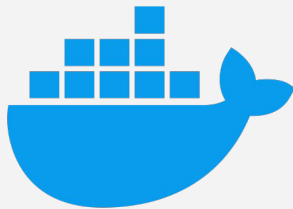


¿Qué es Kubernetes?

Para entender cómo funciona **Kubernetes**, primero es clave conocer los conceptos básicos de los **contenedores**, ya que Kubernetes está diseñado para gestionarlos. Una herramienta como **Docker** permite crear estos contenedores, que son unidades ligeras y portátiles que empaquetan una aplicación junto con todo lo que necesita para ejecutarse (código, dependencias, configuraciones). Piensen en un contenedor como una caja sellada que asegura que tu aplicación funcione igual en cualquier entorno, desde tu laptop hasta un servidor en la nube.

Sin embargo, cuando tienes muchas aplicaciones (o muchos contenedores) corriendo en varios servidores, coordinarlos manualmente se vuelve complicado. Aquí entra **Kubernetes**, una plataforma de orquestación que automatiza la gestión de contenedores a gran escala.



¿Qué es Kubernetes?

- Kubernetes es un sistema de orquestación de contenedores de código abierto, inicialmente desarrollado por Google y ahora mantenido por la Cloud Native Computing Foundation (CNCF).
- Permite automatizar el despliegue, escalado y gestión de aplicaciones contenerizadas, facilitando la administración de múltiples contenedores distribuidos en clusters de nodos.
- También asegura alta disponibilidad y cero tiempo de inactividad mediante mecanismos como la auto-reparación, balanceo de carga y escalado automático, los cuales exploraremos más adelante.



¿Que es un cluster y un nodo?

En Kubernetes, un **cluster** es un conjunto de **nodos**. Cada nodo puede imaginarse como una máquina virtual o física, y dentro de cada nodo se ejecutan **pods**, que son las unidades más pequeñas que contienen contenedores

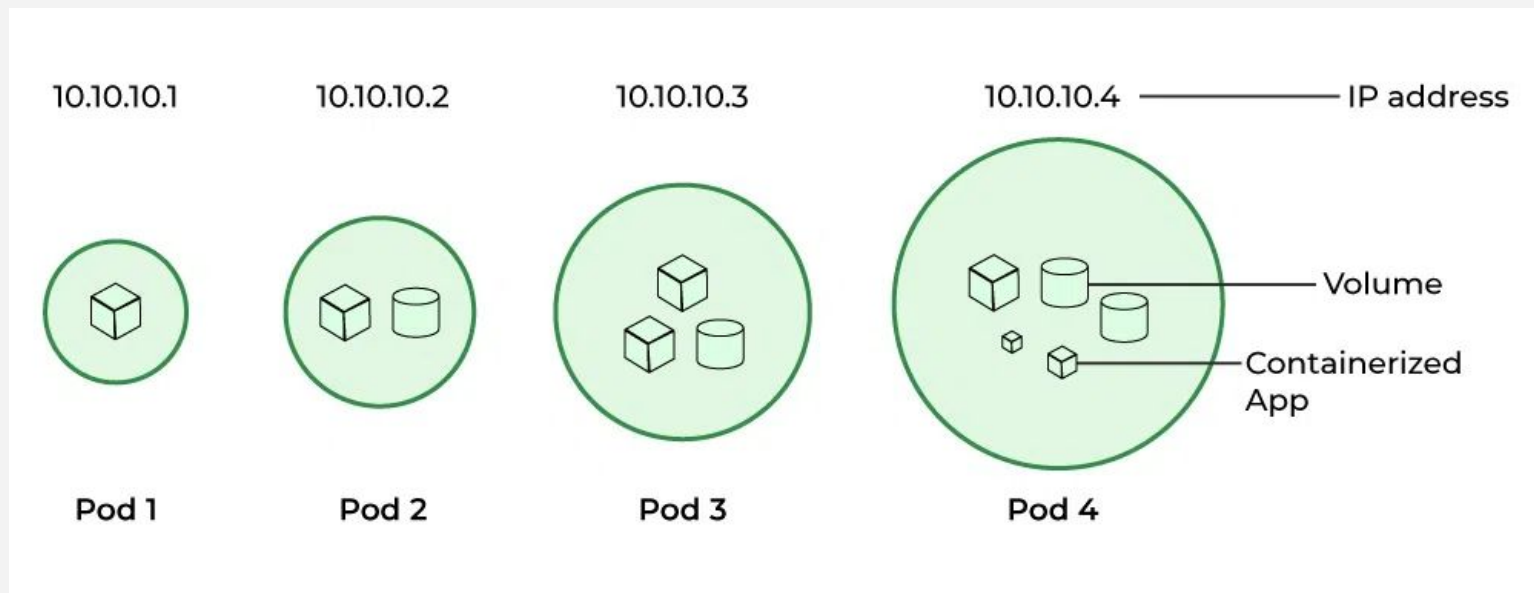


¿Que es un POD?

El Pod es la unidad más pequeña y básica en el modelo de objetos de Kubernetes. Representa una instancia en ejecución de un contenedor o un grupo de contenedores que comparten el mismo espacio de red y almacenamiento.

- ❖ Un Pod puede contener uno o más contenedores (generalmente uno).
- ❖ Los contenedores en un Pod comparten la misma IP, espacio de puertos y almacenamiento.
- ❖ Los Pods son efímeros, es decir, no se espera que vivan para siempre. Kubernetes reemplazará los Pods caídos con nuevos Pods según sea necesario.

PODS



Recordemos que a su vez estos se encuentran dentro de nodos (máquinas virtuales/físicas)



Deployment en Kubernetes

Un Deployment en Kubernetes es un objeto que define cómo se debe desplegar y gestionar una aplicación en un clúster. Controla la creación, actualización y escalado de Pods para garantizar que la aplicación esté siempre disponible y cumpla con los requisitos especificados. Los Deployments aseguran que un número determinado de Pods esté en ejecución, manejan actualizaciones sin interrupciones y permiten revertir cambios si algo falla.

Características principales

- ***Declarativo***: Define el estado deseado de la aplicación en lugar de instrucciones paso a paso.
- ***Rolling updates***: Permite actualizar la aplicación de manera gradual, sin tiempos de inactividad.
- ***Rollback automático***: En caso de problemas, puede revertir automáticamente a la versión anterior.
- ***Escalado automático***: Ajusta el número de réplicas de la aplicación según la carga de trabajo.



Ciclo de vida de un deployment en kubernetes

El ciclo de vida de un **Deployment** en Kubernetes asegura que las aplicaciones se actualicen de manera segura y sin interrupciones:

Creación 🚀

Definir las especificaciones de la aplicación: cuántos **Pods** (unidades que ejecutan la aplicación), la versión, recursos como CPU y memoria, etc.

Actualización ↺

Cambiar la configuración, como actualizar la versión de la aplicación o ajustar recursos.

Revisión ✅

Kubernetes comprueba que los nuevos Pods funcionen correctamente y la aplicación esté estable.

Despliegue Completo 🎯

Kubernetes reemplaza los Pods antiguos por los nuevos de forma gradual (usando un **rolling update**) y, si algo falla, puede revertir a la versión anterior.

Service en Kubernetes

Un **Service** es un recurso que ofrece una dirección estable (IP o nombre DNS) para conectar con un grupo de **Pods**. Permite que las aplicaciones se comuniquen entre sí dentro del clúster o desde el exterior, distribuyendo el tráfico entre los Pods (balanceo de carga). Por ejemplo, un Service puede hacer que una aplicación web acceda a una base de datos usando un nombre fijo, incluso si los Pods cambian.



Tipos principales de Service

- **ClusterIP** 🟢

IP interna al clúster para comunicación entre Pods dentro del mismo clúster.

- **NodePort** 🌐

Expone un puerto fijo en cada nodo del clúster, permitiendo acceso desde fuera del clúster.

- **LoadBalancer** ⚖️

Usa un balanceador de carga externo (por ejemplo, en la nube) para distribuir tráfico externo hacia los Pods.

- **ExternalName** 🔗

Permite usar un alias DNS para acceder a servicios externos fuera del clúster, como bases de datos externas.

- **Headless** 🧠

No asigna una IP única al servicio, permite acceso directo a los Pods individuales (útil para bases de datos distribuidas o aplicaciones stateful).

¿Cómo funciona un Service en Kubernetes?

El Service actúa como intermediario inteligente entre usuarios y Pods

1. Selección de Pods

El Service identifica los Pods adecuados usando etiquetas, asegurando que solo los Pods correctos reciban el tráfico.

2. Balanceo de Carga

Distribuye automáticamente el tráfico entre los Pods seleccionados, evitando sobrecargas y mejorando el rendimiento.

3. Descubrimiento Automático

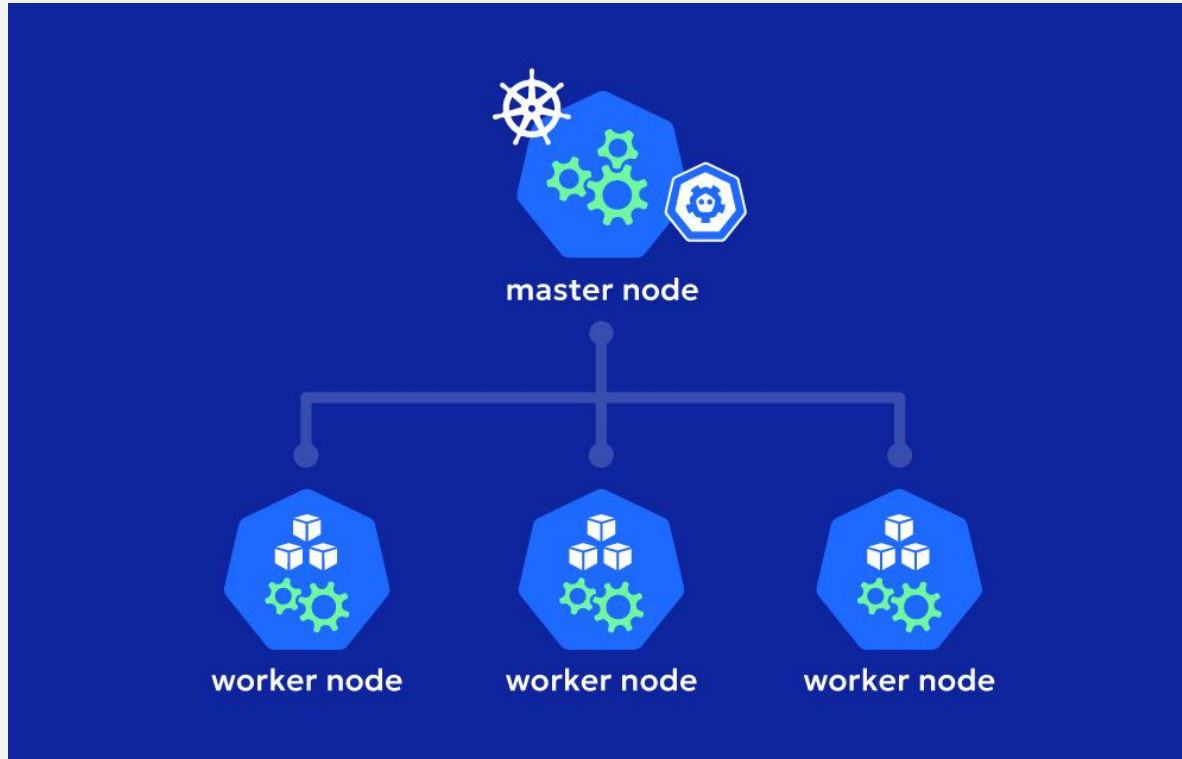
Las aplicaciones encuentran fácilmente otros servicios disponibles en el clúster sin configuraciones manuales complejas.

4. Abstracción de IPs

Proporciona una IP o nombre estable para acceder a los Pods, incluso si estos cambian o se reubican en el clúster.

Arquitectura de Kubernetes





La arquitectura de Kubernetes está compuesta por dos tipos principales de nodos: **nodos maestros** (Master Nodes) y **nodos trabajadores** (Worker Nodes).





Componentes del Master Node




El **Master Node** controla y gestiona el funcionamiento del clúster. Estos son sus componentes clave:

-  **API Server:**
Es el punto central de comunicación. Todas las peticiones hacia Kubernetes pasan por aquí.
-  **etcd:**
Base de datos distribuida que almacena toda la información del estado del clúster.
-  **Controller Manager:**
Monitorea constantemente el clúster para asegurar que siempre coincida con el estado deseado.
-  **Scheduler:**
Asigna y distribuye los Pods eficientemente entre los diferentes nodos disponibles.



Componentes del Worker Node

Los **Worker Nodes** son los encargados de ejecutar tus aplicaciones. Estos son sus componentes principales:

-  **Kubelet:**
Agente que garantiza que los Pods y contenedores se ejecuten correctamente dentro del nodo.
-  **Kube-Proxy:**
Gestiona el tráfico de red entre los Pods y asegura una comunicación efectiva.
-  **Container Runtime:**
Software encargado de ejecutar los contenedores (por ejemplo, Docker o containerd).

¿Vamos bien hasta acá?

Hasta ahora sabemos que Kubernetes es una herramienta de código abierto que sirve para orquestar la gestión de contenedores y hacerla mucho más eficiente. Esto lo logra organizando los contenedores en Pods, que son como pequeños paquetes donde viven uno o más contenedores que trabajan juntos. Estos Pods se alojan en Nodos, que son máquinas (físicas o virtuales) dentro de un clúster (un grupo de nodos conectados). Los nodos se dividen en dos tipos:

- **Master Node:** Es el cerebro del clúster, encargado de manejar toda la gestión, como decidir dónde van los Pods, monitorear el estado y coordinar cambios.
- **Worker Nodes:** Son los que hacen el trabajo pesado, alojando los Pods donde corren nuestras aplicaciones.

Ahora, la gran pregunta: ¿De qué manera Kubernetes orquesta y optimiza mis contenedores?

¿Cómo Kubernetes optimiza nuestros contenedores?

Kubernetes mejora la eficiencia y estabilidad de nuestras aplicaciones gracias a estas características clave:

1- Escalabilidad automática

Kubernetes escala las aplicaciones automáticamente según la carga de trabajo y la disponibilidad de recursos.

2- Alta disponibilidad

Diseñado para mantener las aplicaciones siempre en funcionamiento, incluso si algún componente falla.

3- Despliegues continuos

Permite implementar nuevas versiones sin interrupciones, gracias a estrategias como rolling updates.

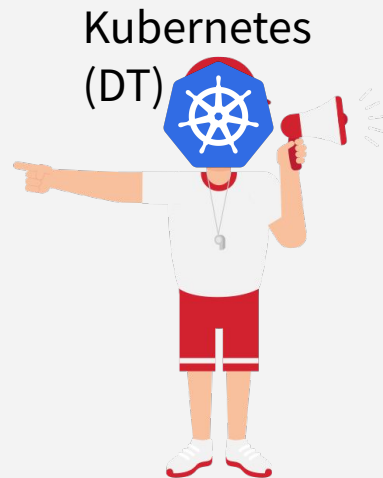
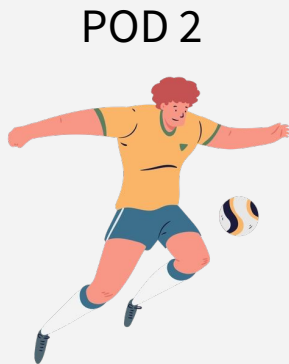
4- Gestión eficiente de recursos

Asegura que cada aplicación reciba solo los recursos necesarios, evitando el desperdicio de hardware.

Kubernetes como director técnico de fútbol

Escalabilidad automática

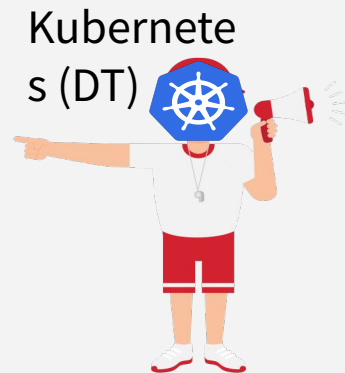
Supongamos que el equipo está jugando un partido y el **delantero estrella** (pod 1) está recibiendo muchas oportunidades de gol porque los aficionados (usuarios) están exigiendo más jugadas ofensivas desde la tribuna. Kubernetes, como entrenador, monitorea los recursos del equipo y las demandas del partido.



Kubernetes como director técnico de fútbol

Escalabilidad automática

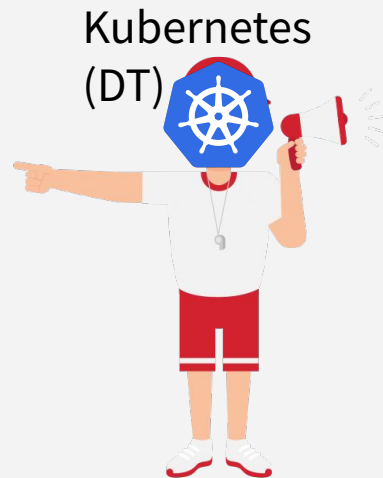
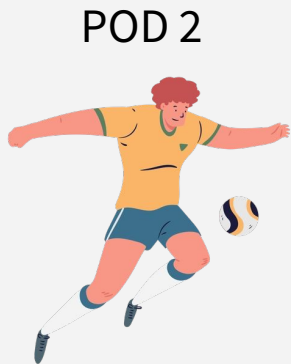
Si el delantero está sobrecargado (alta demanda de solicitudes), Kubernetes decide **escalar** el ataque creando **nuevos delanteros** (réplicas del pod 1) en el campo (otros nodos). Estos nuevos delanteros se suman al partido sin interrupciones, distribuyendo la carga de las jugadas ofensivas. Kubernetes redirige las oportunidades de gol a los distintos delanteros de manera eficiente, asegurando que el equipo siga atacando sin tiempos muertos, sin importar cuántas jugadas demande el público.



Kubernetes como director técnico de fútbol

Alta disponibilidad

Ahora, imagina que el equipo está en pleno partido, con el **mediocampista central** (pod 3) organizando el juego, pero de repente este jugador se lesiona o no puede continuar (falla del pod). Los aficionados no quieren que el partido se detenga, ya que eso arruinaría la experiencia. Kubernetes, como entrenador, ya tiene un plan: siempre mantiene **mediocampistas suplentes** (réplicas del pod 3) listos en otros nodos (banca)

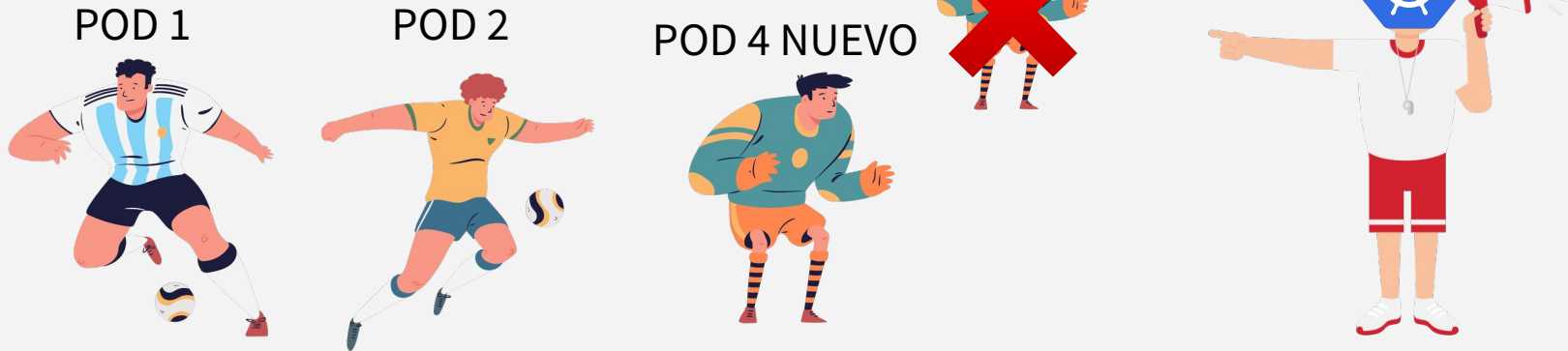


Kubernetes como director técnico de fútbol

Alta disponibilidad

Cuando detecta la falla, Kubernetes descarta al mediocampista lesionado y lo reemplaza inmediatamente por el **mediocampista suplente** (pod 4, una réplica en otro nodo). Este nuevo jugador entra al campo sin que el partido se interrumpa, manteniendo el ritmo del juego y asegurando que los aficionados no noten la falla.

POD 3 (antiguo)



Kubernetes como director técnico de fútbol

Despliegues continuos

El entrenador (Kubernetes) quiere cambiar la táctica del equipo de defensiva (táctica 1) a ofensiva (táctica 2) sin pausar el partido. Usa un **rolling update** para sustituir gradualmente a los jugadores (pods). Por ejemplo, cambia un defensa por un extremo rápido y un mediocampista defensivo por uno creativo, ajustando poco a poco su rol. Los jugadores antiguos salen y los nuevos entran sin que los aficionados noten interrupciones, logrando un equipo completamente ofensivo al final.

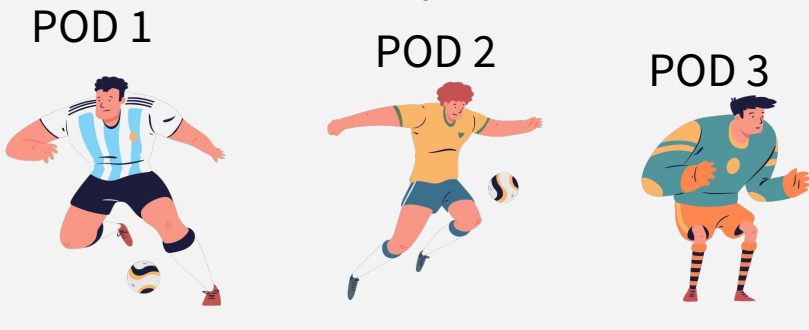
TÁCTICA 2 (OFENSIVA)

Participación 30%

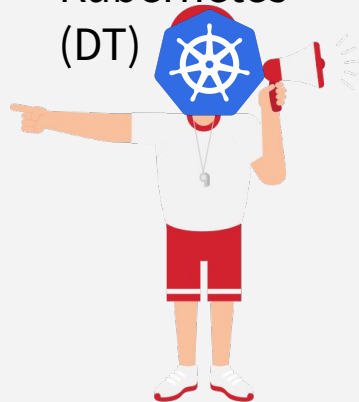


TÁCTICA 1 (DEFENSIVA)

Participación 70%



Kubernetes
(DT)



Kubernetes como director técnico de fútbol

Despliegues continuos

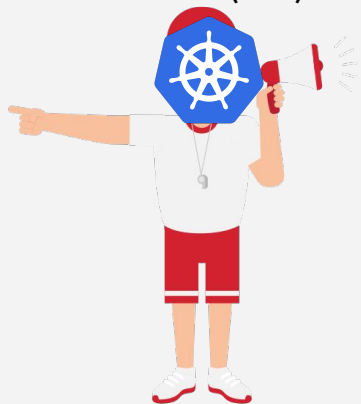
Al final, el equipo está completamente renovado, jugando la nueva táctica ofensiva al 100%, y los jugadores de la táctica anterior han sido retirados. Los aficionados no notaron el cambio, ya que el partido continuó sin interrupciones, y ahora disfrutan de un juego más emocionante.

TÁCTICA 2 (OFENSIVA)

Participación 100%



Kubernetes (DT)



Gestión de recursos

Como vimos antes kubernetes puede gestionar nuestros recursos de las maneras más eficientes, siempre priorizando la efectividad y no tener downtimes.

Es importante tener en cuenta que kubernetes siempre va a estar monitoreando nuestros recursos, por ejemplo si se declara que un POD específico requiere 0.25 de cpu y 256 mb de memoria, Kubernetes se asegurará de que haya siempre los suficientes recursos en los nodos para satisfacer los requisitos.

Kubernetes: El motor detrás de las grandes empresas

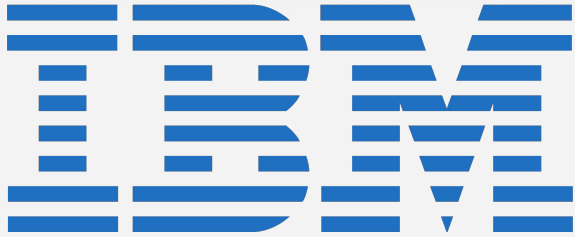
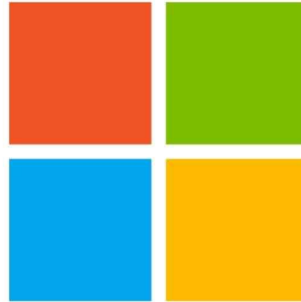
Muchas empresas eligen Kubernetes para organizar sus aplicaciones y servicios de forma eficiente. Es como un administrador inteligente que asegura que todo funcione sin problemas, ahorrando costos al usar solo los recursos necesarios y manteniendo las aplicaciones siempre activas (0 downtime), siempre que esté bien configurado. Por eso, Kubernetes es una herramienta esencial en tecnología, y entender qué es y cómo funciona, aunque sea a nivel teórico, abre muchas puertas en el mundo laboral.

Algunas empresas que usan kubernetes

amazon

The Amazon logo consists of the word "amazon" in a bold, black, sans-serif font. Below the text is a curved orange arrow that starts under the 'a' and points towards the 'n'.

IBM

The IBM logo is the word "IBM" in a bold, blue, sans-serif font. Each letter is composed of eight horizontal blue stripes of equal thickness and height.

GitLab

Virtualización activada en BIOS/UEFI.

1. Reiniciar la PC y entrar al BIOS
 - a. Presionamos la tecla **DEL** o **F2** varias veces (depende del modelo) hasta que veas la pantalla del **BIOS**.
2. Entrar en el **modo EZ** o **Advanced**
 - a. Hacer click en "**EZ mode**" o presionar **F7** para pasar al modo avanzado (Advanced Mode).
3. Activar Virtualizacion
 - a. Ir a la pestaña **Advanced**.
 - b. Entrar en la opción CPU Configuration.
 - c. Buscar la opción llamada:
 - i. Intel Virtualization Technology (VT-x) si tenés un procesador Intel.
 - ii. **SVM Mode** si tenés un procesador AMD.
 - d. Cambiarla a Habilitada (Enabled).
4. Guardar y salir
 - a. Presione **F10** para guardar y salir del BIOS.

Mis favoritos

Principal

Ai Tweaker

Opciones avanzadas

Monitorizar

Arrancar

Monitor hardware

12 Core(s) Running @ 3703 MHz 1100 mV

Max Speed:3700 MHZ

Microcode Patch Level: A201016

----- Caché por núcleo -----

L1 Instruction Cache: 32 KB/8-way

L1 Data Cache: 32 KB/8-way

L2 Cache: 512 KB/8-way

Total L3 Cache per Socket: 64 MB/16-way

Compatibilidad con PSS

Habilitada

Modo NX

Habilitada

Modo SVM

Habilitada

Modo SMT

Automática

Modo de equilibrio de núcleos

Modo automático

CCD Control

Automática

CPU

Frecuencia 3700 MHz Temperatura 53°C

BCLK Freq 100.00 MHz Voltaje principal 1.392 V

Relación 37x

Memoria

Frecuencia 3600 MHz Capacidad 32768 MB

Voltaje

+12V 12.076 V +5V 5.020 V

+3.3V 3.296 V

Habilitar o deshabilitar la actualización de CPU

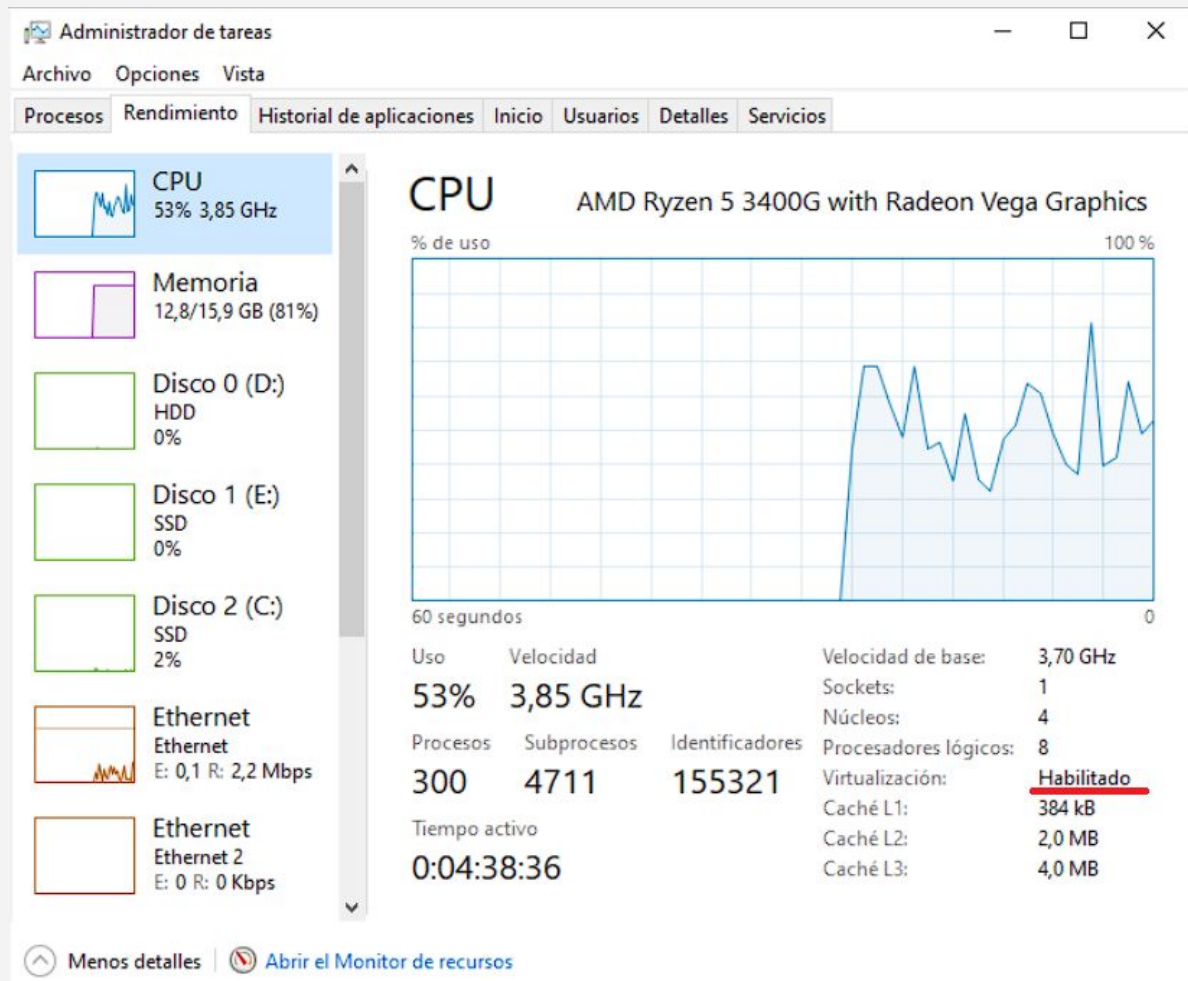
Última modificación

EzMode (F7) | →

Teclas de acceso directo ?

Verificación

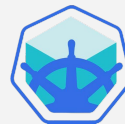
1. Abrir el Administrador de tareas->Rendimiento->CPU
2. Tiene que decir
Virtualización habilitada



Instalación

Descargar e Instalar Minikube, Kubectl y Docker desde:

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

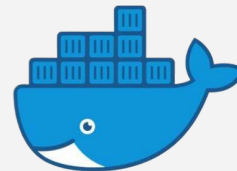


<https://kubernetes.io/docs/tasks/tools/>



<https://minikube.sigs.k8s.io/docs/drivers/docker/>

<https://docs.docker.com/desktop/setup/install/windows-install/>





Minikube

Requisitos:

1. **2 CPUs** o más.
2. **2GB** de memoria ram.
3. **20GB** de espacio en el disco duro.
4. Conexión a internet.
5. alguna de estas máquinas virtuales: **Docker**, QEMU, Hyperkit, Hyper-V, KVM, Parallels, Podman, VirtualBox, or VMware Fusion/Workstation.

Actualmente está disponible la versión **1.35.0**



Documentation

Get Started!

Handbook

Basic controls

Deploying apps

Kubectl

Accessing apps

Addons

Configuration

Dashboard

Pushing images

Proxies and VPNs

Registries

Certificates

Offline usage

Host access

Network Policy

Persistent Volumes

Mounting filesystems

File Sync

Troubleshooting

FAQ

Presentations

1 Installation

Click on the buttons that describe your target platform. For other architectures, see [the release page](#) for a complete list of minikube binaries.

Operating system

Linux

macOS

Windows

Architecture

x86-64

Release type

Stable

Installer type

.exe download

Windows Package Manager

Chocolatey

To install the latest minikube **stable** release on **x86-64 Windows** using **.exe download**:

1. Download and run the installer for the [latest release](#).

Or if using PowerShell, use this command:

```
New-Item -Path 'c:\' -Name 'minikube' -ItemType Directory -Force
Invoke-WebRequest -OutFile 'c:\minikube\minikube.exe' -Uri 'https://github.com/kubernetes/minikube/releases/latest/download/minikube-windows-amd64.exe' -UseBasicParsing
```

2. Add the `minikube.exe` binary to your `PATH`.

Make sure to run PowerShell as Administrator.

```
$oldPath = [Environment]::GetEnvironmentVariable('Path', [EnvironmentVariableTarget]::Machine)
if ($oldPath.Split(';') -notcontains 'C:\minikube'){
    [Environment]::SetEnvironmentVariable('Path', $('{0};C:\minikube' -f $oldPath), [EnvironmentVariableTarget]::Machine)
}
```

If you used a terminal (like powershell) for the installation, please close the terminal and reopen it before running minikube.



Kubectl

Opciones:

1. Instalar kubectl en Linux
2. Instalar kubectl en macOS
3. Instalar kubectl en Windows:
 - a. Instalar kubectl a través de binario en windows (Descarga directa o curl).
 - b. Instalar kubectl usando Chocolatey (gestor de paquetes)

Actualmente está disponible la versión **1.32**

🔍 Search this site

▶ Documentation

▶ Getting started

▶ Concepts

▼ Tasks

▼ Install Tools

Install and Set Up kubectl on
Linux

Install and Set Up kubectl on
macOS

**Install and Set Up kubectl on
Windows**

▶ Administer a Cluster

▶ Configure Pods and Containers

▶ Monitoring, Logging, and Debugging

▶ Manage Kubernetes Objects

▶ Managing Secrets

▶ Inject Data Into Applications

▶ Run Applications

▶ Run Jobs

▶ Access Applications in a Cluster

▶ Extend Kubernetes

▶ TLS

▶ Manage Cluster Daemons

▶ Networking

Extend kubectl with plugins

Manage HugePages

Schedule CRI

Install kubectl on Windows

The following methods exist for installing kubectl on Windows:

- [Install kubectl binary on Windows \(via direct download or curl\)](#)
- [Install on Windows using Chocolatey, Scoop, or winget](#)

Install kubectl binary on Windows (via direct download or curl)

1. You have two options for installing kubectl on your Windows device

◦ Direct download:

Download the latest 1.32 patch release binary directly for your specific architecture by visiting the [Kubernetes release page](#). Be sure to select the correct binary for your architecture (e.g., amd64, arm64, etc.).

◦ Using curl:

If you have `curl` installed, use this command:

```
curl.exe -LO "https://dl.k8s.io/release/v1.32.0/bin/windows/amd64/kubectl.exe"
```

Note:

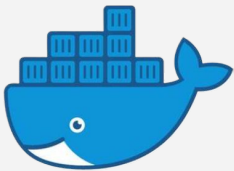
To find out the latest stable version (for example, for scripting), take a look at <https://dl.k8s.io/release/stable.txt>.

2. Validate the binary (optional)

Download the `kubectl` checksum file:

```
curl.exe -LO "https://dl.k8s.io/v1.32.0/bin/windows/amd64/kubectl.exe.sha256"
```

Enable shell autocompletion
Install kubectl convert plugin
What's next



Docker Desktop

Instalamos Docker Desktop

Disponible para Mac, **Windows** o Linux

Requisitos del Sistema:

1. WSL version 1.1.3.0 o posterior.
2. Windows 10 64-bit.
3. Window 11 64-bit.
4. 4GB RAM
5. Virtualización habilitada.

Actualmente está disponible la versión **4.40.0**

Install Docker Desktop on Windows

Page options ▾

Docker Desktop terms

Commercial use of Docker Desktop in larger enterprises (more than 250 employees OR more than \$10 million USD in annual revenue) requires a [paid subscription](#).

This page provides download links, system requirements, and step-by-step installation instructions for Docker Desktop on Windows.

[Docker Desktop for Windows - x86_64](#)

[Docker Desktop for Windows - Arm \(Beta\)](#)

For checksums, see [Release notes](#)

System requirements




Tip

Should I use Hyper-V or WSL?

Docker Desktop's functionality remains consistent on both WSL and Hyper-V, without a preference for either architecture. Hyper-V and WSL have their own advantages and disadvantages, depending on your specific setup and your planned use case.

[WSL 2 backend, x86_64](#) [Hyper-V backend, x86_64](#) [WSL 2 backend, Arm \(Beta\)](#)

- WSL version 1.1.3.0 or later.

 [Edit this page](#)

✓ [Request changes](#)

Table of contents

System requirements

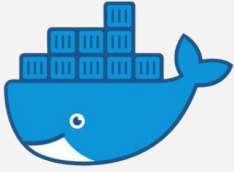
[Install Docker Desktop on Windows](#)

[Install interactively](#)

[Install from the command line](#)

[Start Docker Desktop](#)

[Where to go next](#)



Docker Desktop

Abrimos el panel de control de **Docker Desktop**

Activamos la opción de **Kubernetes**:

1. Ir a Docker Desktop->Settings->kubernetes
2. Activar la casilla "**Enable Kubernetes**"
3. Hacer click en "**Apply & restart** "

General

Resources

Docker Engine

Builders

Kubernetes

Software updates

Extensions

Features in development

Notifications

Kubernetes



Enable Kubernetes

Start a Kubernetes single or multi-node cluster when starting Docker Desktop.

Cluster



docker-desktop

kubeadm, 1 node, v1.32.2



Running

Started 4 hours ago

Reset cluster

Cluster settings

Choose cluster provisioning method



Kubeadm

Create a single-node cluster with kubeadm.

Version: v1.32.2



kind

[SIGN IN REQUIRED](#)

Create a cluster containing one or more nodes with kind. Requires the [containerd image store](#)



Show system containers (advanced)

Show Kubernetes internal containers when using Docker commands.

Cancel

Apply & restart



Engine running



Kubernetes running

RAM 5.66 GB CPU 10.29% Disk: 6.30 GB used (limit 1006.85 GB)

>_ Terminal

✓ v4.40.0