**run --name mongo -d mongo**

run the mongo container if it is not present it will download from internet

**docker container ls –a**

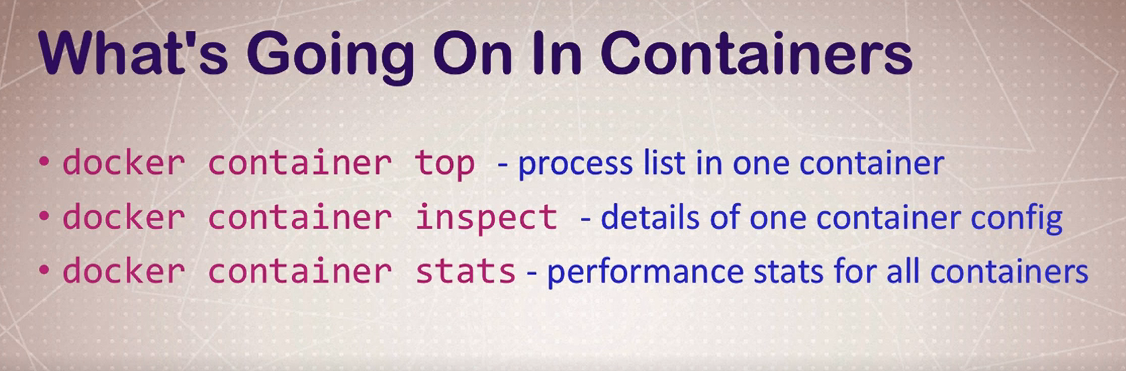
list all the docker container present

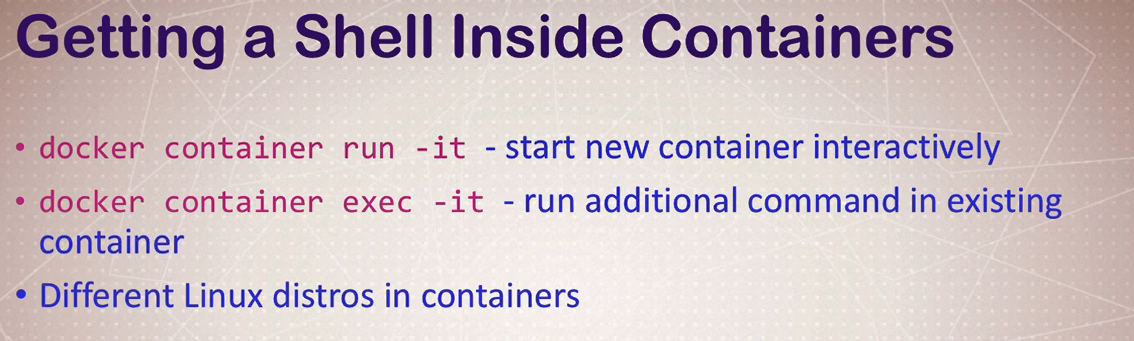
**docker container top**

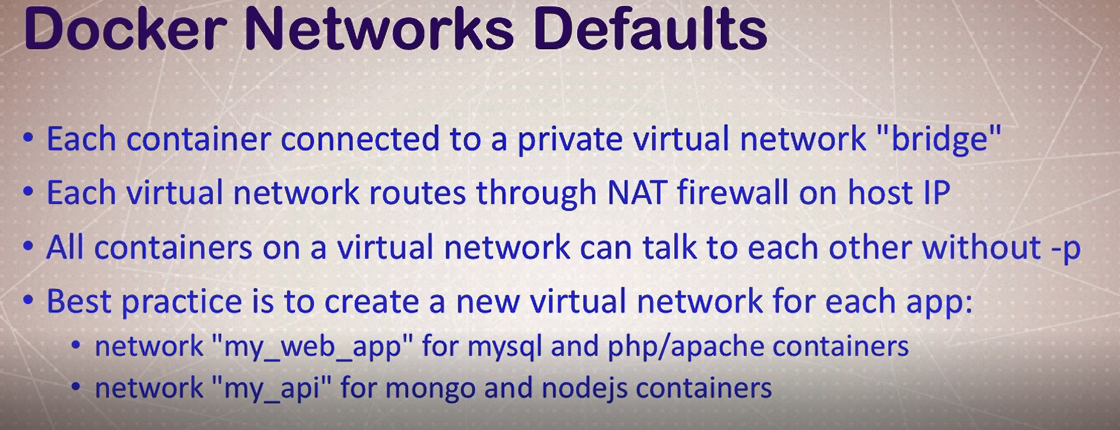
list all the container currently running

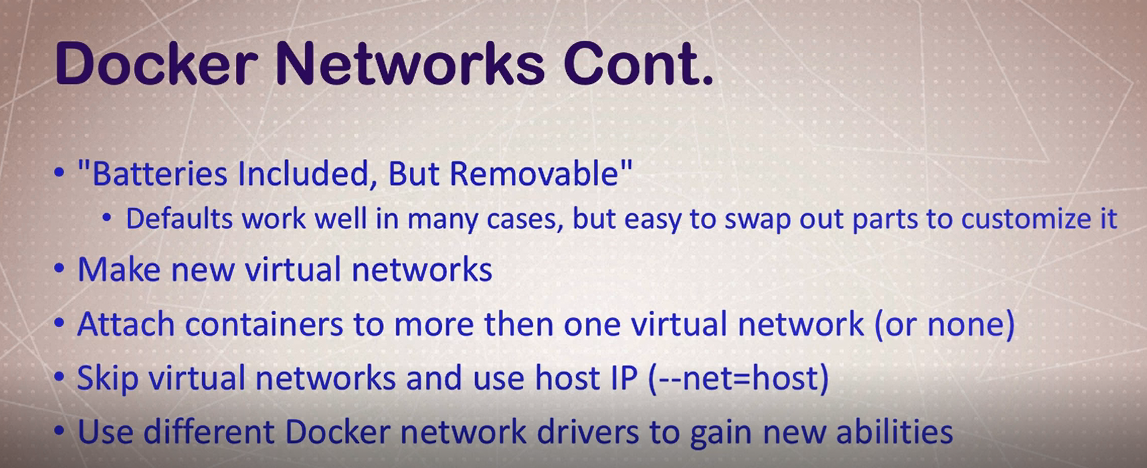
**docker container rm -f 1fcc**

to stop a specific container f1fcc is unique name of container









**docker container run -p 80:80 --name webhost -d nginx**

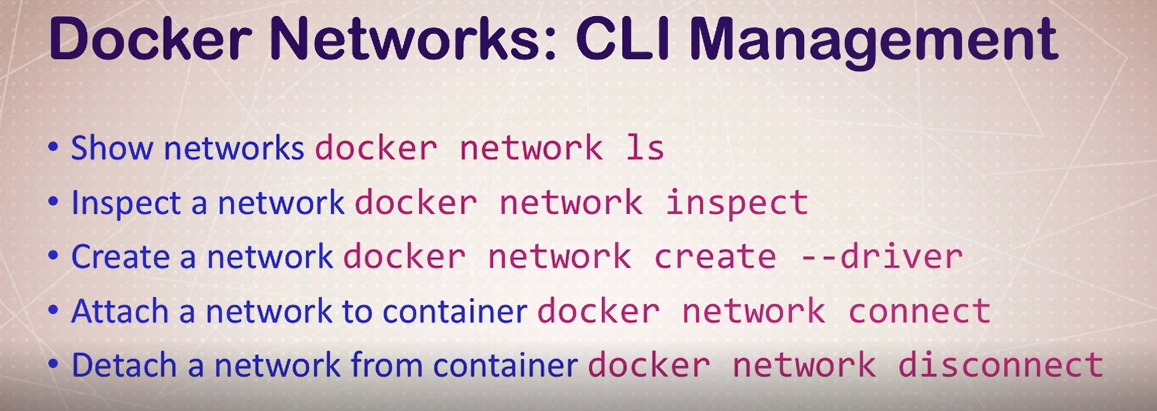
where –p is publish

**docker container port webhost**

o- 80/tcp -> 0.0.0.0:80

**docker container inspect --format '{{.NetworkSettings.IPAddress }}' webhost**

to get the ipaddress of container



**docker container run -d --name new\_nginx --network my\_app\_net nginx**

**docker container run --rm -it ubuntu:14.04 bash**

**docker container run --rm -it centos:7 bash**

**yum update curl**

for centos

apt-get update && apt-get install –y curl

**curl –version**

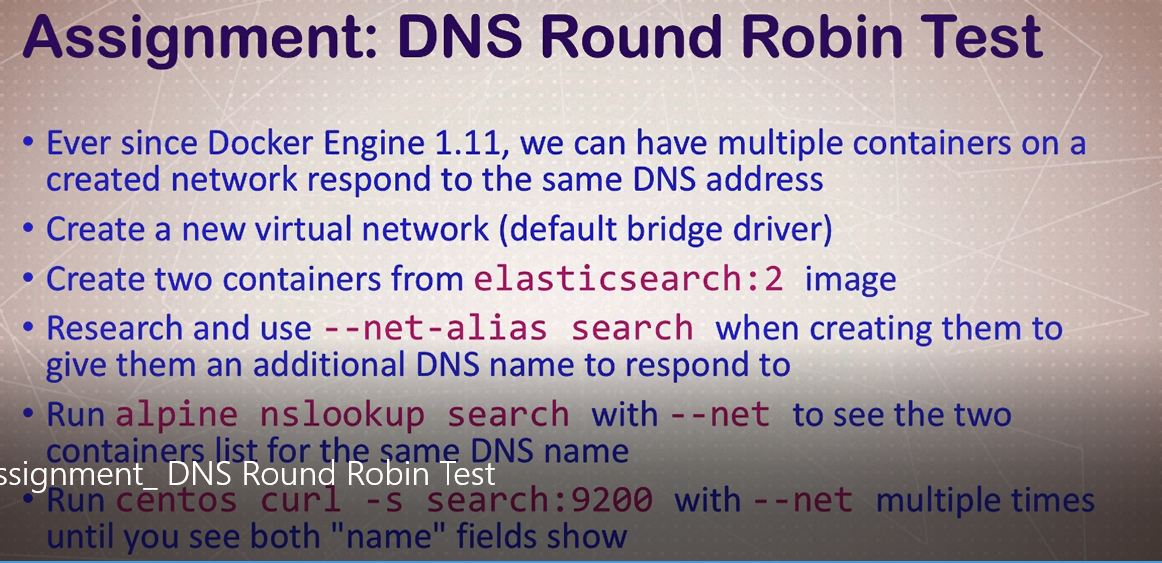
to get the current version of curl

**on exit command the container is stopped**

**DNS Round robin test**

It is a concept of having two different host with dns aliasis that respond to same dns name

It is used by tech giant to make their server always on



**Docker network commands**

**docker network ls**

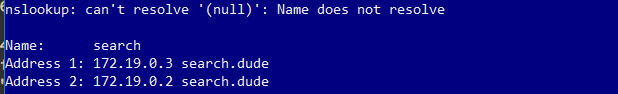
shows all the available virtual network

**docker network create dude**

create a network hub

**docker container run -d --net dude --net-alias search elasticsearch:2**

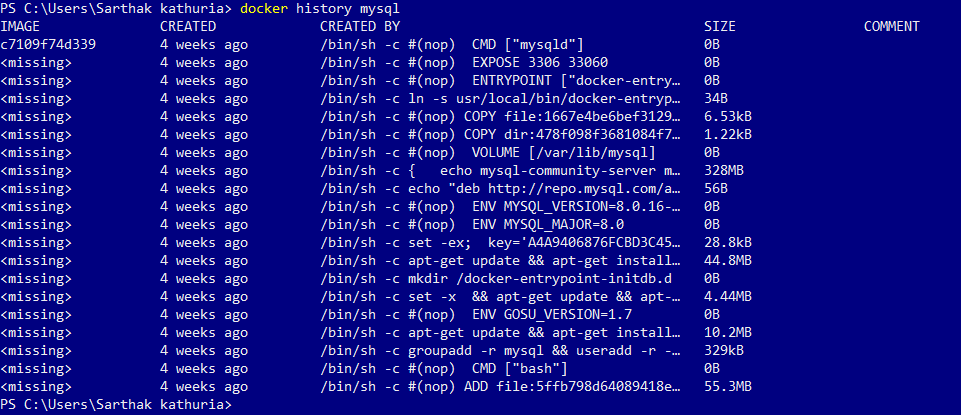
**docker container run --rm --net dude alpine nslookup search**



**docker container run --rm --net dude centos curl -s search:9200**

docker history nginx

it gives the history of image layer



Containers are formed on top of image and there can be multiple container on a single image

Image is formed by combining all the required features layer by layer like for Ubuntu image we can include Ubuntu-apt-env variables all are clustered to form a single image

Main benefit of docker is we are never storing the same image data more than once with the help of cache memory

**docker image build -t nginx-with-html .**

**container run -p 80:80 --rm nginx-with-html**

**FROM node:6-alpine**

**EXPOSE 3000**

**RUN apk add --update tini**

**RUN mkdir -p /usr/src/app**

**WORKDIR /usr/src/app**

**COPY package.json package.json**

**RUN npm install && npm cache clean**

**COPY ..**

**CMD ["tini","--","node","./bin/www"]**

**docker build -t testnode .**

**docker tag testnode sarthakkathuria/testing-node**

to change the tag

**docker push sarthakkathuria/testing-node**