Docker and Kubernetes

Session 3

Docker networking

Agenda

- Networking overview
- Docker network types
- Disable networking for a container
- How to use networks

Docker networking

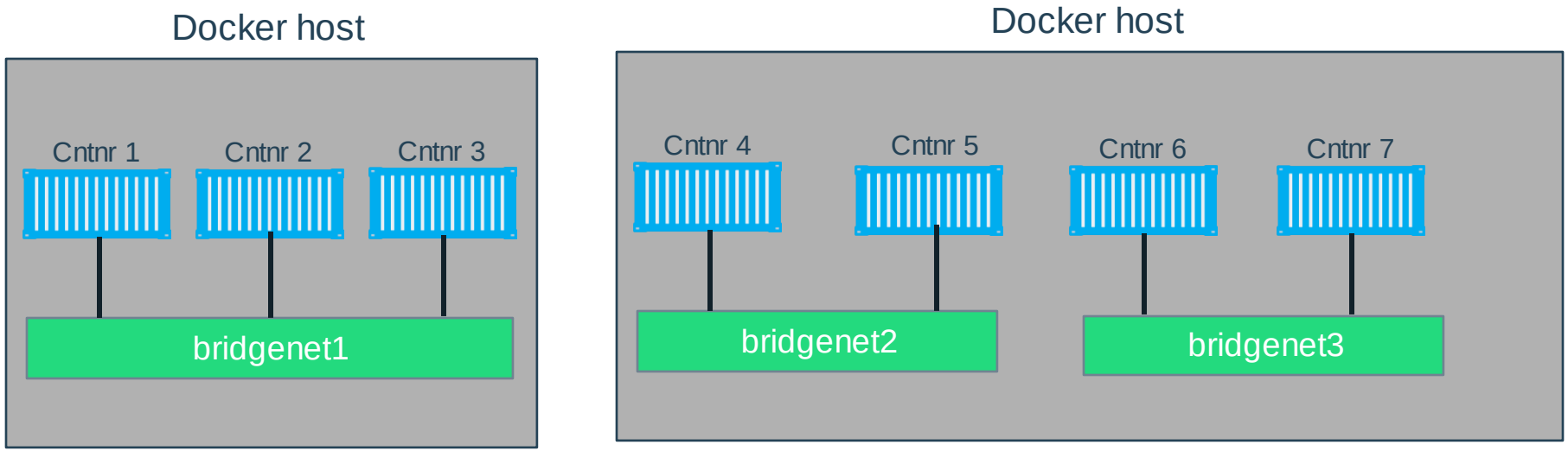
Network driver:

- Bridge
- Host
- None
- Overlay
- Macvlan
- Network plugins: 3rd network plugins

Network driver summary

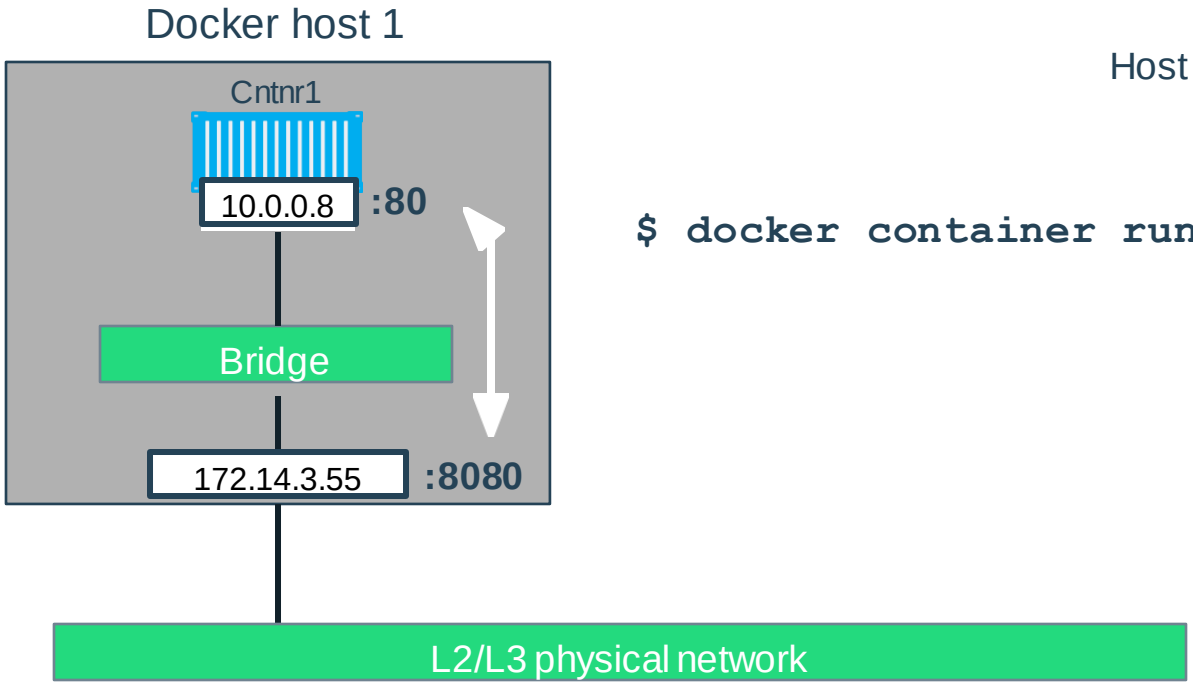
- **User-defined bridge networks** are best when you need multiple containers to communicate on the same Docker host.
- **Host networks** are best when the network stack should not be isolated from the Docker host, but you want other aspects of the container to be isolated.
- **Overlay networks** are best when you need containers running on different Docker hosts to communicate, or when multiple applications work together using swarm services.
- **Macvlan networks** are best when you are migrating from a VM setup or need your containers to look like physical hosts on your network, each with a unique MAC address.
- **Third-party network plugins** allow you to integrate Docker with specialized network stacks.

What is Docker Bridge Networking




```
docker network create -d bridge --name bridgenet1
```

Docker Bridge Networking and Port Mapping



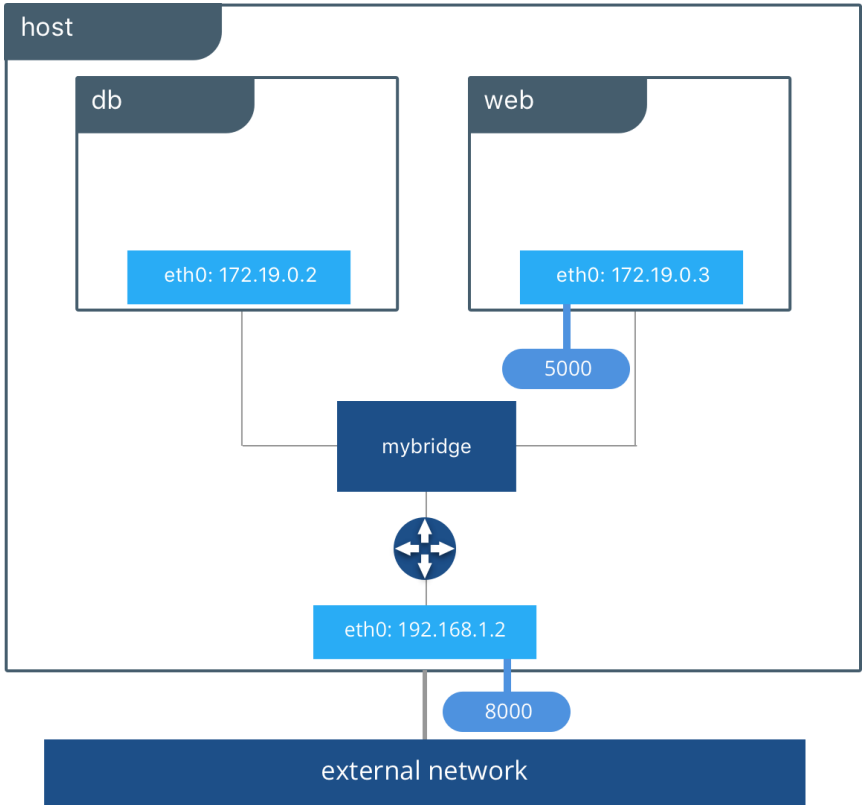
Host port

Container port

```
$ docker container run -p 8080:80 ...
```

Docker Bridge Networking and Port Mapping

With no extra configuration the Docker Engine does the necessary wiring, provides service discovery for the containers, and configures security rules to prevent communication to other networks



Differences between user-defined bridges and the default bridge

- User-defined bridges provide automatic DNS resolution between containers
- User-defined bridges provide better isolation
- Containers can be attached and detached from user-defined networks on the fly
- Each user-defined network creates a configurable bridge
- Linked containers on the default bridge network share environment variables.

Manage a user-defined bridge

Use the **docker network create** command to create a user-defined bridge network.

```
$ docker network create my-net
```

You can specify the subnet, the IP address range, the gateway, and other options

Use the **docker network rm** command to remove a user-defined bridge network. If containers are currently connected to the network, disconnect them first.

```
$ docker network rm my-net
```

Connect a container to a user-defined bridge

When you create a new container, you can specify one or more --network flags

```
$ docker create --name my-nginx --network my-net --publish 8080:80 nginx:latest
```

To connect a running container to an existing user-defined bridge, use the **docker network connect** command

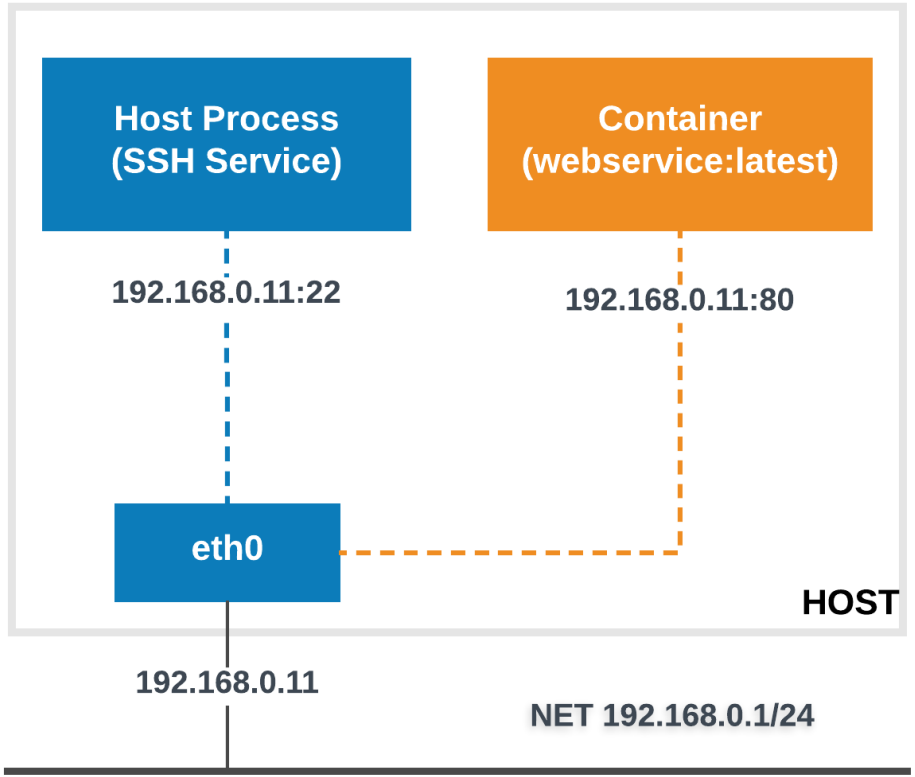
```
$ docker network connect my-net my-nginx
```

Disconnect a container to a user-defined bridge

To disconnect a running container from a user-defined bridge, use the **docker network disconnect** command

```
$ docker network disconnect my-net my-nginx
```

Docker host networking



Use host networking

If you use the host network mode for a container, that container's network stack is not isolated from the Docker host (the container shares the host's networking namespace), and the container does not get its own IP-address allocated.

> Note: Given that the container does not have its own IP-address when using `host` mode networking, port-mapping does not take effect, and the `-p`, `--publish`, `-P`, and `--publish-all` option are ignored, producing a warning instead:

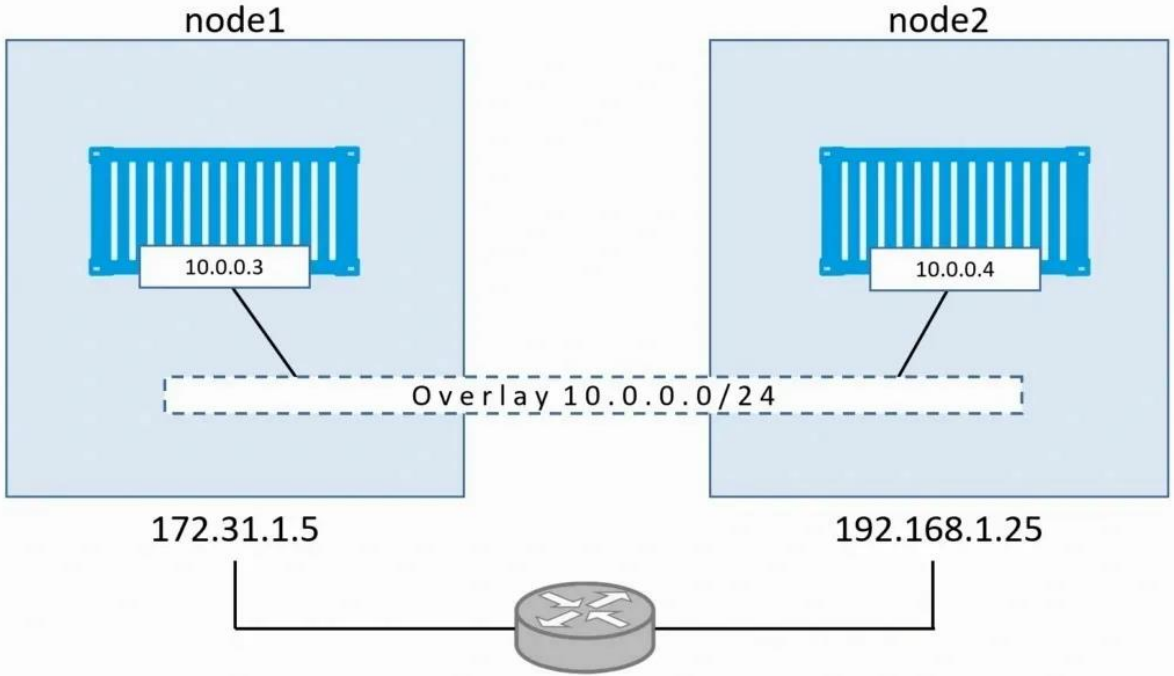
```
WARNING: Published ports are discarded when using host network mode
```

Docker overlay networking

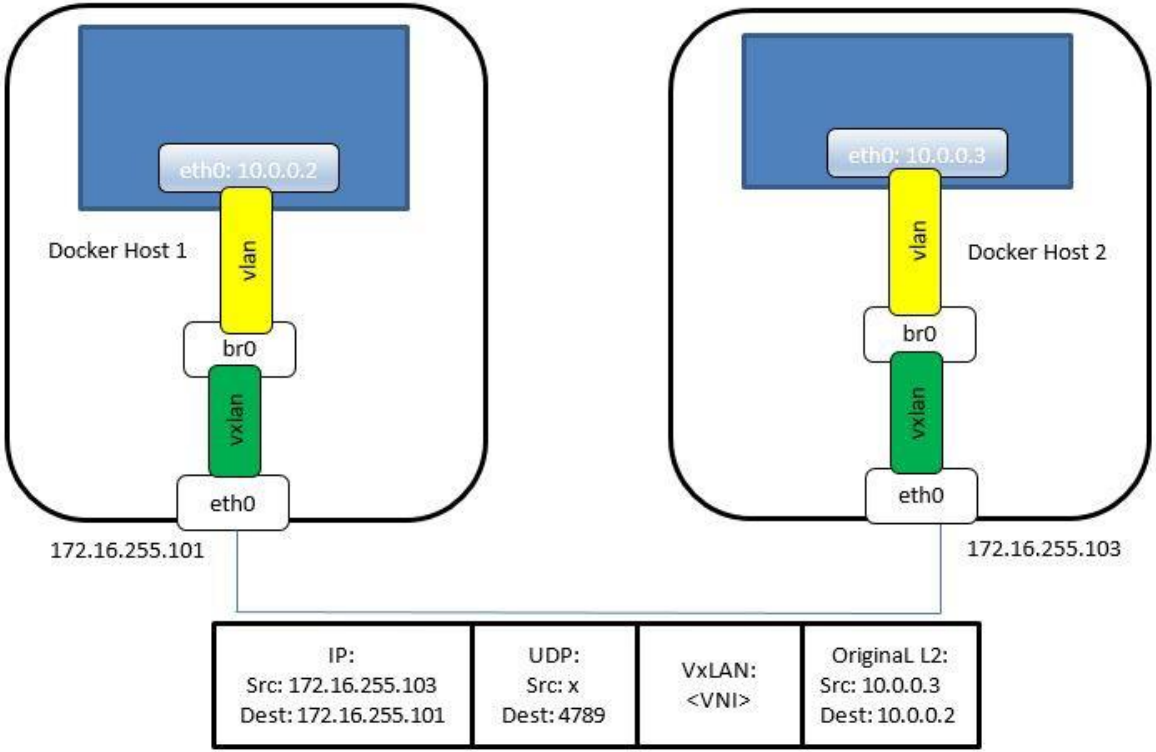
The **overlay network** driver creates a distributed network among multiple Docker daemon hosts.

This network sits on top of (overlays) the host-specific networks, allowing containers connected to it (including swarm service containers) to communicate securely when encryption is enabled.

Docker overlay networking

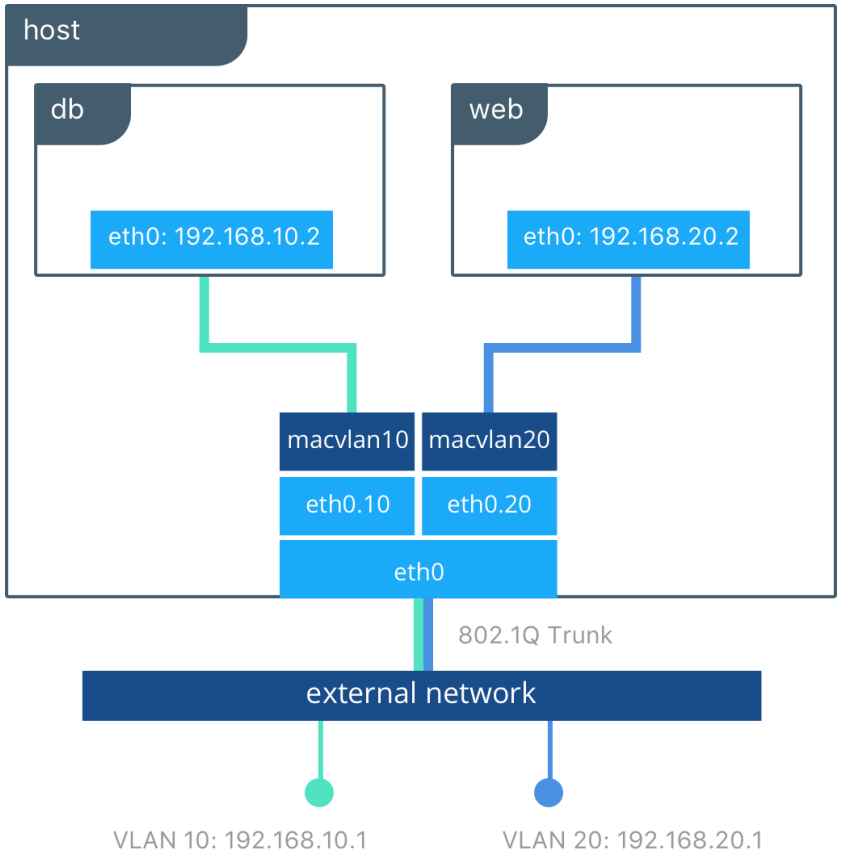


Docker overlay networking

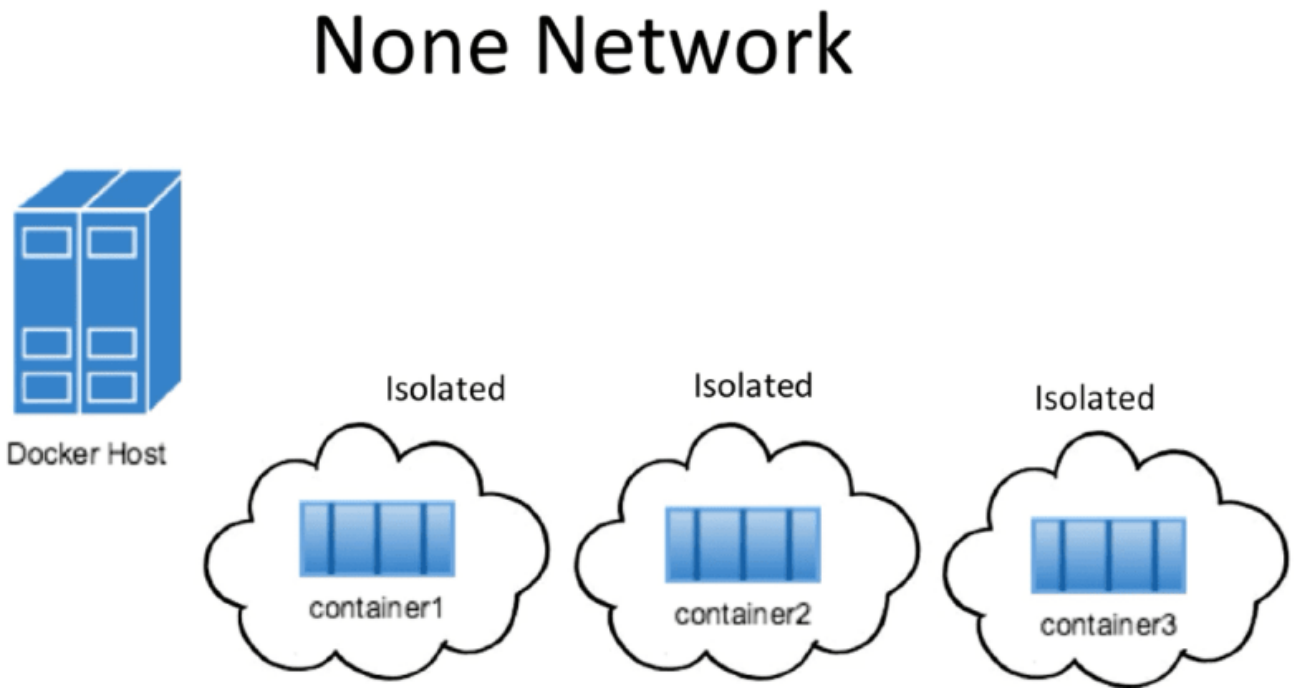


Docker macvlan networking

Some applications, especially legacy applications or applications which monitor network traffic, expect to be directly connected to the physical network.



Docker none network



PRACTICE DOCKER NETWORKING

Q&A