Clase 8- Contenedores, Docker Compose, Podman, estandarización

miércoles, 7 de mayo de 2025 16:41

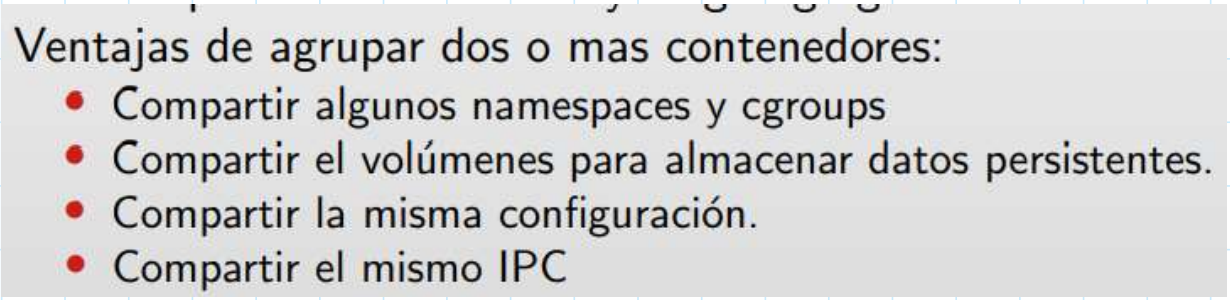
Para docker compose un contenedor es un servicio
Se usa un YAML de configuración para crear servicios, almacenamiento y red
Por default, Compose establece una red default a la que todos los contenedores se unen. Pero es posible definir redes propias para cada Compose

Podman-> pod manager
Es un container engine daemonless para desarrollar, administrar y ejecutar contenedores OCI (Open Container Initiative) en sistemas Linux.
Es un motor de contenedores que no depende de un daemon, a diferencia de Docker. Esto lo hace más seguro y flexible, ya que los contenedores se ejecutan directamente como procesos individuales en lugar de depender de un proceso centralizado. Además, es compatible con el estándar OCI, lo que le permite trabajar con imágenes de Docker y otros contenedores OCI.

Docker es un container engine

Un pod comprende uno o m´as contenedores que comparten los mismos namespaces
En docker compose no comaprten el namespace

Podman no necesita un demonio central para administrar los conentedores. Es como un proceso normal solo que aislado con namespaces y cgoups.



IPC (Inter-Process Communication) es un mecanismo que permite a los procesos en un sistema operativo **comunicarse entre si** y **compartir datos**.

• Por cada contenedor dentro del pod existe un proceso conmon.
• Conmon es un programa C liviano que monitorea un contenedor hasta que finaliza.
Es una herramienta de comunicaci´on entre el container engine (Podman) y el OCI runtime (runc o crun).

- Por cada contenedor dentro del pod existe un proceso *conmon*.
- *Conmon* es un programa C liviano que monitorea un contenedor hasta que finaliza.
- Es una herramienta de comunicación entre el container engine (Podman) y el OCI runtime (runc o crun).
- Ejecuta el runtime, indicándole donde se encuentra el archivo *OCI spec* y el rootfs (capa que será el punto de montaje en el contenedor).
- Su principal tarea es monitorear el proceso principal del contenedor.
- Salva el código de salida si el contenedor muere.
- Mantiene la tty del contenedor abierta para poder conectarse a él.

- Los **nodos** son **máquinas** (físicas o virtuales) que forman parte de un clúster de orquestación de contenedores.
- Los **contenedores** son **instancias** ejecutándose en los nodos, dentro de un entorno aislado, que contienen aplicaciones.
- Los nodos **no son contenedores**, pero son responsables de ejecutar y gestionar los contenedores dentro del clúster.

- Ayudan a automatizar la creación de un contenedor.
- **Engine:** herramienta de software que acepta y procesa solicitudes de los usuarios para crear un contenedor:
  - CLI y/o interface REST para la interacción de los usuarios.
  - Descarga y extrae las imágenes
  - Interactúa con el correspondiente runtime.
  - Presenta el stack de directorios de la imagen y la capa R/W al runtime.
  - Docker, Podman, CRI-O, rkt, LXD, etc.
- **Runtime:** software de bajo nivel usado por el engine para ejecutar contenedores en un nodo:
  - Inicia los contenedores con la información pasada por el engine
  - Administra la alocación de los cgroups y las políticas de control de acceso (SELinux, AppArmor).
  - Mayoría de los runtimes siguen la especificación OCI.
  - runc, crun, railcar, gVisor, etc.

- Cluster: grupo de nodos interconectados que trabajan en conjunto.
- Permite aprovisionar, desplegar, escalar y administrar automáticamente contenedores sin preocuparse por la infraestructura subyacente.
- Creación de servicios de manera declarativa.
- En general, dos tipos de nodos:
  - **Manager:** encargado de administrar el cluster.
  - **Worker:** encargado de ejecutar las aplicaciones.
- *Service Discovery:* orquestador brinda información para encontrar otro servicio.
- *Routing:* paquetes deben llegar entre servicios ejecutando en diferentes nodos.
- *Load Balancing:* distribuir las cargas de trabajos entre las distintas instancias de un servicio
- *Scaling:* aumentar/disminuir las instancias de un servicios según la carga de trabajo.
- Kubernetes, Docker Swarm, RedHat Openshift, Rancher, etc.

Un kernel común a todos los conteiners pero les hago creer que estàn solos. Segmenta dentro de un mismo so.
Cada uno ocupa poquito: filesystem de la app y lo que consuma la app.
Bootea ràpido
Lo puedo mover fàcil