Docker Network

1. Bridge Network (default)

- **Use case**: Containers on the same host talking to each other.
- **How it works**: Docker creates a private virtual network on the host machine.
- **Example**: Two containers (`web` and `db`) can communicate using their container names.

Bash

```
docker network create my-bridge
docker run -d --name web --network my-bridge nginx
docker run -d --name db --network my-bridge mysql
```

2. Host Network(Assigns current Instance's or Vm's IP)

- **Use case**: When you want the container to use the host's network directly.
- **How it works**: No network isolation between container and host.
- **Example**: Great for performance or when dealing with custom network tools.

Bash

```
docker run --network host nginx
```

Container uses the same IP and ports as the host.

---

3. None Network (Used for IP Isolation)

- **Use case**: Total network isolation.
- **How it works**: No internet, no communication with other containers.
- **Example**: Useful for testing or very secure workloads.

Bash

```
docker run --network none nginx
```

Container has no network access.

---

4. Overlay Network (for Swarm)

- **Use case**: Communication between containers across multiple Docker hosts.
- **How it works**: Docker sets up a virtual network that spans multiple machines.
- **Example**: Used in Docker Swarm mode for services to talk to each other.

---

5. Macvlan Network

- **Use case**: Give containers their own IP address on the local LAN.
- **How it works**: Bypasses Docker's NAT, gives direct access to your LAN.
- **Example**: For legacy systems or network tools needing real IPs.

Helpful Docker Network Commands

- `docker network ls` → List all networks
- `docker network inspect <network>` → View details of a network
- `docker network create <name>` → Create a custom network
- `docker network connect <network> <container>` → Connect container to a network

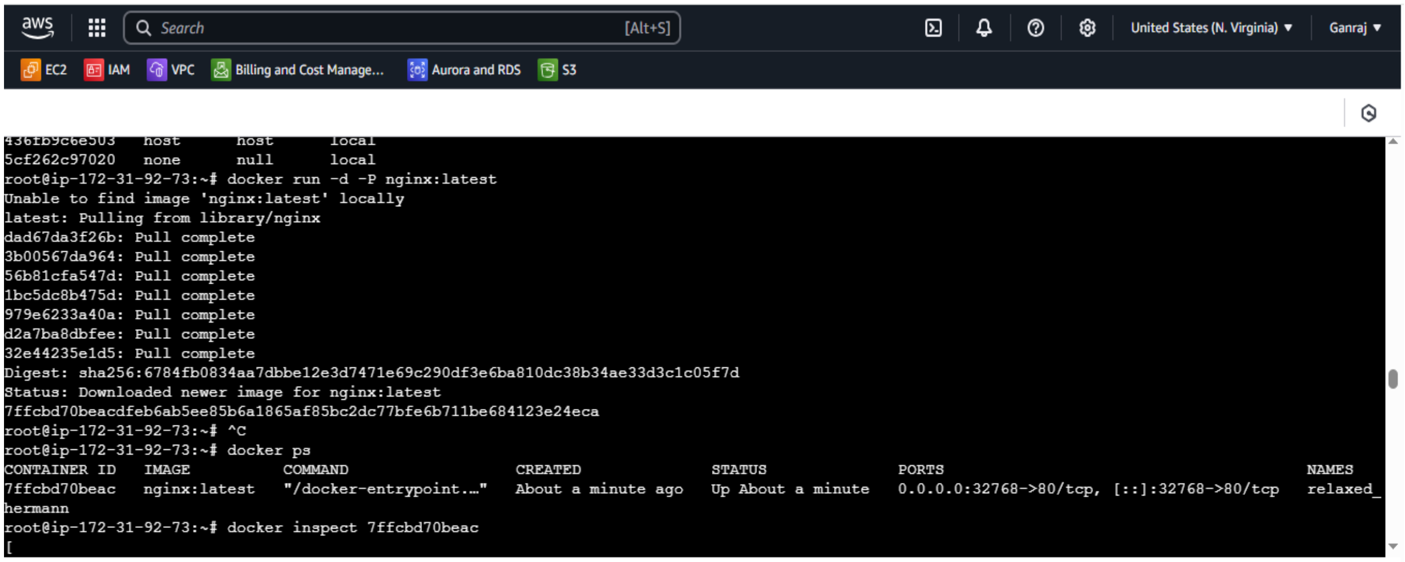
Process Practical : -

I. Bridge.

It is a default network when we create a container without any port or network name mentioned.

docker run -d -P nginx:latest

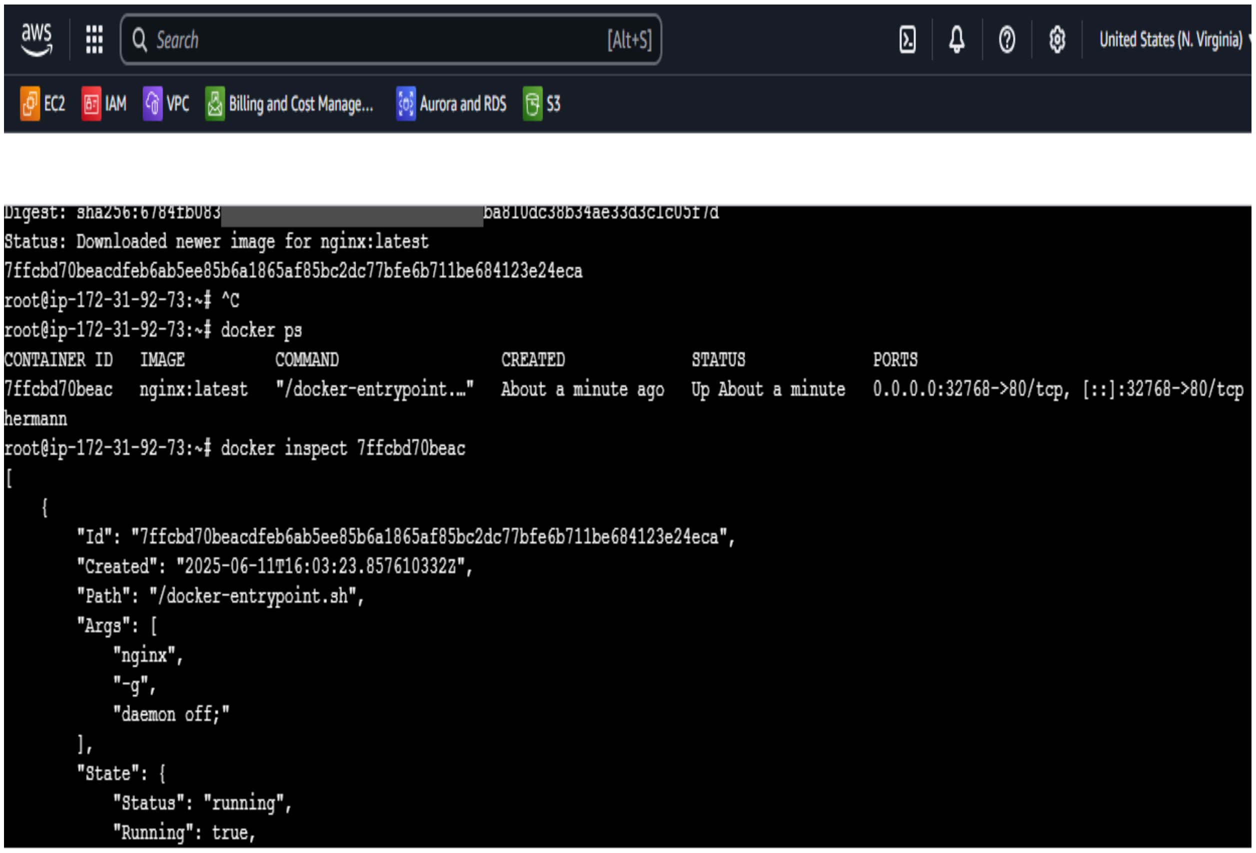
Part	Meaning
`docker run`	Runs a container from an image.
`-d`	Detached mode — runs the container in the background.
`-P`	**Publish all exposed ports** to **random host ports** (dynamic mapping).
`nginx:latest`	Docker image to run (official NGINX image, latest version).



```
436fb9c6e503   host      host      local
5cf262c97020   none      null      local
root@ip-172-31-92-73:~# docker run -d -P nginx:latest
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
dad67da3f26b: Pull complete
3b00567da964: Pull complete
56b81cfa547d: Pull complete
1bc5dc8b475d: Pull complete
979e6233a40a: Pull complete
d2a7ba8dbfee: Pull complete
32e44235e1d5: Pull complete
Digest: sha256:6784fb0834aa7dbbe12e3d7471e69c290df3e6ba810dc38b34ae33d3c1c05f7d
Status: Downloaded newer image for nginx:latest
7ffcbd70beacdfeb6ab5ee85b6a1865af85bc2dc77bfe6b711be684123e24eca
root@ip-172-31-92-73:~# ^C
root@ip-172-31-92-73:~# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED              STATUS              PORTS                                       NAMES
7ffcbd70beac   nginx:latest   "/docker-entrypoint.…"   About a minute ago   Up About a minute   0.0.0.0:32768->80/tcp, [::]:32768->80/tcp   relaxed_
hermann
root@ip-172-31-92-73:~# docker inspect 7ffcbd70beac
[
```

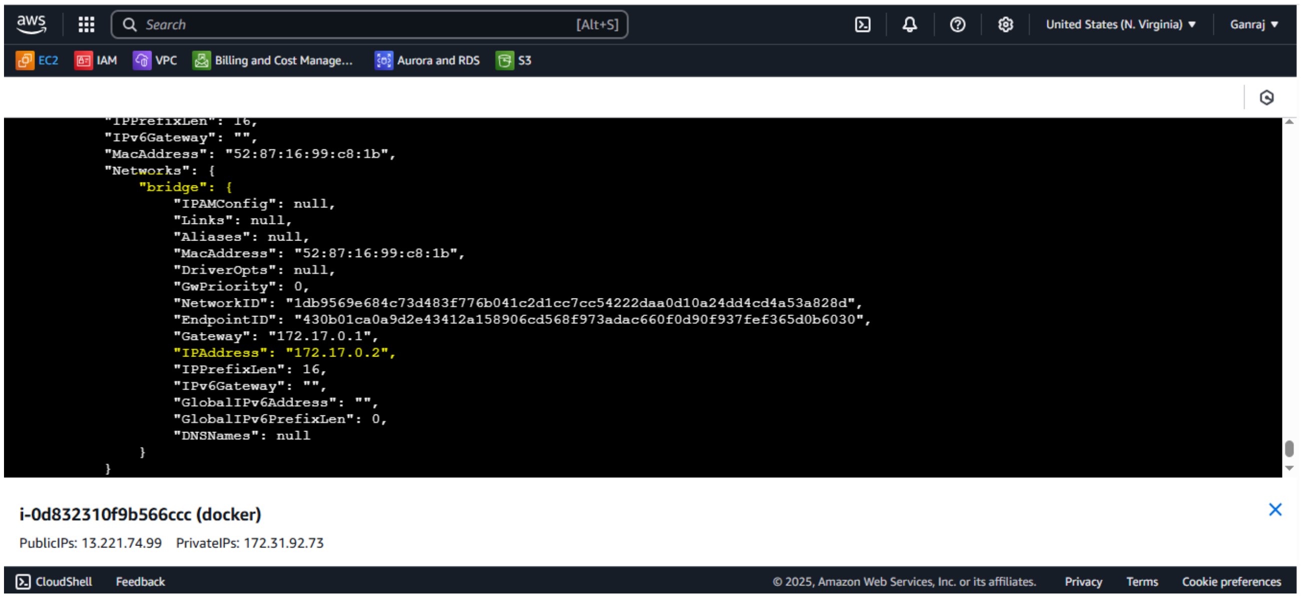
docker inspect 7ffcbd70beac

Using this command we can check every details on that container.



```
Digest: sha256:6784fb083[redacted]ba810dc38b34ae33d3c1c05f7d
Status: Downloaded newer image for nginx:latest
7ffcbd70beacdfeb6ab5ee85b6a1865af85bc2dc77bfe6b711be684123e24eca
root@ip-172-31-92-73:~# ^C
root@ip-172-31-92-73:~# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED              STATUS              PORTS
7ffcbd70beac   nginx:latest   "/docker-entrypoint.…"   About a minute ago   Up About a minute   0.0.0.0:32768->80/tcp, [::]:32768->80/tcp
hermann
root@ip-172-31-92-73:~# docker inspect 7ffcbd70beac
[
    {
        "Id": "7ffcbd70beacdfeb6ab5ee85b6a1865af85bc2dc77bfe6b711be684123e24eca",
        "Created": "2025-06-11T16:03:23.857610332Z",
        "Path": "/docker-entrypoint.sh",
        "Args": [
            "nginx",
            "-g",
            "daemon off;"
        ],
        "State": {
            "Status": "running",
            "Running": true,
```

As you can see in below snap the container created in default bridge network using his own docker engine vpc to assign a IP



```
        "IPPrefixLen": 16,
        "IPv6Gateway": "",
        "MacAddress": "52:87:16:99:c8:1b",
        "Networks": {
            "bridge": {
                "IPAMConfig": null,
                "Links": null,
                "Aliases": null,
                "MacAddress": "52:87:16:99:c8:1b",
                "DriverOpts": null,
                "GwPriority": 0,
                "NetworkID": "1db9569e684c73d483f776b041c2d1cc7cc54222daa0d10a24dd4cd4a53a828d",
                "EndpointID": "430b01ca0a9d2e43412a158906cd568f973adac660f0d90f937fef365d0b6030",
                "Gateway": "172.17.0.1",
                "IPAddress": "172.17.0.2",
                "IPPrefixLen": 16,
                "IPv6Gateway": "",
                "GlobalIPv6Address": "",
                "GlobalIPv6PrefixLen": 0,
                "DNSNames": null
            }
        }
```

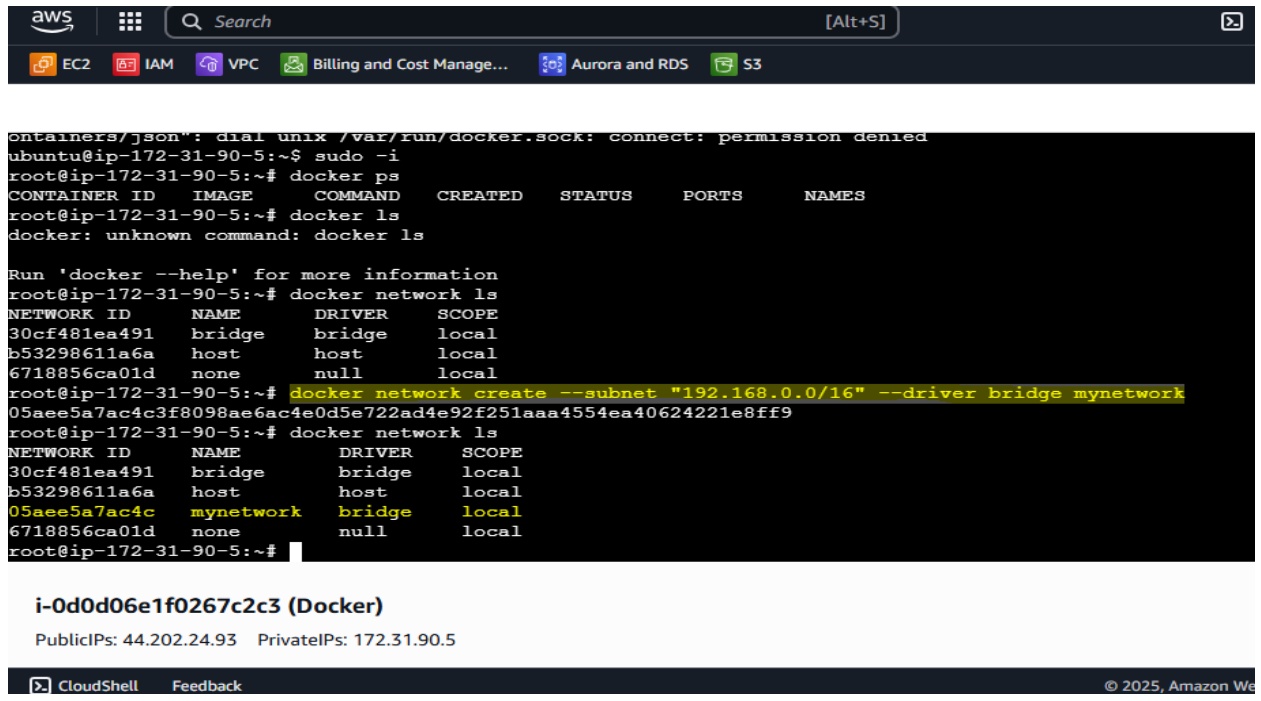
i-0d832310f9b566ccc (docker)

PublicIPs: 13.221.74.99 PrivateIPs: 172.31.92.73

Now see how to create own name bridge driver type network connection which is woek like a VPC will assigns IPs fro m that subnets . So with same Network we will creating containers

docker network create --subnet "192.168.0.0/16" --driver bridge mynetwork

Part	Explanation
`docker network create`	Tells Docker to create a new network
`--subnet "192.168.0.0/16"`	Defines a custom IP address range (subnet) for containers inside this network
`--driver bridge`	Specifies that this is a **bridge** network (default network type for container communication on the same host)
`mynetwork`	The name you're giving to this new network



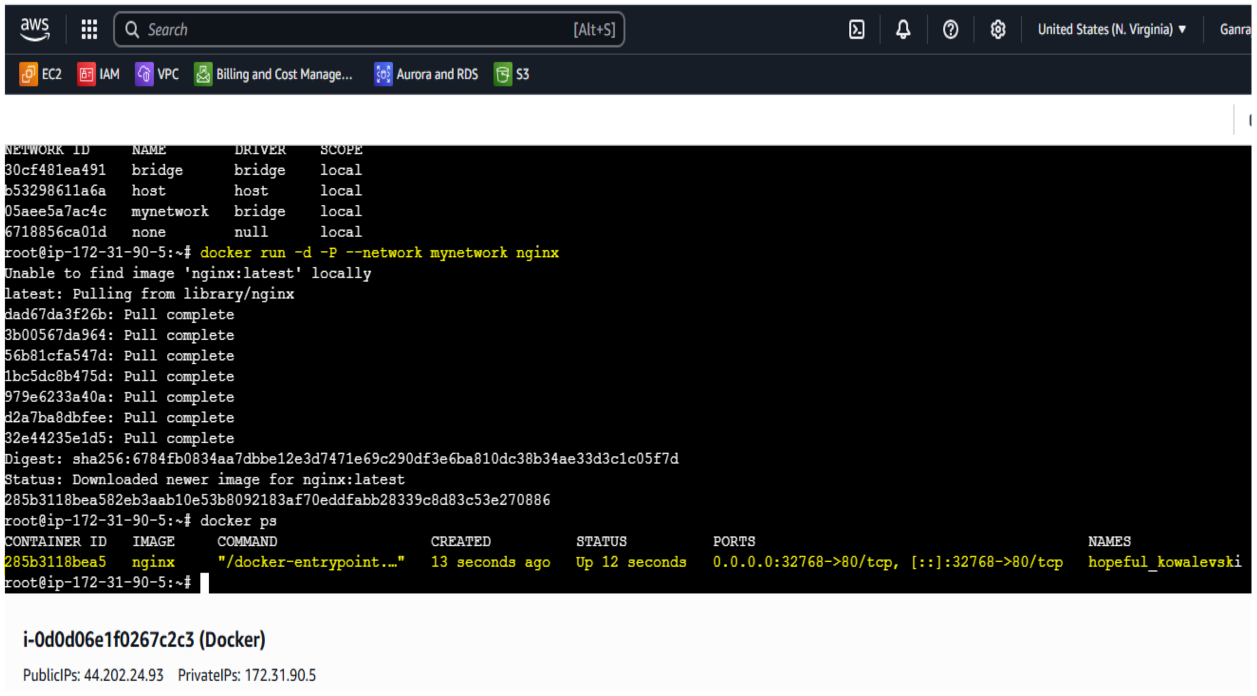
```
ontainers/json": dial unix /var/run/docker.sock: connect: permission denied
ubuntu@ip-172-31-90-5:~$ sudo -i
root@ip-172-31-90-5:~# docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
root@ip-172-31-90-5:~# docker ls
docker: unknown command: docker ls

Run 'docker --help' for more information
root@ip-172-31-90-5:~# docker network ls
NETWORK ID     NAME      DRIVER    SCOPE
30cf481ea491   bridge    bridge    local
b53298611a6a   host      host      local
6718856ca01d   none      null      local
root@ip-172-31-90-5:~# docker network create --subnet "192.168.0.0/16" --driver bridge mynetwork
05aee5a7ac4c3f8098ae6ac4e0d5e722ad4e92f251aaa4554ea40624221e8ff9
root@ip-172-31-90-5:~# docker network ls
NETWORK ID     NAME        DRIVER    SCOPE
30cf481ea491   bridge      bridge    local
b53298611a6a   host        host      local
05aee5a7ac4c   mynetwork   bridge    local
6718856ca01d   none        null      local
root@ip-172-31-90-5:~# 
```

i-0d0d06e1f0267c2c3 (Docker)

PublicIPs: 44.202.24.93 PrivateIPs: 172.31.90.5

Now with that network (mynetwork) creating a docker container nginx

docker run -d -P --network mynetwork nginx

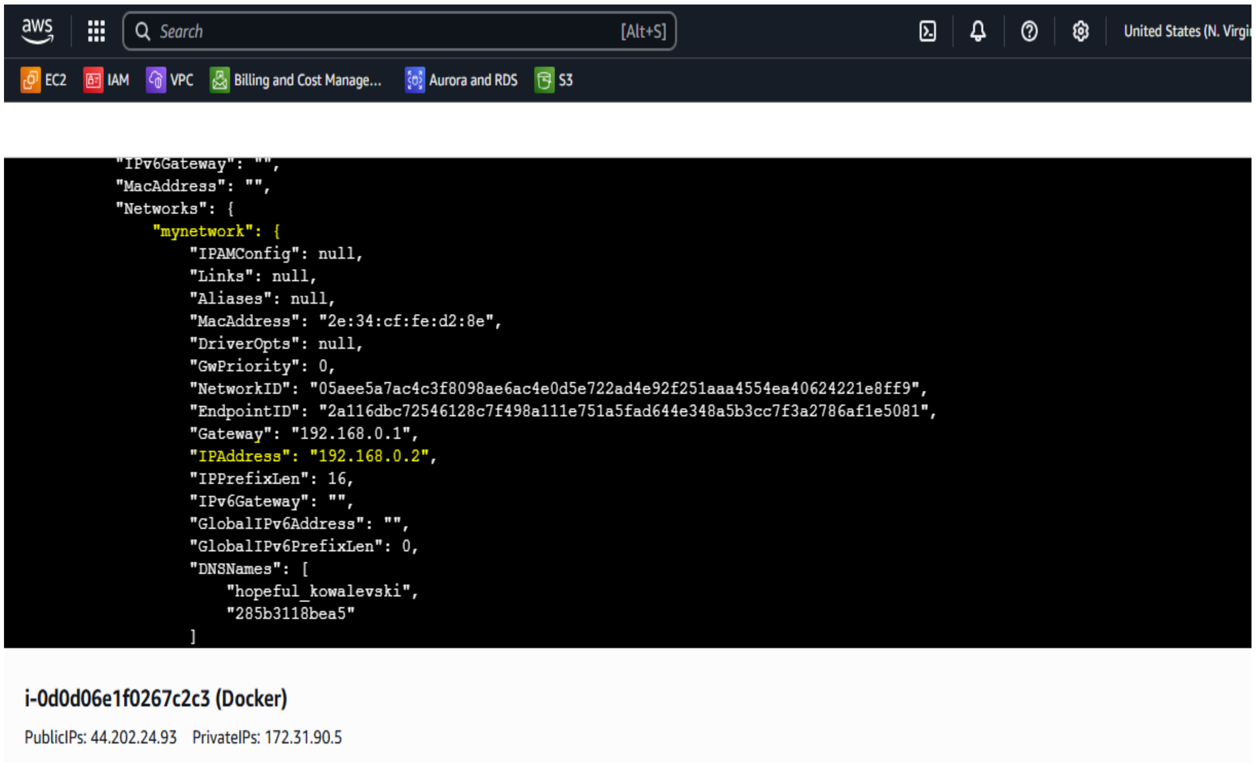


```
NETWORK ID     NAME        DRIVER    SCOPE
30cf481ea491   bridge      bridge    local
b53298611a6a   host        host      local
05aee5a7ac4c   mynetwork   bridge    local
6718856ca01d   none        null      local
root@ip-172-31-90-5:~# docker run -d -P --network mynetwork nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
dad67da3f26b: Pull complete
3b00567da964: Pull complete
56b81cfa547d: Pull complete
1bc5dc8b475d: Pull complete
979e6233a40a: Pull complete
d2a7ba8dbfee: Pull complete
32e44235e1d5: Pull complete
Digest: sha256:6784fb0834aa7dbbe12e3d7471e69c290df3e6ba810dc38b34ae33d3c1c05f7d
Status: Downloaded newer image for nginx:latest
285b3118bea582eb3aab10e53b8092183af70eddfabb28339c8d83c53e270886
root@ip-172-31-90-5:~# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS                                     NAMES
285b3118bea5   nginx     "/docker-entrypoint.…"   13 seconds ago   Up 12 seconds   0.0.0.0:32768->80/tcp, [::]:32768->80/tcp   hopeful_kowalevski
root@ip-172-31-90-5:~# 
```

i-0d0d06e1f0267c2c3 (Docker)

PublicIPs: 44.202.24.93 PrivateIPs: 172.31.90.5

As you can see in snap container taken our network (mynetwork)(bridge)assigned ip from it.



```
        "IPv6Gateway": "",
        "MacAddress": "",
        "Networks": {
            "mynetwork": {
                "IPAMConfig": null,
                "Links": null,
                "Aliases": null,
                "MacAddress": "2e:34:cf:fe:d2:8e",
                "DriverOpts": null,
                "GwPriority": 0,
                "NetworkID": "05aee5a7ac4c3f8098ae6ac4e0d5e722ad4e92f251aaa4554ea40624221e8ff9",
                "EndpointID": "2a116dbc72546128c7f498a111e751a5fad644e348a5b3cc7f3a2786af1e5081",
                "Gateway": "192.168.0.1",
                "IPAddress": "192.168.0.2",
                "IPPrefixLen": 16,
                "IPv6Gateway": "",
                "GlobalIPv6Address": "",
                "GlobalIPv6PrefixLen": 0,
                "DNSNames": [
                    "hopeful_kowalevski",
                    "285b3118bea5"
                ]
```

i-0d0d06e1f0267c2c3 (Docker)

PublicIPs: 44.202.24.93 PrivateIPs: 172.31.90.5

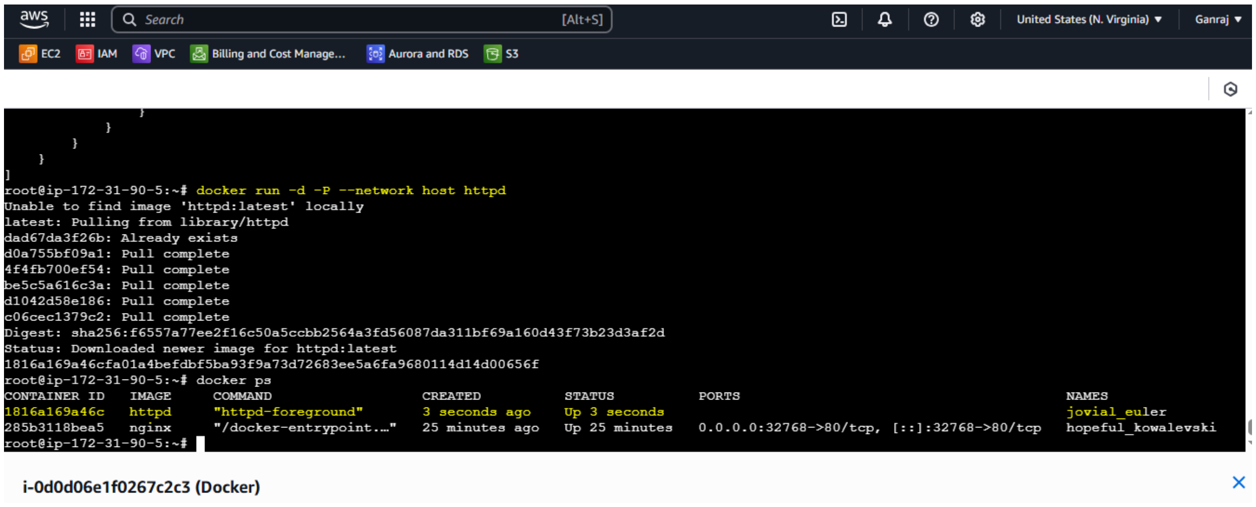
II.HOST

docker run -d -P --network host httpd

Part	Meaning
`docker run`	Starts a new container.
`-d`	Detached mode – runs the container in the background.
`-P`	Publishes all exposed container ports to random ports on the host (only works with default bridge network).
`--network host`	Uses the **host network**, meaning the container shares the host's network stack directly.
`httpd`	The image to use, in this case, Apache HTTP Server

As you can see in below snap when we use host driver network while creating container it creates container with host's (EC2, Vm's) IP in the Snap 2 and it not exposed in ps command you need inspect docker

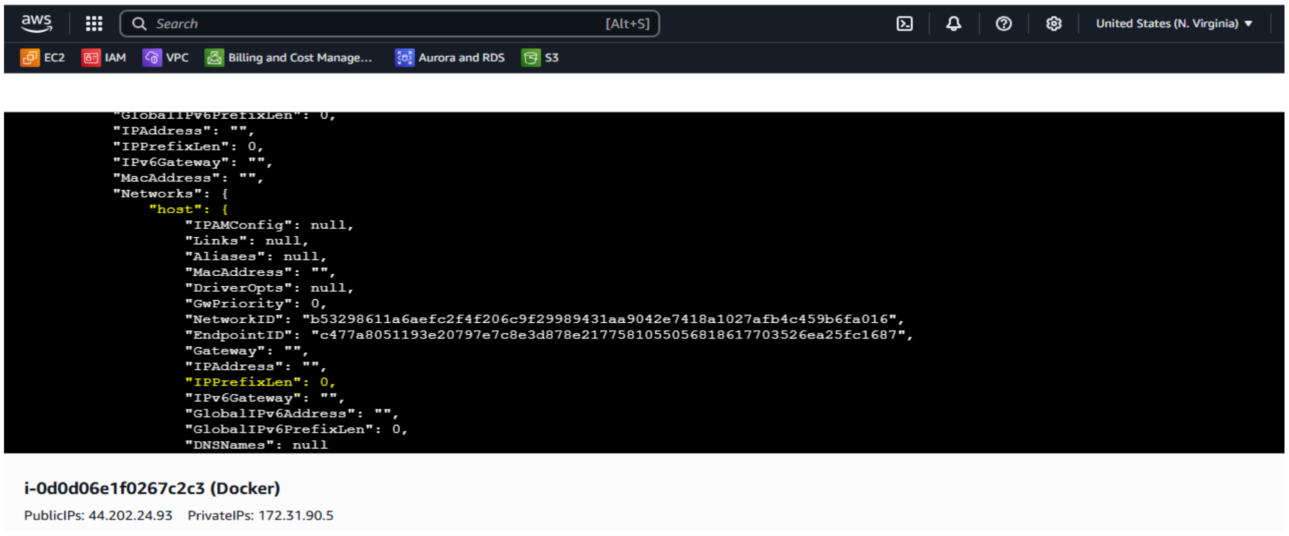
1.



```
                }
            }
        }
    }
]
root@ip-172-31-90-5:~# docker run -d -P --network host httpd
Unable to find image 'httpd:latest' locally
latest: Pulling from library/httpd
dad67da3f26b: Already exists
d0a755bf09a1: Pull complete
4f4fb700ef54: Pull complete
be5c5a616c3a: Pull complete
d1042d58e186: Pull complete
c06cec1379c2: Pull complete
Digest: sha256:f6557a77ee2f16c50a5ccbb2564a3fd56087da311bf69a160d43f73b23d3af2d
Status: Downloaded newer image for httpd:latest
1816a169a46cfa01a4befdbf5ba93f9a73d72683ee5a6fa9680114d14d00656f
root@ip-172-31-90-5:~# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS                                     NAMES
1816a169a46c   httpd     "httpd-foreground"       3 seconds ago    Up 3 seconds                                              jovial_euler
285b3118bea5   nginx     "/docker-entrypoint.…"   25 minutes ago   Up 25 minutes   0.0.0.0:32768->80/tcp, [::]:32768->80/tcp   hopeful_kowalevski
root@ip-172-31-90-5:~#
```

i-0d0d06e1f0267c2c3 (Docker)

2.

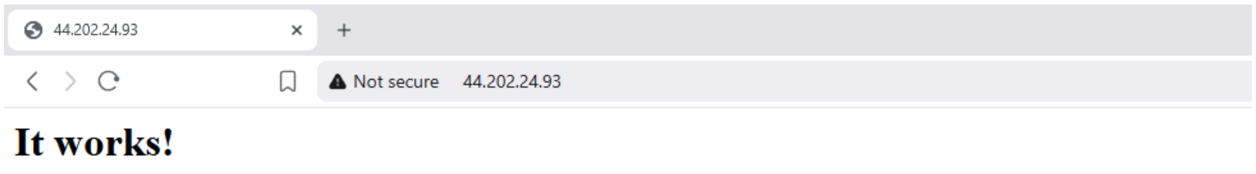


```
            "GlobalIPv6PrefixLen": 0,
            "IPAddress": "",
            "IPPrefixLen": 0,
            "IPv6Gateway": "",
            "MacAddress": "",
            "Networks": {
                "host": {
                    "IPAMConfig": null,
                    "Links": null,
                    "Aliases": null,
                    "MacAddress": "",
                    "DriverOpts": null,
                    "GwPriority": 0,
                    "NetworkID": "b53298611a6aefc2f4f206c9f29989431aa9042e7418a1027afb4c459b6fa016",
                    "EndpointID": "c477a8051193e20797e7c8e3d878e2177581055056818617703526ea25fc1687",
                    "Gateway": "",
                    "IPAddress": "",
                    "IPPrefixLen": 0,
                    "IPv6Gateway": "",
                    "GlobalIPv6Address": "",
                    "GlobalIPv6PrefixLen": 0,
                    "DNSNames": null
```

i-0d0d06e1f0267c2c3 (Docker)

PublicIPs: 44.202.24.93   PrivateIPs: 172.31.90.5

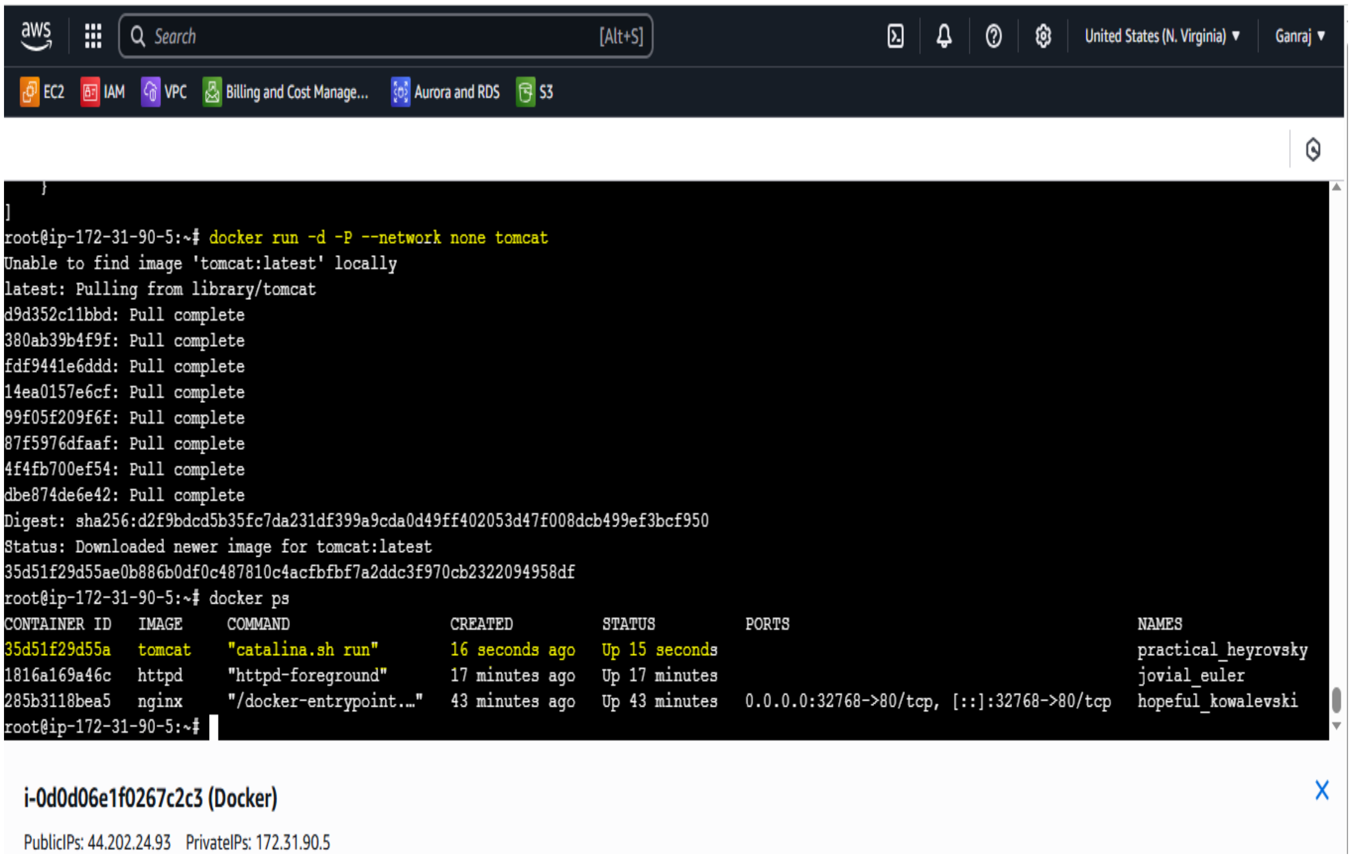
3. when we access our public ip of EC2's(vm) it works because in host driver network takes our EC2's public IP



It works!

III.NONE

docker run -d -P --network none tomcat



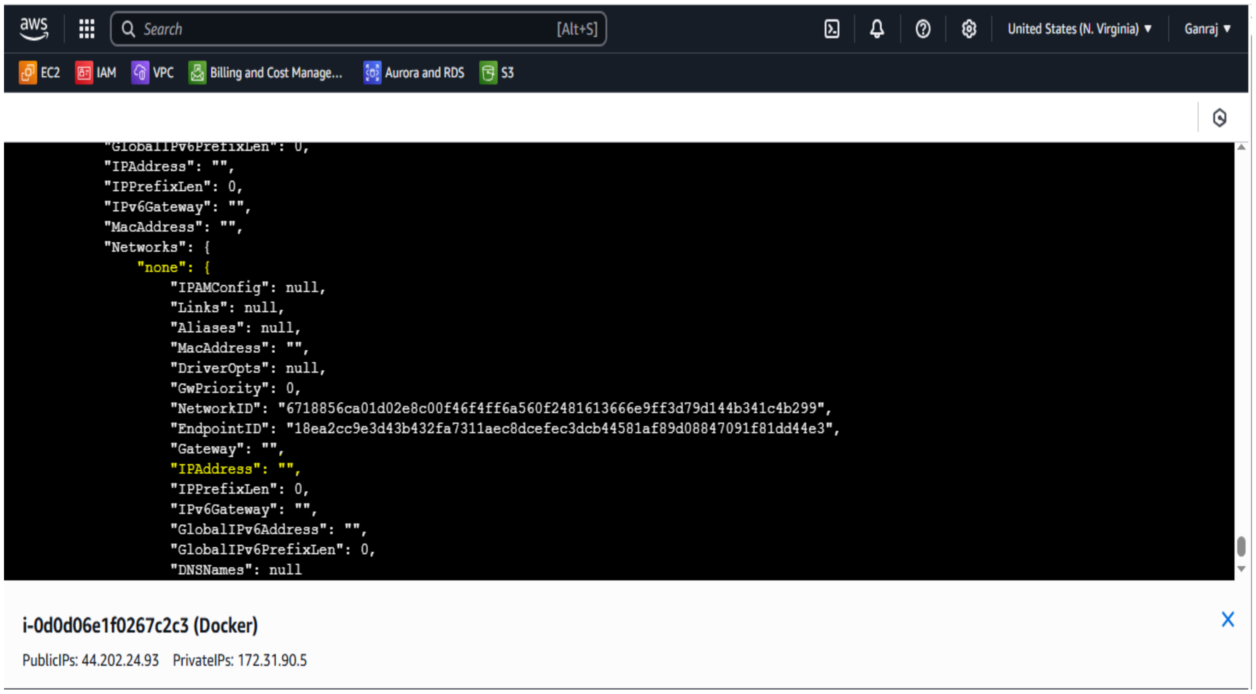
```
   }
]
root@ip-172-31-90-5:~# docker run -d -P --network none tomcat
Unable to find image 'tomcat:latest' locally
latest: Pulling from library/tomcat
d9d352c11bbd: Pull complete
380ab39b4f9f: Pull complete
fdf9441e6ddd: Pull complete
14ea0157e6cf: Pull complete
99f05f209f6f: Pull complete
87f5976dfaaf: Pull complete
4f4fb700ef54: Pull complete
dbe874de6e42: Pull complete
Digest: sha256:d2f9bdcd5b35fc7da231df399a9cda0d49ff402053d47f008dcb499ef3bcf950
Status: Downloaded newer image for tomcat:latest
35d51f29d55ae0b886b0df0c487810c4acfbfbf7a2ddc3f970cb2322094958df
root@ip-172-31-90-5:~# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS                                   NAMES
35d51f29d55a   tomcat    "catalina.sh run"        16 seconds ago   Up 15 seconds                                           practical_heyrovsky
1816a169a46c   httpd     "httpd-foreground"       17 minutes ago   Up 17 minutes                                           jovial_euler
285b3118bea5   nginx     "/docker-entrypoint.…"   43 minutes ago   Up 43 minutes   0.0.0.0:32768->80/tcp, [::]:32768->80/tcp   hopeful_kowalevski
root@ip-172-31-90-5:~#
```

i-0d0d06e1f0267c2c3 (Docker)

PublicIPs: 44.202.24.93   PrivateIPs: 172.31.90.5

Below screeshot shows none driver network is Isolated which means no ip assigned in it.

It is only used for backup process(database)



```
            "GlobalIPv6PrefixLen": 0,
            "IPAddress": "",
            "IPPrefixLen": 0,
            "IPv6Gateway": "",
            "MacAddress": "",
            "Networks": {
                "none": {
                    "IPAMConfig": null,
                    "Links": null,
                    "Aliases": null,
                    "MacAddress": "",
                    "DriverOpts": null,
                    "GwPriority": 0,
                    "NetworkID": "6718856ca01d02e8c00f46f4ff6a560f2481613666e9ff3d79d144b341c4b299",
                    "EndpointID": "18ea2cc9e3d43b432fa7311aec8dcefec3dcb44581af89d08847091f81dd44e3",
                    "Gateway": "",
                    "IPAddress": "",
                    "IPPrefixLen": 0,
                    "IPv6Gateway": "",
                    "GlobalIPv6Address": "",
                    "GlobalIPv6PrefixLen": 0,
                    "DNSNames": null
```

i-0d0d06e1f0267c2c3 (Docker)

PublicIPs: 44.202.24.93 PrivateIPs: 172.31.90.5