Conceptos básicos de Docker

Autor: Alberto Herrera Poza
Github: https://github.com/AlbertoIHP

Introducción

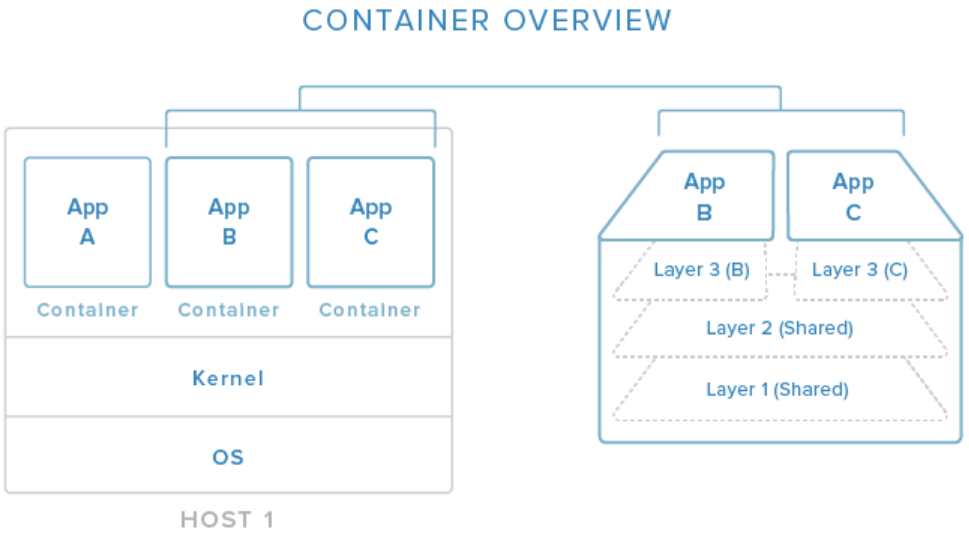
El presente documento tiene la finalidad de definir los conceptos básicos para el uso de Docker por cualquier miembro del equipo de desarrollo.

1. *Contenedorización*

La contenedorización es el proceso de distribución e implementación de aplicaciones de forma portátil y predecible. Lo logra mediante el empaquetado de componentes y sus dependencias en entornos de proceso estandarizados, aislados y livianos llamados contenedores. Se puedan implementar fácilmente en sistemas distribuidos, permitiendo que el sistema se escale fácilmente y sobreviva a fallas de máquinas y aplicaciones.

2. *Docker y Contenedorización*

Docker es el software de contenedorización más común en uso hoy en día. Si bien existen otros sistemas de contenedorización, Docker simplifica la creación y gestión de contenedores y se integra con muchos proyectos de código abierto.



En la arquitectura de contenedores de Docker, el primer análisis que se puede hacer es el de **Host/Apps**, donde, es posible levantar tantos contenedores (apps) como sea necesario en nuestro sistema Host. Por otro lado, la relación **Apps/Layers**, permite que cada

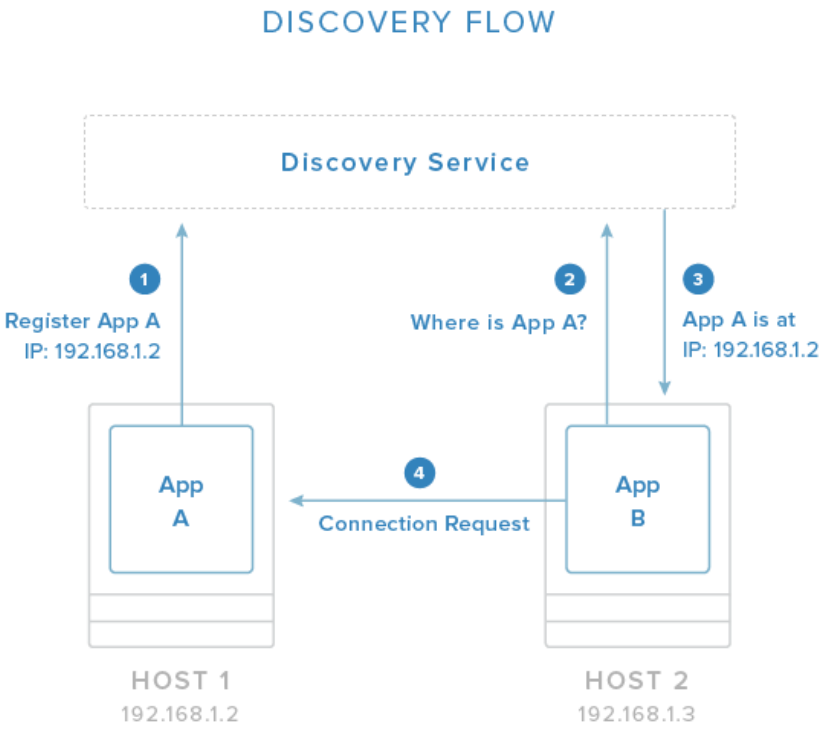
contenedor (app) se cree en base a distintas capas (layers), esto, con el principal objetivo de reducir los recursos utilizados por parte del Host.

Las ventajas que los usuarios Docker tienen, son:

- Utilización ligera de recursos: **en lugar de virtualizar** un sistema operativo completo, los contenedores se aíslan en el nivel del proceso y **usan el núcleo del host.**
- Portabilidad: todas las dependencias para una aplicación en contenedor están agrupadas dentro del contenedor, lo que **le permite ejecutarse en cualquier host Docker**.
- Previsibilidad: al host no le importa lo que se está ejecutando dentro del contenedor y al contenedor no le importa en qué host se está ejecutando. **Las interfaces están estandarizadas y las interacciones son predecibles.**

3. Descubrimiento de servicios y almacenes de configuración global

El descubrimiento de servicios se utiliza para que los contenedores puedan conocer el entorno en el que se han introducido sin la intervención del administrador.



En la imagen anterior, puede ver un flujo de ejemplo en el que una aplicación registra su información de conexión con el sistema de servicio de descubrimiento. Una vez registrados, otras aplicaciones pueden consultar el servicio de descubrimiento para averiguar cómo conectarse a la aplicación. En general, los almacenes de valores clave proporcionan una API HTTP para acceder y establecer valores. Algunas herramientas de descubrimiento de servicios populares y proyectos relacionados son:

- **etcd** : descubrimiento de servicios / almacén de valores-clave distribuidos globalmente
- **cónsul** : descubrimiento de servicios / almacén de valores clave distribuido globalmente
- **zookeeper** : descubrimiento de servicios / tienda de valor-clave distribuida globalmente
- **crypt** : proyecto para encriptar entradas etcd
- **confd** : busca cambios en el almacén de valores clave y activa la reconfiguración de servicios con nuevos valores

4. Herramientas de red

Docker proporciona las estructuras de red básicas necesarias para la comunicación de contenedor a contenedor y de contenedor a host.

- Exponer los puertos de un contenedor y, opcionalmente, asignar al sistema host para el enrutamiento externo. Puede seleccionar el puerto de host para asignar o permitir que Docker elija aleatoriamente un puerto alto no utilizado.
- Permitir que los contenedores se comuniquen utilizando los "enlaces" de Docker. Un contenedor vinculado obtendrá información de conexión sobre su contraparte, lo que le permitirá conectarse automáticamente si está configurado para prestar atención a esas variables. Esto permite el contacto entre contenedores en el mismo host sin tener que saber de antemano el puerto o la dirección donde se ubicará el servicio.

Aparte de estas dos opciones que ofrece el Core de Docker, existen otras alternativas.

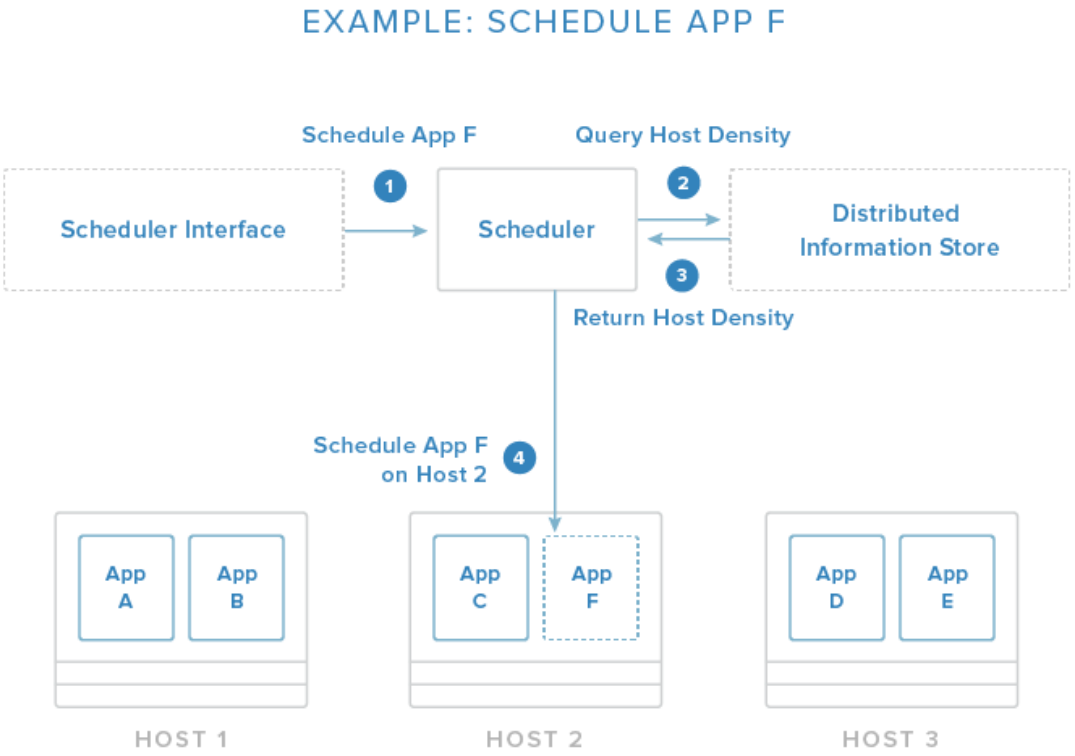
- Superposición de redes para simplificar y unificar el espacio de direcciones en varios hosts.
- Redes privadas virtuales adaptadas para proporcionar comunicación segura entre varios componentes.
- Asignación de subredes por host o por aplicación
- Establecimiento de interfaces macvlan para la comunicación.
- Configurar direcciones MAC, puertas de enlace, etc. personalizadas para sus contenedores

Algunos proyectos que están involucrados con la mejora de las redes de Docker son:

- **flannel** : red superpuesta que proporciona a cada host una subred separada.
- **weave** : red superpuesta que retrata todos los contenedores en una sola red.
- **pipework** : kit de herramientas de red avanzado para configuraciones de red arbitrariamente avanzadas.

5. *Programación, gestión de clústeres y orquestación*

Otro componente necesario al construir un entorno de contenedor agrupado es un planificador. Los programadores son responsables de iniciar los contenedores en los hosts disponibles.



- La solicitud se entrega a través de una API o herramienta de administración.
- El planificador evalúa las condiciones de la solicitud y el estado de los hosts disponibles.
- Extrae información sobre la densidad del contenedor de un servicio distribuido de almacenamiento / descubrimiento de datos (como se discutió anteriormente) para que pueda colocar la nueva aplicación en el host menos ocupado.

Este proceso de selección de host es una de las principales responsabilidades del planificador. Por lo general, tiene funciones que automatizan este proceso con el administrador que tiene la opción de especificar ciertas restricciones. Algunas de estas restricciones pueden ser:

- Programe el contenedor en el mismo host que otro contenedor dado.
- Asegúrese de que el contenedor no esté ubicado en el mismo host que otro contenedor dado.
- Coloque el contenedor en un host con una etiqueta o metadatos coincidentes.
- Coloque el contenedor en el host menos ocupado.
- Ejecute el contenedor en cada host del clúster.

El planificador es responsable de cargar contenedores en los hosts relevantes y de iniciar, detener y administrar el ciclo de vida del proceso.

Algunos proyectos populares que funcionan como programadores y herramientas de gestión de flotas son:

- **fleet** : programador y herramienta de gestión de clúster.
- **marathon** : programador y herramienta de gestión de servicios.
- **Swarm** : programador y herramienta de gestión de servicios.
- **mesos** : servicio de abstracción de host que consolida los recursos de host para el planificador.
- **kubernetes** : planificador avanzado capaz de gestionar grupos de contenedores.
- **compose** : herramienta de orquestación de contenedores para crear grupos de contenedores.