

Host and None Network commands

1. This command is used to create a host network

```
docker run -d --network host --name <network-name>
```

```
docker ps
```

```
ubuntu@ip-172-31-32-116:~$ sudo su
root@ip-172-31-32-116:/home/ubuntu# docker run -it -d --name container-A --network host nginx
5fa2bab36f3b59c1bdc99a6e46f3742e8f2e515673028167ff7ac0903ee624d0
root@ip-172-31-32-116:/home/ubuntu# docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
5fa2bab36f3b	nginx	"/docker-entrypoint..."	8 seconds ago	Up 8 seconds		container-A
c55dc138882f	nginx:latest	"/docker-entrypoint..."	17 minutes ago	Up 17 minutes	80/tcp	service-B.1.ewi7fat9i9gl9jo0i3ocqww3z
61059e3e9724	nginx:latest	"/docker-entrypoint..."	17 minutes ago	Up 17 minutes	80/tcp	service-A.2.dmkgn9fgmx8d6pb0ak2b6y6n7
dd9e9e4d0634	nginx:latest	"/docker-entrypoint..."	17 minutes ago	Up 17 minutes	80/tcp	service-c.1.lcg68j3x0czahjfaln25rhoyh

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

i-01f2abbdb0d592e12 (Manager)
PublicIPs: 54.248.6.222 PrivateIPs: 172.31.32.116

2. To check if the host network is working or not we used nginx as the image for the container, which means that if the container is connected to docker's host then we should get nginx welcome page when we curl the localhost as it maps directly to localhost.

```
curl localhost
```

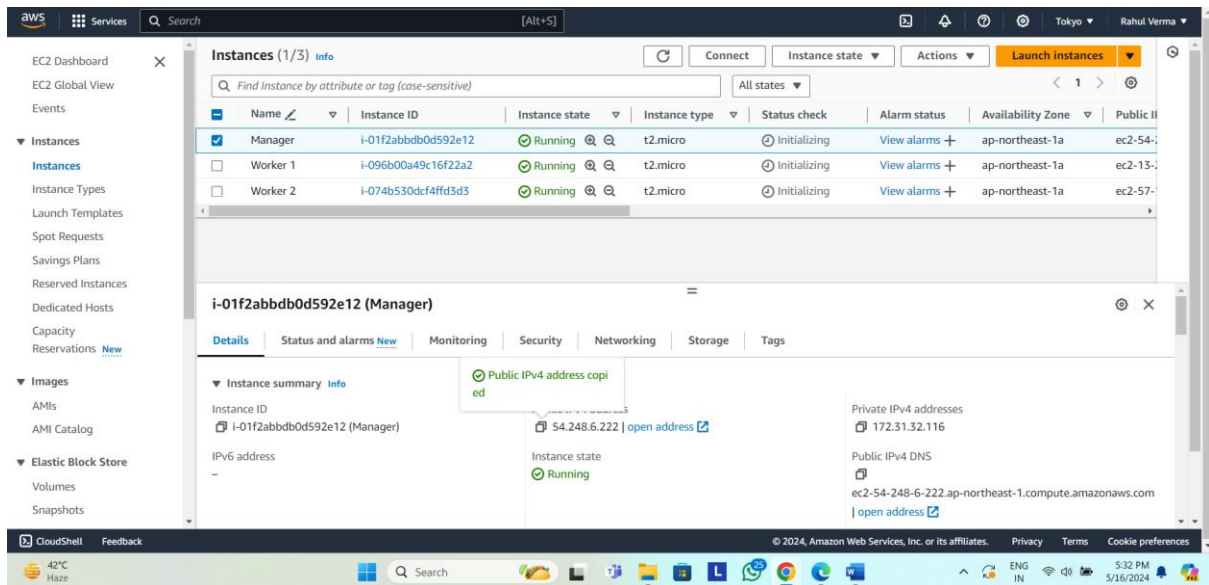
```
aws Services Search [Alt+S]
curl: (6) Could not resolve host: local
curl: (6) Could not resolve host: host
root@ip-172-31-32-116:/home/ubuntu# curl localhost
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
root@ip-172-31-32-116:/home/ubuntu#
```

i-01f2abbdb0d592e12 (Manager)
PublicIPs: 54.248.6.222 PrivateIPs: 172.31.32.116

Copy public ip address

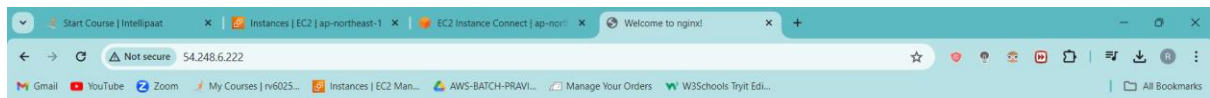


The screenshot shows the AWS Management Console interface. On the left, the navigation menu includes 'Instances', 'Images', and 'Elastic Block Store'. The main panel displays a list of EC2 instances under the 'Instances (1/3)' tab. The instances are:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP
Manager	i-01f2abdb0d592e12	Running	t2.micro	Initializing	View alarms +	ap-northeast-1a	ec2-54-248-6-222
Worker 1	i-096b00a49c16f22a2	Running	t2.micro	Initializing	View alarms +	ap-northeast-1a	ec2-13-54-248-6-222
Worker 2	i-074b530dcf4ffd3d3	Running	t2.micro	Initializing	View alarms +	ap-northeast-1a	ec2-57-248-6-222

The 'Manager' instance is selected, and its details are shown in the 'Details' tab. The 'Instance summary' section shows the instance ID, name, and state. The 'Public IPv4 address' is highlighted and copied, with a tooltip indicating 'Public IPv4 address copied'. The address is 54.248.6.222. Other details include the private IPv4 address (172.31.32.116) and the public IPv4 DNS (ec2-54-248-6-222.ap-northeast-1.compute.amazonaws.com).

Paste it in web browser



The screenshot shows a web browser window with the address bar displaying '54.248.6.222'. The page content is a 'Welcome to nginx!' message, indicating that the nginx web server is successfully installed and working. The message includes instructions for further configuration and links to online documentation and support.

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.



The screenshot shows the Windows taskbar at the bottom of the screen. The system clock displays '5:33 PM' and '5/16/2024'. The network status icon shows 'ENG IN' and '54.248.6.222'.

NONE NETWORK COMMAND

1. This command will completely disable all networking capabilities of this container and it won't have any connections when it starts.

```
docker run -dit --network none --name <network-name> <image>
```

```
root@ip-172-31-31-183: /home/ubuntu
root@ip-172-31-31-183:/home/ubuntu# docker run -dit --network none --name no-net-alpine alpine:latest
Unable to find image 'alpine:latest' locally
latest: Pulling from library/alpine
89d9c30c1d48: Pull complete
Digest: sha256:c19173c5ada610a5989151111163d28a67368362762534d8a8121ce95cf2bd5a
Status: Downloaded newer image for alpine:latest
d0fc86ca9e5f2911a5dcadbb21593dc55c02f7fed8fed07134396f23d0380993
root@ip-172-31-31-183:/home/ubuntu#
```

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

```
docker exec <network-name> ip route
```

```
root@ip-172-31-31-183: /home/ubuntu
root@ip-172-31-31-183:/home/ubuntu# docker exec no-net-alpine ip route
root@ip-172-31-31-183:/home/ubuntu#
```