Networking in Docker

When we say docker container run --publish 90:90 , there is lot of extra work which is done in background.

1.Each container is connected to a virtual network called as “bridge network”

2.each virtual network routes through NAT firewall on host IP

3.all containers run on virtual network and can communicate with each other with out -P

For example

If we have 2 containers in one container PHP/apache and in other container we have Node js

And both the containers are on same network , then both the containers can communicate with each other.

We can make our own virtual network, depending on security requirements.

1. **Docker container port [container\_id | container\_name]:**

Above command will return the exposed ports of docker container. And port of a host from which it is listening to

1. **Docker container ip --format ‘{{.NetworkSettings.IPAddress}}’ [container\_name | container\_id]**

Above command will return the virtual IP address of the container

1. **Docker network ls** --- this shows all of the network created

1. **Docker network inspect [network name]** - returns the configuration of network in JSON format, the IPAM tag displays the IP address and subnet mask of the virtual network assigned to containers on that network

**Bridge network is the network specially designed to skip the virtual networking of docker and attaches the container directly to host interface.**

1. **Docker network create [network\_name]:** this will create a new network with driver of bridge which by default driver.
2. **Docker network -- help :** will give all options we can specify while creating a network.
3. **Docker container run --name [container\_name] -d --network [network\_name] image name**

This will create a new container on out newly created virtual network , just need to specify --network option in our command.

1. **Docker network connect [network name] [container name] --** This command will connect and existing container to a mentioned network
2. **Docker network disconnect [network name] [container name] --** This command will disconnect and existing container to a mentioned network

Docker DNS naming:

Docker daemon has inbuilt DNS server which container uses by default

Just consider their are two containers on same network , and as we cant rely on IP as IP address changes frequently there might be possibility we loose the container, hence container on same network identify each other by their name.

So if we execute the ping command from our container1 to container 2 it should give us a response

Remember to use exec command to execute any command in the container from above .

**Docker container exec -it [container\_name | container\_id] ping [container\_name | container\_id]**

Above command will execute the ping command inside container one to connect to container 2