Redes en Docker

Antonio Espín Herranz

Contenidos

- Introducción
- Redes predefinidas
- Crear Redes
- Rangos autogenerados
- Rangos específicos
- Especificar redes al crear contenedores
- Inspeccionar redes
- Conectar y desconectar contenedor a una red.

Introducción

- Las redes se componen de:
  - El modelo **CNM** (modelo de red de contenedores)
    - Arquitectura modular para crear redes de contenedores
    - Se divide en:
      - Sandbox: lleva la configuración: IP, tabla de enrutamiento y configuración DNS.
      - EndPoint: red virtual de un contenedor para conectar una sandbox a una red.
      - Network: Grupo de endpoints para comunicarse entre si
  - El componente **libnetwork**
    - Sirve para gestionar redes en Docker, provee compatibilidad con drivers de red y plugins.
  - Los **drivers** de red
    - Para comunicar contenedores entre ellos y el exterior.

Redes predefinidas

- Para contenedores **Linux**:
  - **bridge** = Se utiliza por defecto. No se puede eliminar. Se pueden crear personalizadas (para poder aislar contenedores que se ejecutan dentro del mismo host de Docker). Cada contenedor tiene su IP.
  - **host** = El contenedor no tiene su propia IP. Sólo para Docker en Linux.
  - **overlay** = Red distribuida para contenedores que se ejecutan en diferentes hosts de Docker.
  - **macvlan** = se puede asignar una dirección MAC a cada una de las interfaces de red virtuales de un contenedor.
  - **null** = Red llamada none. Tiene deshabilitada las funciones de red.
- Para contenedores **Windows**:
  - **nat** = bridge
  - **overlay** = overlay
  - **transparent** = macvlan
  - **null** = null
  - **12bridge** = Se conecta a la misma red que el host de Docker. El contenedor tendrá su IP.
  - **12tunnel** = Similar a 12bridge, el tráfico de la red se envía al host de Docker

Crear redes

- Las redes de **tipo bridge** se comportan como **redes privadas** que permiten a los contenedores que se están ejecutando en el mismo host de Docker puedan comunicarse entre sí.

- A nivel interno tienen todos los puertos accesibles, pero a nivel externo hay que publicarlos.

- Al crear el contenedor se indican los **puertos**:

  -p, --publish

  -P, --publish-all

Crear redes

- Redes bridge **predefinidas**:
  - Se crea de forma predefinida
  - No se puede eliminar
  - La utilizan los contenedores por defecto, que se encuentran en el mismo host.

- Redes bridge **definidas** por el usuario:
  - Se crean de forma explícita por el usuario.
  - Proporcionan mejor aislamiento
  - Se crea un DNS automática para que los contenedores de la misma red puedan acceder a los otros (de su red) a través del nombre, pero en la red predefinida tienen que acceder con su dirección IP o con la opción --link

Tipo Bridge

- Es la red que más se suele utilizar en Docker por su simplicidad:

- **Aislamiento por defecto:**
  - Cada red bridge está aislada de otras redes y del host, lo que proporciona una capa básica de seguridad.
  - Los contenedores en la misma red bridge pueden comunicarse directamente entre sí utilizando sus direcciones IP internas.

- **Facilidad de configuración:**
  - Es el tipo de red que Docker configura automáticamente cuando no se especifica una red personalizada.
  - No requiere configuraciones adicionales para funcionar en la mayoría de los casos.

- **Mapeo de puertos:**
  - Puedes exponer servicios en los contenedores al host a través del mapeo de puertos (-p o --publish), lo cual es ideal para escenarios simples.

- **Uso típico:**
  - Se utiliza mucho en entornos locales y en configuraciones de desarrollo debido a su simplicidad.
  - Ejemplo: cuando ejecutas un contenedor como docker run -p 8080:80 nginx, estás usando una red bridge predeterminada para que el contenedor sea accesible desde el puerto 8080 del host.

Crear redes

- **docker network**
  - **connect** conectar un contenedor a la red
  - **create** crear una red
  - **disconnect** desconectar de una red
  - **inspect** visualizar los detalles de la red
  - **ls** listar redes
  - **prune** borrar las redes no utilizadas
  - **rm** borrar redes

Crear redes

- **docker network create nombre_red**

- Contenedores de Linux:
  - **docker network create –d bridge nombre_red**

- Contenedores de Windows:
  - **docker network create –d nat nombre_red**

Listar redes

- **docker network ls**
- Muestra de cada red:
  - NETWORK_ID, NAME, DRIVER, SCOPE

- Ejemplos:
  - docker network ls --no-trunc
  - docker network ls --filter driver=bridge

```
C:\Users\Anton>docker network ls
NETWORK ID     NAME           DRIVER    SCOPE
fdbbd7a75658   bridge         bridge    local
2ff0ff815611   host           host      local
64ef527c1d9d   none           null      local
fbde254455da   wordpress_net  bridge    local
```

Rangos autogenerados

- **Redes de tipo bridge:**
  - Docker asigna automáticamente un rango de subred que no entre en conflicto con otras subredes ya en uso en el sistema.
  - Estas direcciones suelen ser algo como 172.17.0.0/16 u otras variantes similares, dependiendo de las configuraciones previas.

  - Si generamos otra red le asigna: 172.18.0.0/16
  - 172.19.0.0/16 así sucesivamente…

  - Por defecto, /16 → son 16 bits para el prefijo de red y 32 – 16 = 16 bits para las direcciones IP 2^16 → 65536 direcciones IP

Rangos específicos

- Redes con rangos específicos de direcciones IP:
  - Opción **--subnet** al crear una red con docker network create.

- Permite especificar tanto la subred como otros parámetros, como el rango de direcciones IP dentro de la red.
  - docker network create **--subnet=192.168.1.0/24** mi_red_personalizada
  - Son 32 – 24 (prefijo de red) = 8 bits
  - 2^8 → 256 direcciones IP de: 192.168.1.**0** a 192.168.1.**255**

Rangos específicos

- Especificar un rango de **IPs** dentro de la subred:
  - docker network create --subnet=192.168.1.0/24 --ip--range=192.168.1.100/28 mi_red_con_rango

  - --ip-range=192.168.1.100/28: Este rango específico limita las direcciones IP asignables a las IPs entre 192.168.1.100 y 192.168.1.115, dejando el resto de la subred libre.

  - 32 – 28 = 4 bits → 2^4 = 16 direcciones IP

Inspeccionar Redes

- **docker network inspect id_nombre_red**
  - **ID de la red**: El identificador único de la red.
  - **Nombre**: El nombre que le diste a la red al crearla.
  - **Driver**: El tipo de controlador utilizado (por ejemplo, bridge, overlay, etc.).
  - **Scope**: El ámbito de la red (local o global).
  - **Subredes** (IPAM Config): Información sobre la asignación de direcciones IP, como las subredes y las puertas de enlace (gateways).
  - **Contenedores conectados**: Lista de los contenedores que están actualmente conectados a la red, incluyendo:
    - ID del contenedor.
    - Nombre del contenedor.
    - Dirección IP asignada dentro de la red.
    - Dirección MAC.

Conectar contenedores a una red

- Para conectar:
  - docker network connect
  - Ejemplo:
    - **docker network connect mi_red mi_contenedor**

  - Al crear el contenedor utilizar la opción –network (docker run)
  - La red habría que crearla antes:
    - Ejemplo:
      - **docker network create mi_red**
      - **docker run --network mi_red --name mi_contenedor nginx**

Desconectar contenedores a una red

- docker network **disconnect** mi_red mi_contenedor