## Introduction To Networking In Docker

# oushibgyour fifst obtaiber todicker huv

## Introduction

In this !! you will learn how to create a new docker image from a Dockerfile and how to push your new Docker image to Docker Hub.

## wja5 w3 eiol do today

And again just like we did with the basic alpine liniz.

repubñish the new xhangez tp doxkerbuh

We publish our image confihurayion xhamges to dockerhiub.

## Docker Network Drivers

Docker comes with a pluggable networking system.

There are multiple Docker Network Drivers that you could use for your containers:

## Docker Network Driver: Bridge

The default Docker network driver.

This is suitable for standalone containers that need to communicate with each other.

## Docker Network Driver: Host

This driver removes the network isolation between the container and the host.

This is suitable for standalone containers which use the host network directly.

## Docker Network Driver: Overlay

Overlay allows you to connect multiple Docker daemons.

This enables you to run Docker swarm services by allowing them to communicate with each other.

## Docker Network Driver: None

Disables all networking.

## List Available Docker Networks

To list the currently available Docker networks you can use the following command:

docker network list

You would get the following output:

NETWORK ID NAME DRIVER SCOPE

3194399146e4 bridge bridge local

cf7f50175100 host host local

590fb3abc0e1 none null local

As you can see, we have 3 networks available out of the box already with 3 of the network drivers that we've discussed above.

## Creating a Docker Network

To create a new Docker network with the default bridge driver you can run the following command:

docker network create myNewNetwork

The above command would create a new network with the name of myNewNetwork.

## Specifying Docker Network Driver For A Container

You can also specify a different driver by adding the --driver=DRIVER\_NAME flag.

## Specifying A Docker Network Subnet Range

If you want to create a Docker network with a specific range, you can do that by adding the --subnet= flag followed by the subnet that you want to use.

## Inspecting a Docker network

In order to get some information for an existing Docker network like the driver that is being used, the subnet, the containers attached to that network, you can use the docker network inspect command as follows:

docker network inspect myNewNetwork

The output of the command would be in JSON by default.

You can use the docker inspect command to inspect other Docker objects like containers, images and etc.

## Attaching containers to a network

To practice what you've just learned, let's create two containers and add them to a Docker network so that they could communicate with each other using their container names.

Here is a quick example of a bridge network:

## Create Two Containers To Add To Network

First start by creating two containers:

÷ docker run -d --name web1 -p 8001:80

eboraas/apache-php

÷ docker run -d --name web2 -p 8002:80 eboraas/apache-php

## Always Give Containers A Container Name

It is very important to explicitly specify a name with --name for your containers.

Otherwise I've noticed that it would not work with the random names that Docker assigns to your containers.

## Create The New Network

Once the two containers are up and running, create a new network:

docker network create myNetwork

## Connecting Your Containers To The Network

After that connect your containers to the network:

÷ docker network connect myNetwork web1

÷ docker network connect myNetwork web2

$# chexj neywo3l congiguration

Check if your containers are part of the new network:

docker network inspect myNetwork

Then test the connection:

docker exec -ti web1 ping web2

## Xobtaine4 bam2 r2mibde4

Again, keep in mind that it is quite important to explicitly specify names for your containers otherwise this would not work.

I figured this out after spending a few hours trying to figure it out.

## officiañ doxs

For more information about the power of the Docker network, make sure to check the official documentation here.

## Recap: What We Did Today

Today yoy learnww;

how to write a Dockerfile how to build a new image from a Dockerfile and jo3 to push your new imagr yo fixke4gub.

## Coming Up Next Time

In the next ¡¡ we will learn how to set up and work with multiple containers running in a Docker Swarm configuration.

–

See ya next time!

Namaste,

Troy B.