DOCKER NETWORK – DEFAULT BRIDGE NETWORK

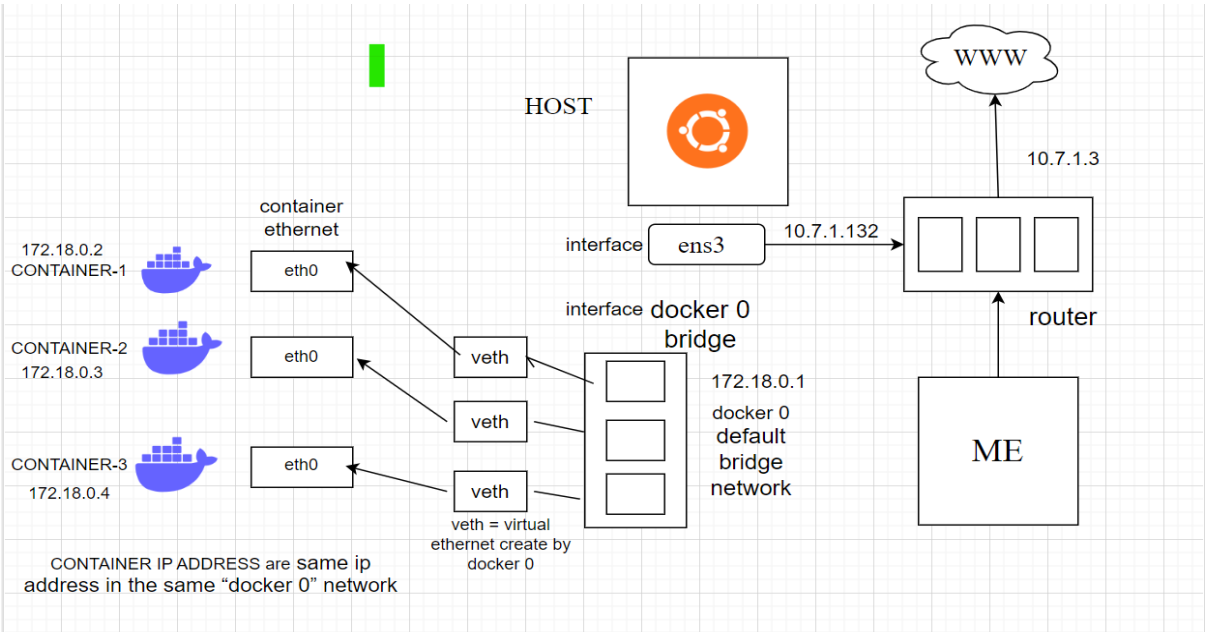
What is Docker Network?

Docker network is a virtual network that allows containers to communicate with each other and with the outside world. Docker provides a networking feature that enables containers to be connected to one or more networks, each with its own unique IP address.

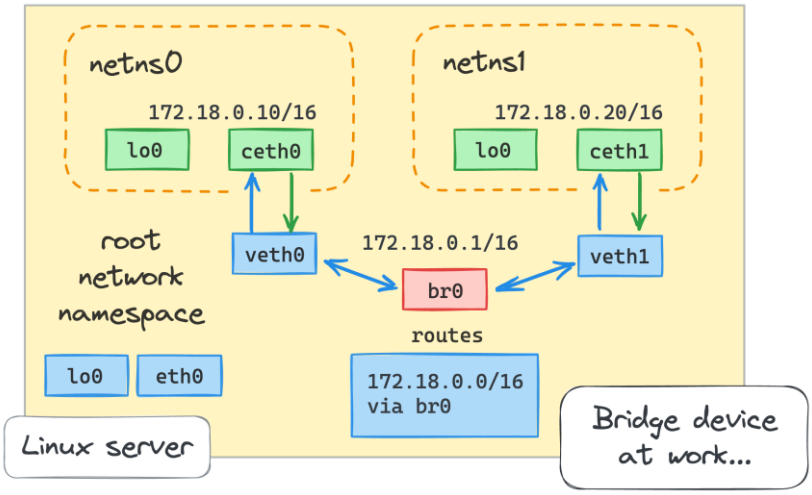
1. **Default Bridge Network:**
   The default bridge network in Docker is a basic network mode created automatically when Docker is installed. It allows containers to communicate with each other and the host using simple network routing.

Characteristics of the Default Bridge Network:

- **Limited Inter-Container Communication**:
  Containers connected to the default bridge network **cannot communicate with each other by name** (i.e., using container names). They can only communicate by IP address unless you link them explicitly.
- **IP Addressing**:
  Each container connected to the default bridge network gets an IP address assigned automatically from the internal Docker subnet range (usually something like 191.18.0.0/24).
- **Isolation from Host**:
  By default, containers on the default bridge network are isolated from the host's network, except for explicitly exposed ports. To allow external access to a container, you need to map ports from the host to the container using the -p or -P option.
- **No DNS Resolution**:
  Containers on the default bridge network cannot resolve each other by container name. You would need to use user-defined networks (such as custom bridge networks) for name-based resolution.
- **Simple Routing**:
  Containers can communicate with each other via IP addresses and ports, but this is basic networking. If more advanced routing is required (e.g., routing to multiple networks), a user-defined network is usually better.



DOCKER NETWORK – DEFAULT BRIDGE NETWORK



Creating the container using default bridge network and manually exposing the port 80:80

```
root@manoj:~#
root@manoj:~# docker run -itd --rm -p 80:80 --name docker_container_1 nginx
8454fb1c2b3d72c36beb6b8e90c70d1df774cffa53d28[redacted]
root@manoj:~#
root@manoj:~#
root@manoj:~# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS         PORTS
   NAMES
8454[redacted]   nginx     "/docker-entrypoint.…"   7 seconds ago   Up 6 seconds   0.0.0.0:80->80/tcp, :::80->80/tcp
   docker_container_1
root@manoj:~#
root@manoj:~#
```

Now we can see that container got default ip address from the docker

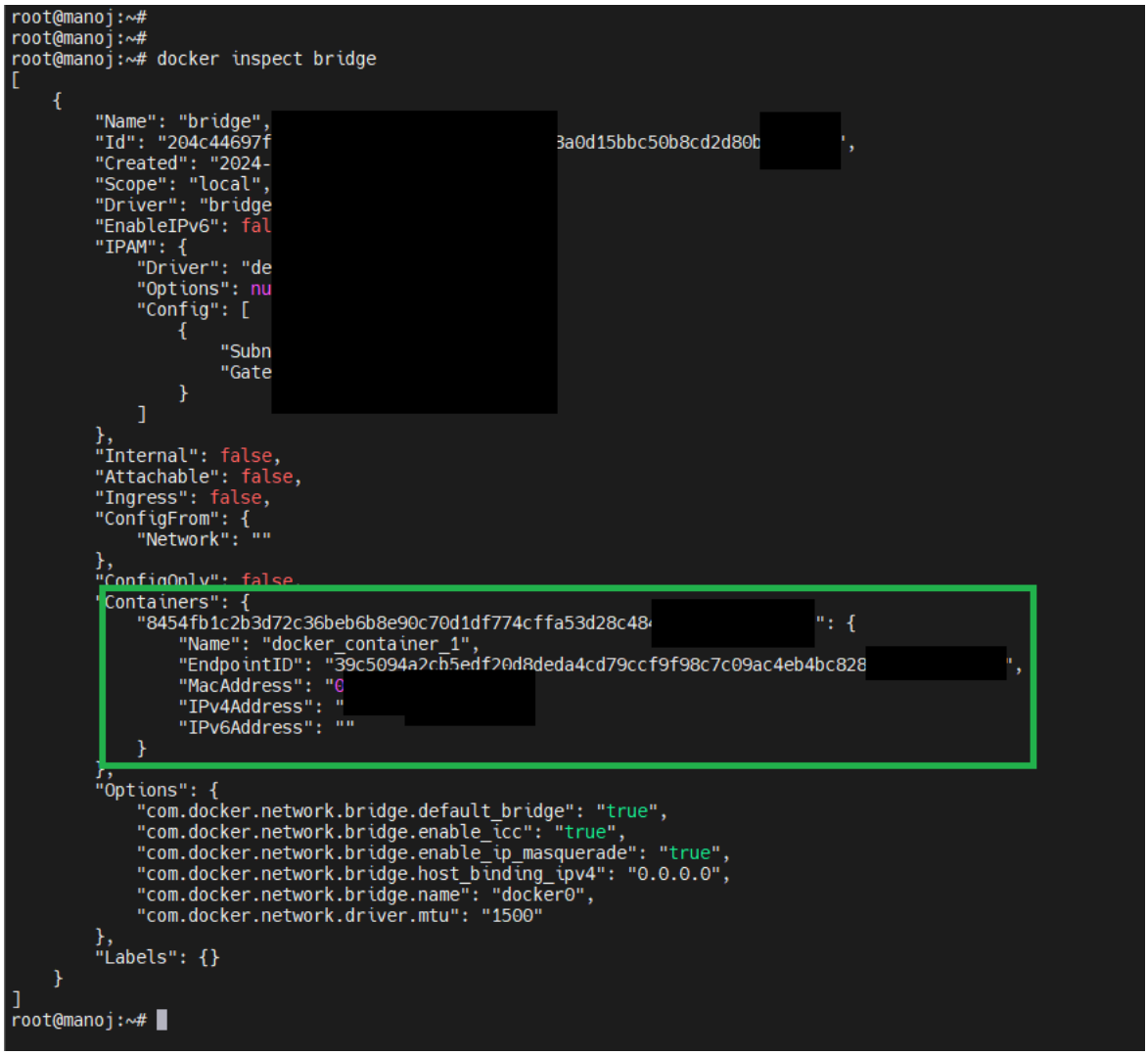
```
root@manoj:~#
root@manoj:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/[redacted]
    inet [redacted]
       va
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: [redacted] <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc pfifo_fast state UP group default qlen 1000
    link/[redacted]
    inet [redacted] cope global dynamic enX0
       va
    inet6 [redacted]
       valid_lft forever preferred_lft forever
3: docker0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/[redacted] ff:ff:ff:ff
    inet [redacted] cope global docker0
       va [redacted] ver
    inet6 [redacted] ink
       valid_lft forever preferred_lft forever
11: veth208d809@if10: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master docker0 state UP group defau
lt
    link/ether 4a:5b:da:5b:7a:36 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet6 fe80::485b:daff:fe5b:7a36/64 scope link
       valid_lft forever preferred_lft forever
root@manoj:~#
```

Bridge link show us containers are connected to “DOCKER 0” i.e is default network

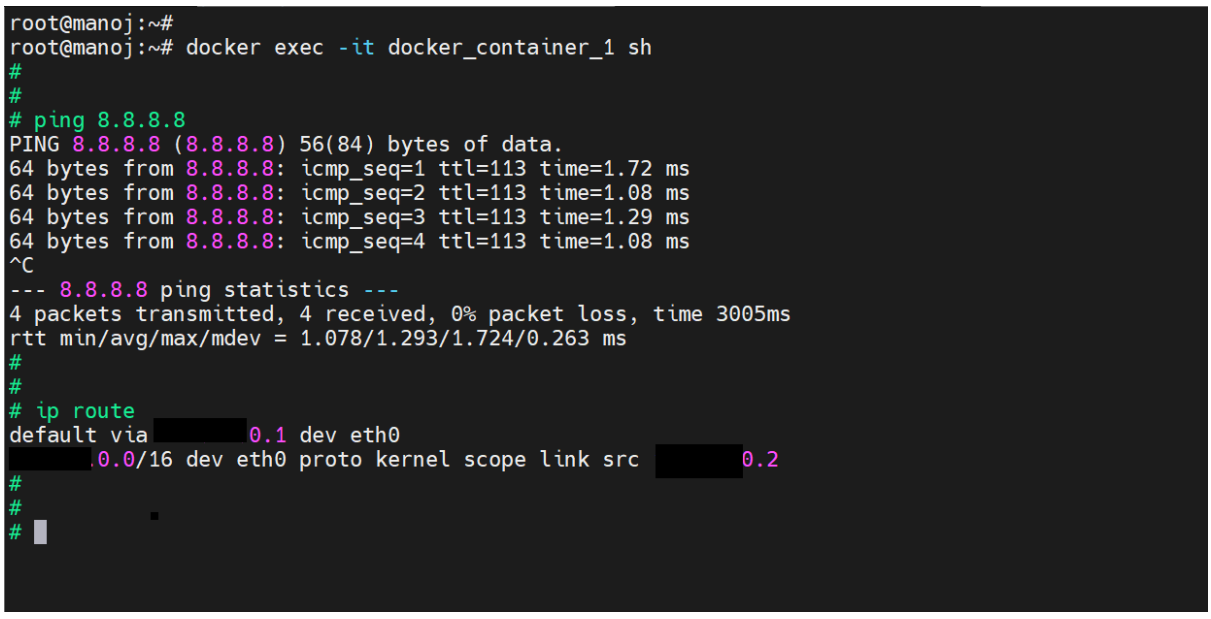
```
root@manoj:~#
root@manoj:~# bridge link
11: veth208d809@if10: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 master docker0 state forwarding priority 32 cost
2
root@manoj:~#
root@manoj:~#
root@manoj:~#
```

DOCKER NETWORK – DEFAULT BRIDGE NETWORK

We can see “docker_container_1” with same ip address in the same “docker 0” network



We can see container can ping the internet



DOCKER NETWORK – DEFAULT BRIDGE NETWORK

Creating another container name “docker_container_2”

```
root@manoj:~#
root@manoj:~# docker run -itd --rm --name docker_container_2 busybox
Unable to find image 'busybox:latest' locally
latest: Pulling from library/busybox
3d1a87f2317d: Pull complete
Digest: sha256:34b191d63fbc93e25e275bfccf1b5365664e5ac28f06d974[redacted]
Status: Downloaded newer image for busybox:latest
16752b030ad74be9eb9d7bf6090681a22fa00ecb71168a9271c[redacted]
root@manoj:~#
root@manoj:~#
root@manoj:~#
```

We can see “docker_container_1 and docker_container_2 ” has same ip address in the same “docker 0” network

```
root@manoj:~#
root@manoj:~# docker inspect bridge
[
    {
        "Name": "bridge",
        "Id": "204c44697f14c299ed435b7daa1f7c8d3b8838a0d15bb[redacted]",
        "Created": "2024-09-09T14:19:33.3[redacted]",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": [
                {
                    "Subnet": "[redacted]/16",
                    "Gateway": "[redacted].1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {
            "16752b030ad74be9eb9d7bf6090681a22fa00ecb71168a9271c13[redacted]
                "Name": "docker_container_2",
                "EndpointID": "4c0a2c0ece0ef16ec7171cdd790a5170700[redacted]",
                "MacAddress": "[redacted]:11:00:03",
                "IPv4Address": "[redacted]0.3/16",
                "IPv6Address": ""
            },
            "8454fb1c2b3d72c36beb6b8e90c70d1df774cffa53d28c484f62d4b6[redacted]
                "Name": "docker_container_1",
                "EndpointID": "39c5094a2cb5edf20d8deda4cd79ccf9f98c7c[redacted]
                "MacAddress": "[redacted]:11:00:02",
                "IPv4Address": "[redacted]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": {}
    }
]
root@manoj:~#
```

DOCKER NETWORK – DEFAULT BRIDGE NETWORK

```
root@manoj:~#
root@manoj:~#
root@manoj:~# docker exec -it docker_container_1 sh
#
#
# ping [redacted]7.0.3
PING [redacted].0.3 (172.17.0.3) 56(84) bytes of data.
64 bytes from [redacted].0.3: icmp_seq=1 ttl=64 time=0.044 ms
64 bytes from [redacted].0.3: icmp_seq=2 ttl=64 time=0.055 ms
64 bytes from [redacted].0.3: icmp_seq=3 ttl=64 time=0.056 ms
64 bytes from [redacted].0.3: icmp_seq=4 ttl=64 time=0.051 ms
64 bytes from [redacted].0.3: icmp_seq=5 ttl=64 time=0.059 ms
64 bytes from [redacted].0.3: icmp_seq=6 ttl=64 time=0.055 ms
64 bytes from [redacted].0.3: icmp_seq=7 ttl=64 time=0.056 ms
^C
--- [redacted].0.3 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6142ms
rtt min/avg/max/mdev = 0.044/0.053/0.059/0.004 ms
#
#
#
```

By EXPOSING the port 80 of the container to port 80 of my host, manually we can see the website

```
root@manoj:~#
root@manoj:~# docker images
REPOSITORY   TAG       IMAGE ID       CREATED         SIZE
nginx        latest    39286[redacted]  3 weeks ago     188MB
busybox      latest    87ff7[redacted]  15 months ago   4.26MB
node         14        1d124[redacted]  17 months ago   912MB
root@manoj:~#
root@manoj:~# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS                                   NAMES
16752b03[redacted]  busybox   "sh"                     2 minutes ago    Up 2 minutes                                            docker_container_2
8454fb1c[redacted]  nginx     "/docker-entrypoint.…"   15 minutes ago   Up 15 minutes   0.0.0.0:80->80/tcp, :::80->80/tcp       docker_container_1
root@manoj:~#
root@manoj:~#
root@manoj:~#
```