

# Connecting a container to the bridge network

1. This command can be used to create a bridge network named 'bridge-net'

```
docker network create --driver bridge <network_name>
```

```
ubuntu@ip-172-31-32-116:~$ sudo su
root@ip-172-31-32-116:/home/ubuntu# docker network create --driver bridge bridge-net
b9f60ae33fa4f2f0ff04920b69044481b43254bc9e3aa4601d2771dd697d3445
root@ip-172-31-32-116:/home/ubuntu#
```

i-01f2abdb0d592e12 (Manager)

PublicIPs: 52.195.209.161 PrivateIPs: 172.31.32.116

To check the network

```
docker network ls
```

```
root@ip-172-31-32-116:/home/ubuntu# docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
40804148129b        bridge              bridge              local
b9f60ae33fa4        bridge-net          bridge              local
e9c8bdda68ec        docker_gwbridge     bridge              local
174394cf6e9c        host                host                local
tqjyxdouwobe        ingress             overlay             swarm
4754405da1a8        none                null                local
root@ip-172-31-32-116:/home/ubuntu#
```

i-01f2abdb0d592e12 (Manager)

PublicIPs: 52.195.209.161 PrivateIPs: 172.31.32.116

2. Now Create a nginx container attached to bridge network we created.

```
docker run -d --name <container_name> --network <network_name> -p 8080:80 <image>
```

```
root@ip-172-31-32-116:/home/ubuntu# docker run -d --name container-A --network bridge-net -p 8080:80 nginx:latest
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
09f376ebb190: Pull complete
a11fc495bafd: Pull complete
933cc8470577: Pull complete
999643392fb7: Pull complete
971bb7f4fb12: Pull complete
45337c09cd57: Pull complete
de3b062c0af7: Pull complete
Digest: sha256:a484819eb60211f5299034ac80f6a681b06f89e65866ce91f356ed7c72af059c
Status: Downloaded newer image for nginx:latest
afb4a3321e40cd2b384d77013174b5bb352dbfc6f4261ed0f1968b5adc77a52
root@ip-172-31-32-116:/home/ubuntu#
```

i-01f2abdb0d592e12 (Manager)

PublicIPs: 52.195.209.161 PrivateIPs: 172.31.32.116

To check container

**docker ps**

```
root@ip-172-31-32-116:/home/ubuntu# docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
efb4a3321e40	nginx:latest	"/docker-entrypoint..."	About a minute ago	Up About a minute	0.0.0.0:8080->80/tcp, :::8080->80/tcp	container-A

```
root@ip-172-31-32-116:/home/ubuntu#
```

i-01f2abdbdb0d592e12 (Manager)

PublicIPs: 52.195.209.161 PrivateIPs: 172.31.32.116

Let me show you the network by inspect command

**docker inspect <container\_name>**

```
    "ContainerIDFile": "",
    "LogConfig": {
      "Type": "json-file",
      "Config": {}
    },
    "NetworkMode": "bridge-net",
    "PortBindings": {
      "80/tcp": [
        {
          "HostIp": "",
          "HostPort": "8080"
        }
      ]
    },
    "RestartPolicy": {
      "Name": "no",
```

i-01f2abdbdb0d592e12 (Manager)

PublicIPs: 52.195.209.161 PrivateIPs: 172.31.32.116

So it is using our bridge-net only

3. now if a container is already running and we want to attach the network to it for that will use the following commands

First create container with adding --network

```
docker run -d --name container-B -p 80:80 nginx:latest
```

```
root@ip-172-31-32-116:/home/ubuntu# docker run -d --name container-B -p 80:80 nginx:latest
6ae70df2867fe1e0c9cb2a0b8da408df4ed7abb91406ca9c766ba54fdc119705
root@ip-172-31-32-116:/home/ubuntu#
```

i-01f2abdb0d592e12 (Manager)

PublicIPs: 52.195.209.161 PrivateIPs: 172.31.32.116

Now our second container is also created

```
docker ps
```

```
root@ip-172-31-32-116:/home/ubuntu# docker run -d --name container-B -p 80:80 nginx:latest
6ae70df2867fe1e0c9cb2a0b8da408df4ed7abb91406ca9c766ba54fdc119705
root@ip-172-31-32-116:/home/ubuntu# docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS                               NAMES
6ae70df2867f   nginx:latest "/docker-entrypoint..." 21 seconds ago Up 19 seconds 0.0.0.0:80->80/tcp, :::80->80/tcp   container-B
efb4a3321e40   nginx:latest "/docker-entrypoint..." 11 minutes ago Up 11 minutes  0.0.0.0:8080->80/tcp, :::8080->80/tcp container-A
root@ip-172-31-32-116:/home/ubuntu#
```

i-01f2abdb0d592e12 (Manager)

PublicIPs: 52.195.209.161 PrivateIPs: 172.31.32.116

Let's inspect our container to check it's network

So here we don't have any network

```
"IPPrefixLen": 16,
"IPv6Gateway": "",
"MacAddress": "02:42:ac:11:00:02",
"Networks": {
  "bridge": {
    "IPAMConfig": null,
    "Links": null,
    "Aliases": null,
    "NetworkID": "40804148129ba97f3d94e13d30084675c60cb90916a6d463fb9ea1c2cc74c6b3",
    "EndpointID": "966319548b3f56b140ba33894561f5650503037a37ab28f8b42578714608f603",
    "Gateway": "172.17.0.1",
```

4. Now to connect our second container (Container-B) with our bridge-net

```
docker network connect <network_name> <container_name>
```

```
docker inspect container-B
```

```
      "DriverOpts": null
    },
    "bridge-net": {
      "IPAMConfig": {},
      "Links": null,
      "Aliases": [
        "6ae70df2867f"
      ],
      "NetworkID": "b9f60ae33fa4f2f0ff04920b69044481b43254bc9e3aa4601d2771dd697d3445",
      "EndpointID": "159e5b060cac3f306240bb86447ce4983c2d4d09c3fdc98deb23269974a5a8d2",
      "Gateway": "172.19.0.1",
      "IPAddress": "172.19.0.3",
      "IPPrefixLen": 16
    }
  ]
}
```

5. Now we will learn how to Disconnect the network from the container

```
docker network disconnect <network_name> <container_name>
```

```
docker inspect <container_name>
```

```
      "StopSignal": "SIGQUIT"
    },
    "NetworkSettings": {
      "Bridge": "",
      "SandboxID": "d7830203674d58ae5d5f9b5c6fca909bf94aa7e885bf42284e3931f98f1b120f",
      "HairpinMode": false,
      "LinkLocalIPv6Address": "",
      "LinkLocalIPv6PrefixLen": 0,
      "Ports": {},
      "SandboxKey": "/var/run/docker/netns/d7830203674d",
      "SecondaryIPAddresses": null,
      "SecondaryIPv6Addresses": null
    }
  ]
}
```

6. This command removes the bridge network you created

```
docker network rm <network-name>
```

```
root@ip-172-31-32-116:/home/ubuntu# docker network rm bridge-net
Error response from daemon: error while removing network: network bridge-net id b9f60ae33fa4f2f0ff04920b69044481b43254bc9e3aa4601d2771dd697d3445 has active endpoints
root@ip-172-31-32-116:/home/ubuntu#
```

i-01f2abdb0d592e12 (Manager)

PublicIPs: 52.195.209.161 PrivateIPs: 172.31.32.116

So will remove those containers first

This will remove all the containers

```
docker rm -f $(sudo docker ps -a -q)
```

```
root@ip-172-31-32-116:/home/ubuntu# docker rm -f $(sudo docker ps -a -q)
6ae70df2867f
efb4a3321e40
root@ip-172-31-32-116:/home/ubuntu#
```

i-01f2abdb0d592e12 (Manager)

PublicIPs: 52.195.209.161 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
root@ip-172-31-32-116:/home/ubuntu#
```

i-01f2abdb0d592e12 (Manager)

PublicIPs: 52.195.209.161 PrivateIPs: 172.31.32.116

And now we can remove our network

```
docker network rm <network-name>
```

```
root@ip-172-31-32-116:/home/ubuntu# docker network rm bridge-net
bridge-net
root@ip-172-31-32-116:/home/ubuntu#
```

i-01f2abdb0d592e12 (Manager)

PublicIPs: 52.195.209.161 PrivateIPs: 172.31.32.116

And we can verify also by using

**docker network ls**

```
root@ip-172-31-32-116:/home/ubuntu# docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
40804148129b        bridge              bridge              local
e9c8bdda68ec        docker_gwbridge     bridge              local
174394cf6e9c        host                host                local
tqjyxdouwobe        ingress             overlay             swarm
4754405da1a8        none                null                local
root@ip-172-31-32-116:/home/ubuntu#
```

i-01f2abbdb0d592e12 (Manager)

PublicIPs: 52.195.209.161 PrivateIPs: 172.31.32.116