Checking the connection between two services in an Overlay Network

We need at least two nodes for this Hands-on

docker node ls

```
root@ip-172-31-32-116:/home/ubuntu# docker node ls
ID                            HOSTNAME           STATUS    AVAILABILITY   MANAGER STATUS   ENGINE VERSION
2sgx9mu165qzxlzrsnzm9z8m9 *   ip-172-31-32-116   Ready     Active         Leader           24.0.5
50ickfymzwrg961fjcqulthiu     ip-172-31-38-39    Ready     Active                          24.0.5
root@ip-172-31-32-116:/home/ubuntu#
```

i-01f2abbdb0d592e12 (Manager)

PublicIPs: 18.179.178.175 PrivateIPs: 172.31.32.116

1. Use the below command to drain Manager node, so that when we launch our services it only gets launched to one node i.e worker 1.

docker node update –availability drain < Node ID>

docker node ls

```
root@ip-172-31-32-116:/home/ubuntu# docker node update --availability drain 2sgx9mu165qzxlzrsnzm9z8m9
2sgx9mu165qzxlzrsnzm9z8m9
root@ip-172-31-32-116:/home/ubuntu# docker node ls
ID                            HOSTNAME           STATUS    AVAILABILITY   MANAGER STATUS   ENGINE VERSION
2sgx9mu165qzxlzrsnzm9z8m9 *   ip-172-31-32-116   Ready     Drain          Leader           24.0.5
50ickfymzwrg961fjcqulthiu     ip-172-31-38-39    Ready     Active                          24.0.5
root@ip-172-31-32-116:/home/ubuntu#
```

i-01f2abbdb0d592e12 (Manager)

PublicIPs: 18.179.178.175 PrivateIPs: 172.31.32.116

2. Create a new Overlay Network

docker network create --driver overlay <network_name>

```
root@ip-172-31-32-116:/home/ubuntu# docker network create --driver overlay overlay-net
xi8n9w3b95v2ynh5qle1vbn2t
root@ip-172-31-32-116:/home/ubuntu#
```

i-01f2abbdb0d592e12 (Manager)

PublicIPs: 18.179.178.175 PrivateIPs: 172.31.32.116

3. Then create a new service attached to the user defined overlay network

docker service create --name <service_name> --network <network_name> --replicas 2 <image>

```
root@ip-172-31-32-116:/home/ubuntu# docker service create --name service-A --replicas 2 --network overlay-net nginx
qkgac6zpnbhbwgca1kn87g5x9
overall progress: 2 out of 2 tasks
1/2: running   [==================================================>]
2/2: running   [==================================================>]
verify: Service converged
root@ip-172-31-32-116:/home/ubuntu#
```

i-01f2abbdb0d592e12 (Manager)

PublicIPs: 18.179.178.175 PrivateIPs: 172.31.32.116

Now if I check any container is there

Manager node (there is no container)

docker ps

```
root@ip-172-31-32-116:/home/ubuntu# docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
root@ip-172-31-32-116:/home/ubuntu#
```

i-01f2abbdb0d592e12 (Manager)

PublicIPs: 18.179.178.175 PrivateIPs: 172.31.32.116

Now go to the worker node and type in the below commands to verify if the services were deployed only in one of the nodes.

docker ps

```
root@ip-172-31-38-39:/home/ubuntu# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED         STATUS         PORTS     NAMES
521fe24fe0e3   nginx:latest   "/docker-entrypoint.…"   2 minutes ago   Up 2 minutes   80/tcp    service-A.2.v1vi751w0cl9yz7pwlejcc6in
5095d66d7456   nginx:latest   "/docker-entrypoint.…"   2 minutes ago   Up 2 minutes   80/tcp    service-A.1.0qa5qql9yqzmisw3d5cbxsy81
root@ip-172-31-38-39:/home/ubuntu#
```

i-096b00a49c16f22a2 (Worker 1)

PublicIPs: 57.181.39.145 PrivateIPs: 172.31.38.39

4. Now change the availability status of the first node back to “active” so that we can launch a new service.

docker node update --availability active <node_id>

docker node ls

```
root@ip-172-31-32-116:/home/ubuntu# docker node update --availability active 2sgx9mu165qzxlzrsnzm9z8m9
2sgx9mu165qzxlzrsnzm9z8m9
root@ip-172-31-32-116:/home/ubuntu# docker node ls
ID                            HOSTNAME           STATUS    AVAILABILITY   MANAGER STATUS   ENGINE VERSION
2sgx9mu165qzxlzrsnzm9z8m9 *   ip-172-31-32-116   Ready     Active         Leader           24.0.5
50ickfymzwrg961fjcqulthiu     ip-172-31-38-39    Ready     Active                          24.0.5
root@ip-172-31-32-116:/home/ubuntu#
```

i-01f2abbdb0d592e12 (Manager)

PublicIPs: 18.179.178.175 PrivateIPs: 172.31.32.116

5. Now create the Second Service attached to the same user defined overlay network

docker service create --name <service_name> --replicas <no> --network <network_name> <image>

```
root@ip-172-31-32-116:/home/ubuntu# docker service create --name service-B --replicas 2 --network overlay-net nginx
c7g7i94afo562mzkruiyjv4xe
overall progress: 2 out of 2 tasks
1/2: running   [==================================================>]
2/2: running   [==================================================>]
verify: Service converged
root@ip-172-31-32-116:/home/ubuntu#
```

i-01f2abbdb0d592e12 (Manager)

PublicIPs: 18.179.178.175 PrivateIPs: 172.31.32.116

If you check the manager node you can see that one of the replicas of the service is deployed here, now we can properly check the connection of the overlay connection across the nodes. And make sure to copy the IP address of this container.

```
root@ip-172-31-32-116:/home/ubuntu# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED              STATUS              PORTS     NAMES
5a7de2d54f68   nginx:latest   "/docker-entrypoint.…"   About a minute ago   Up About a minute   80/tcp    service-B.1.79902zrzcew8nv4tbl82zylk8
root@ip-172-31-32-116:/home/ubuntu#
```

i-01f2abbdb0d592e12 (Manager)

PublicIPs: 18.179.178.175 PrivateIPs: 172.31.32.116

Check the worker or the other node again and you will notice one of the replicas was deployed here. No worries, just make sure that one replica is deployed on the manager node or thee first node.

```
root@ip-172-31-38-39:/home/ubuntu# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS          PORTS     NAMES
077f8821c23c   nginx:latest   "/docker-entrypoint.…"   3 minutes ago    Up 3 minutes    80/tcp    service-B.2.tnp2grsec4s7l0nc79ou8o2yv
521fe24fe0e3   nginx:latest   "/docker-entrypoint.…"   11 minutes ago   Up 11 minutes   80/tcp    service-A.2.v1vi75lw0cl9yz7pwlejcc6in
5095d66d7456   nginx:latest   "/docker-entrypoint.…"   11 minutes ago   Up 11 minutes   80/tcp    service-A.1.0qa5qql9yqzmisw3d5cbxsy81
root@ip-172-31-38-39:/home/ubuntu#
```

i-096b00a49c16f22a2 (Worker 1)

PublicIPs: 57.181.39.145 PrivateIPs: 172.31.38.39

6. Copy the IP address of this container. from Manager node

docker inspect <container_id> | grep “IP”

```
root@ip-172-31-32-116:/home/ubuntu# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED              STATUS              PORTS     NAMES
5a7de2d54f68   nginx:latest   "/docker-entrypoint.…"   About a minute ago   Up About a minute   80/tcp    service-B.1.79902zrzcew8nv4tbl82zylk8
root@ip-172-31-32-116:/home/ubuntu# docker inspect 5a7de2d54f68 | grep "IP"
            "LinkLocalIPv6Address": "",
            "LinkLocalIPv6PrefixLen": 0,
            "SecondaryIPAddresses": null,
            "SecondaryIPv6Addresses": null,
            "GlobalIPv6Address": "",
            "GlobalIPv6PrefixLen": 0,
            "IPAddress": "",
            "IPPrefixLen": 0,
            "IPv6Gateway": "",
                    "IPAMConfig": {
                        "IPv4Address": "10.0.1.7"
                    "IPAddress": "10.0.1.7",
                    "IPPrefixLen": 24,
                    "IPv6Gateway": "",
                    "GlobalIPv6Address": "",
                    "GlobalIPv6PrefixLen": 0,
root@ip-172-31-32-116:/home/ubuntu#
```

i-01f2abbdb0d592e12 (Manager)

PublicIPs: 18.179.178.175 PrivateIPs: 172.31.32.116

7. Now go inside Service A’s container in the worker node and install the ping utility.

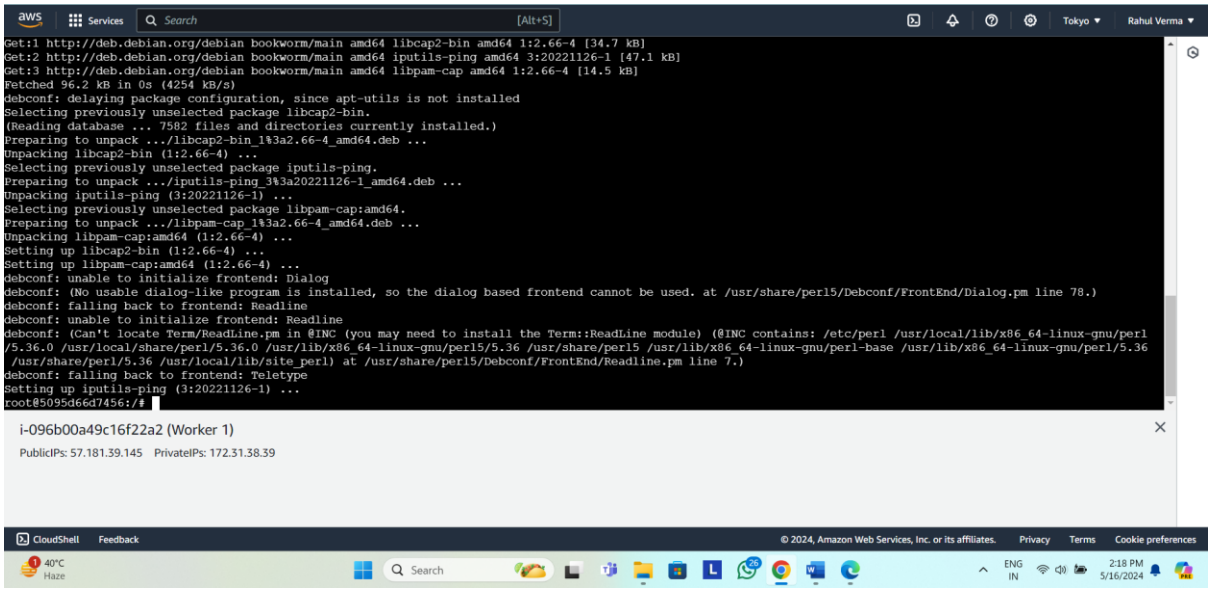
docker exec -it <container_id> bash

```
root@ip-172-31-38-39:/home/ubuntu# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS          PORTS     NAMES
077f8821c23c   nginx:latest   "/docker-entrypoint.…"   3 minutes ago    Up 3 minutes    80/tcp    service-B.2.tnp2grsec4s7l0nc79ou8o2yv
521fe24fe0e3   nginx:latest   "/docker-entrypoint.…"   11 minutes ago   Up 11 minutes   80/tcp    service-A.2.v1vi75lw0cl9yz7pwlejcc6in
5095d66d7456   nginx:latest   "/docker-entrypoint.…"   11 minutes ago   Up 11 minutes   80/tcp    service-A.1.0qa5qql9yqzmisw3d5cbxsy81
root@ip-172-31-38-39:/home/ubuntu# docker exec -it 5095d66d7456 bash
root@5095d66d7456:/#
```

i-096b00a49c16f22a2 (Worker 1)

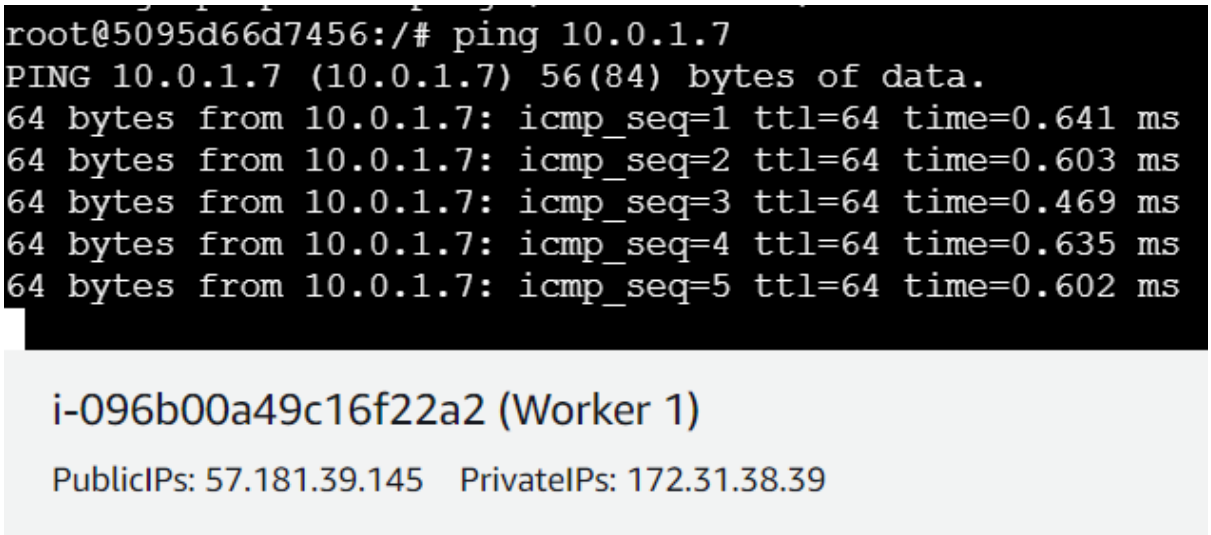
PublicIPs: 57.181.39.145 PrivateIPs: 172.31.38.39

sudo apt-get update && sudo apt-get install -y iputils-ping



Now ping the IP of the Service B's replica on the manager node to see if the Overlay connection works across containers of different services and across different nodes.

ping < Service B Manager Node replica IP>



```
64 bytes from 10.0.1.7: icmp_seq=122 ttl=64 time=0.590 ms
64 bytes from 10.0.1.7: icmp_seq=123 ttl=64 time=0.603 ms
64 bytes from 10.0.1.7: icmp_seq=124 ttl=64 time=0.516 ms
^C
--- 10.0.1.7 ping statistics ---
124 packets transmitted, 124 received, 0% packet loss, time 125893ms
rtt min/avg/max/mdev = 0.460/0.598/2.991/0.223 ms
root@5095d66d7456:/#
```

i-096b00a49c16f22a2 (Worker 1)

PublicIPs: 57.181.39.145 PrivateIPs: 172.31.38.39

Since the Ping was successful we can conclude that a container on service B can talk to container on service A even though both of them are on different node.

8. Now create a Third Service and DO NOT attach it to the User defined Overlay connection.

docker service create --name <service_name> <image>

```
root@ip-172-31-32-116:/home/ubuntu# docker service create --name service-c nginx
bk73x8asmbffouhlr3m09w8u2
overall progress: 1 out of 1 tasks
1/1: running   [==================================================>]
verify: Service converged
root@ip-172-31-32-116:/home/ubuntu#
```

i-01f2abbdb0d592e12 (Manager)

PublicIPs: 18.179.178.175 PrivateIPs: 172.31.32.116

Docker ps

```
root@ip-172-31-32-116:/home/ubuntu# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED              STATUS              PORTS     NAMES
bca5c8953cfb   nginx:latest   "/docker-entrypoint.…"   About a minute ago   Up About a minute   80/tcp    service-c.1.mdbyszb95jmn88n90vuvgl3gq
5a7de2d54f68   nginx:latest   "/docker-entrypoint.…"   25 minutes ago       Up 25 minutes       80/tcp    service-B.1.79902zrzcew8nv4tbl82zylk8
root@ip-172-31-32-116:/home/ubuntu#
```

i-01f2abbdb0d592e12 (Manager)

PublicIPs: 18.179.178.175 PrivateIPs: 172.31.32.116

9. Now go inside Service C's container in the manager node and install the ping utility.

docker exec -it <container id> bash

```
root@ip-172-31-32-116:/home/ubuntu# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS          PORTS     NAMES
bca5c8953cfb   nginx:latest   "/docker-entrypoint.…"   5 minutes ago    Up 5 minutes    80/tcp    service-c.1.mdbyszb95jmn88n90vuvgl3gq
5a7de2d54f68   nginx:latest   "/docker-entrypoint.…"   29 minutes ago   Up 29 minutes   80/tcp    service-B.1.79902zrzcew8nv4tbl82zylk8
root@ip-172-31-32-116:/home/ubuntu# docker exec -it bca5c8953cfb bash
root@bca5c8953cfb:/#
```

i-01f2abbdb0d592e12 (Manager)

PublicIPs: 18.179.178.175 PrivateIPs: 172.31.32.116

apt-get update && apt-get install -y iputils-ping

```
root@bca5c8953cfb:/# apt-get update && apt-get install -y iputils-ping
Get:1 http://deb.debian.org/debian bookworm InRelease [151 kB]
Get:2 http://deb.debian.org/debian bookworm-updates InRelease [55.4 kB]
Get:3 http://deb.debian.org/debian-security bookworm-security InRelease [48.0 kB]
Get:4 http://deb.debian.org/debian bookworm/main amd64 Packages [8786 kB]
Get:5 http://deb.debian.org/debian bookworm-updates/main amd64 Packages [13.8 kB]
Get:6 http://deb.debian.org/debian-security bookworm-security/main amd64 Packages [156 kB]
Fetched 9210 kB in 2s (5925 kB/s)
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libcap2-bin libpam-cap
The following NEW packages will be installed:
  iputils-ping libcap2-bin libpam-cap
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 96.2 kB of archives.
After this operation, 311 kB of additional disk space will be used.
Get:1 http://deb.debian.org/debian bookworm/main amd64 libcap2-bin amd64 1:2.66-4 [34.7 kB]
Get:2 http://deb.debian.org/debian bookworm/main amd64 iputils-ping amd64 3:20221126-1 [47.1 kB]
Get:3 http://deb.debian.org/debian bookworm/main amd64 libpam-cap amd64 1:2.66-4 [14.5 kB]
Fetched 96.2 kB in 0s (4055 kB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package libcap2-bin.
(Reading database ... 7582 files and directories currently installed.)
```

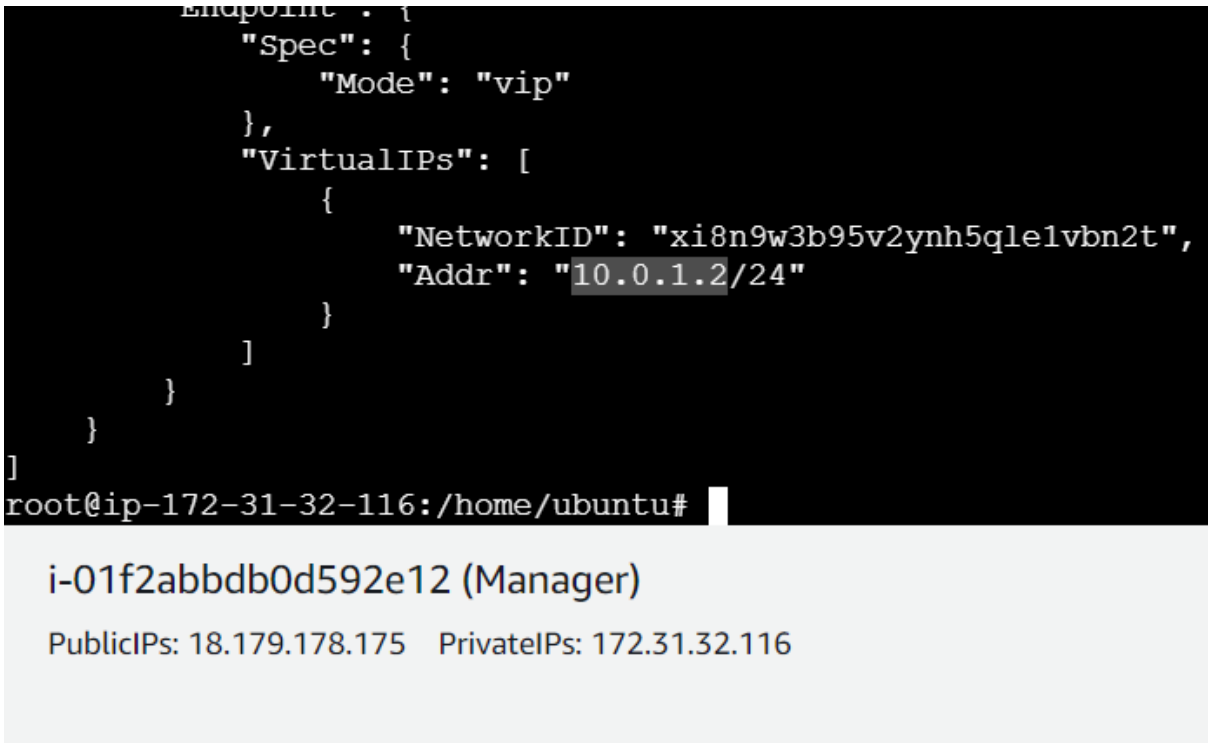
i-01f2abbdb0d592e12 (Manager)

PublicIPs: 18.179.178.175 PrivateIPs: 172.31.32.116

Now ping the IP of one of other services itself to see if the new service cann communicate with the service

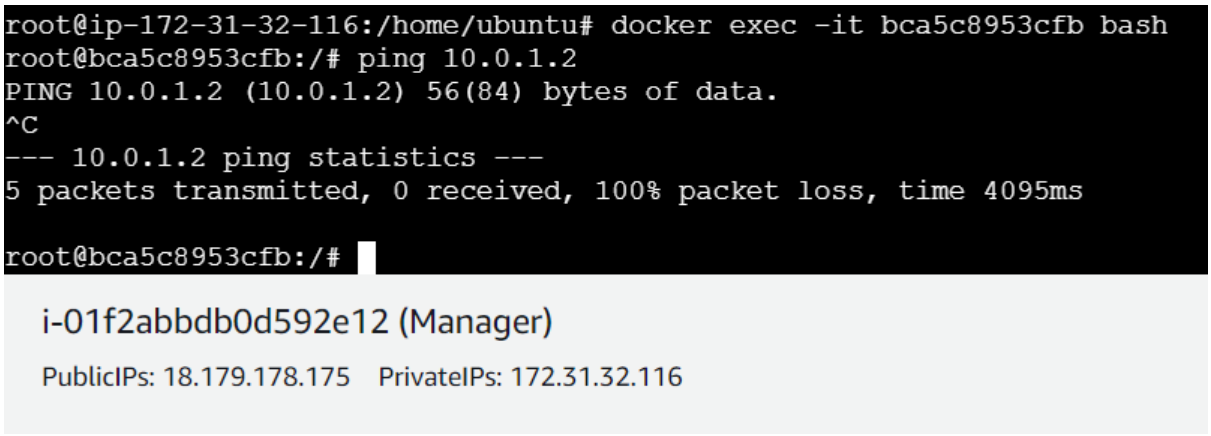
exit

docker inspect <service-id> (service-A)



Now get inside the service-c container again

ping < Service A/Service B 's IP>



As you can see here, A Service that is not a part of the Overlay connection cannot communicate with any of the services within that service.