Université Sorbonne Paris Nord
École d'ingénieur Sup Galilée

Du développement au déploiement d'applications web

Différents types de réseaux

Réalisé par :

Célian WIRTZ (celian.wirtz@edu.univ-paris13.fr)

3ème année
Année universitaire : 2025/2026

Réseaux Docker : concepts et types

Docker connecte des conteneurs via des “drivers” réseau (modes de réseau), qui déterminent **comment** un conteneur communique avec les autres conteneurs et avec l’extérieur.

Bridge (par défaut)

- C’est le réseau le plus courant en local : Docker met automatiquement un conteneur dans le réseau bridge si aucun réseau n’est précisé.
- Principe : Docker crée un réseau virtuel “interne” sur la machine hôte, et les conteneurs communiquent entre eux via IP interne (et souvent via DNS si réseau bridge “custom”).
- Commande utile : inspecter la config du bridge (subnet, gateway, conteneurs attachés) via “docker network inspect bridge”.

Host

- Le conteneur n’a pas de NAT propre et n’a pas d’IP dédiée au conteneur.
- Avantage : performance et simplicité quand on veut accéder directement aux ports de l’hôte.

Le conteneur utilise le réseau de la machine sur lequel s'exécute Docker, donc « docker run --network host ngix » ne marche pas si la machine n'est pas un linux.

None

- Isolation totale : le conteneur n’a pas d’interface réseau, donc il ne peut pas communiquer avec d’autres conteneurs ni avec l’extérieur.
- Cas d’usage : tâches qui n’ont pas besoin d’I/O réseau.

Réseau personnalisé (bridge “custom”)

- Bonne pratique : créer un réseau bridge dédié à une application plutôt que d’utiliser le bridge par défaut.
- Exemples vus :
    - Lancer un conteneur directement dans un réseau : docker run -d --name site --network monreseau nginx.
    - Lancer un autre conteneur dans le même réseau (ex. un conteneur “api” en alpine) pour qu’ils puissent communiquer : docker run -d --name api --network monreseau alpine sleep 1000.

- Attacher/détacher un conteneur après coup :
    - Attacher : docker network connect monreseau site.
    - Détacher : docker network disconnect monreseau site.

Publier un service vers l'extérieur (-p)

- En bridge, rendre un conteneur accessible depuis l'hôte se fait par mapping de ports : docker run -d -p 8080:80 nginx (port hôte 8080 → port conteneur 80).
- Idée clé : sans -p, le service reste "interne" au réseau Docker et n'est pas directement joignable depuis la machine (hors cas particuliers).

Docker Compose : décrire une architecture multi-conteneurs

Docker Compose sert à définir et lancer une application composée de plusieurs services (ex : front + API + DB), via un fichier docker-compose.yml qui décrit tout l'architecture.

Ce que Compose automatise

- Au lieu de taper plusieurs docker run, on fait docker compose up pour :
    - La mise en réseau des services (réseaux dédiés, découverte par noms).
    - La création et le démarrage des conteneurs.
    - Les volumes (persistance et montages de code).

Structure typique du docker-compose.yml

- services : chaque service correspond à un conteneur (ou un groupe de conteneurs identiques selon le scaling, même si Compose n'est pas Kubernetes).
- Champs classiques :
    - image (utiliser une image existante) ou build (construire depuis un Dockerfile).
    - ports (publier des ports).
    - volumes (monter du code ou stocker des données).
    - depends_on (gérer l'ordre de démarrage "logique" entre services).
- networks : définir des réseaux virtuels pour isoler et connecter les services.
- volumes : définir des volumes nommés pour persister des données (important pour DB).

Exemple : “product API” + “website”

Le cours décrit un mini-projet avec deux dossiers (ex. product/ et website/) et un docker-compose.yml à la racine.

- Service API (backend)
- Service Website (frontend simple)
- Compose relie tout :
    - Publication de ports (ex. exposer l’API et le site sur des ports différents côté hôte).
    - Dépendance : le site dépend du service API (depends_on) car il doit pouvoir l’appeler.
    - Réseau : les services communiquent via le nom du service (ex. appeler http://product-service/... depuis le conteneur website).