**Docker Network**

**What is Docker Networking?**

* By default, **Docker containers are isolated**, but
* **Docker network** allows **containers** to communicate with
  + each other,
  + the host system,
  + and the external world (internet)

It’s essential for building **distributed**, **microservices-based**, and **scalable** applications.

**Types of Docker Networks (main are listed below, others are Macvlan, 3rd-party plugins (e.g., Calico, Weave))**

* **Bridge**
  + default network
  + created automatically
  + used for standalone containers on a single host
  + containers get private IPs
  + containers can ping each other via their IPs (or container name)
  + **Host machine** and other containers **outside the bridge** can't reach containers unless ports are published.
  + can define our own bridge networks for name resolution 🡪 Custom Bridge network  
      
    *docker network create --driver bridge my-bridge*
* **Host**
  + shares host's network
  + No network isolation — container can access host ports directly.
  + No need to publish ports — container services are available on host ports.
  + used when container needs high performance or access to host ports
    - *docker run --network host myimage*
  + Only works on Linux.
  + Can cause port conflicts.

* **None**
  + no networking for container
  + useful for highly isolated containers (for security or testing)
  + *docker run --network none myimage*
* **Overlay**
  + used in Docker Swarm
  + containers across different hosts to communicate
  + *docker network create --driver overlay my-overlay*

**Custom Bridge Network (Recommended)**

**Example:**

* Create custom bridge network: *docker network create my-bridge*

*Or,  
docker network create --driver bridge my-bridge*

* Run containers in custom network:  
  *docker run -d --name db-app\_container -h db-app -p 5433:5432 -e POSTGRES\_USER=postgres -e POSTGRES\_PASSWORD=postgres --network my-bridge db-app:v1.0  
    
  docker run -d --name writer-app\_container -h writer-app -p 8080:5000 --network my-bridge writer-app:v1.0  
    
  docker run -d --name reader-app\_container -h reader-app -p 8081:5001 --network my-bridge reader-app:v1.0*
* Now, the frontend container can talk to backend using the **service name** (container name)
  + Test:
    - Verify network:
      * docker inspect container db-app\_container
      * docker inspect container writer-app\_container
      * docker inspect container reader-app\_container
    - Ping:  
      ping writer-app\_container after entering to each container  
      * docker exec -it db-app\_container /bin/bash
      * docker exec -it writer-app\_container /bin/bash
      * docker exec -it reader-app\_container /bin/bash

If error come – ‘Error: bash: ping: command not found’ then run command inside container - apt update && apt install -y iputils-ping

Other commands:

* List networks
  + docker network ls
* Inspect a network
  + docker network inspect my-network
* Connect a container to another network
  + docker network connect my-network my-container
* Disconnect a container
  + docker network disconnect my-app-net my-container
* Remove a network
  + docker network rm my-network

Practise:

**How do we achieve the below objectives?**

* frontend1 ↔️ db ✅
* frontend2 ↔️ db ✅
* frontend1 ❌ frontend2 (isolated)

Create **two separate bridge networks**:

* writer-db-net
* reader-db-net

Then connect the db-app container to **both networks**, and each frontend to only its respective network.

Create networks:

* docker network create writer-db-net
* docker network create reader -db-net

Run DB container and connect to both networks

* docker run -d --name db-app\_container -h db-app -p 5433:5432 -e POSTGRES\_USER=postgres -e POSTGRES\_PASSWORD=postgres --network writer-db-net db-app:v1.0

Then connect DB to second network:

* docker network connect reader-db-net db-app\_container

Run writer-app container (only on writer-db-net):

* *docker run -d --name writer-app\_container -h writer-app -p 8080:5000 --network* writer-db-net *writer-app:v1.0*

Run reader-app container (only on reader-db-net):

* *docker run -d --name reader-app\_container -h reader-app -p 8081:5001 --network* reader-db-net *reader-app:v1.0*

**Docker Compose for Connecting Containers Easily**

A screenshot of a computer program

AI-generated content may be incorrect.

**Summary**

To connect two Docker containers:

* Use a **custom bridge network** for same-host communication.
* Use **Docker Compose** for ease and readability.
* Use **Docker Swarm services + overlay network** for multi-host setups.