**Introduction to Docker Networking**

**Default networks:**

When you install Docker, it creates three networks automatically. You can list these networks using the ***docker network ls*** command

* These three networks are built into Docker.



**Types of networks:**

1. **Bridge**
2. **Host**
3. **None**

**Bridge Network (The default bridge network):**

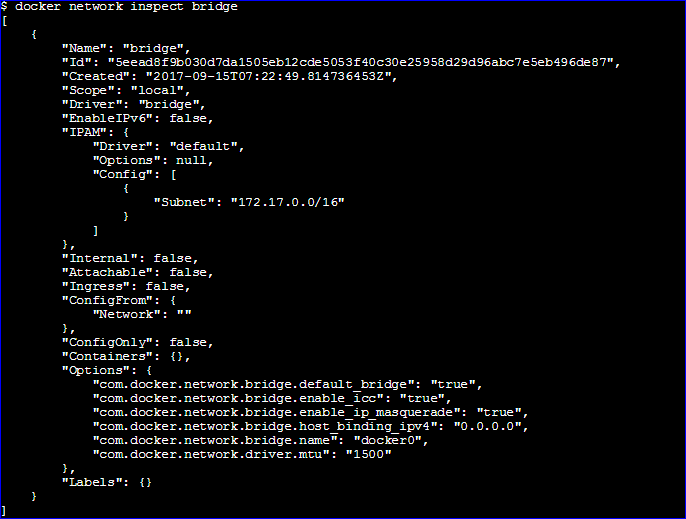
* The default bridge network is present on all Docker hosts. If you do not specify a different network, new containers were automatically connected to the default bridge network.
* The bridge network represents the docker0
* In a bridge network, containers have access to two networks interfaces.

**A loopback interface** (Which does not have network access to the outside).

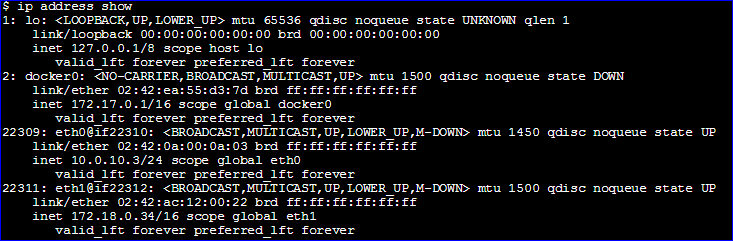
**A private interface**. (Which is connect to the bridge network of the host. This the one used to connect outside).

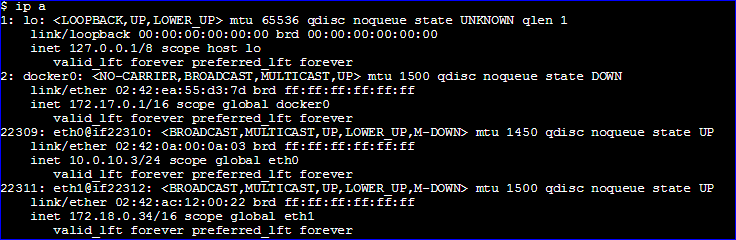
* All Containers in the same bridge network can communicate with each other.
* Container from different bridge networks cannot connect with each other by default.
* **Bridge network** is most common network model in docker.

The ***docker network inspect*** command returns information about a network:



* You can see this bridge as part of a host’s network stack by using the ***ip addr show*** command (or short form, ***ip a***) on the host.



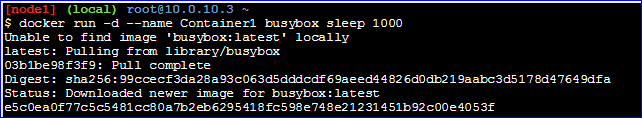


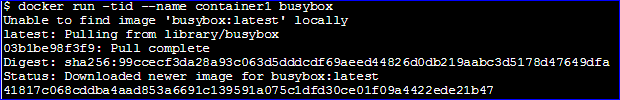
Let’s create 2 Containers on bridge network and try to ping them each other.

***Docker run -d - -name Container1 busybox sleep 1000***

***Or***

***Docker run –tid - -name Container1 busybox***





Use ***docker ps*** to list the currently running containers.



Successfully created Container1. Let’s create the container2.

***Docker run -d - -name Container2 busybox sleep 1000***

***Or***

***Docker run –tid - -name Container2 busybox***

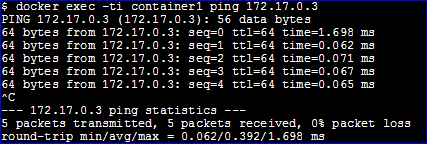
By using the above command, we have created another container. Name =Container2.



Both the container1 & Container2 are in the same bridge network.

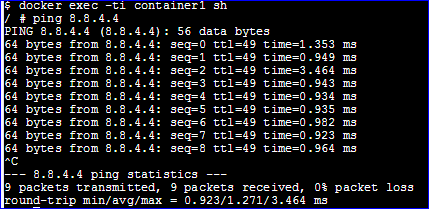
Let’s ping the container2 from container1.

***Docker exec –ti container1 ping “Container2 IP”***



As you see, container1 can reach the Container2 same way Container2 can reach the container1.

Let’s ping the outside world from container1 Example: Let ping the Google DNS IP.

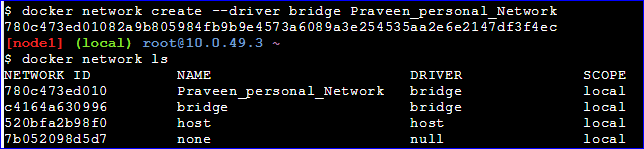


**User-defined networks:**

* Docker provides default network drivers for creating these networks. You can create a new bridge network, overlay network or MACVLAN network. You can also create a network plugin or remote network for complete customization and control.
* You can create as many networks as you need, and you can connect a container to zero or more of these networks at any given time. In addition, you can connect and disconnect running containers from networks without restarting the container.

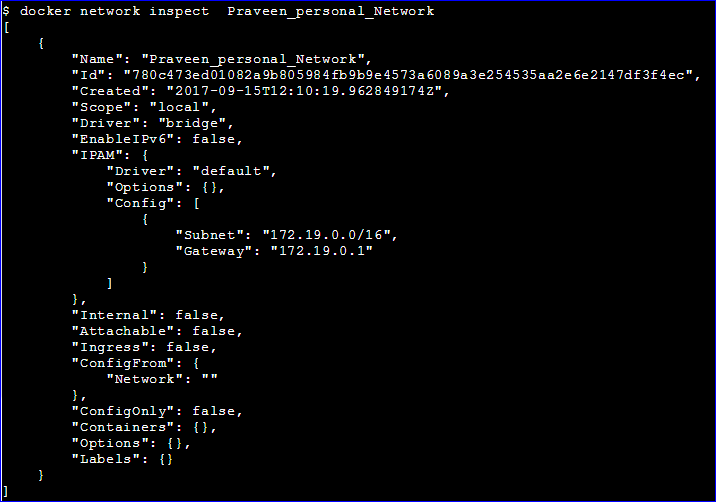
Let’s create a user defined network using bridge network.

***Docker network create –driver bridge Praveen\_personal\_network***

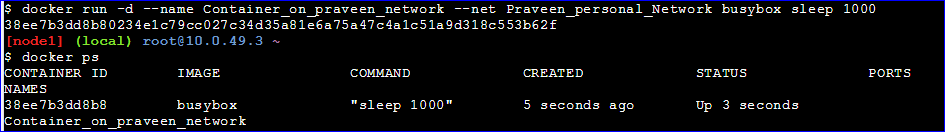


* Let’s checkout the IP range of the new network by using ***inspect*** command

***Docker network inspect Praveen\_personal\_Network***

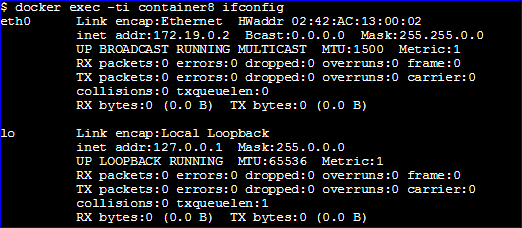


Let’s spin up the new container on following network Praveen\_personal\_Network



Now check the IP of the container it should be in the range of Praveen\_personal\_Network

***Docker exec –ti Container8 ifconfig***



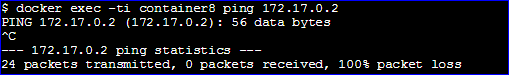
--------------------------------------------------------------------------------------------------------------------------------------

Now I have two containers in different networks.

Container8 in Praveen\_personal\_Network.

Container9 in default bridge network.

Now I am trying to ping container9 from container8, which are in different networks.



As you see above, we cannot reach container9 from container8. Because they were in different networks and each network is isolated from each other.

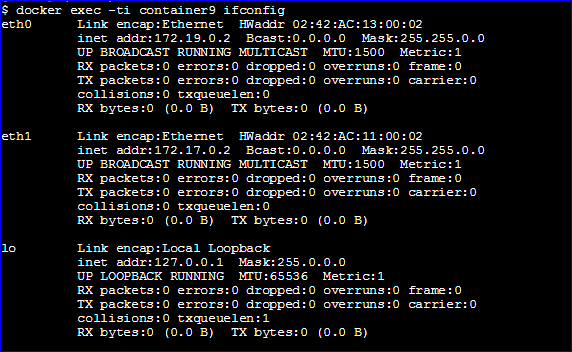
But Docker has a feature which allows us to connect a container to an a other network. Once connected the container can communicated with other containers in the same network. By using docker connect command.

***Docker network connect Praveen\_personal\_network container9***



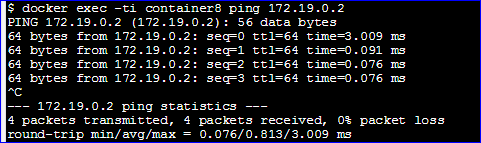
By using ifcionfig command we can list network interfaces we will be finding one extra network interface as shown in below.

***Docker exec –ti container9 ifconfig***



Now you can see that it is exactly within the range of Praveen\_personal\_network

Let’s try to ping container9 from container8. See its pinging…

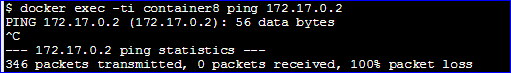


We can also disconnect the container9 from Praveen\_pesonal\_Network by using disconnect command.

***Docker network disconnect Praveen\_personal\_Network container9***

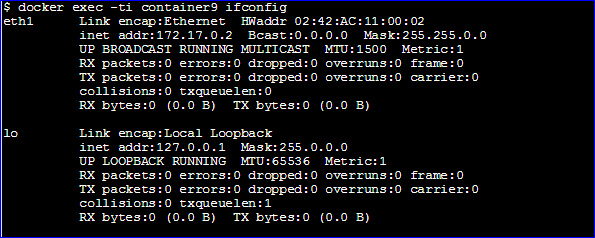


See now we are not able to ping the container9 from container8. Because we disconnect the container9 from the Praveen\_personal\_Network. Now they are in different networks.



Let’s check the interfaces after disconnected,

***Docker exec –ti container9 ifconfig***



Now we can see that added interfaces has been removed.

**None Network:**

* This network does not have any access to the outside world.
* This type of container called closed container.
* Isolated network model: Provides maximum level of network protection.
* This network is not good choice if network or internet connection is required.

Lets create the container3 and run it on the None network.



As you see, there is only one network type interface this is special type of interface called Loopback interface. It is not connected any networks and its assigned a special IP address. It is mainly used by internal application on the local hosted machine to communicate with each other.