KPLABS Course

Docker Certified Associate 2020

Networking

ISSUED BY
Zeal Vora

REPRESENTATIVE
instructors@kplabs.in

Module 1: Overview of Docker Networking

Docker takes care of the networking aspects so that containers can communicate with other containers and also with the Docker Host.

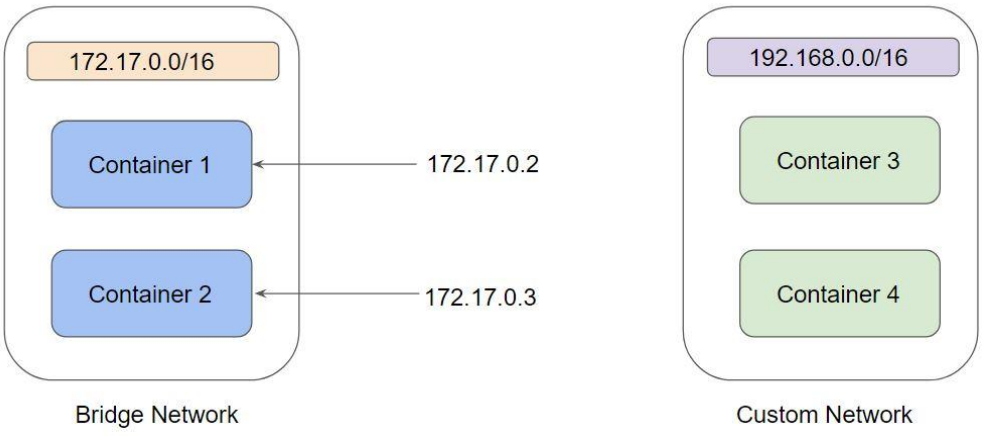
```
[root@docker-demo ~]# ifconfig
docker0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 0.0.0.0
        inet6 fe80::42:b6ff:fea6:f62b  prefixlen 64  scopeid 0x20<link>
        ether 02:42:b6:a6:f6:2b  txqueuelen 0  (Ethernet)
        RX packets 147774  bytes 42989090 (40.9 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 153893  bytes 273579092 (260.9 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 139.59.82.72  netmask 255.255.240.0  broadcast 139.59.95.255
        inet6 fe80::8838:79ff:feed:1335  prefixlen 64  scopeid 0x20<link>
        ether 8a:38:79:ed:13:35  txqueuelen 1000  (Ethernet)
        RX packets 822197  bytes 2191970977 (2.0 GiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 634774  bytes 44815091 (42.7 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Docker networking subsystem is pluggable, using drivers.

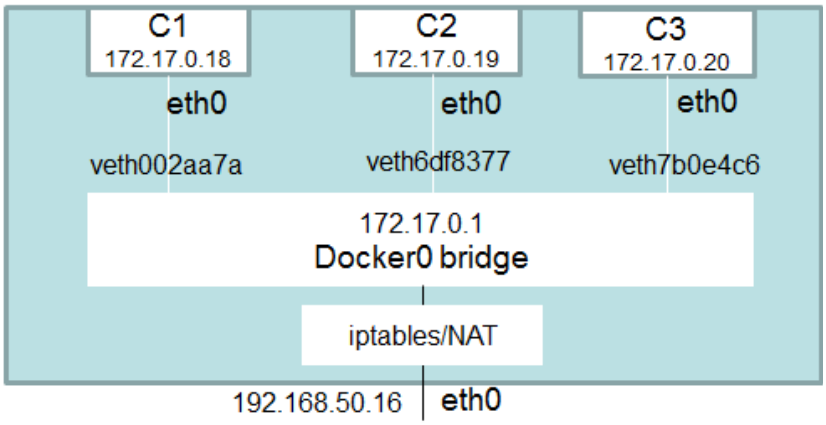
There are several drivers available by default and provides core networking functionality.

- bridge
- host
- overlay
- macvlan
- none



Module 2: Bridge Networks

A bridge network uses a software bridge that allows containers connected to the same bridge network to communicate while providing isolation from containers which are not connected to that bridge network.



Bridge is the default network driver for Docker.

If we do not specify a driver, this is the type of network you are creating.

When you start Docker, a default bridge network (also called a bridge) is created automatically, and newly-started containers connect to it unless otherwise specified.

We also can create a User-Defined Bridge Network which is superior to the default bridge.

Module 3: Host Network

This driver removes the network isolation between the docker host and the docker containers to use the host's networking directly.

For instance, if you run a container that binds to port 80 and you use host networking, the container's application will be available on port 80 on the host's IP address.

Module 4: None Network

If you want to completely disable the networking stack on a container, you can use the none network.

This mode will not configure any IP for the container and doesn't have any access to the external network as well as for other containers.

Module 5: Publishing Exposed Ports of Container

We were discussing an approach to publishing container port to host.

docker container run -dt --name webserver -p 80:80 nginx

This is also referred to as a publish list as it publishes the only a list of the port specified.

There is also a second approach to publish all the exposed ports of the container.

docker container run -dt --name webserver -P nginx

This is also referred to as a publish all.

In this approach, all exposed ports are published to random ports of the host.

Module 6: Legacy Approach for Linking Containers

Before the Docker networks feature, you could use the Docker link feature to allow containers to discover each other and securely transfer information about one container to another container.

The --link flag is a legacy feature of Docker. It may eventually be removed. Unless you absolutely need to continue using it, we recommend that you use user-defined networks to facilitate communication between two containers instead of using --link

Join Our Discord Community

We invite you to join our Discord community, where you can interact with our support team for any course-based technical queries and connect with other students who are doing the same course.

Joining URL:

http://kplabs.in/chat