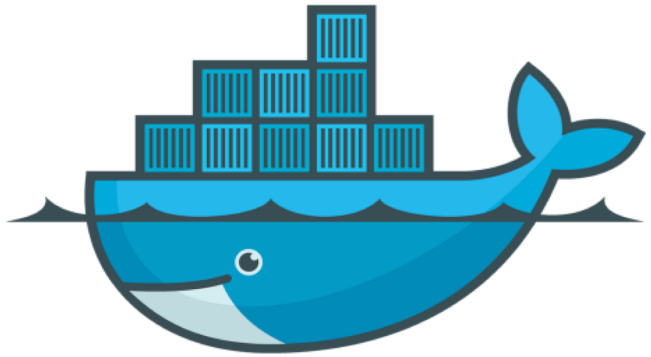


Docker JumpStart

Container Networking Demystified

Agenda

- Working with Ports
- Bridge Networking
- Macvlan Networking
- Overlay Networking

Working with Ports

Docker Networking Basics - Ports

- Most containers will provide some sort of network service such as HTTP or other protocol
- Use `-p` parameter when running a container to control how ports on the host are mapped to ports inside the container.
- If you let Docker engine dynamically assign a port, the ephemeral port range is used, typically ranging from **32768 to 61000**
- If you statically assign ports you run the risk of clashing with an existing bound port

Map container port 80 to a dynamic *ephemeral port* on the host

```
$ docker run -p 80 nginx
```

Map container port 80 to port 80 on the host.
Note. We will get an error is port 80 is already taken

```
$ docker run -p 80:80 nginx
```

Map container port 80 to port 8181 on the host.
Note. We will get an error is port 8181 is already taken

```
$ docker run -p 8181:80 nginx
```

Map container port 2222 to port 22 on host &
map container port 3000 to a dynamic port

```
$ docker run -p 22:2222 -p 3000 my-nodeapp
```

Bridge Networking

Docker Networking Basics - Networks

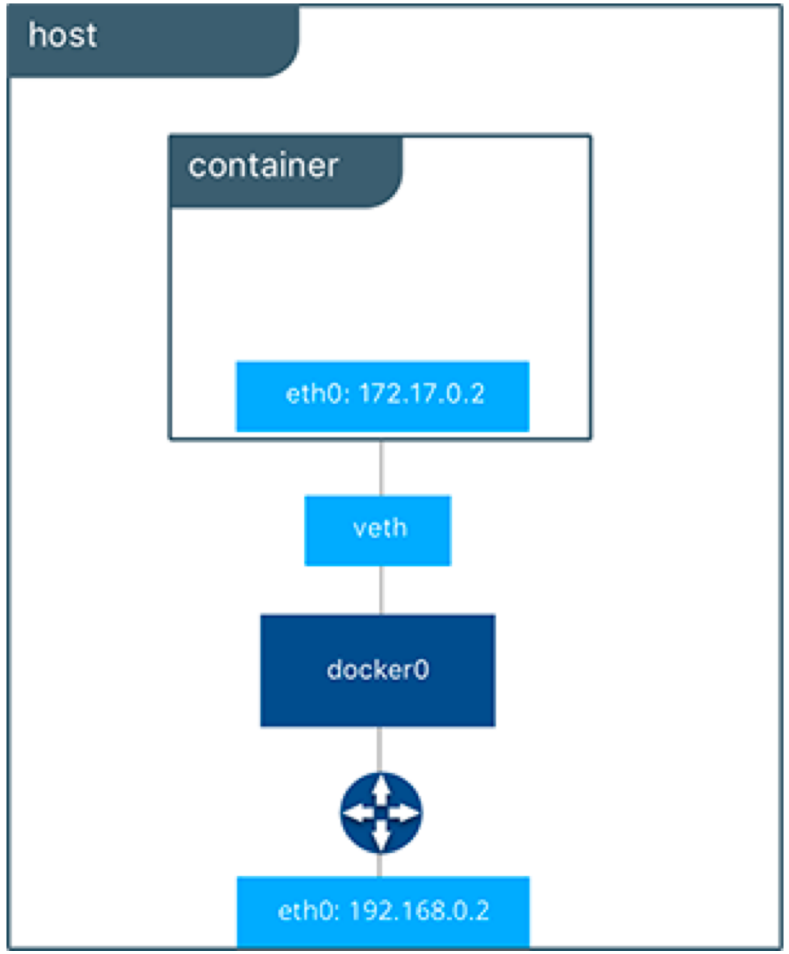
- Docker engine provides 3 networks by default. The **bridge** network is the most commonly used
- The **bridge** network provides a bridge between the host's network and the containers via a virtual NIC on the host
- Default **bridge** network subnet is **172.17.0.0/16**

- Linking containers using the **bridge** network and `--link`, name of container will resolve to IP address

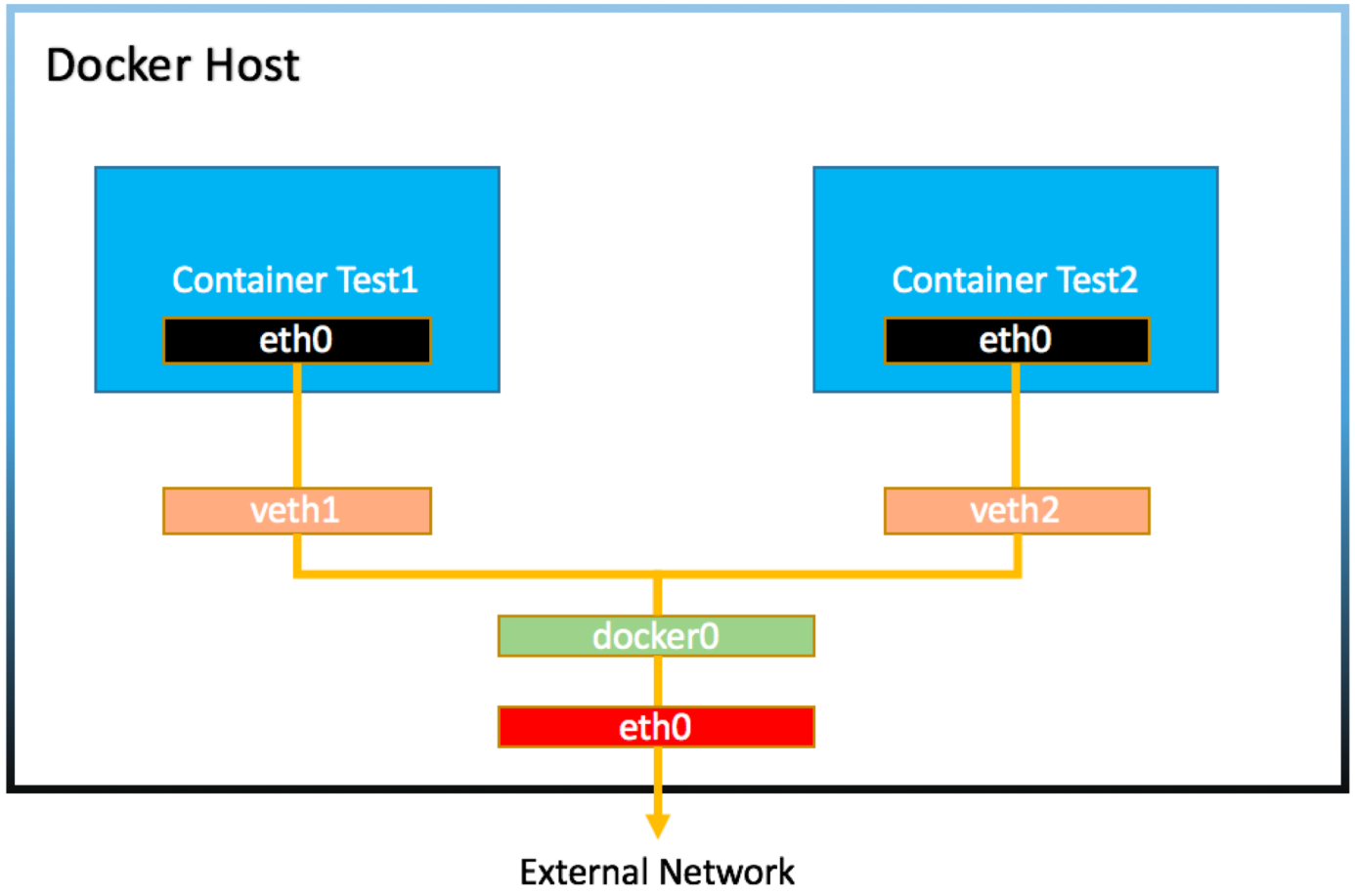
```
$ docker run -d --name db1 mydatabase
$ docker run --it --link db1 ubuntu
/# ping db1
PING db1 (172.17.0.2) 56(84) bytes of data.
64 bytes from db1 (172.17.0.2): icmp_seq=1 ttl=64 time=0.116 ms
```

Note. No ports need to be exposed with **-p** for the two containers to communicate, as they are both on the same 172.17.0.0 private subnet

Bridge Networking

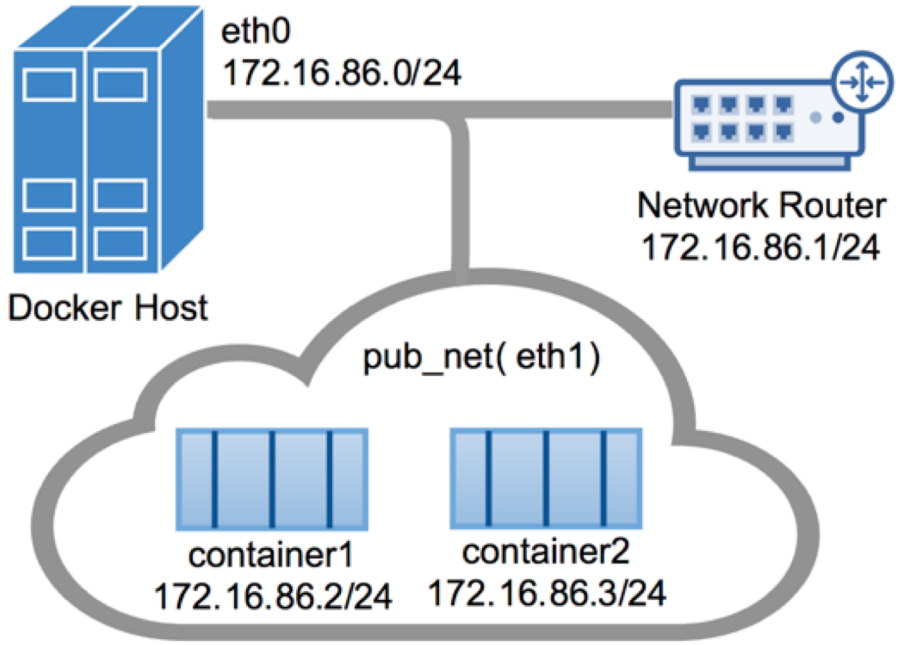


Bridge Networking



Macvlan Networking

Macvlan Networking

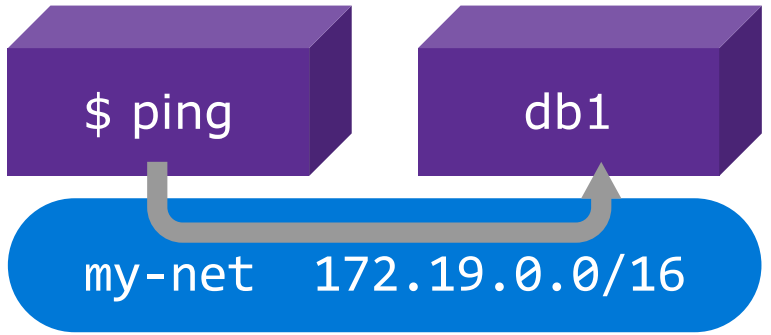


Overlay Networking

Docker Networking - Advanced

- Docker allows you to define your **own networks**
- Assigning a container to a network allows it to see all other containers on that network
  - No need to use `--link` and name resolution is handled automatically

```
$ docker network create my-net
$ docker run -d --name db1 --network my-net mydatabase
$ docker run --it --network my-net ubuntu
/# ping db1
PING db1 (172.19.0.2) 56(84) bytes of data.
64 bytes from db1 (172.19.0.2): icmp_seq=1 ttl=64 time=0.112 ms
```



- **Overlay networks** allow you to combine multiple separate Docker hosts into one logical network, and containers can communicate across hosts

Summary

- Use the -p flag to control how host ports are mapped to container ports
- Bridge networking allows the host and container to share a network connection via a virtual interface
- Macvlan networking allows containers to be directly connected to the physical network
- Overlay networking allows containers to run in the same virtual network across a fleet of host machines