Docker Hands-on:4

Here we will discuss the Docker networking
What is docker networking?
All docker container is having the different IP address

```
root@ip-172-31-8-163:~# docker inspect cb6e0f7db160 | grep -i ipaddr
            "SecondaryIPAddresses": null,
            "IPAddress": "172.17.0.3",
                    "IPAddress": "172.17.0.3",
root@ip-172-31-8-163:~# docker inspect 3b136c0f8937 | grep -i ipaddr
            "SecondaryIPAddresses": null,
            "IPAddress": "172.17.0.2",
                    "IPAddress": "172.17.0.2",
root@ip-172-31-8-163:~#
```

If two or more docker container wants to communicate each other then in this picture docker networking coming in the picture.

Every system get assign the localhost ipaddress as 127.0.0.1

So,by the command “curl localhost:container_portno” or “curl localhost_ip:container_portno.” by that we can communicate.

For communicating more then 2 server or system we generally use the devices called
Router ->Physical networking device it is helping us to all the devices and also unknown thing that is internet
Switch/hub->If we want to use the private network i.e called lAN ans its coming with 100 port number
These switches/hub/router is having their CIDR value

How could docker container assigned the IP addresses
Docker services which is running internally will assign the IP to the container
Systemctl status docker.service

Docker is having it’s own internal network configuration which is installed by default.

Docker network ls

```
devops@master-node:~$ docker network ls
NETWORK ID     NAME      DRIVER    SCOPE
3ded9beafdba   bridge    bridge    local
56f6edea22b3   host      host      local
fba18cf3b0f9   none      null      local
devops@master-node:~$
```

Here Bridges are like a default VPC which is created by default

Docker network inspect bridge

```
devops@master-node:~$ docker network inspect bridge
[
    {
        "Name": "bridge",
        "Id": "3ded9beafdba545081221905ab50a960b6a714660c3e9e69ce6341ea56f1296b",
        "Created": "2024-01-24T01:24:01.658741366Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": [
                {
                    "Subnet": "172.17.0.0/16",
                    "Gateway": "172.17.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {},
        "Options": {
            "com.docker.network.bridge.default_bridge": "true",
            "com.docker.network.bridge.enable_icc": "true",
            "com.docker.network.bridge.enable_ip_masquerade": "true",
            "com.docker.network.bridge.host_binding_ipv4": "0.0.0.0",
            "com.docker.network.bridge.name": "docker0",
            "com.docker.network.driver.mtu": "1500"
        },
        "Labels": {}
```

By this we will see the Gateway,subnet and bridgename as docker0

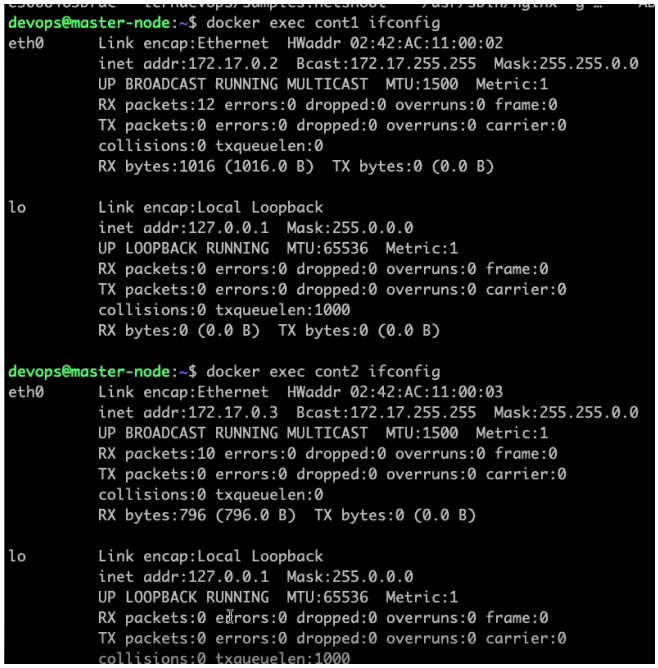
Docker zero network is the default network created by the docker software automatically when we install.

Now i'm creating the two container with the below command:

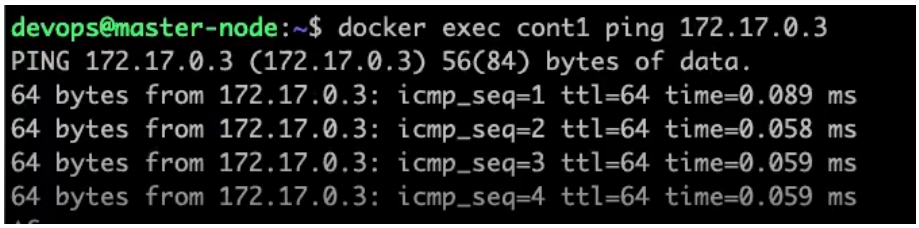
```
root@ip-172-31-8-163:~# docker run -d --name cont1 harshit407/cizaar:v1
fab19c31616536df9c582e400cc12e7d0a15b1427791a7ba1d5b0491d68bd511
root@ip-172-31-8-163:~# docker run -d --name cont2 harshit407/cizaar:v1
fff9ad5eaae0b4e6d8c9b108bf874164bda8210634d6d43727b13be5cb9a6671
root@ip-172-31-8-163:~# docker container ps
CONTAINER ID   IMAGE                  COMMAND                  CREATED          STATUS          PORTS      NAMES
fff9ad5eaae0   harshit407/cizaar:v1   "/opt/tomcat/bin/cat…"   6 seconds ago    Up 5 seconds    8080/tcp   cont2
fab19c316165   harshit407/cizaar:v1   "/opt/tomcat/bin/cat…"   12 seconds ago   Up 11 seconds   8080/tcp   cont1
root@ip-172-31-8-163:~#
```

NOw if we want to check the IP of the container we can check by the inspect command
If we check the output of the inspect we got to know that both are having the same gateway connected means both are connected with the same network
docker container inspect "container ID"

We can also check the below details:



By the below command we can check the connection between both the container cont1 to con2 and vise versa by ping



Docker would allow us to customize the bridge network
So for that we can take the help of the CLI for the command:
“docker network create –help”
“docker network create mynet --driver bridge --subnet 19.16.0.0/16 --gateway 19.16.0.1”
In the above command we can gave anysubnet as per the requirement and with this we are creating our customer network

NO if we want to check our created network configuration we can check with the below command:
“docker network inspect mynet “

If we want ot create our custom network with the container we can use the below command:
Docker run -d --name cont3 --network mynet harshit407/cizaar:v1
Docker run -d --name cont4 --network mynet harshit407/cizaar:v1

In the above statement custom network means the network which we/user created on that network we can create our container.

```
root@ip-172-31-8-163:~# docker run -d --name cont4 --network mynet harshit407/cizaar:v1
35439f20cb3f114c2eff13bb6985517b9dcbf88833d378bb388e138d0a70d357
root@ip-172-31-8-163:~# docker run -d --name cont3 --network mynet harshit407/cizaar:v1
f9a1831d198223f700dd8540fa148e3fa36660ae63164af9e33ddd4c6085d006
root@ip-172-31-8-163:~# docker container ps
CONTAINER ID   IMAGE                  COMMAND                  CREATED          STATUS          PORTS      NAMES
f9a1831d1982   harshit407/cizaar:v1   "/opt/tomcat/bin/cat…"   6 seconds ago    Up 5 seconds    8080/tcp   cont3
35439f20cb3f   harshit407/cizaar:v1   "/opt/tomcat/bin/cat…"   53 seconds ago   Up 52 seconds   8080/tcp   cont4
fff9ad5eaae0   harshit407/cizaar:v1   "/opt/tomcat/bin/cat…"   2 hours ago      Up 2 hours      8080/tcp   cont2
fab19c316165   harshit407/cizaar:v1   "/opt/tomcat/bin/cat…"   2 hours ago      Up 2 hours      8080/tcp   cont1
root@ip-172-31-8-163:~#
```

In the above output snip cont1, cont2 belongs to the default network and cont3,cont4 are belongs to the custom user defined network

```
root@ip-172-31-8-163:~# docker exec cont3 hostname -i
19.16.0.3
root@ip-172-31-8-163:~# docker exec cont4 hostname -i
19.16.0.2
root@ip-172-31-8-163:~# docker exec cont2 hostname -i
172.17.0.3
root@ip-172-31-8-163:~# docker exec cont1 hostname -i
172.17.0.2
```

From here we can see same network can talk to each other and vise versa true

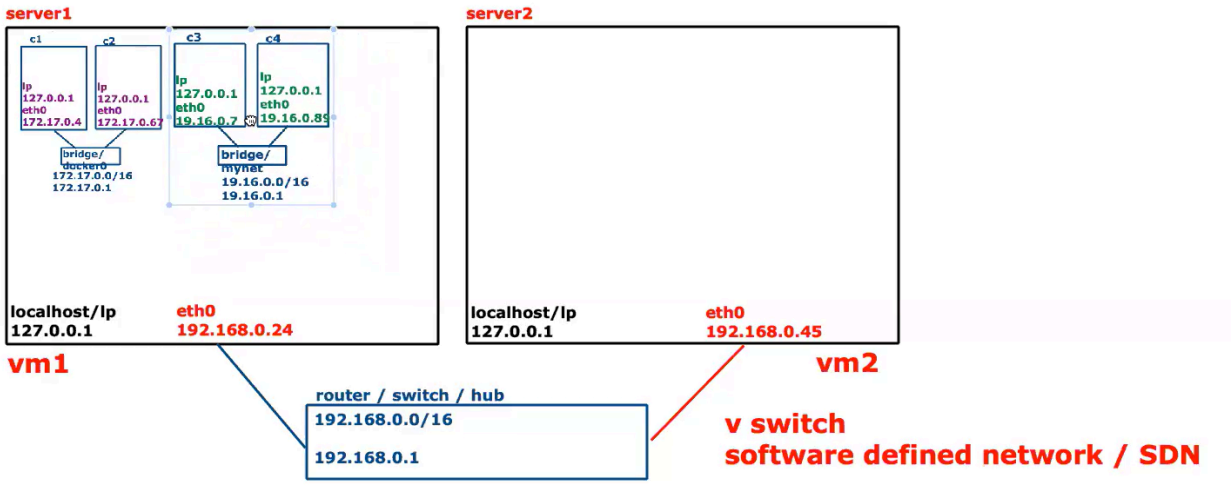
If we are running this command and if we got this below error means ping is not installed in the container so,open the container and install the ping also follow the below steps for reference.

```
root@ip-172-31-8-163:~# docker image ls
REPOSITORY          TAG            IMAGE ID       CREATED         SIZE
dockerfile          v1             3fe8620610b1   47 hours ago    361MB
harshit407/cizaar   v1             3fe8620610b1   47 hours ago    361MB
cizaar              v1             3fe8620610b1   47 hours ago    361MB
docker              v1             3fe8620610b1   47 hours ago    361MB
harshit407/cizaar   nginx-latest   92b11f67642b   7 weeks ago     187MB
nginx               latest         92b11f67642b   7 weeks ago     187MB
ubuntu              18.04          f9a80a55f492   10 months ago   63.2MB
root@ip-172-31-8-163:~# docker container ps
CONTAINER ID   IMAGE                  COMMAND                  CREATED          STATUS          PORTS      NAMES
f9a1831d1982   harshit407/cizaar:v1   "/opt/tomcat/bin/cat…"   20 minutes ago   Up 20 minutes   8080/tcp   cont3
35439f20cb3f   harshit407/cizaar:v1   "/opt/tomcat/bin/cat…"   21 minutes ago   Up 21 minutes   8080/tcp   cont4
fff9ad5eaae0   harshit407/cizaar:v1   "/opt/tomcat/bin/cat…"   3 hours ago      Up 3 hours      8080/tcp   cont2
fab19c316165   harshit407/cizaar:v1   "/opt/tomcat/bin/cat…"   3 hours ago      Up 3 hours      8080/tcp   cont1
root@ip-172-31-8-163:~# docker exec -it f9a1831d1982 /bin/bash
root@f9a1831d1982:/#
```

Then try to ping with the below command:
ex:Docker exec cont3 ping 172.17.0.2

Now if we want to connect the container cont3 with cont1 and cont2 then follow the below process:
Docker network connect bridge cont3 (by this command now cont3 will become the part of both custom and default bridge)

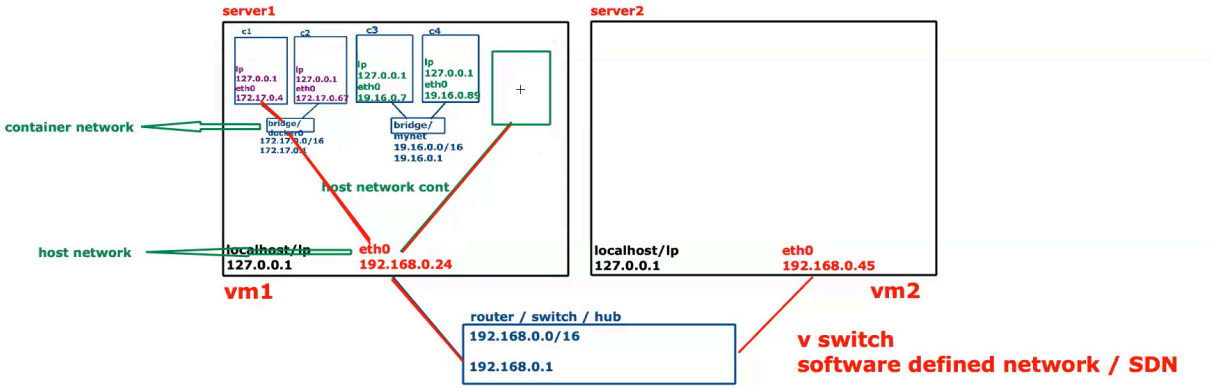
If we try to connect cont3 to cont1,2 it will connect smoothly.



Host network container -> The container which is not connected with the containers but it was connected directly to the main host network .It is like we installing jenkins in VM.We can directly access with host public IP addres in our browser with IP:defaultport ex:(Not important)
Docker network ls
Ex:docker run -d --name cont5 --network host harshit407/cizaar:v1

In the below output we can see the different as every container is having the IP address but then host/cont5 container is not having it due to it is connected directly with the host network

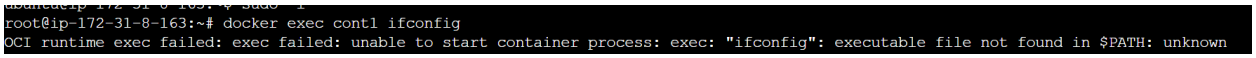
```
root@ip-172-31-8-163:~# docker network ls
NETWORK ID     NAME      DRIVER    SCOPE
9f8f8ec884f2   bridge    bridge    local
e5cf631ac5a6   host      host      local
9bb9eb5efc53   mynet     bridge    local
4033b4d70ae3   none      null      local
root@ip-172-31-8-163:~# docker run -d --name  cont5 --network host harshit407/cizaar:v1
e096c2439bf80db7e52e19377bae9d08ac6684214aa7645a3b40d6cb2216e258
root@ip-172-31-8-163:~# docker container ps
CONTAINER ID   IMAGE                  COMMAND                  CREATED          STATUS          PORTS      NAMES
e096c2439bf8   harshit407/cizaar:v1   "/opt/tomcat/bin/cat…"   8 seconds ago    Up 7 seconds               cont5
f9a1831d1982   harshit407/cizaar:v1   "/opt/tomcat/bin/cat…"   52 minutes ago   Up 52 minutes   8080/tcp   cont3
35439f20cb3f   harshit407/cizaar:v1   "/opt/tomcat/bin/cat…"   53 minutes ago   Up 53 minutes   8080/tcp   cont4
fff9ad5eaae0   harshit407/cizaar:v1   "/opt/tomcat/bin/cat…"   3 hours ago      Up 3 hours      8080/tcp   cont2
fab19c316165   harshit407/cizaar:v1   "/opt/tomcat/bin/cat…"   3 hours ago      Up 3 hours      8080/tcp   cont1
root@ip-172-31-8-163:~#
```



Normally if we search something inside the container and which is not in that container then it will find like the below sequence:
container->bridge->hostnetwork-> router->internet container network

container->hostnetwork-> router->internet host network

Again if we got the below error while running the command



Do install the ifconfig in the container with the below command:
Apt-get update
apt install net-tools

Last concept is None in this::

Docker network ls
None networ is only comes when we diattach the network

So if any container with in a same network type like default and from in that anyone compromised by the hacker

Then we can quickly disconnect the bridge connection of the container with the below command:
Docker network disconnect bridge cont1
By the above command it will disconnected with the network now no one access it and later on we can check and if required we can reconnect it again.This is like you are exist but you don't have any address so,no body can track you now.

==========END==========