



# TRABAJANDO CON KUBERNETES AVANZADO

## MÓDULO 1

# Módulo 1: Deploy de Kubernetes mediante KinD

Introducción a los **componentes y objetos** de Kubernetes

Uso de **clústeres** de desarrollo

Instalación de **KinD**

**Creación de un clúster** KinD

**Revisando su clúster** KinD

Agregar un **balanceador de carga** personalizado para **Ingress**

# Introducción a los componentes y objetos de Kubernetes

Componente	Descripción
Control Plane	<b>API-Server:</b> Acepta las peticiones de los clientes <b>Kube-Scheduler:</b> Asigna los Workloads a los nodos <b>Etc</b> d: Base de datos con todos los datos del clúster <b>Kube-Controller-Manager:</b> Observa los nodos, replicas, endpoints, servicios, cuentas y tokens
Nodo	<b>Kubelet:</b> El agente que ejecuta los Pods basado en las instrucciones del Control Plane <b>Kube-Proxy:</b> Crea y elimina reglas de red para la comunicación entre Pods <b>Container-Runtime:</b> Es el responsable de ejecutar los contenedores

# Introducción a los componentes y objetos de Kubernetes

Objeto	Descripción
Container	Es una <b>imagen inmutable</b> que contiene todo lo necesario para ejecutar una aplicación
Pod	Es el <b>objeto más pequeño</b> que puede controlarse en Kubernetes. Este mantiene uno o múltiples contenedores. Todos los contenedores en el Pod son programados en el mismo servidor con un contexto compartido (el <i>nodo</i> )
Deployment	Despliega una aplicación en un <b>estado deseado</b> , incluyendo <i>replicas</i> del Pod y configuración sobre el <i>rolling update</i>

# Introducción a los componentes y objetos de Kubernetes

Objeto	Descripción
Storage Class	Define un <b>proveedor de almacenamiento</b> y se lo presenta al clúster
Persistent Volume (PV)	Provee un <b>objetivo de almacenamiento</b> que puede ser reclamado por un <i>Persistent Volume Request</i>
Persistent Volume Claim (PVC)	<b>Conecta (reclama)</b> un <i>Persistent Volume</i> que pueda ser usado dentro del Pod

# Introducción a los componentes y objetos de Kubernetes

Objeto	Descripción
Container Network Interface (CNI)	Provee las <b>conexiones de red</b> para los Pods, por ejemplo, Flannel y Calico
Container Storage Interface (CSI)	Provee la conexión entre los Pods y los <b>sistemas de almacenamiento</b>

## Uso de clústeres de desarrollo

Cluster	Descripción
Docker Swarm	Está <b>integrado a Docker</b> y permite administrar nodos del tipo Manager-Worker que pueden ser distribuidos en diferentes Hosts
Minikube	Consiste en un <b>administrador de Kubernetes</b> de un <b>único Nodo</b> y sirve para probar el funcionamiento y ejemplos limitados
Kubeadm	Consiste en un <b>administrador de Kubernetes</b> con <b>nodos del tipo Master-Worker</b> que pueden ser distribuidos en diferentes Hosts
KinD	"Kubernetes in Docker" está basado en crear un <b>ecosistema completo de Kubernetes</b> usando <b>contenedores de Docker como base</b> y así poder diseñar múltiples nodos en un único Host o PC personal para pruebas avanzadas <b>sin consumir muchos recursos</b> o servidores

# Instalación de KinD (1)

[illegible]

```
$ sudo snap install docker
```

```
$ sudo addgroup --system docker
```

```
$ sudo usermod -aG docker $USER
```

```
$ sudo snap disable docker
```

```
$ sudo snap enable docker
```

[ LOGOUT ]

```
$ docker version
```

%%%%%%%%%



## Instalación de KinD (2)

%%% Install Kubectl %%%%%%%%%%%%%%

```
$ sudo snap install kubectl --classic
```

```
$ kubectl version --output yaml
```

%%%%%%%%%



## Instalación de KinD (4)

%%% Install Kind %%%%%%%%%%%%%%

```
$ go install sigs.k8s.io/kind@latest
```

## \$ kind version

%%%%%%%%%

# Creación de un clúster KinD (1)

```
%%% Create KinD cluster %%%%%%%%%%%%%%
```

```
$ kind create cluster --name cluster101
```

```
$ kubectl cluster-info
```

```
--- Inspect KinD cluster -----
```

```
$ kind get clusters
```

```
$ kubectl get nodes
```

%%%%%%%%%



## Creación de un clúster KinD (3)

--- List Resources -----

```
$ kubectl api-resources
```

--- Inspect Pods -----

```
$ kubectl get pods --all-namespaces
```

--- Delete KinD cluster -----

```
$ kind delete cluster -n <name>
```

-----

## Revisando su clúster KinD

--- List Nodes -----

```
$ kubectl get nodes
```

```
$ kubectl get csinodes
```

--- Describe Nodes -----

```
$ kubectl describe node <name>
```

```
$ kubectl describe csinode <name>
```

-----

## Revisando su clúster KinD

--- List Storage Drivers -----

```
$ kubectl get csidrivers
```

--- List Storage Classes -----

```
$ kubectl get storageclasses
```

```
~ kubectl get sc
```

-----



## Revisando su clúster KinD

--- List Persistent Volume & Claim -----

\$ kubectl **get pv**

\$ kubectl **get pvc**

-----

## Trabajo 301: Crear un PVC de prueba (1)

```
=== test-pvc.yaml =====
```

```
apiVersion: v1  
kind: PersistentVolumeClaim
```

```
metadata:  
  name: test-claim
```

```
spec:  
  accessModes:  
    - ReadWriteOnce  
  resources:  
    requests:  
      storage: 1Mi
```

```
=====
```

## Trabajo 301: Crear un PVC de prueba (2)

=== test-pvc-pod.yaml =====

```
apiVersion: v1
kind: Pod

metadata:
  name: test-pvc-pod

spec:
  containers:
    ...
  volumes:
    ...
  restartPolicy: "Never"
```

=====

```
////////////////////////////////////

containers:
  - name: test-pod
    image: busybox
    command:
      - "/bin/sh"
    args:
      - "-c"
      - "touch /mnt/test && exit 0 || exit 1"
    volumeMounts:
      - name: test-pvc
        mountPath: "/mnt"
  volumes:
    - name: test-pvc
      persistentVolumeClaim:
        claimName: test-pvc

////////////////////////////////////
```

## Trabajo 301: Crear un PVC de prueba (3)

--- Run PVC -----

```
$ kubectl create -f test-pvc.yaml
```

--- Inspect PVC -----

```
$ kubectl get pvc
```

```
$ kubectl describe pvc test-pvc
```

-----

## Trabajo 301: Crear un PVC de prueba (4)

--- Run Pod claims PVC -----

```
$ kubectl create -f test-pvc-pod.yaml
```

--- Inspect Pod -----

```
$ kubectl get pods
```

```
$ kubectl describe pod test-pvc-pod
```

-----

## Trabajo 301: Crear un PVC de prueba (5)

--- Inspect Pod and PV/C -----

```
$ kubectl get po,pv,pvc
```

```
$ kubectl describe persistentvolumeclaim/test-pvc
```

```
$ kubectl describe persistentvolume/pvc-<<xxxx>>
```

```
$ kubectl describe pod/test-pvc-pod
```

--- Inspect Node Worker -----

```
$ kubectl get pod test-pvc-pod --output wide
```

```
$ docker exec -it <<worker-name>> ls -l \  
/var/local-path-provisioner/pvc-<<xxxx>>_default_test-pvc
```

-----

Agregar un balanceador de carga personalizado para Ingress

... Manos a la obra ...

[https://github.com/dragonnomada/kubelab/blob/main  
/d301-kind-docker-cluster.txt](https://github.com/dragonnomada/kubelab/blob/main/d301-kind-docker-cluster.txt)