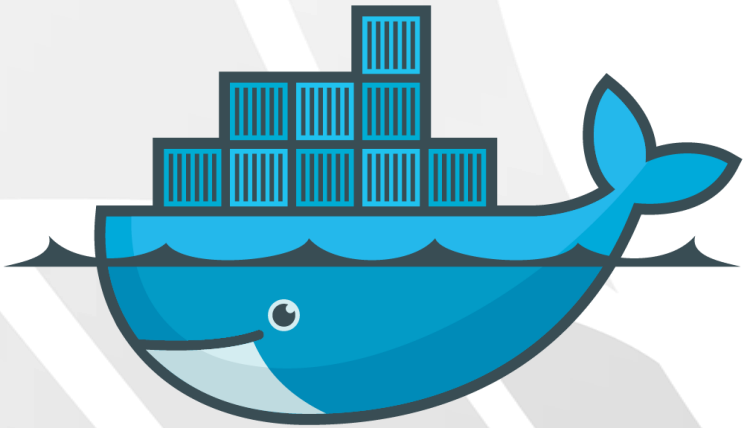
SELA

Module 08: Volumes & Networks

Docker Workshop

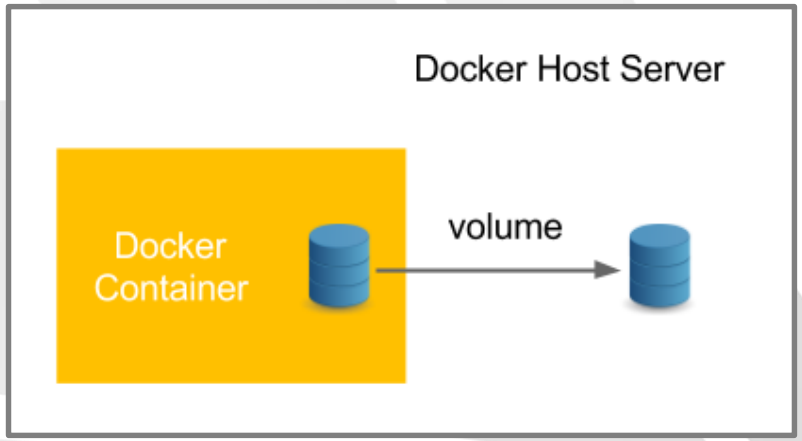


Agenda

- Docker Volumes
- Lab 07: Using Volumes
- Docker Networking
- Lab 08: Docker Networks

Docker Volumes

- When a container dies all the data it has created (logs, database records, etc) dies with it (and remember containers are ephemeral).
- Volumes are external storage areas used to store data produced by a Docker container.
- Volumes can be located on the Docker host or even on remote machines



Docker Volumes

- By default volumes are not deleted when the container is stopped.
- Data volumes can be shared across containers.
- Volumes could be mounted in read-only mode.
- **--volume:** Create a file or directory if it doesn't exist on the Docker host
- **--mount:** Does not automatically create it for you, but generates an error

Docker Volumes - Syntax

```
$ docker run [OPTIONS] -v "volume_name:/container/path" [IMAGE]
```

```
$ docker run [OPT] --mount "type=bind,source=host/path,target=contain/path" [IMAGE]
```

- Optionally you can create a volume with a default name:

```
$ docker run [OPTIONS] -v "container/path" [IMAGE]
```

Docker Volumes – Dockerfile

VOLUME

- Create a new volume with any data that exists at the specified location within the base image.
- Anything after the VOLUME instruction will not be able to make changes to that volume.

```
FROM microsoft/iis
RUN powershell -NoProfile -Command
Remove-Item -Recurse
C:\inetpub\wwwroot\*
WORKDIR /inetpub/wwwroot
COPY . .
VOLUME c:/inetpub/wwwroot
EXPOSE 80
```

Docker Volumes – Managing Volumes

- Create a volume:

```
$ docker volume create volume-name
```

```
$ docker volume create demo-volume
demo-volume
```

- List volumes:

```
$ docker volume ls
```

```
$docker volume ls
DRIVER          VOLUME NAME
local           demo-volume
local           ed702f0a2c8b6ceb56...
local           my_volume
```

Docker Volumes – Managing Volumes

- Inspect a volume:

```
$ docker volume inspect volume-name
```

```
$ docker volume inspect demo-volume
[
  {
      "CreatedAt": "2018-06-07T23:39:20+03:00",
      "Driver": "local",
      "Labels": {},
      "Mountpoint":
"/var/lib/docker/volumes/demo-volume/_data",
      "Name": "demo-volume",
      "Options": {},
      "Scope": "local"
  }
]
```

- Remove a volume:

```
$ docker volume rm volume-name
```

```
$ docker volume rm demo-volume
demo-volume
```

Questions

Lab 07: Using Volumes

Lab

https://gitlab.com/sela-docker-workshop/lab-07

Docker Networking

- Docker includes support for networking containers through the use of network drivers.
- By default, the container is assigned an IP address for every Docker network it connects to (the Docker daemon acts as a DHCP server).
- By default, a container inherits the DNS settings of the Docker daemon (can be overridden on a per-container basis).

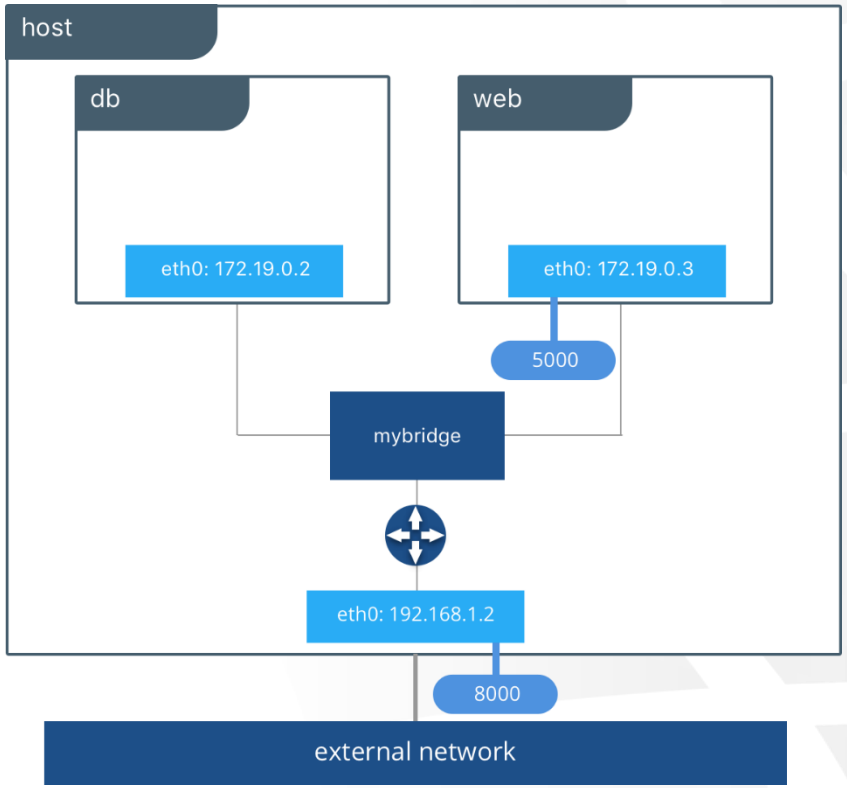
Network Drivers

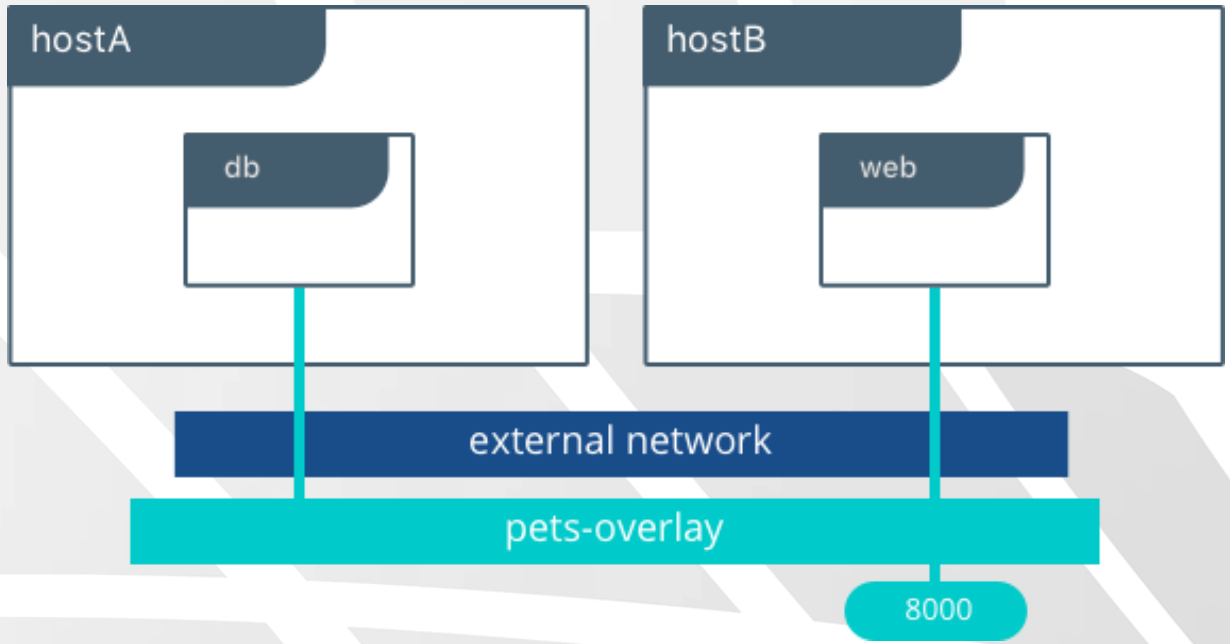
- **bridge:** Allows containers connected to the same bridge network to communicate, while providing isolation from containers which are not connected to that bridge network.
- **host:** For standalone containers, remove network isolation between the container and the Docker host, and use the host's networking directly.
- **overlay:** Creates a distributed network among multiple Docker hosts. (available only using swarm mode)

Network Drivers

- **macvlan:** Macvlan networks allow you to assign a MAC address to a container, making it appear as a physical device on your network.
- **none:** disable all networking for the container.
- **Network Plugins:** You can install and use third-party network plugins with Docker. (Develop or download from the Docker Store)

Bridge vs Overlay Networks





Default Networks

- Every installation of the Docker Engine automatically includes three default networks:

```
$ docker network ls

NETWORK ID          NAME                DRIVER
18a2866682b8        none                null
c288470c46f6        host                host
7b369448dccb        bridge              bridge
```

- Unless you tell it otherwise, Docker always launches your containers in the “bridge” network

Docker Networks – Managing Networks

- Create a network:

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

- Delete a network:

```
$ docker network rm network-name
```

Docker Networks – Managing Networks

- Run a container adding it to an specific network:

```
$ docker run [OPTIONS] --network=network-name [IMAGE]
```

- Add running container to a network:

```
$ docker connect network-name [CONTAINER]
```

Docker Networks – Managing Networks

- Disconnect container from a network:

```
$ docker disconnect network-name [CONTAINER]
```

- Inspect networks:

```
$ docker network inspect network-name
```

Questions

Lab 08: Docker Networks

Lab

https://gitlab.com/sela-docker-workshop/lab-08