Single-Node Docker Bridge Networking

Submitted by:

1. Dileep Kumar (ERP: 18255)
2. Muhammad Arsalan Mubeen (ERP: 23394)

Single-Node Docker Bridge Networking

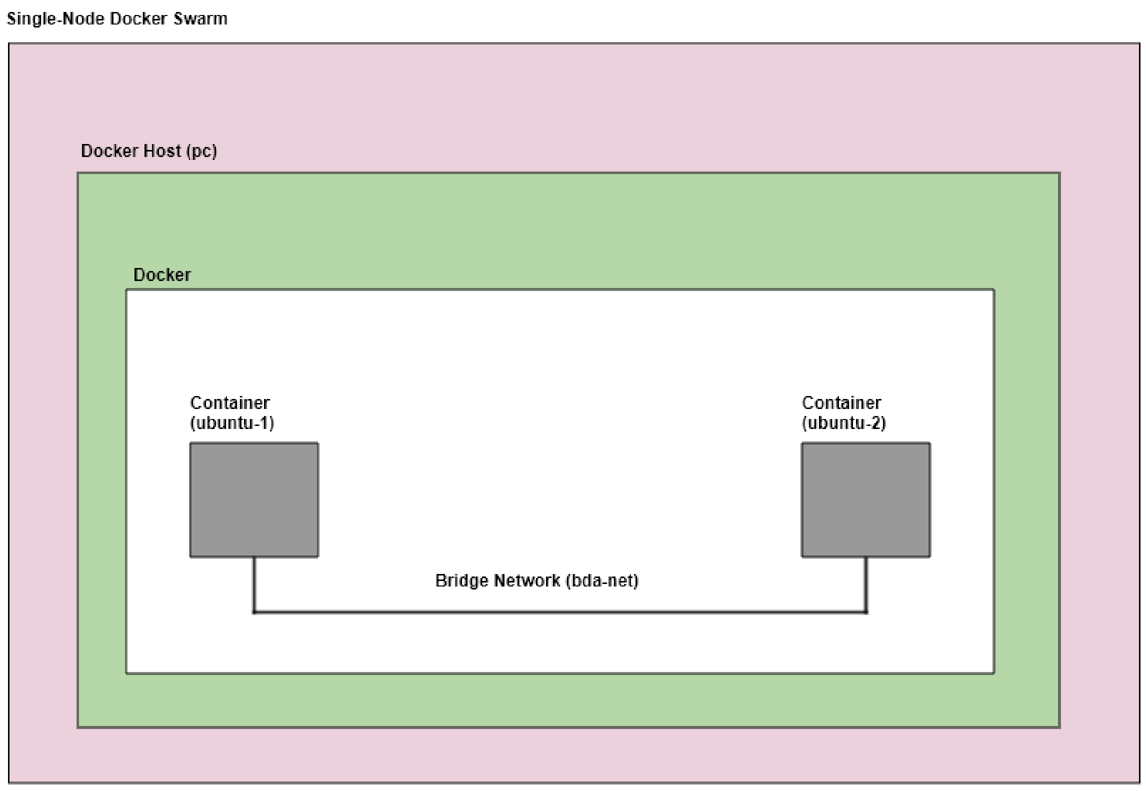
Contents:

Architecture

We have implemented single-node docker bridge networking with following architecture.

We have created two containers (ubuntu-1, ubuntu-2) on the single Docker host. Then we created a bridge network (**bda-net**) and connected both containers through bda-net. Finally we tested network connectivity between both containers.

Architectural Block Diagram:



Single-Node Docker Bridge Networking

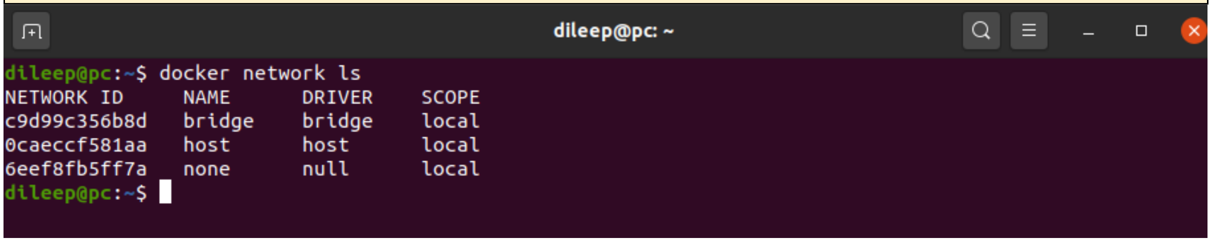
Step 1: Installing bridge utilities and checking network information

Docker comes with a pre-built default network called **bridge**.

All networks created with the *bridge* driver are based on a Linux bridge (virtual switch).

Verify available networks on the docker host

```
docker network ls
```

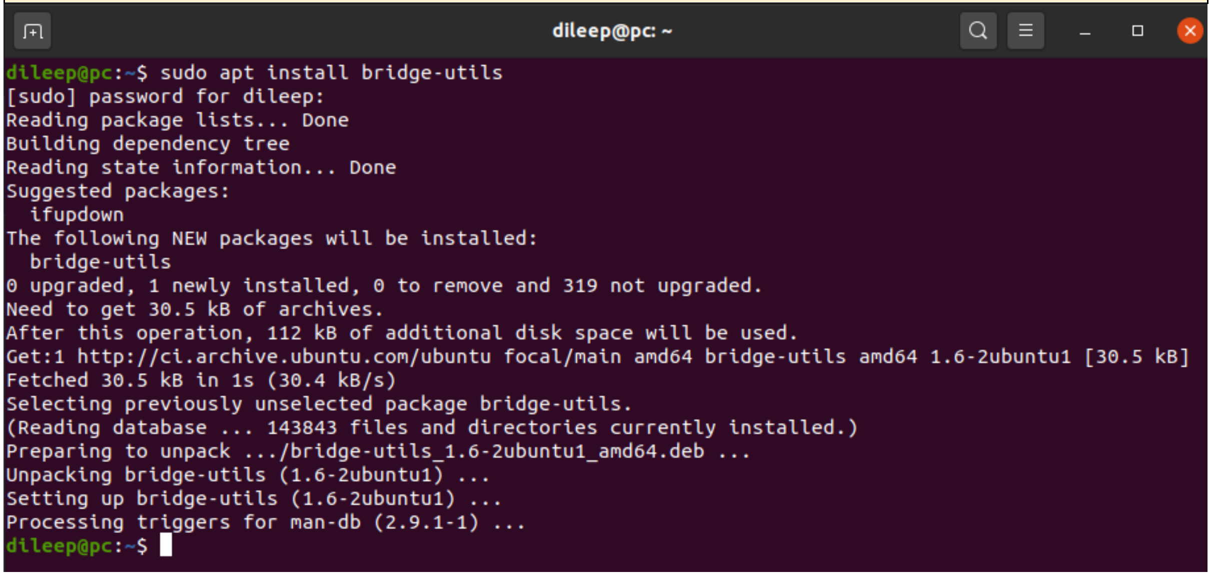


dileep@pc: ~

```
dileep@pc:~$ docker network ls
NETWORK ID     NAME      DRIVER    SCOPE
c9d99c356b8d   bridge    bridge    local
0caeccf581aa   host      host      local
6eef8fb5ff7a   none      null      local
dileep@pc:~$
```

Install the brctl utility to use the Linux bridges on Docker host

```
apt-get install bridge-utils
```

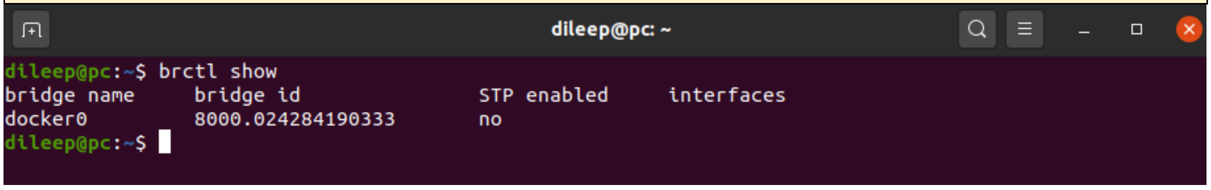


dileep@pc: ~

```
dileep@pc:~$ sudo apt install bridge-utils
[sudo] password for dileep:
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  ifupdown
The following NEW packages will be installed:
  bridge-utils
0 upgraded, 1 newly installed, 0 to remove and 319 not upgraded.
Need to get 30.5 kB of archives.
After this operation, 112 kB of additional disk space will be used.
Get:1 http://ci.archive.ubuntu.com/ubuntu focal/main amd64 bridge-utils amd64 1.6-2ubuntu1 [30.5 kB]
Fetched 30.5 kB in 1s (30.4 kB/s)
Selecting previously unselected package bridge-utils.
(Reading database ... 143843 files and directories currently installed.)
Preparing to unpack .../bridge-utils_1.6-2ubuntu1_amd64.deb ...
Unpacking bridge-utils (1.6-2ubuntu1) ...
Setting up bridge-utils (1.6-2ubuntu1) ...
Processing triggers for man-db (2.9.1-1) ...
dileep@pc:~$
```

List the bridges on Docker host

```
brctl show
```

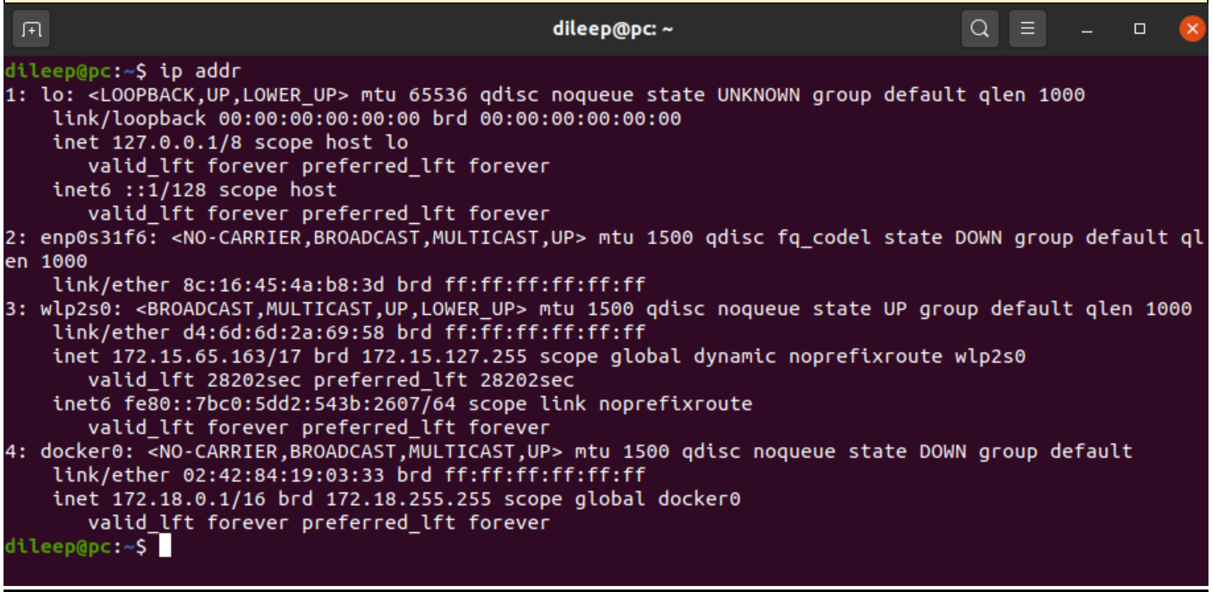


dileep@pc: ~

```
dileep@pc:~$ brctl show
bridge name     bridge id               STP enabled     interfaces
docker0         8000.024284190333       no
dileep@pc:~$
```

View the detailed info of the docker0 bridge

```
ip addr
```



dileep@pc: ~

```
dileep@pc:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s31f6: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel state DOWN group default ql
en 1000
    link/ether 8c:16:45:4a:b8:3d brd ff:ff:ff:ff:ff:ff
3: wlp2s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether d4:6d:6d:2a:69:58 brd ff:ff:ff:ff:ff:ff
    inet 172.15.65.163/17 brd 172.15.127.255 scope global dynamic noprefixroute wlp2s0
       valid_lft 28202sec preferred_lft 28202sec
    inet6 fe80::7bc0:5dd2:543b:2607/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
4: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:84:19:03:33 brd ff:ff:ff:ff:ff:ff
    inet 172.18.0.1/16 brd 172.18.255.255 scope global docker0
       valid_lft forever preferred_lft forever
dileep@pc:~$
```

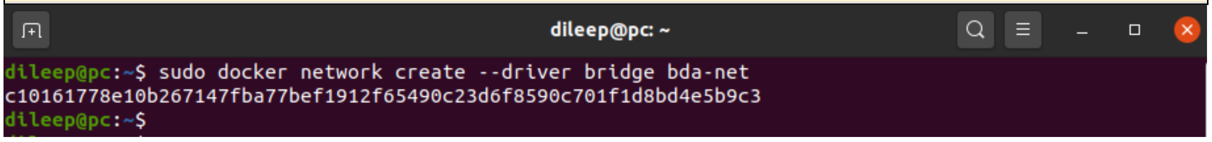
Step 2: Creating a user defined bridge network

Create a Bridge Network

Let’s create a bridge network called **bda-net** using the following command.

This returns the network ID of the created network.

```
sudo docker network create −−driver bridge bda−net
```

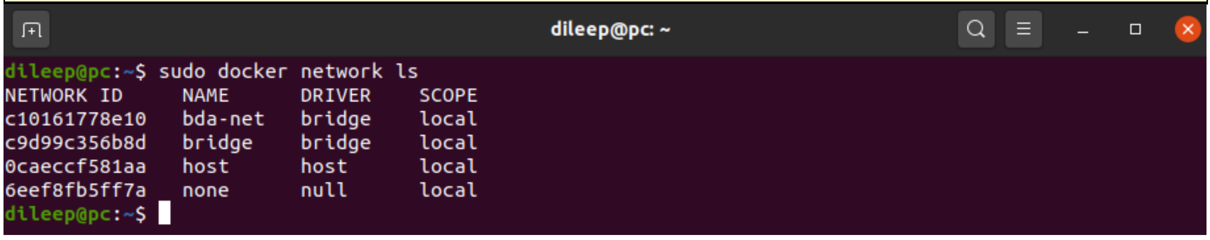


```
dileep@pc:~$ sudo docker network create --driver bridge bda-net
c10161778e10b267147fba77bef1912f65490c23d6f8590c701f1d8bd4e5b9c3
dileep@pc:~$
```

Verify the recently created bda−net bridge network

To check whether the bda-net bridge network has been created successfully or not, list all the networks.

```
sudo docker network ls
```

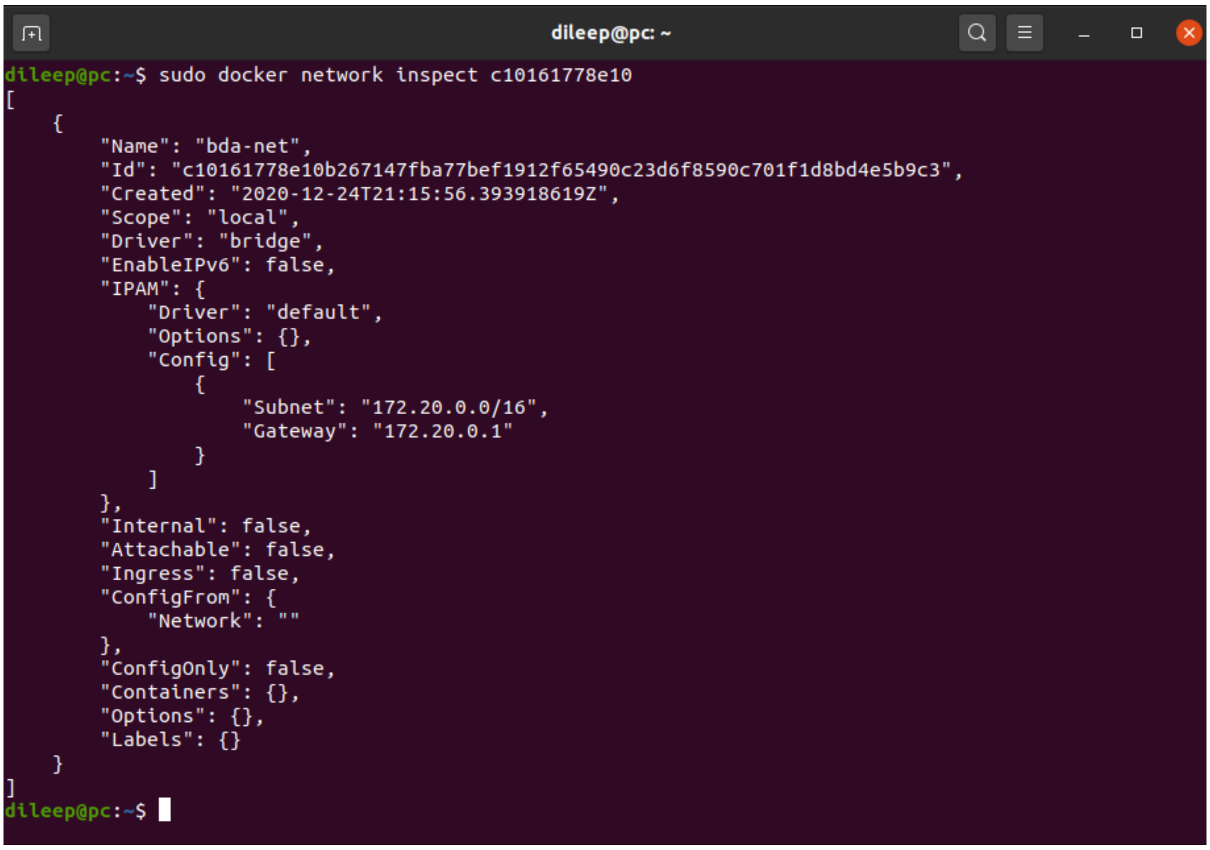


```
dileep@pc:~$ sudo docker network ls
NETWORK ID     NAME      DRIVER    SCOPE
c10161778e10   bda-net   bridge    local
c9d99c356b8d   bridge    bridge    local
0caeccf581aa   host      host      local
6eef8fb5ff7a   none      null      local
dileep@pc:~$
```

Inspect the bda−net

The network currently does not contain any container associated with it.

```
sudo docker network inspect bda−net
```



```
dileep@pc:~$ sudo docker network inspect c10161778e10
[
    {
        "Name": "bda-net",
        "Id": "c10161778e10b267147fba77bef1912f65490c23d6f8590c701f1d8bd4e5b9c3",
        "Created": "2020-12-24T21:15:56.393918619Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.20.0.0/16",
                    "Gateway": "172.20.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {},
        "Options": {},
        "Labels": {}
    }
]
dileep@pc:~$
```

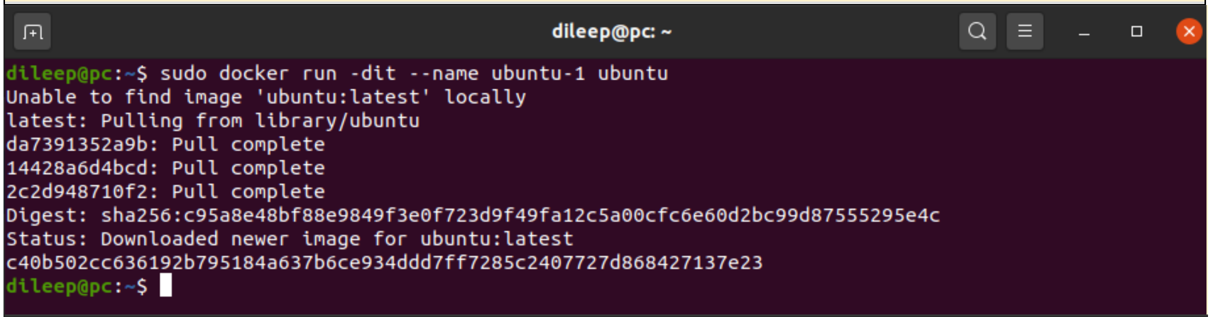
Create 2 Containers

Let's try to create Containers and associate them with different network specifications.

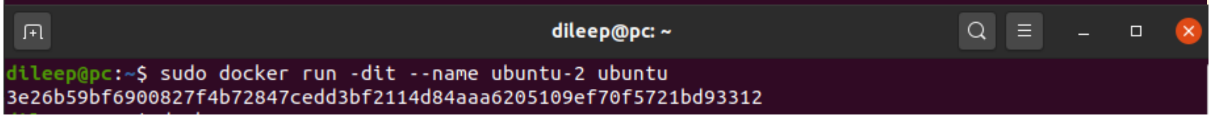
We will create 2 ubuntu Containers here.

- Create a container called "**ubuntu-1**" which is to be connected to the **bda-net** bridge network.
- Create another container called "**ubuntu-2**" which is to be connected to the **bda-net** bridge network.

```
sudo docker run –dit ––name ubuntu-1 ubuntu
sudo docker run –dit ––name ubuntu-2 ubuntu
```



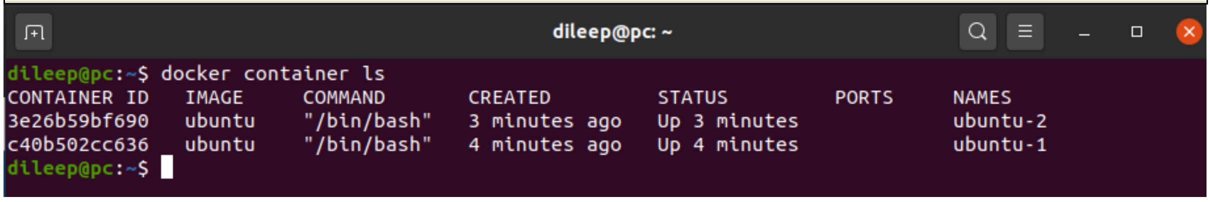
```
dileep@pc:~$ sudo docker run -dit --name ubuntu-1 ubuntu
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
da7391352a9b: Pull complete
14428a6d4bcd: Pull complete
2c2d948710f2: Pull complete
Digest: sha256:c95a8e48bf88e9849f3e0f723d9f49fa12c5a00cfc6e60d2bc99d87555295e4c
Status: Downloaded newer image for ubuntu:latest
c40b502cc636192b795184a637b6ce934ddd7ff7285c2407727d868427137e23
dileep@pc:~$
```



```
dileep@pc:~$ sudo docker run -dit --name ubuntu-2 ubuntu
3e26b59bf6900827f4b72847cedd3bf2114d84aaa6205109ef70f5721bd93312
```

Verify the containers are created

```
sudo docker network ls
```

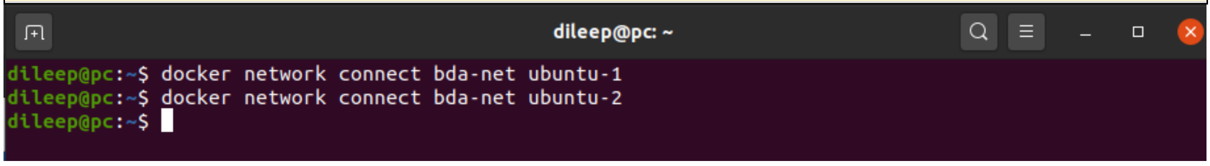


```
dileep@pc:~$ docker container ls
CONTAINER ID   IMAGE     COMMAND       CREATED         STATUS         PORTS     NAMES
3e26b59bf690   ubuntu    "/bin/bash"   3 minutes ago   Up 3 minutes             ubuntu-2
c40b502cc636   ubuntu    "/bin/bash"   4 minutes ago   Up 4 minutes             ubuntu-1
dileep@pc:~$
```

Connect both containers (ubuntu-1 and ubuntu-2) to bridge network (bda-net)

The following command connects an already-running containers (ubuntu-1, and ubuntu-2) to an already-existing bda-net network:

```
docker network connect bda-net ubuntu-1
docker network connect bda-net ubuntu-2
```



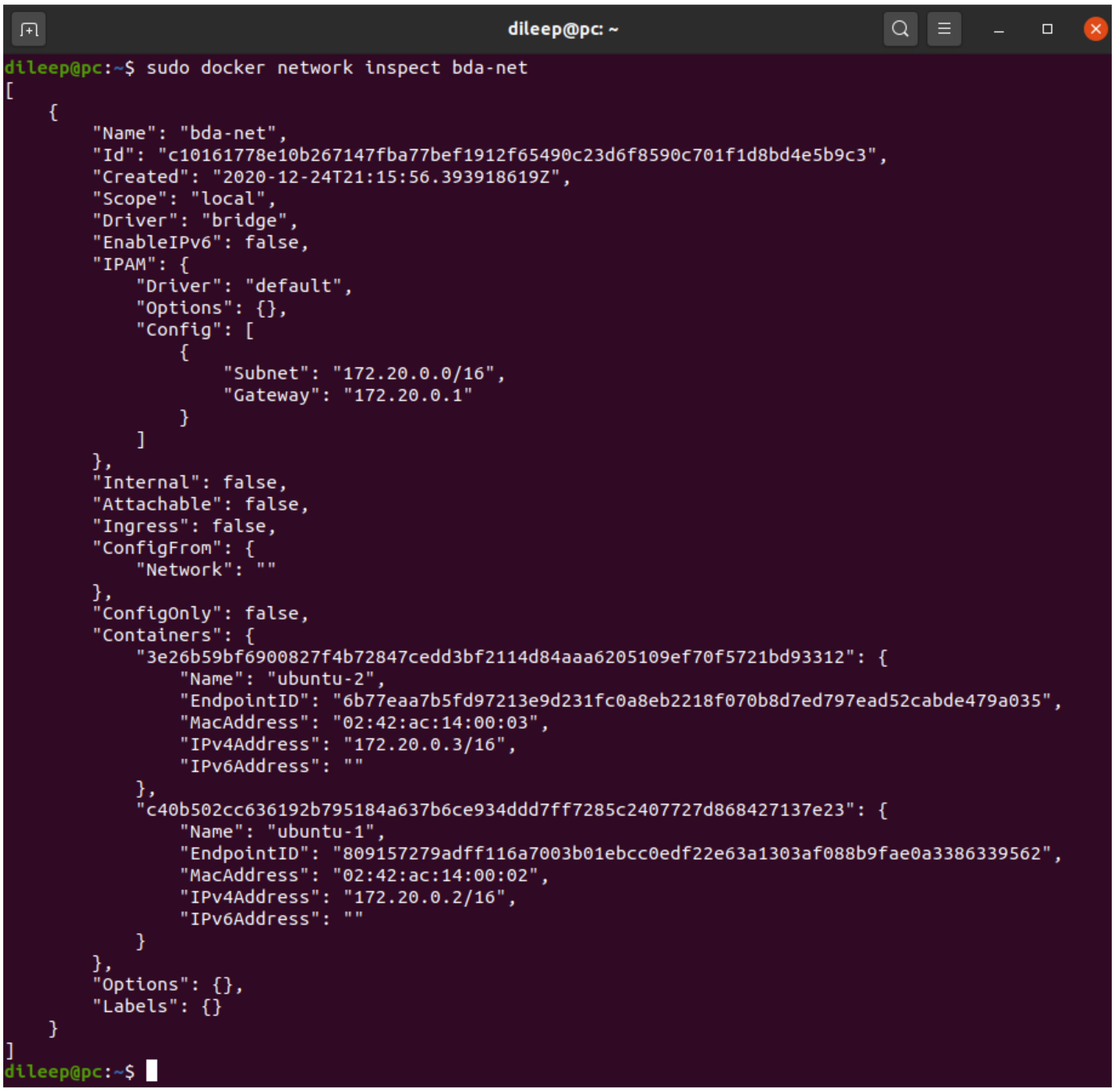
```
dileep@pc:~$ docker network connect bda-net ubuntu-1
dileep@pc:~$ docker network connect bda-net ubuntu-2
dileep@pc:~$
```

Verify both containers are connected with bda-net

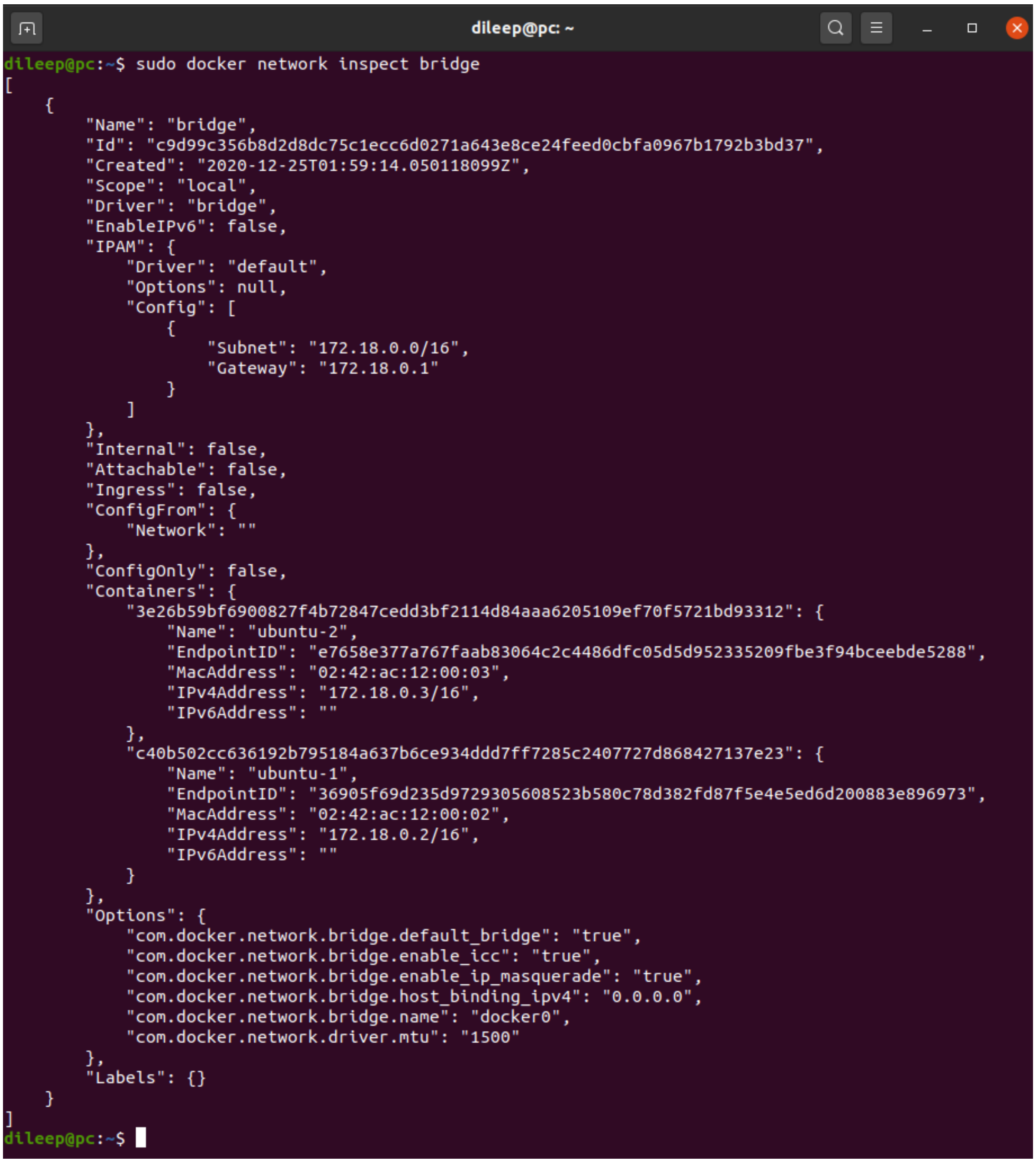
After we have created the Docker Containers and associated them with the networks, we can now inspect the networks.

To verify ubuntu-1 and ubuntu-2 containers are connected with the bda-net bridge network.

```
sudo docker network inspect bda−net
sudo docker network inspect bridge
```



```
dileep@pc:~$ sudo docker network inspect bda-net
[
    {
        "Name": "bda-net",
        "Id": "c10161778e10b267147fba77bef1912f65490c23d6f8590c701f1d8bd4e5b9c3",
        "Created": "2020-12-24T21:15:56.393918619Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.20.0.0/16",
                    "Gateway": "172.20.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {
            "3e26b59bf6900827f4b72847cedd3bf2114d84aaa6205109ef70f5721bd93312": {
                "Name": "ubuntu-2",
                "EndpointID": "6b77eaa7b5fd97213e9d231fc0a8eb2218f070b8d7ed797ead52cabde479a035",
                "MacAddress": "02:42:ac:14:00:03",
                "IPv4Address": "172.20.0.3/16",
                "IPv6Address": ""
            },
            "c40b502cc636192b795184a637b6ce934ddd7ff7285c2407727d868427137e23": {
                "Name": "ubuntu-1",
                "EndpointID": "809157279adff116a7003b01ebcc0edf22e63a1303af088b9fae0a3386339562",
                "MacAddress": "02:42:ac:14:00:02",
                "IPv4Address": "172.20.0.2/16",
                "IPv6Address": ""
            }
        },
        "Options": {},
        "Labels": {}
    }
]
dileep@pc:~$
```


dileep@pc: ~

```
dileep@pc:~$ sudo docker network inspect bridge
[
    {
        "Name": "bridge",
        "Id": "c9d99c356b8d2d8dc75c1ecc6d0271a643e8ce24feed0cbfa0967b1792b3bd37",
        "Created": "2020-12-25T01:59:14.050118099Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": [
                {
                    "Subnet": "172.18.0.0/16",
                    "Gateway": "172.18.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {
            "3e26b59bf6900827f4b72847cedd3bf2114d84aaa6205109ef70f5721bd93312": {
                "Name": "ubuntu-2",
                "EndpointID": "e7658e377a767faab83064c2c4486dfc05d5d952335209fbe3f94bceebde5288",
                "MacAddress": "02:42:ac:12:00:03",
                "IPv4Address": "172.18.0.3/16",
                "IPv6Address": ""
            },
            "c40b502cc636192b795184a637b6ce934ddd7ff7285c2407727d868427137e23": {
                "Name": "ubuntu-1",
                "EndpointID": "36905f69d235d9729305608523b580c78d382fd87f5e4e5ed6d200883e896973",
                "MacAddress": "02:42:ac:12:00:02",
                "IPv4Address": "172.18.0.2/16",
                "IPv6Address": ""
            }
        },
        "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": {}
    }
]
dileep@pc:~$ 
```

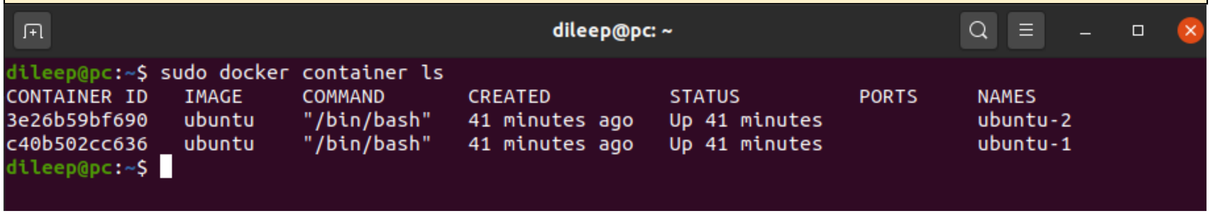
Step 3: Testing Network Connectivity:

We can go to any one of these containers and ping the other using the IP address of other container.

Follow the following steps to ping from one container to other container.

Get the ID of the container

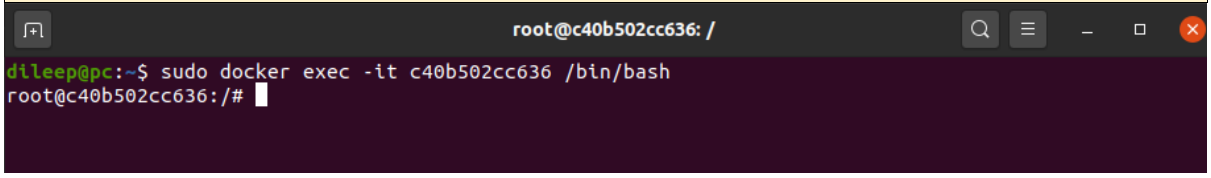
```
sudo docker container ls
```



```
dileep@pc:~$ sudo docker container ls
CONTAINER ID   IMAGE     COMMAND       CREATED          STATUS          PORTS     NAMES
3e26b59bf690   ubuntu    "/bin/bash"   41 minutes ago   Up 41 minutes             ubuntu-2
c40b502cc636   ubuntu    "/bin/bash"   41 minutes ago   Up 41 minutes             ubuntu-1
dileep@pc:~$
```

Exec into the container

```
sudo docker exec -it 6dd93d6cdc80 /bin/bash
```

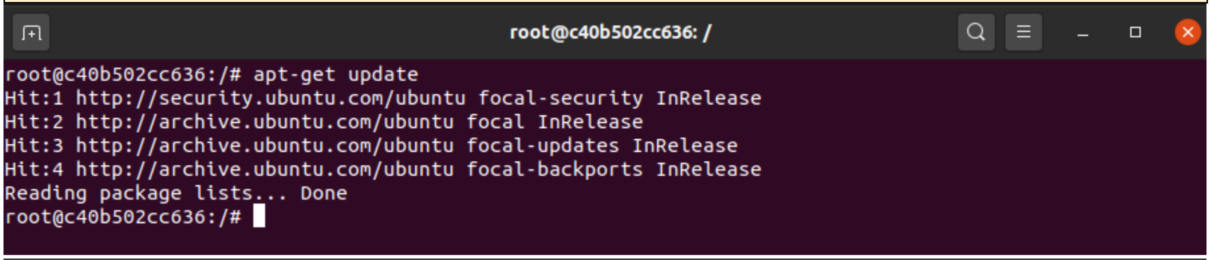


```
dileep@pc:~$ sudo docker exec -it c40b502cc636 /bin/bash
root@c40b502cc636:/#
```

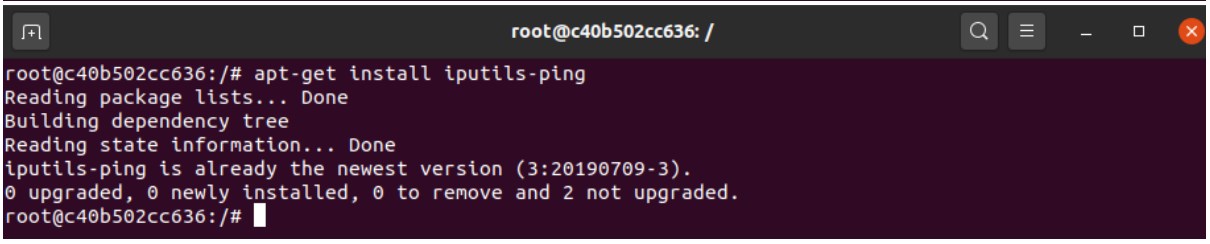
Update and Install the iputils-ping package in the container

```
apt-get update

apt-get install iputils-ping
```



```
root@c40b502cc636:/# apt-get update
Hit:1 http://security.ubuntu.com/ubuntu focal-security InRelease
Hit:2 http://archive.ubuntu.com/ubuntu focal InRelease
Hit:3 http://archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:4 http://archive.ubuntu.com/ubuntu focal-backports InRelease
Reading package lists... Done
root@c40b502cc636:/#
```

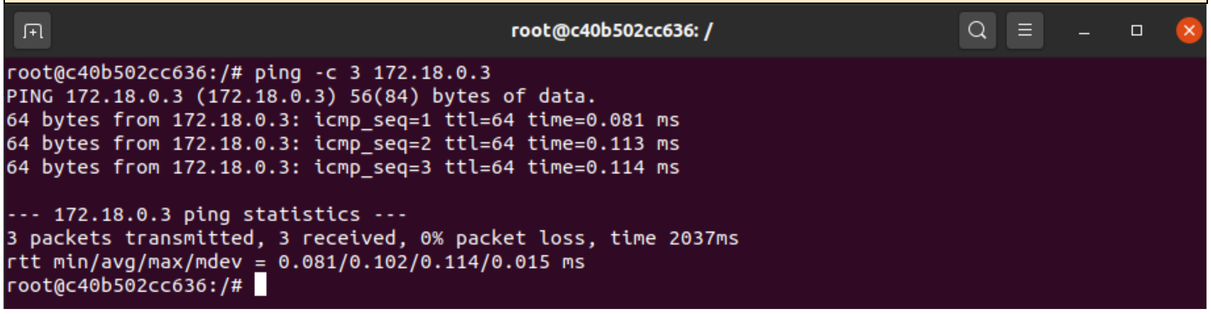


```
root@c40b502cc636:/# apt-get install iputils-ping
Reading package lists... Done
Building dependency tree
Reading state information... Done
iputils-ping is already the newest version (3:20190709-3).
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
root@c40b502cc636:/#
```

Verify communication between containers (ubuntu-1 and ubuntu-2)

Use the ping from ubuntu-1 and provide IP address of ubuntu-2

```
ping -c 3 172.18.0.3
```



root@c40b502cc636: /

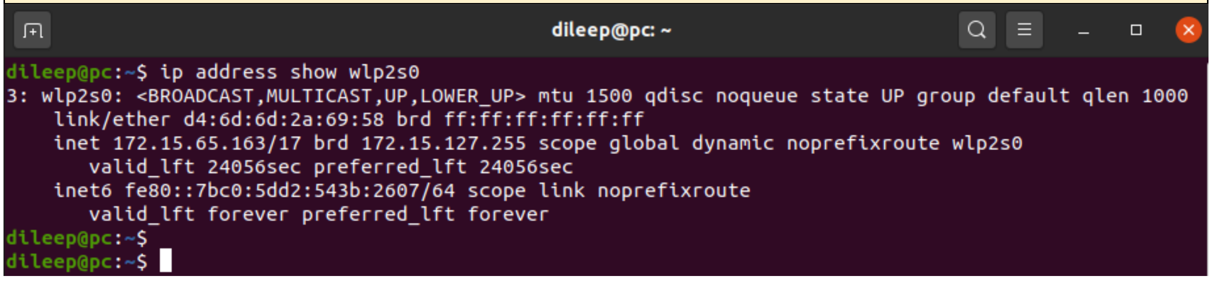
```
root@c40b502cc636:/# ping -c 3 172.18.0.3
PING 172.18.0.3 (172.18.0.3) 56(84) bytes of data.
64 bytes from 172.18.0.3: icmp_seq=1 ttl=64 time=0.081 ms
64 bytes from 172.18.0.3: icmp_seq=2 ttl=64 time=0.113 ms
64 bytes from 172.18.0.3: icmp_seq=3 ttl=64 time=0.114 ms

--- 172.18.0.3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2037ms
rtt min/avg/max/mdev = 0.081/0.102/0.114/0.015 ms
root@c40b502cc636:/#
```

Verify Communication between host and containers:

First we will check the IP address of our docker host

```
ip address show wlp2s0
```

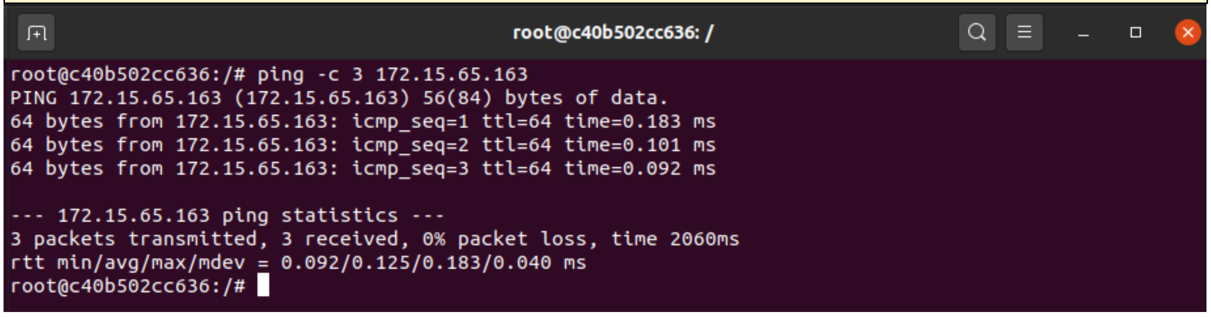


dileep@pc: ~

```
dileep@pc:~$ ip address show wlp2s0
3: wlp2s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether d4:6d:6d:2a:69:58 brd ff:ff:ff:ff:ff:ff
    inet 172.15.65.163/17 brd 172.15.127.255 scope global dynamic noprefixroute wlp2s0
       valid_lft 24056sec preferred_lft 24056sec
    inet6 fe80::7bc0:5dd2:543b:2607/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
dileep@pc:~$
dileep@pc:~$
```

Ping the IP address of host from within the container

```
Ping -c 3 172.15.65.163
```



root@c40b502cc636: /

```
root@c40b502cc636:/# ping -c 3 172.15.65.163
PING 172.15.65.163 (172.15.65.163) 56(84) bytes of data.
64 bytes from 172.15.65.163: icmp_seq=1 ttl=64 time=0.183 ms
64 bytes from 172.15.65.163: icmp_seq=2 ttl=64 time=0.101 ms
64 bytes from 172.15.65.163: icmp_seq=3 ttl=64 time=0.092 ms

--- 172.15.65.163 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2060ms
rtt min/avg/max/mdev = 0.092/0.125/0.183/0.040 ms
root@c40b502cc636:/#
```