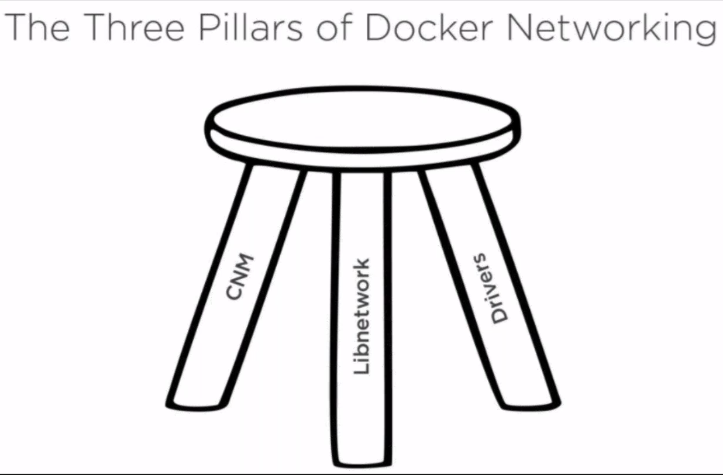
**Networking:**

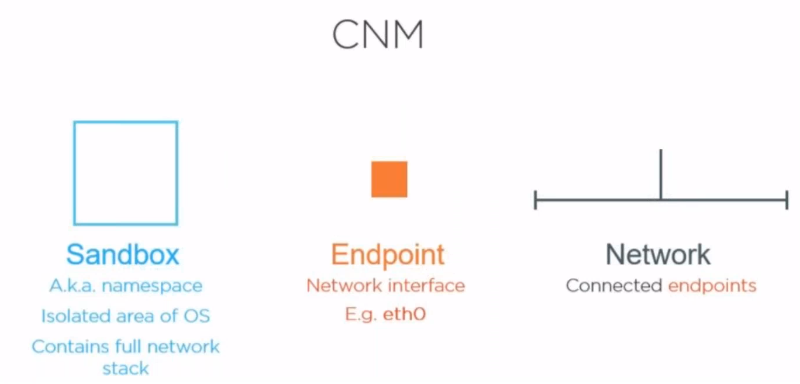
* Docker was not that great in networking from starting
* Network was part of docker engine
* Now, it is separate component of engine



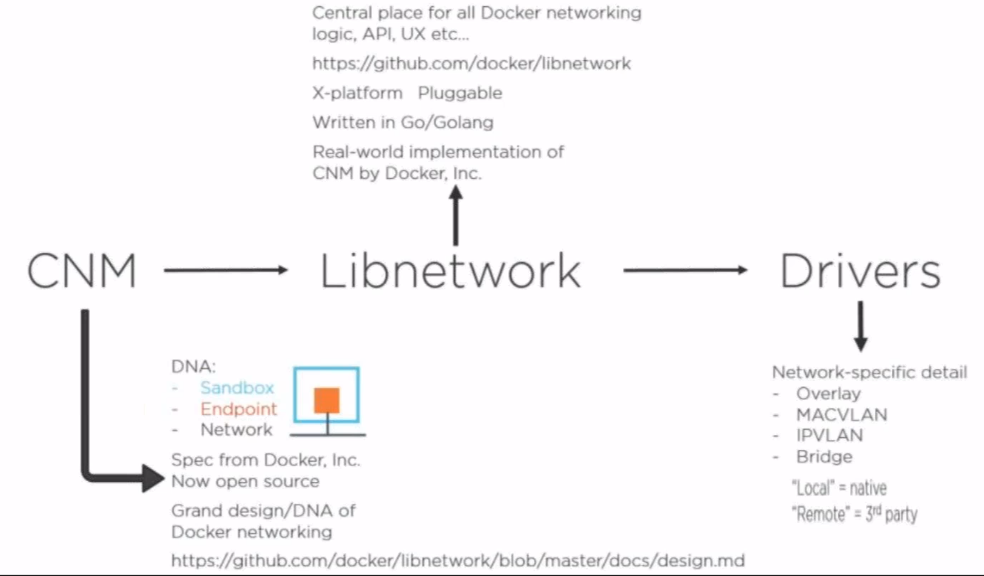
* CNM (container network model)
* CNM is basical designed document, it is required for containers networking. Container model is visible here. It is open and available at github/docker
* Libnetwork is based out of the designed by CNM
* CNM is designed document, all the docker network is based on this
* Libnetwork is implementation of CNM. Just like CNM is document and libnetwork is application of that
* Drivers is the extreme frontend where we can plugin. There are certain drives provided by docker
* If we want, we can write own drivers
* To supports multiple networks, they have written a component called as drivers



* CNI is more matured than CNM



* In CNM, these are major components
* Endpoint is nothing but the drivers in system. In windows, we see network adapters
* Every driver has its own way of connection
* Sandbox is about the namespaces which gives isolated networking for container. It contains full network stack implementation like tcp, http etc.
* Sandbox is what gets into OS, and the result would be endpoint and we can connect other endpoints is called network
* Every container has endpoint. When we connect or communicate with all those, we form network





Libnetwork is implemented in golang like docker. This is the language from google where we can run executables, we no need to install any software

* **Docker network ls 🡪 to check the list**

If we check the inspect command, we can see the below details





* Bridge driver is default driver
* For example, if we want to create a network where we can ping another container, we can do that

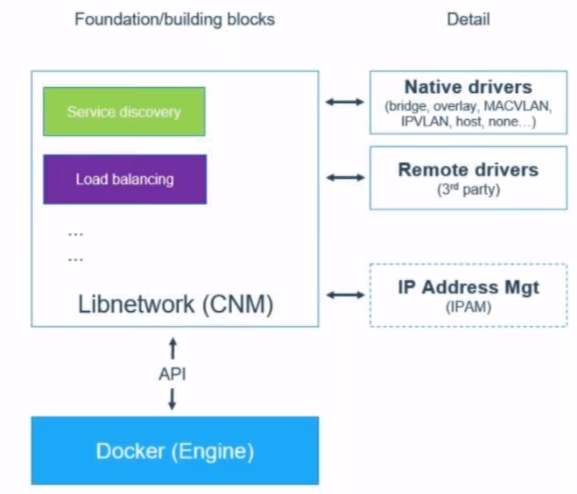
With the default driver, from one container, we can ping the other container with its id not with its name

* **Ping <container ip address>**

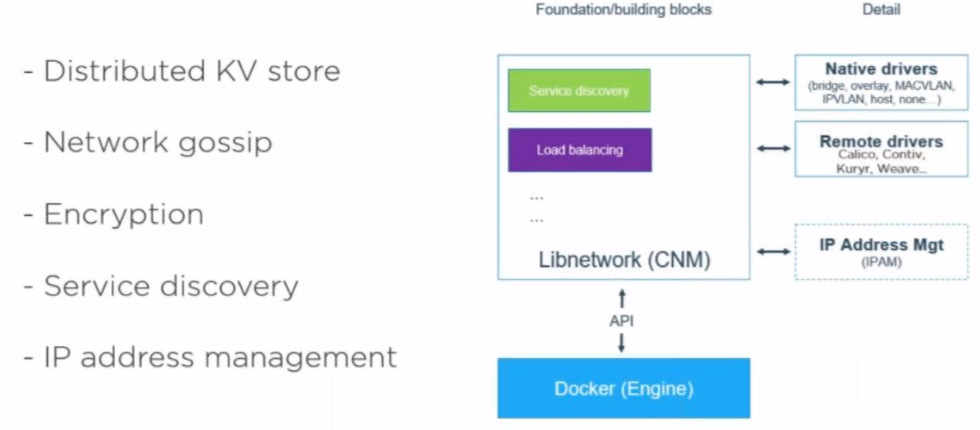
With service discovery, we can ping other containers with its name, but with the default network, we can’t do it

In real time, after restarting the server, ip address may get changes, so it is good to have the connection with the name not with the ip

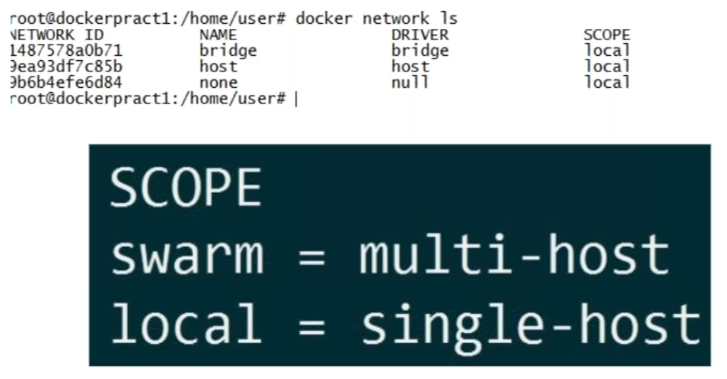
* Bridged and overlay networking are important
* Docker has separated the network component which is libnetwork
* It has some native and remote drivers. If we know golang, we can write the network



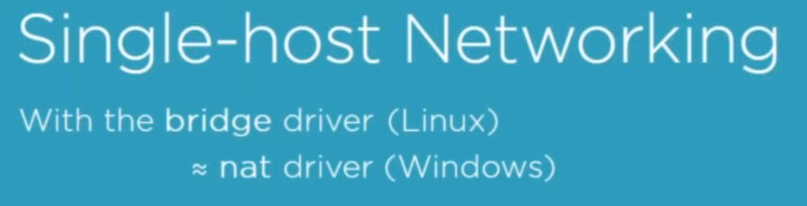
* Iibnetwork comes with service discovery and load balancer. It has many services, out of that, these two are important. It also has ip management system
* Suppose we increased our app server, we no need to have load balancer, libnetwork has load balancer already
* If we want to use libnetwork load balancer, we use docker swarm which is equals to kubernetes
* Drivers are not part of the libnetwork, they plugin to make the stuff workout

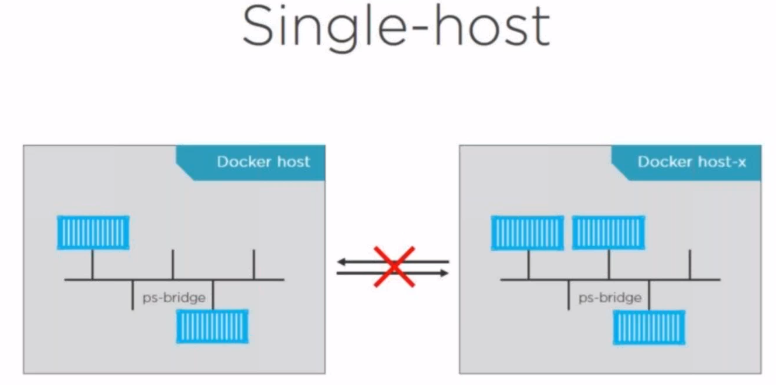


* They store keys and values in that
* On above image, last 3 are important
* For example, if we want to broadcast something to other thing, docker also supports something called as whisper protocol, it sends the info to nearest nodes and those things sends to others, this is how gossip protocol works

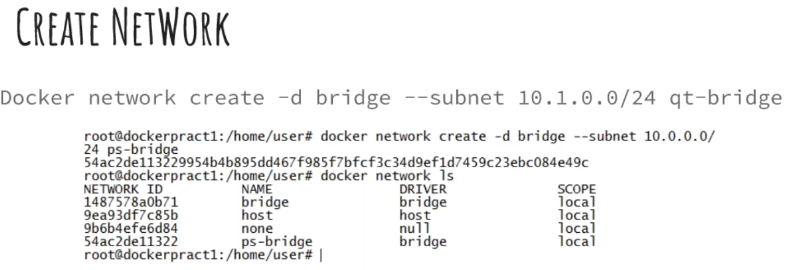


* Whenever we see the scope as local, that is speaking of single machine network, network between our machine and container
* Scope as swarm, means the speaking of containers in multiple machines
* If a container on one machine needs to communicate with the container of another machine, we use swarm for that





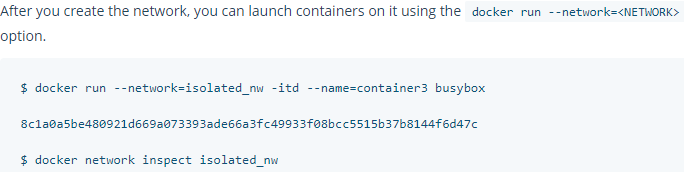
* Which can communicate multiple containers that are launched within the system



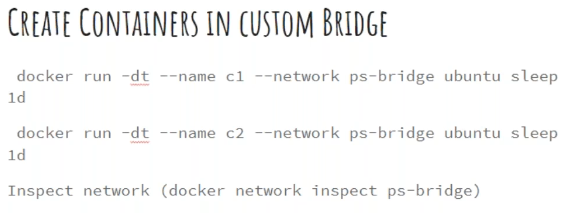
* -d means driver, qt-bridge is the name
* Subnet is what is our network size
* Private subnet should start either 10 or 192 or 171 or 169. These are meant for the private ip addresses



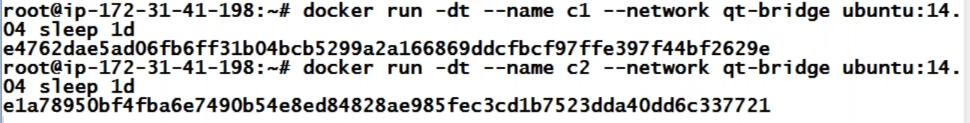
* Above is the error we get when we try to create network with same ip range



* Machines which are launched by default bridge network will not be able to communicate with my bridge



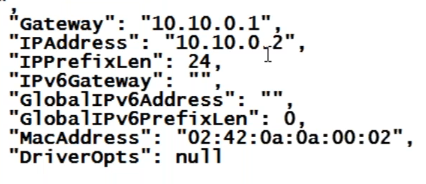
* With below, we can run docker container with the network which we have created
* If we don’t give - -network, it will go to default network



* Sleep is a command in docker which will not do anything, but will show it as running process
* C1 is the name which we gave, if we don’t give, docker will give
* This we never do on production

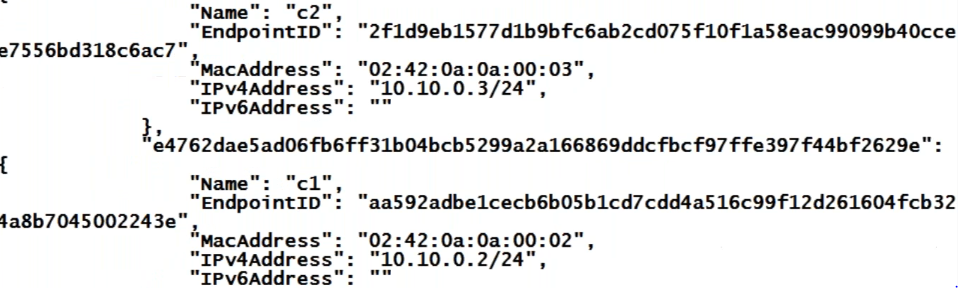
We can check with below command

* **Docker inspect c1**



* **Docker network inspect qt-bridge**

This will also tell what the containers are running on the network. The ip addresses will be assigned as per our range



Get into any one of the machines

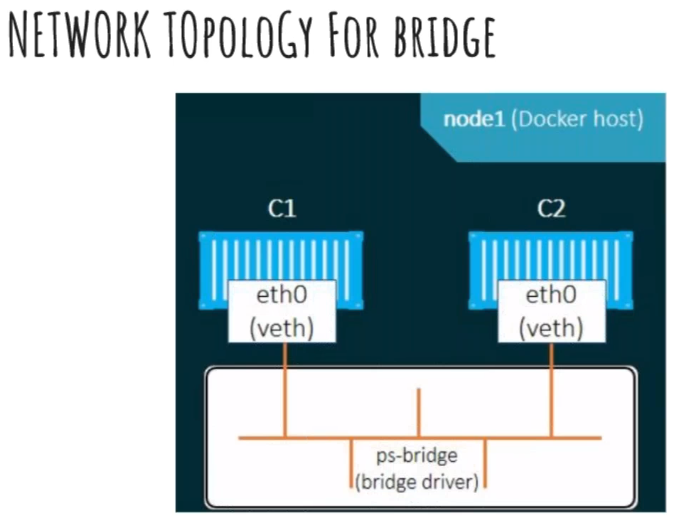
* **Docker exec -it c1 /bin/bash**

And try to ping with name, it will work

* **Ping c2**

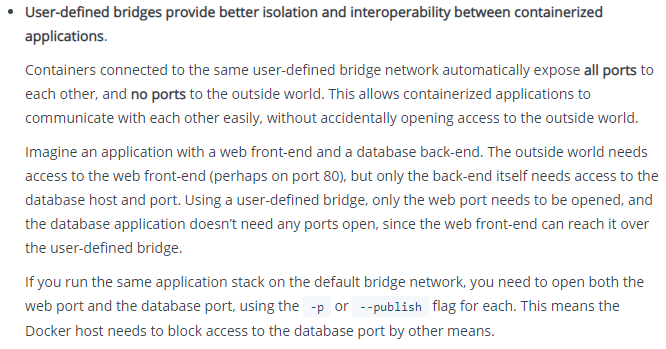
We have to create our own bridge network to do this, we can’t do it with default network

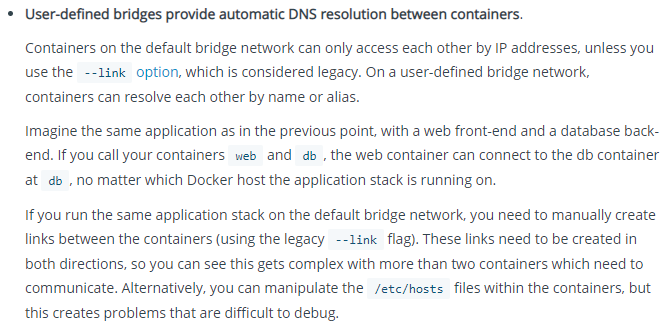
* Below is what we did

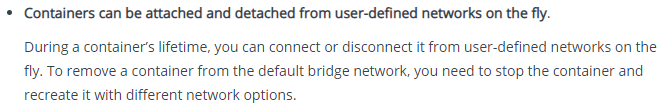


* Here sandbox is our libnetwork. Endpoints are eth0 on c1 and c2 container

**Difference between user defined and default bridge:**



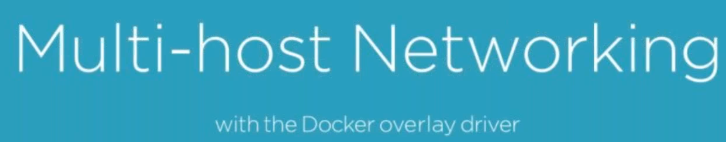


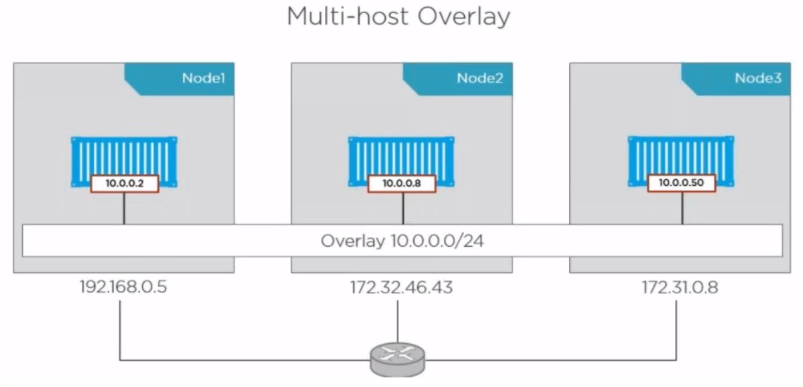




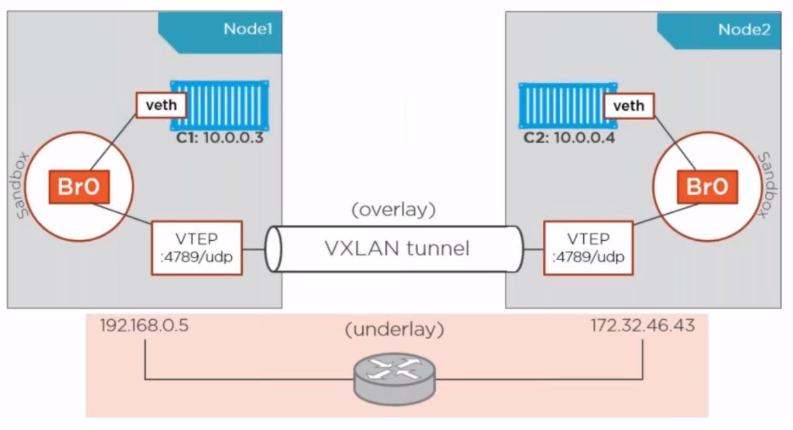
**Multi-host networking:**

* As of now, we have done the network with in one machine
* For the network between multiple hosts, we use overlay driver





* For this, we have to create overlay network
* To do this, the machined should be reachable to each other
* Overlay is logical communication, but the actual communication happened with underlay
* Logical it shows communication happens with overlay but actually it happens with underlay



* On sandbox, we create containers which connected to a same
* Underlay is physical communication

