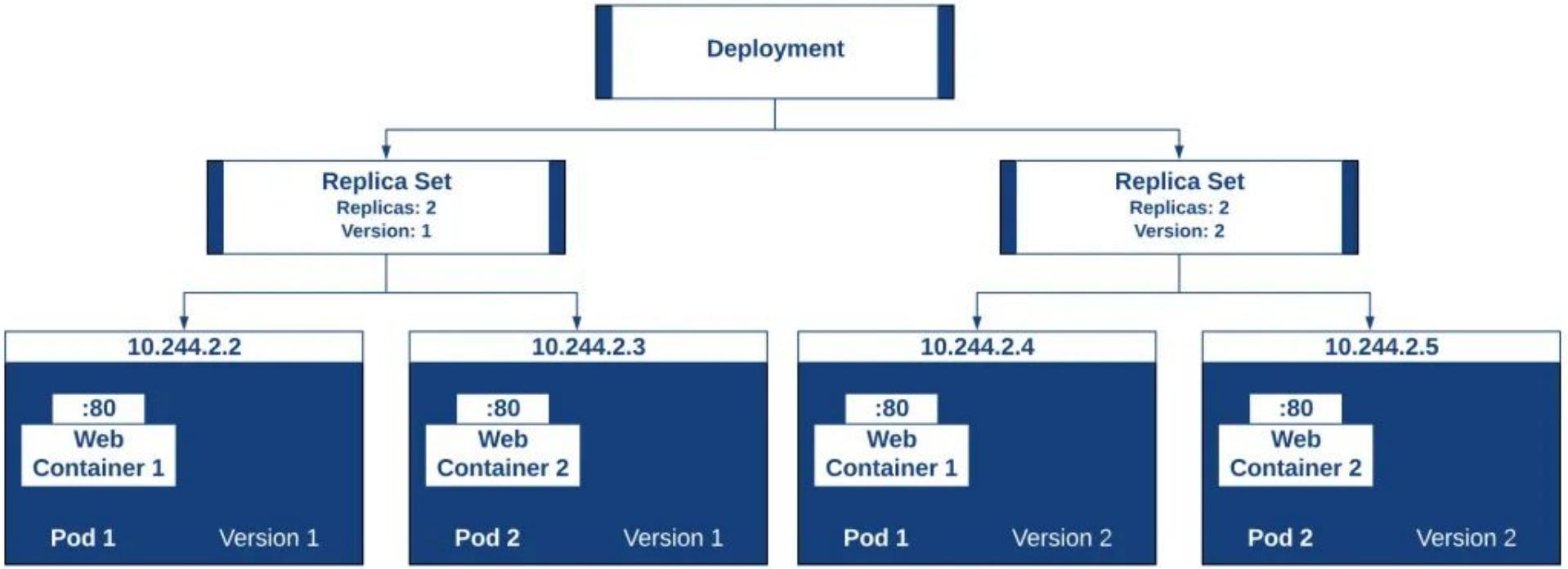
Es el responsable de gestionar la vida de los pods y el encargado de mantener arrancados los pods que se hayan indicado en la configuración.
Permite escalar de forma muy sencilla los sistemas y maneja la recreación de un pod cuando ocurre algún tipo de fallo, en las últimas versiones nos recomienda utilizar replica Sets.



GeeksHubs academy _

Conceptos: Replica Sets

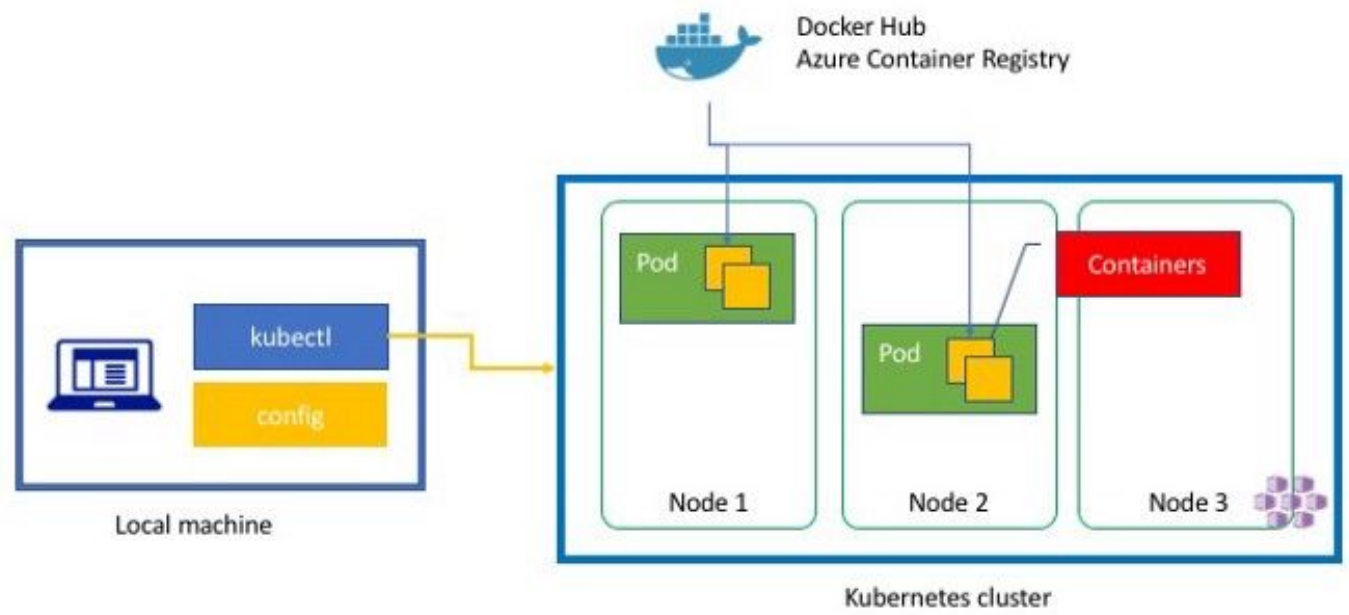
Es la nueva generación de Replication Controller, con nuevas funcionalidades. Una de las funcionalidades destacadas que nos permite desplegar pods en función de los **labels y selectors.**



GeeksHubs
academy _

Conceptos: Deployments

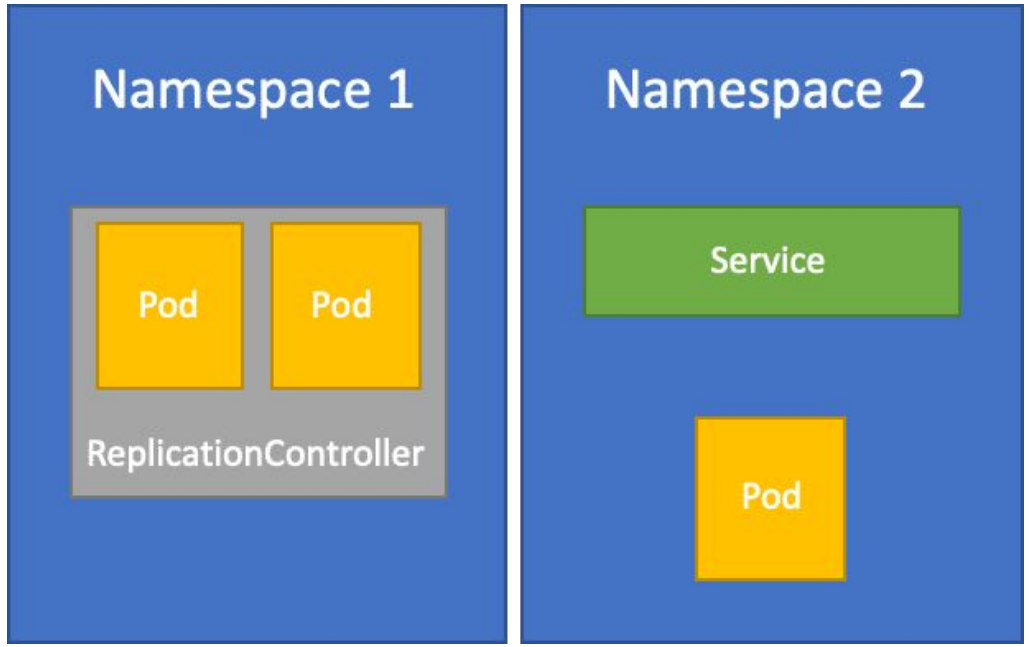
Es una funcionalidad más avanzada que los Replication Controller y nos permite especificar la cantidad de réplicas de pods que tendremos en el sistema.



GeeksHubs academy _

Conceptos: Namespaces

Son agrupaciones de la infraestructura que nos permite diferenciar espacios de trabajo, por ejemplo desarrollo, producción, etc

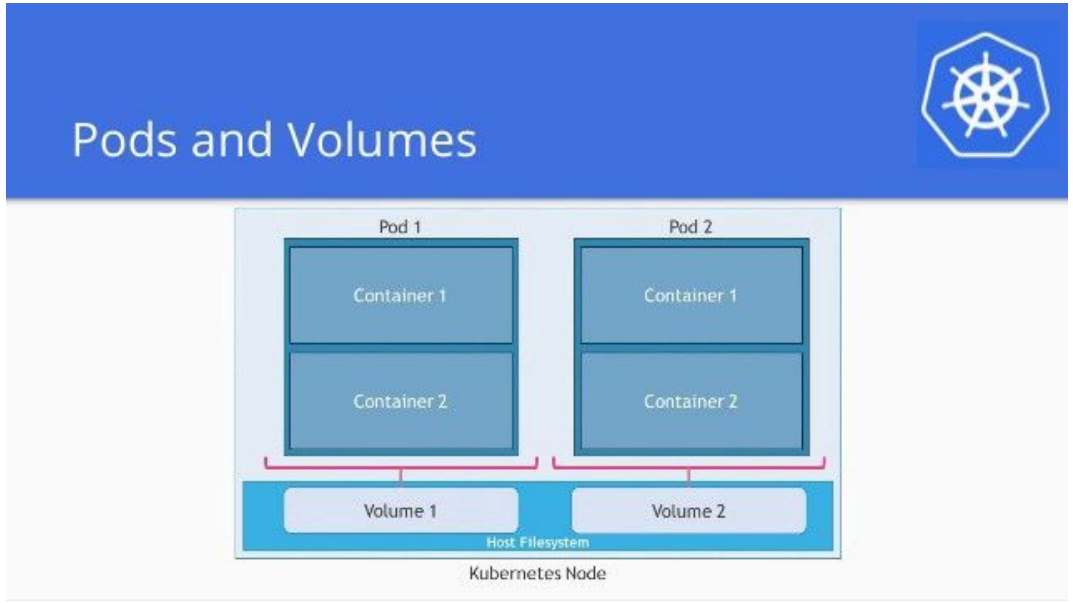


GeeksHubs academy _

Conceptos: Volumes y Secrets

Volumes
La gestión del sistema de almacenamiento de nuestros pods.

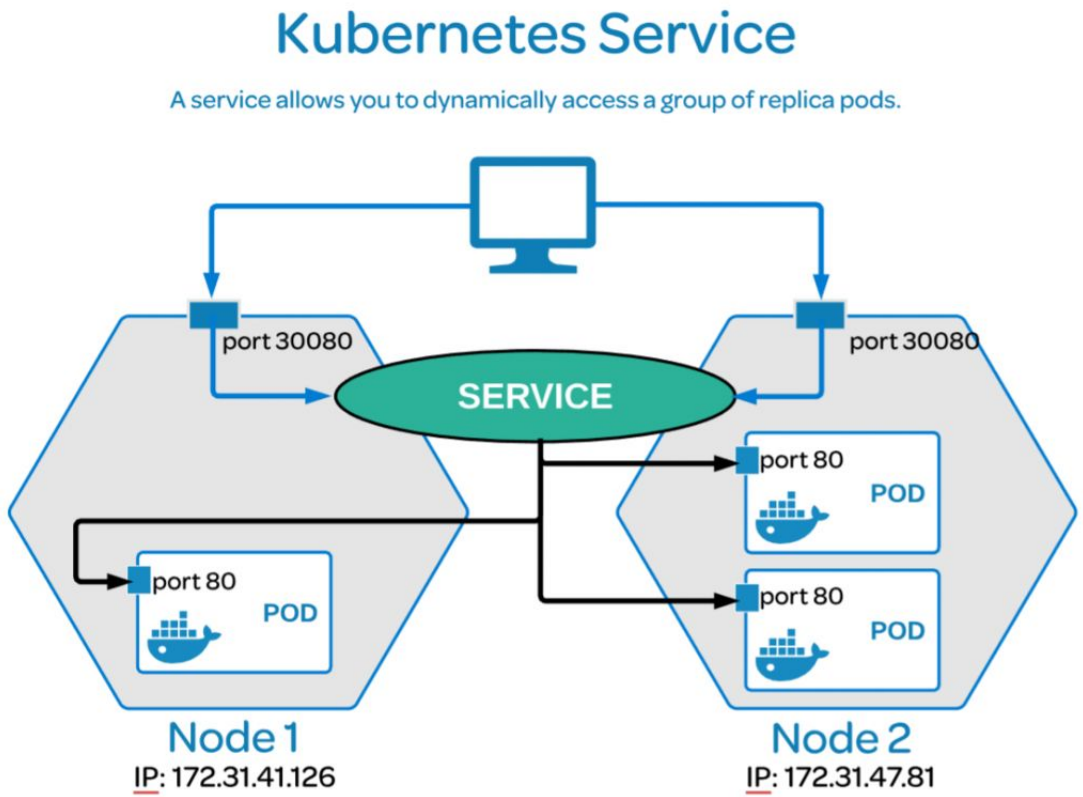
Secrets
Es donde se guarda la información confidencial como usuario y passwords para acceder a los recursos.



GeeksHubs academy _

Conceptos: Service

Es la política de acceso a los pods. Lo podríamos definir como la abstracción que dene un conjunto de pods y la lógica para poder acceder a ellos.



GeeksHubs
academy _

Arquitectura de kubernetes.

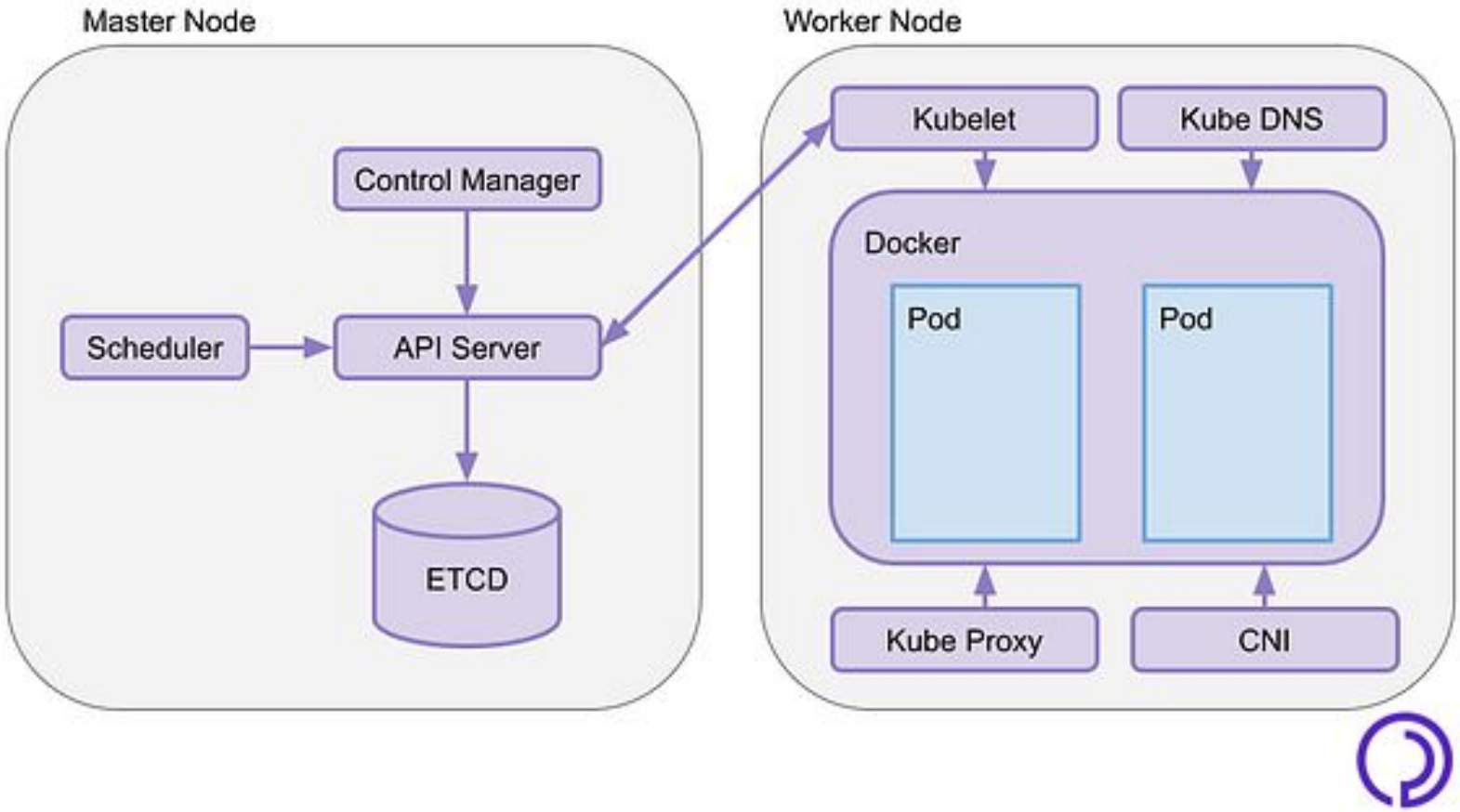
Kubernetes gestiona mediante su nodo maestro para ello dispone de un Scheduler que es el planificador de las acciones y trabajo de los distintos nodos workers.

Después disponer del **Controller** nos permite controlar el estado de los diferentes nodos.

También nos expone un **Api-Server** es decir nos expone un rest donde poder realizar el control de los diferentes nodos. Y por último dispone del **ETCD ALMACEN CLAVE VALOR** de esta manera obtener elementos mediante su clave y valor.

GeeksHubs academy _

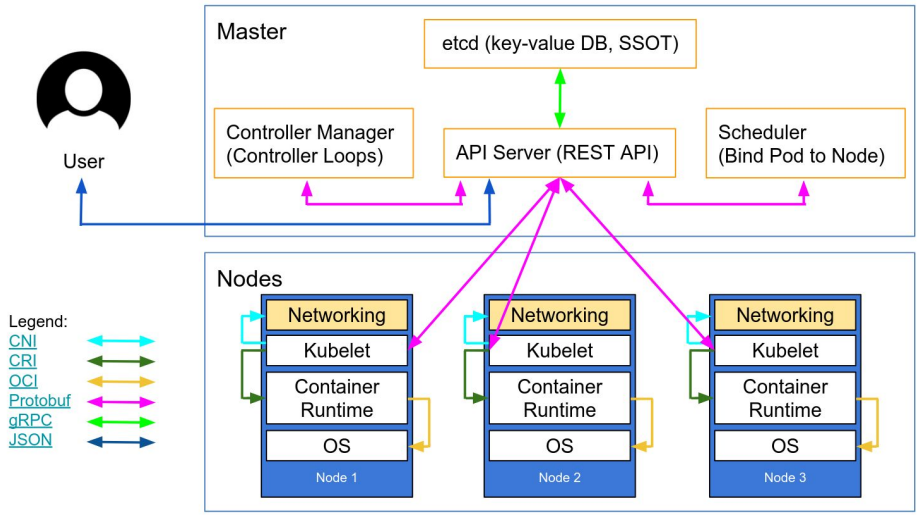
Arquitectura de kubernetes.



GeeksHubs academy

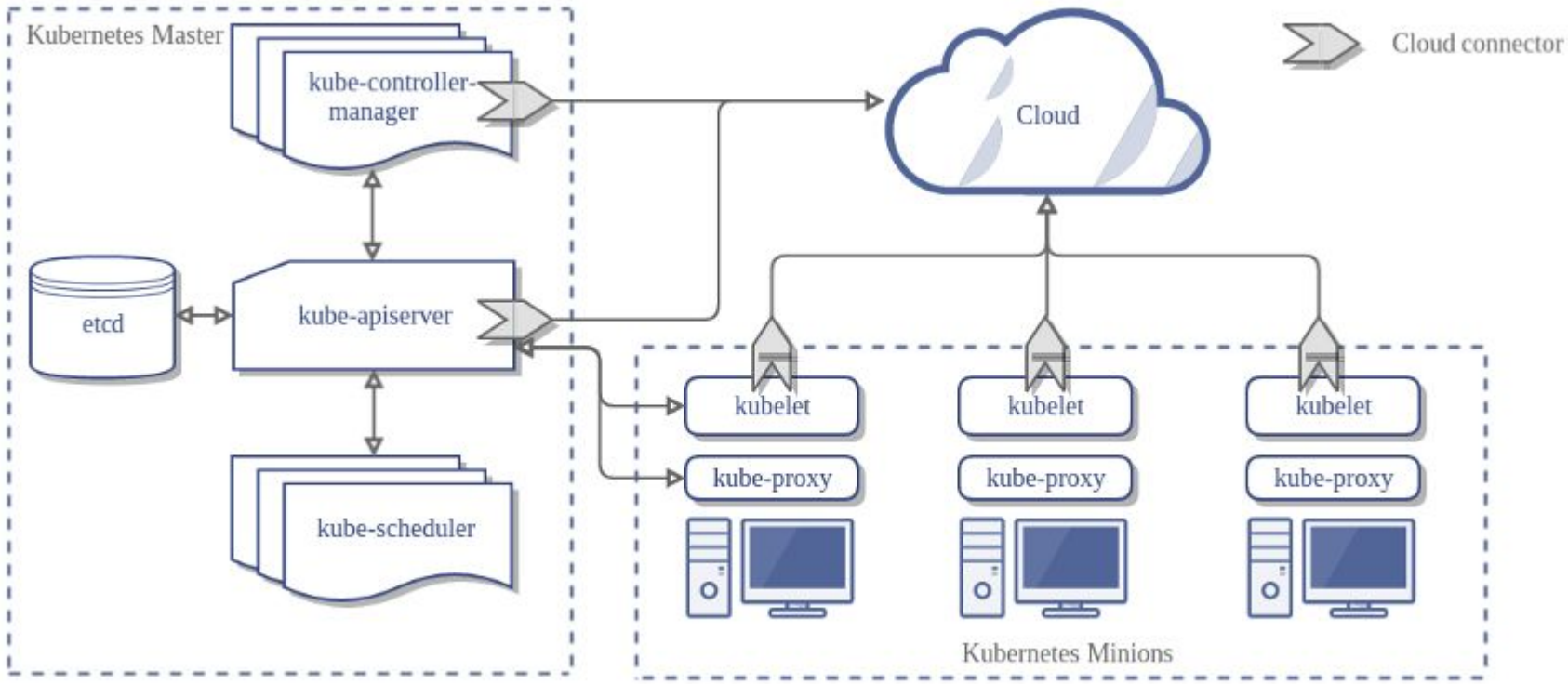
Arquitectura de kubernetes.

En el nodo Worker dispone de un Container Runtime, teniendo en cuenta que no sólo ejecuta docker, sino cualquier otro contenedor, al nal dispone de componentes como los pod, que son el elemento básico que utilizamos en el cluster. Y después dispone de un agente que se llama **kubelet** que se comunica con el maestro. Por último se encuentra en **Kube-proxy** que se utiliza para trabajar por los pods, y se utiliza para gestionar y como puntos de entradas a las llamadas de los pods.



GeeksHubs academy _

Arquitectura de kubernetes.



GeeksHubs
academy _

Arquitectura de kubernetes.

- **Etcd**: Almacena los datos de configuración a los que puede acceder el Servidor API de Kubernetes Master utilizando http simple o API JSON.
- **Kube-Apiserver**: es el centro de gestión para el nodo Master, facilita la comunicación entre los diversos componentes manteniendo con ello la salud del clúster.
- **Kube-Controller-Manager**: es el encargado de asegurar la coincidencia entre el estado deseado del clúster y el estado actual, esto lo consigue escalando cargas de trabajo hacia arriba y hacia abajo.
- **Kube-Scheduler**: coloca la carga de trabajo en el nodo que corresponde; en este diagrama particular, todas las cargas de trabajo se ubican localmente en su host.
- **Kubelet**: recibe las especificaciones del pod del servidor API y administra los pods que se ejecutan en el host.

GeeksHubs academy _

Docker Swarm VS Kubernetes

	Kubernetes	Docker Swarm
Definición de la aplicación	Combinación de pods, implementaciones y servicios	Como servicios en un clúster Swarm
Construcciones de escalabilidad de aplicaciones	Escalar cuando se administra mediante una implementación	Escalar utilizando plantillas Docker Compose YAML
Alta Disponibilidad	Pods se distribuyan entre los nodos para proporcionar AD	Los administradores de Swarm son responsables de todo el clúster y administran los recursos de los nodos de trabajadores
Balanceo de carga	Los pods de Kubernetes se exponen a través de un servicio, que se puede usar como un equilibrador de carga dentro del clúster.	En el modo Swarm tiene un componente DNS que se puede usar para distribuir solicitudes entrantes a un nombre de servicio.

GeeksHubs
academy _

Instalación y configuración de Kubernetes

Existe diferentes formas y soluciones para instalar kubernetes, desde un entorno en local, hasta en la nube, etc.. Para ello podemos verlo en su página Kubernetes.io. Nosotros para aprender utilizaremos MINIKUBE,
Minikube puede instalarse en nuestro equipo de forma local. Mas adelante veremos como instalarlo, pero de momento y con el fin de practicar y afianzar conocimiento vamos a realizar una serie de tutoriales usando un escenario virtual que se ejecutara en tu navegador mediante Katakoda y Packer.

GeeksHubs academy

Escenario para realizar prácticas.

Hello Minikube

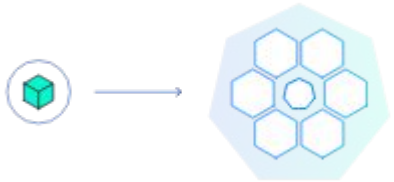
Este tutorial muestra como ejecutar una aplicación Node.js
Hola Mundo en Kubernetes utilizando Minikubey Katacoda.
Katacoda provee un ambiente de Kubernetes desde el navegador.

https://kubernetes.io/docs/tutorials/hello-minikube/

GeeksHubs academy

PRACTICANDO LO BÁSICO

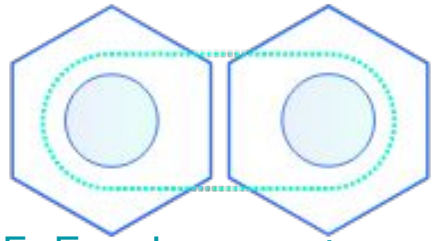
1. Crear un cluster de K8S



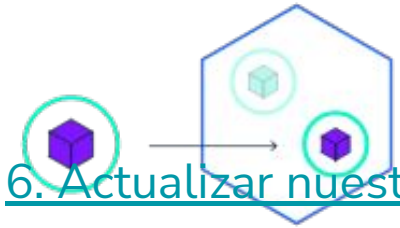
2. Desplegar una app

3. Explorar nuestra app

4. Publicar nuestra app



5. Escalar nuestra app



6. Actualizar nuestra app

GeeksHubs academy _

Instalación y configuración de Kubernetes [MINIKUBE]

1º Instalamos Kubectl que es una herramienta que nos permite trabajar en modo línea con nuestros clusters independientemente del sistema instalado.

2º Instalamos minikube, en este caso el sistema operativo anfitrión sera determinante y tendremos diferentes maneras de proceder en cada caso

https://minikube.sigs.k8s.io/docs/start/

GeeksHubs academy _

Uso de Minikube

Para arrancar nuestro cluster minikube ejecutaremos:

$ minikube start

Para pararlo simplemente:

$ minikube stop

GeeksHubs
academy _

Uso Minikube

El uso de Kubectl es exactamente igual usando Minikube como en un cluster real desplegado en la nube o en nuestros propios servidores, es todo igual salvo el componente “Dashboard” que en el caso de minikube viene incluido con un ejecutable propio que provee la el despliegue con credenciales automatizadas mientras que en un cluster real tenemos que instalar nosotros los providers necesarios si queremos usar el dashboard.

$ minikube dashboard

GeeksHubs academy _

Trabajando con [Workloads] Pods

En el siguiente enlace encontraremos una guía con algunos ejemplos de operaciones relativas a nuestros pods con su relativa explicación...

¿ Practicamos ?

https://github.com/DevOpsOnline1-Edicion/3.1-Kubernetes#trabajando-con-pods

GeeksHubs academy _

Escribiendo nuestro cluster

Las operaciones con Kubectl son una herramienta de gestión de nuestro cluster, pero la principal motivación de Kubernetes es la de poder asegurarnos la gobernabilidad integral de todo cuanto acontence al cluster **COMO CÓDIGO**

XML, JSON o YAML

XML

```
<Servers>
  <Server>
    <name>Server1</name>
    <owner>Fouzi</owner>
    <created>28092019</
created>
    <status>active</status>
  </Server>
</Servers>
```

JSON

```
{
   Servers: [
      {
      name : Server1,
      owner : Fouzi,
      created : 28092019,
      status : active,
      }
   ]
}
```

YAML

```
Servers :
   -   name : Server1
       owner : Fouzi
       created : 28092019
       status : active
```

GeeksHubs
academy _

[Workloads] Controladores

Los objetos básicos de Kubernetes incluyen:

Pod
Service
Volume
Namespace

Además, Kubernetes contiene abstracciones de nivel superior llamadas Controladores. Los Controladores se basan en los objetos básicos y proporcionan funcionalidades adicionales sobre ellos. Incluyen:

ReplicaSet
Deployment
StatefulSet
DaemonSet
Job

GeeksHubs academy _

Mecanismos de Configuración y Organización

Fundamentalmente el principal mecanismo de configuración de Kubernetes se basa en Etiquetas o labels:

Labels

Además, Kubernetes contiene abstracciones como son los ConfigMaps y los Secrets que nos permiten organizar, almacenar y gestionar la información que van a usar las cargas de trabajo en su funcionamiento de manera segura y compartida

ConfigMaps
Secrets

Contamos con los namespaces, que nos dan la capacidad de organizar los despliegues en nuestro cluster de manera modular.

Namespaces

GeeksHubs academy _

Almacenamiento y Persistencia

Fundamentalmente el principal mecanismo de configuración de Kubernetes se basa en Etiquetas o labels:

Labels

Además, Kubernetes contiene abstracciones como son los ConfigMaps y los Secrets que nos permiten organizar, almacenar y gestionar la información que van a usar las cargas de trabajo en su funcionamiento de manera segura y compartida

Persistent Volumes

Secrets

Contamos con los namespaces, que nos dan la capacidad de organizar los despliegues en nuestro cluster de manera modular.

Namespaces

GeeksHubs academy

Tráfico y visibilidad entre servicios

Cada uno de los pods de nuestro cluster recibirá una dirección IP y podrá ser accedido mediante la misma en cualquiera de las subredes que se van desplegando en nuestro cluster, sin embargo este mecanismo es poco eficaz, ya que estas ips son asignadas de manera dinámica y no suelen ser persistentes.
Cuando un servicio debe ser accesado por otro definiremos un Service, los services son la abstracción que usaremos para dotar de nombre a cualquier despliegue y que pueda ser accesado por algún puerto.

Service

Por otra parte contamos con los Ingress, estos son los mecanismos mediante los cuales determinamos el mapeo de rutas no solo de nombres a los diferentes despliegues

Ingress

GeeksHubs academy _

Acceder a un servicio desde internet

Tanto los Services como los Ingreses son los mecanismos que facilitaran esta posibilidad, según la infraestructura donde nuestro cluster este desplegado necesitaremos conectar con componentes de la infraestructura que nos permitan acceder mediante internet a un servicio o publicar un servicio.

Estos componentes suelen tener un coste, por lo que evitaremos disponer de uno de estos componentes para cada servicio y configuraremos un mecanismo de traficado. El más habitual Nginx como Proxy Inverso

Nginx Proxy Inverse

GeeksHubs
academy _

Gestión de Recursos

Nuestros nodos son finitos, en tanto a memoria y capacidad de CPU por tanto nuestros despliegues deben definir una correcta estrategia de gestión de estos recursos para ello utilizaremos las reservas y los límites de recursos

Resource requests and limits of Pod and Container

GeeksHubs
academy _

Kubernetes en la nube.

KOPS - Kuber
netes Operations
Amazon EKS - Elastic Kubernetes Service
Google GKE – Google Kubernetes Engine
DOK – Digital Ocean Kubernetes

DESPLEGAR UN CLUSTER CON KOPS
[aprende como]

GeeksHubs academy _

Componentes habituales en Kubernetes

Desplegar y configurar Proxy Inverso.

Habilitar un Cert Manager.

Monitorizar con Prometheus / Grafana

Gestionar logs con Kibana