AI-03029 Docker Networking Notes

Owner	Ⓡ Rigel Arcayan
Tags	Project
Created time	@September 24, 2025 7:12 PM

In the topic of Docker Networking, explain Docker Networking and components. What are the 7 types of Docker Networks. List all the sudo commands when working in Docker Networks. In using docker-compose, how can I employ or use the type of network? List a docker-compose.yaml with the 7 types of docker network example.

Got it — let's break this down systematically into **concepts, types, commands, docker-compose usage, and an example YAML file.**

---

1. Docker Networking Explained

Docker networking enables **containers** to communicate with each other, with the host machine, and with the outside world. By default, each container runs in isolation, but Docker provides networking features to connect them flexibly.

Key Components

- **Container** – the isolated unit running an application.
- **Network Driver** – a plugin that defines how networking works (e.g., bridge, overlay).
- **Endpoint** – the interface that connects a container to a network.
- **Network Namespace** – the Linux kernel construct that isolates network resources per container.

- **IPAM (IP Address Management)** – allocates and manages IP addresses for containers.

---

2. The 7 Types of Docker Networks

Docker provides several built-in network drivers:

1. **bridge** – Default for standalone containers; containers share a private bridge network on the host.
2. **host** – Removes network isolation between the container and the Docker host (shares host's network stack).
3. **none** – Container has no networking (only loopback).
4. **overlay** – Connects multiple Docker daemons, useful for Swarm mode, multi-host communication.
5. **macvlan** – Assigns a MAC address from the physical network to the container (appears as a real device on LAN).
6. **ipvlan** – Similar to macvlan but with L2/L3 routing modes (more efficient).
7. **network plugins** – Third-party/custom drivers (e.g., Calico, Weave, Cilium).

---

3. Common `sudo` Commands for Docker Networks

```
# List all networks
sudo docker network ls

# Inspect a network
sudo docker network inspect <network_name>

# Create a network
sudo docker network create <network_name>

# Create a network with a specific driver
sudo docker network create --driver <driver_type> <network_name>
```

```
# Connect a container to a network
sudo docker network connect <network_name> <container_name>

# Disconnect a container from a network
sudo docker network disconnect <network_name> <container_name>

# Remove a network
sudo docker network rm <network_name>

# Remove all unused networks
sudo docker network prune
```

---

4. Using Networks in Docker Compose

In `docker-compose.yaml`, you can:

- Define networks under the **networks** section.
- Assign containers (services) to specific networks.
- Use `driver` to choose the network type.

---

5. Example: `docker-compose.yaml` with 7 Network Types

Here's a **demo compose file** showing one service attached to each network type:

```
version: "3.9"

services:
  app_bridge:
    image: nginx
    networks:
      - net_bridge

  app_host:
    image: nginx
    network_mode: host   # uses host network directly
```

```
  app_none:
    image: nginx
    network_mode: none   # no network

  app_overlay:
    image: nginx
    networks:
      - net_overlay

  app_macvlan:
    image: nginx
    networks:
      - net_macvlan

  app_ipvlan:
    image: nginx
    networks:
      - net_ipvlan

  app_plugin:
    image: nginx
    networks:
      - net_plugin

networks:
  net_bridge:
    driver: bridge

  net_overlay:
    driver: overlay

  net_macvlan:
    driver: macvlan
    driver_opts:
      parent: eth0   # host interface
```

```
    ipam:
      config:
        - subnet: 192.168.1.0/24
          gateway: 192.168.1.1

  net_ipvlan:
    driver: ipvlan
    driver_opts:
      parent: eth0
      ipvlan_mode: l2
    ipam:
      config:
        - subnet: 192.168.2.0/24
          gateway: 192.168.2.1

  net_plugin:
    external: true  # assume third-party plugin already installed
    driver: my_custom_plugin
```

⚠️ **Notes:**

- `host` and `none` networks are declared at the **service level** (`network_mode`) and not under `networks`.
- `overlay` requires Docker Swarm (multi-host).
- `macvlan/ipvlan` need proper parent interface & IPAM config.
- `plugin` assumes you installed a network driver plugin.

✅ **Direct Answer Summary:**

Docker networking connects containers, hosts, and external networks using **drivers**. There are 7 main types: bridge, host, none, overlay, macvlan, ipvlan, and plugin. You manage networks using `docker network` commands. In Docker Compose, networks are defined under `networks:` and services attach via `networks:` or `network_mode:`. The example YAML demonstrates all 7 network types.

Would you like me to also **draw a diagram** showing visually how each network type connects containers and hosts, so it’s easier to remember?