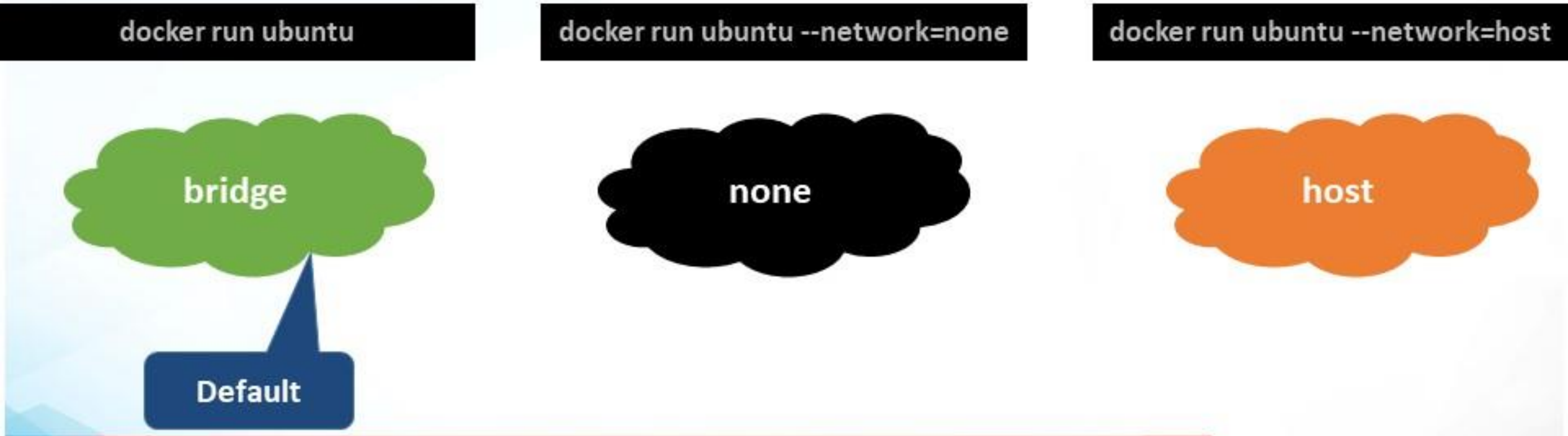
Docker Networking

Default Networks

- When you install Docker it creates three networks automatically
- Bridge is the default network for containers
- To associate the container with any other network, use the network command line parameter

```
docker run ubuntu --network=<network_name>
```



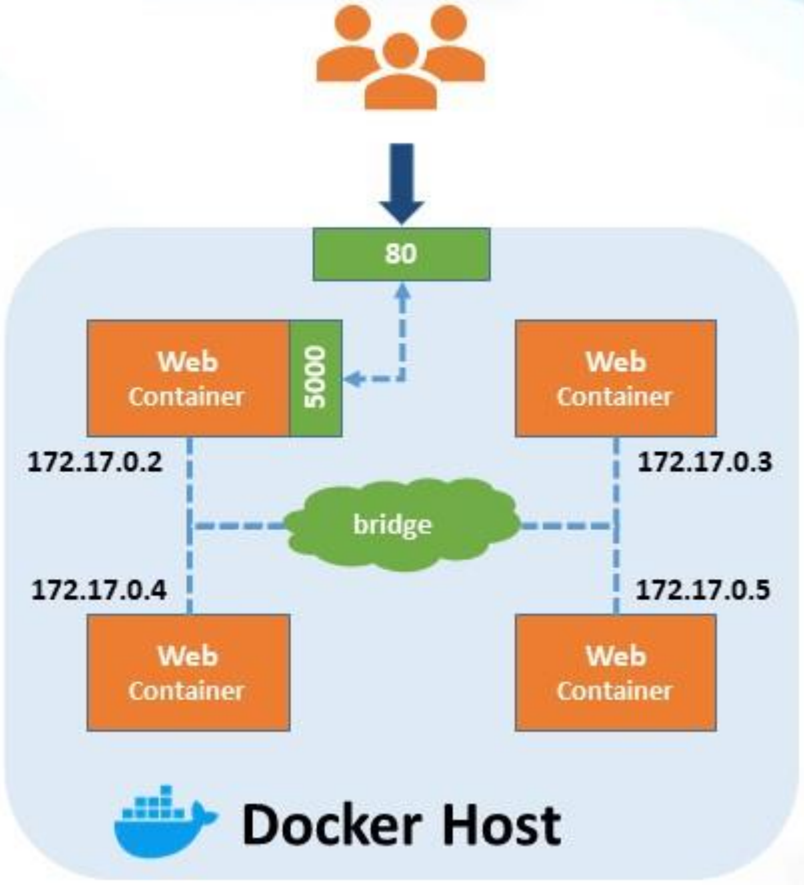
```
docker run ubuntu
```

```
docker run ubuntu --network=none
```

```
docker run ubuntu --network=host
```

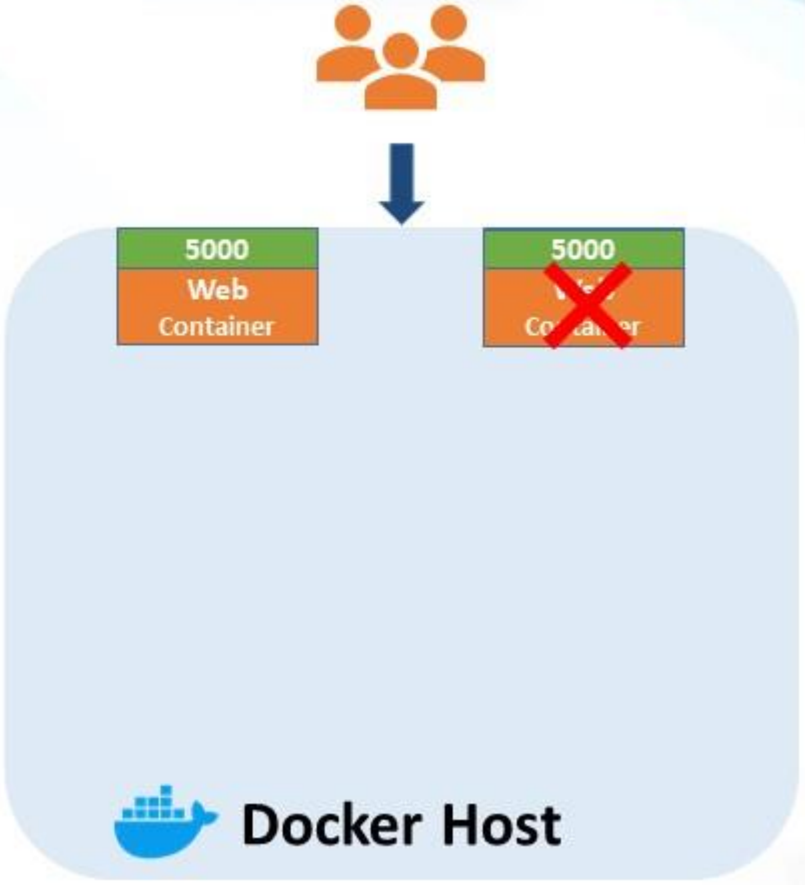
Default Networks - Bridge

- Bridge network is a private internal network created by Docker on the host.
- All containers attached to this network get an internal IP address usually in the range, 172.17.X.X
- Containers can access each other using these internal IP address if required
- To access these containers from outside, we map the ports of these containers to ports on the docker host



Default Networks - Host

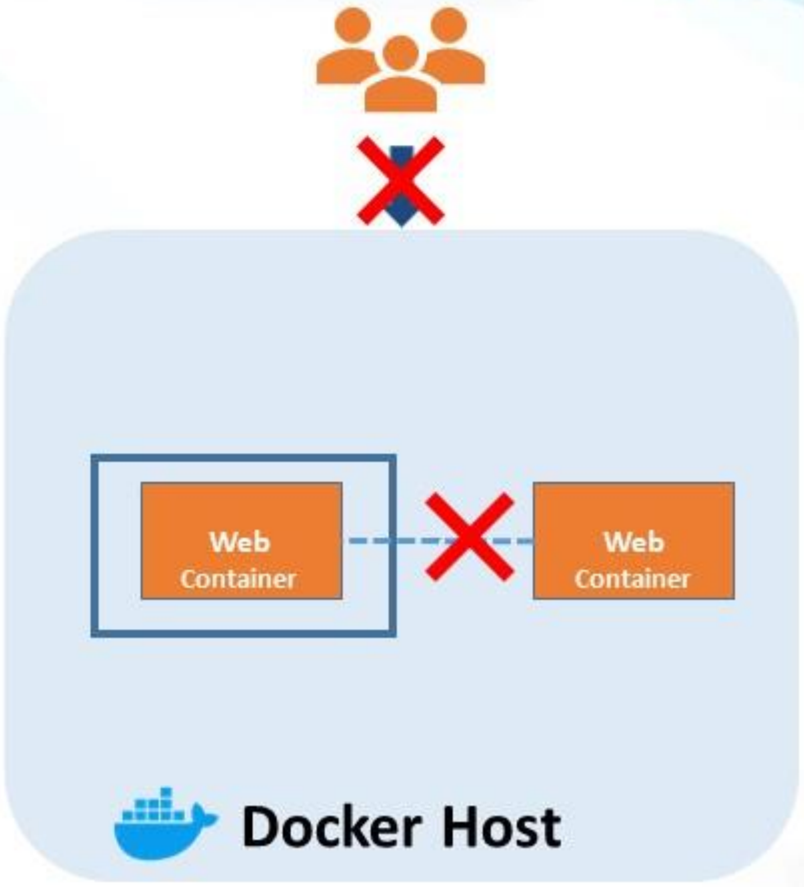
- Another way to access the containers externally is to associate the container to the host network.
- It takes out any network isolation between the docker host and the docker container.
- If you run a container, it is automatically accessible externally on the same port
- With this network, we will not be able to run multiple containers on the same host on the same port



SkillAssure

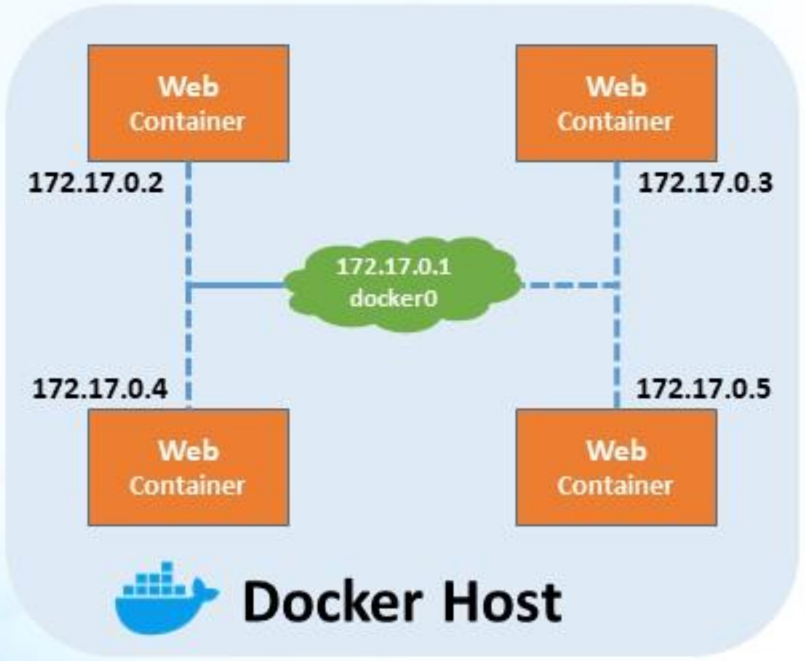
Default Networks - None

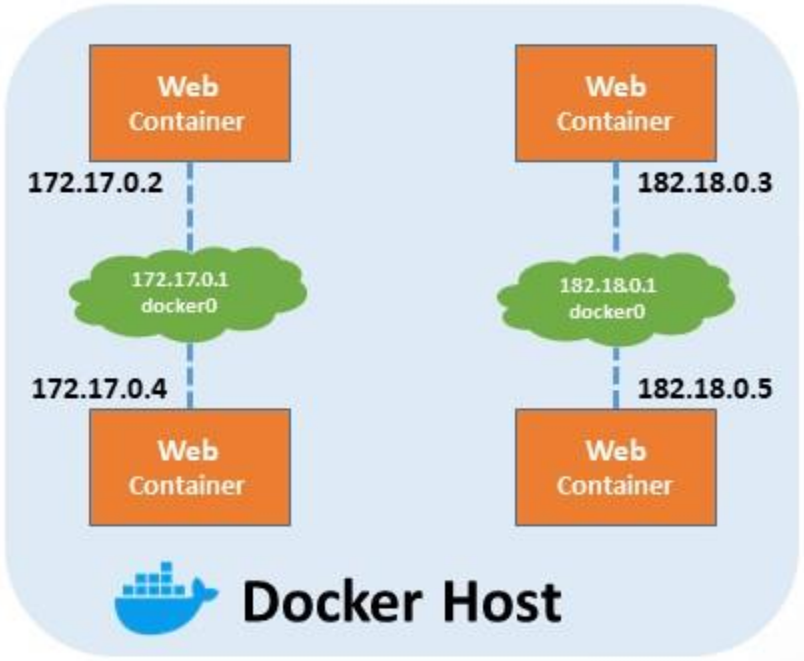
- The containers are not attached to any network
- Containers do not have any access to the external network
- Containers even cannot communicate with other containers
- They run in an isolated network.



User Defined Networks

- By default docker creates only one internal bridge network.
- If we wish to isolate the containers within the docker host, e.g.
    - first 2 containers on internal network 172.X.X.X
    - second 2 containers on a different internal network 182.X.X.X





SkillAssure

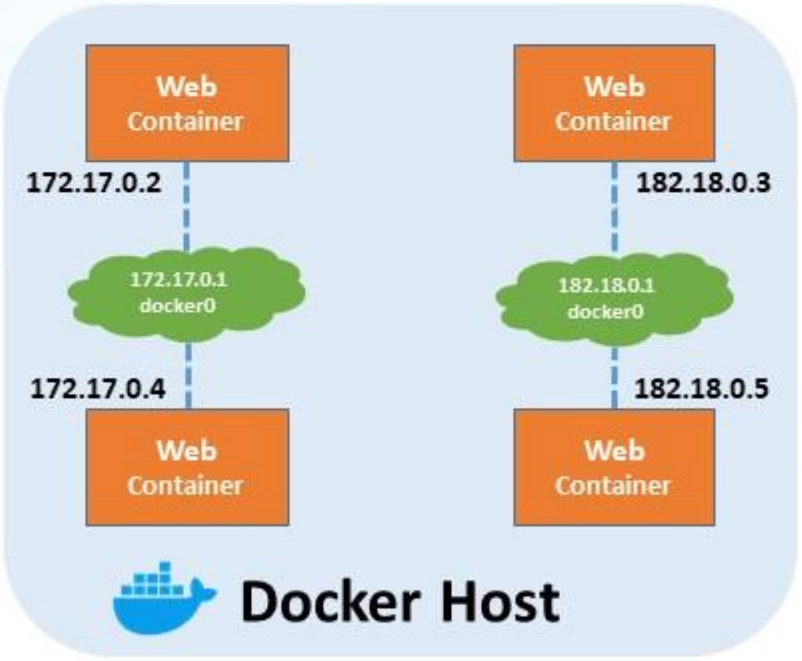
User Defined Networks

- We can create our own internal network

```
docker network create --driver bridge --subnet 182.18.0.0/16 <network_name>
```

- List all networks

```
docker network ls
```



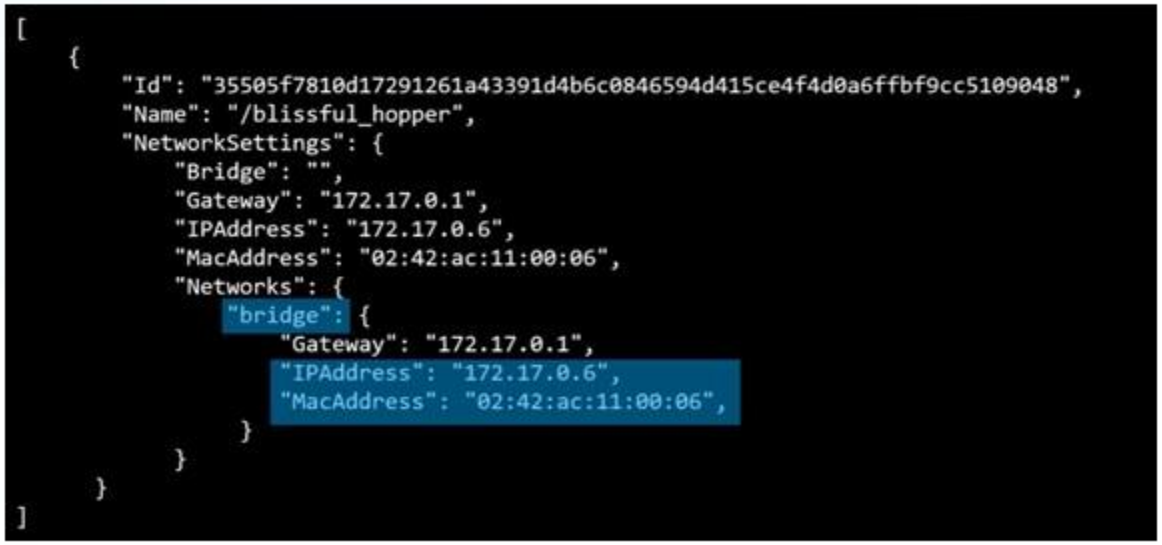
SkillAssure

Inspect Network

- See the network settings and the IP address assigned to an existing container

```
docker inspect <container-name>
```

- Find a section on network settings
- You can see the type of network, internal IP address, mac address, etc.

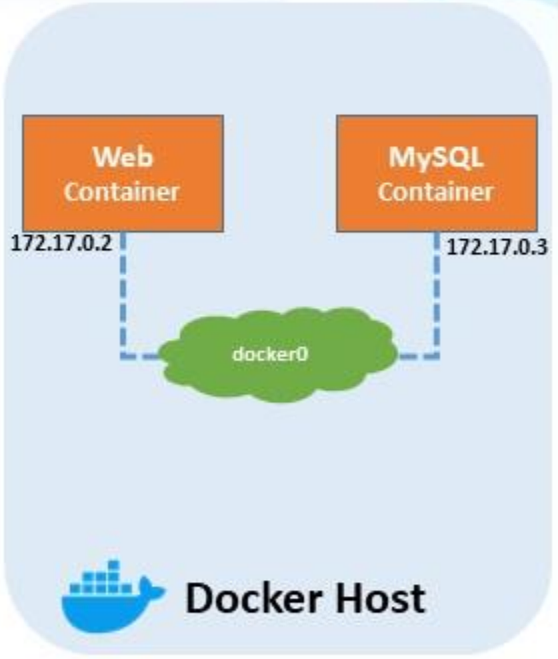


```
[
    {
        "Id": "35505f7810d17291261a43391d4b6c0846594d415ce4f4d0a6ffbf9cc5109048",
        "Name": "/blissful_hopper",
        "NetworkSettings": {
            "Bridge": "",
            "Gateway": "172.17.0.1",
            "IPAddress": "172.17.0.6",
            "MacAddress": "02:42:ac:11:00:06",
            "Networks": {
                "bridge": {
                    "Gateway": "172.17.0.1",
                    "IPAddress": "172.17.0.6",
                    "MacAddress": "02:42:ac:11:00:06",
                }
            }
     }
]
```

SkillAssure

Embedded DNS

- Let’s say you have 2 containers running on the same node
  - Web
  - MySQL



- How can my web server access the database server?

```
mysql.connect(            )
```

- Maybe using container IP Address

```
mysql.connect(  172.17.0.3 )
```

- But container IP may change if system reboots.

- The right way to do it is to use the container name.

```
mysql.connect(  mysql      )
```

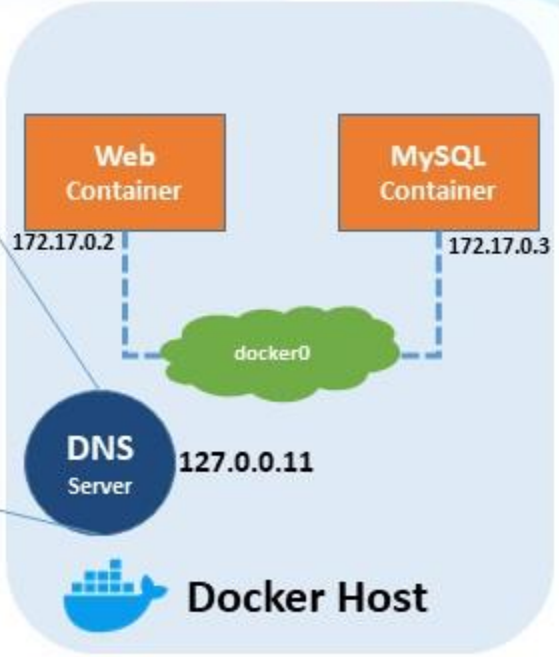
- Containers can reach each other using their names.

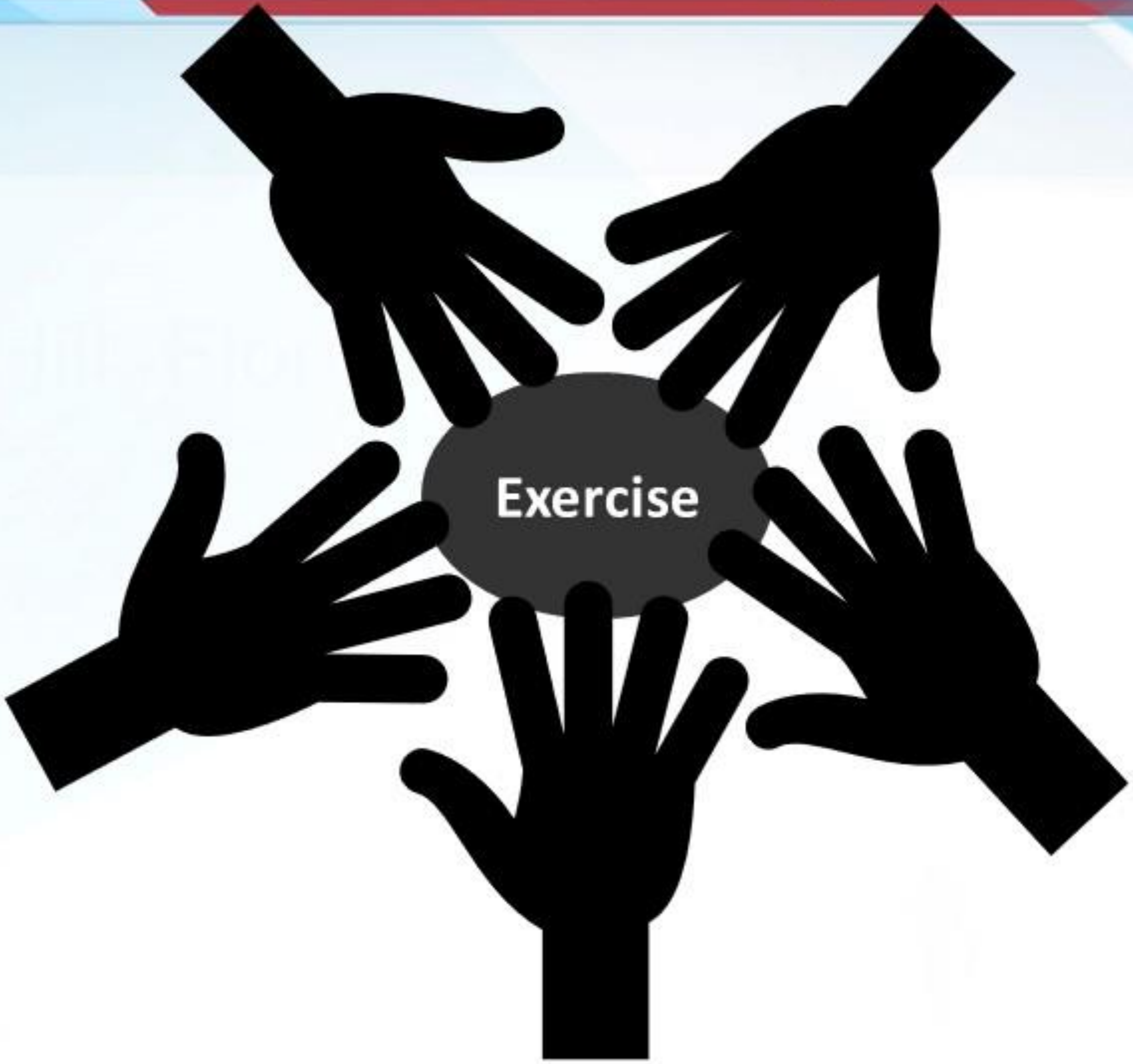
SkillAssure

Embedded DNS

- All containers in a docker host can resolve each other with the name of the container
- Docker has a built-in DNS server.
- It helps the containers to resolve each other using the container name.
- Built-in DNS server always runs at address 127.0.0.11.

Host	IP
web	172.17.0.2
mysql	172.17.0.3




Exercise