Connecting a network in swarm

We don't have any service

docker service Is

```
root@ip-172-31-23-36:/home/ubuntu# docker service ls
ID NAME MODE REPLICAS IMAGE PORTS
root@ip-172-31-23-36:/home/ubuntu#

i-Oefc1c15a6ac477fa (Manager)
PublicIPs: 54.221.85.119 PrivateIPs: 172.31.23.36
```

1. Create a new network so that you can attach it to our service through updating

docker network create --driver overlay <network_name>

```
root@ip-172-31-23-36:/home/ubuntu# docker network create --driver overlay my-net
t4j9znz15mvwjyzzcz0nnxnu2
root@ip-172-31-23-36:/home/ubuntu#

i-Oefc1c15a6ac477fa (Manager)
PublicIPs: 54.221.85.119 PrivateIPs: 172.31.23.36
```

2. now create a service

docker service create --replicas 3 --name < name of service >< name of image > e.g- docker service create --replicas 3 --network my-net --name test nginx:latest

Now let's inspect our service

docker service inspect --pretty <service_name>

```
Services Q Search
                                                                            [Alt+S]
root@ip-172-31-23-36:/home/ubuntu# docker service inspect --pretty test
ID:
                921bqoc9kuddb4b75qe4uv4x6
Name:
                test
Service Mode:
                Replicated
Replicas:
Placement:
UpdateConfig:
Parallelism:
On failure: pause
Monitoring Period: 5s
Max failure ratio: 0
Update order:
                    stop-first
RollbackConfig:
Parallelism:
On failure:
               pause
Monitoring Period: 5s
Max failure ratio: 0
Rollback order:
                    stop-first
ontainerSpec:
                nginx:latest@sha256:e3ffd9d807cce9d9f973faff2e420b05243b49fd241b576a3de929bb3362cb60
Image:
Init:
Resources:
Networks: my-net
Endpoint Mode: vip
root@ip-172-31-23-36:/home/ubuntu#
  i-0efc1c15a6ac477fa (Manager)
  PublicIPs: 54.221.85.119 PrivateIPs: 172.31.23.36
```

In the above image we can see Networks: my-net

3. Now we will try to update the new service with the network we created For this will create a service without network flag

docker service create --replicas 3 --name demo nginx:latest

Let's inspect this service

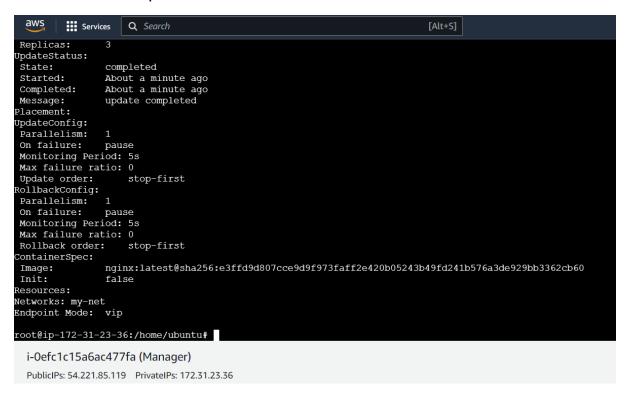
docker service inspect --pretty <service_name>

```
aws
         Services Q Search
                                                                          [Alt+S]
verify: Service converged
root@ip-172-31-23-36:/home/ubuntu# docker service inspect --pretty demo
                2tx4vp2rnptdl33nlshrjwamd
ID:
Name:
                demo
Service Mode:
                Replicated
Replicas:
Placement:
JpdateConfig:
Parallelism:
On failure:
               pause
Monitoring Period: 5s
Max failure ratio: 0
                    stop-first
Update order:
RollbackConfig:
Parallelism: 1
On failure:
               pause
Monitoring Period: 5s
Max failure ratio: 0
Rollback order:
                    stop-first
ContainerSpec:
                nginx:latest@sha256:e3ffd9d807cce9d9f973faff2e420b05243b49fd241b576a3de929bb3362cb60
Image:
Init:
                false
Resources:
Endpoint Mode: vip
root@ip-172-31-23-36:/home/ubuntu#
  i-0efc1c15a6ac477fa (Manager)
  PublicIPs: 54.221.85.119 PrivateIPs: 172.31.23.36
```

So there is no network mentioned over here for this particular service

docker service update --network-add < network name >< service name >

And now if we inspect this service we will be able to see network over there



4. if you want to update the service again by removing the network.

docker service update --network-rm< network name >< service name >

Let's verify this (and it's removed)

