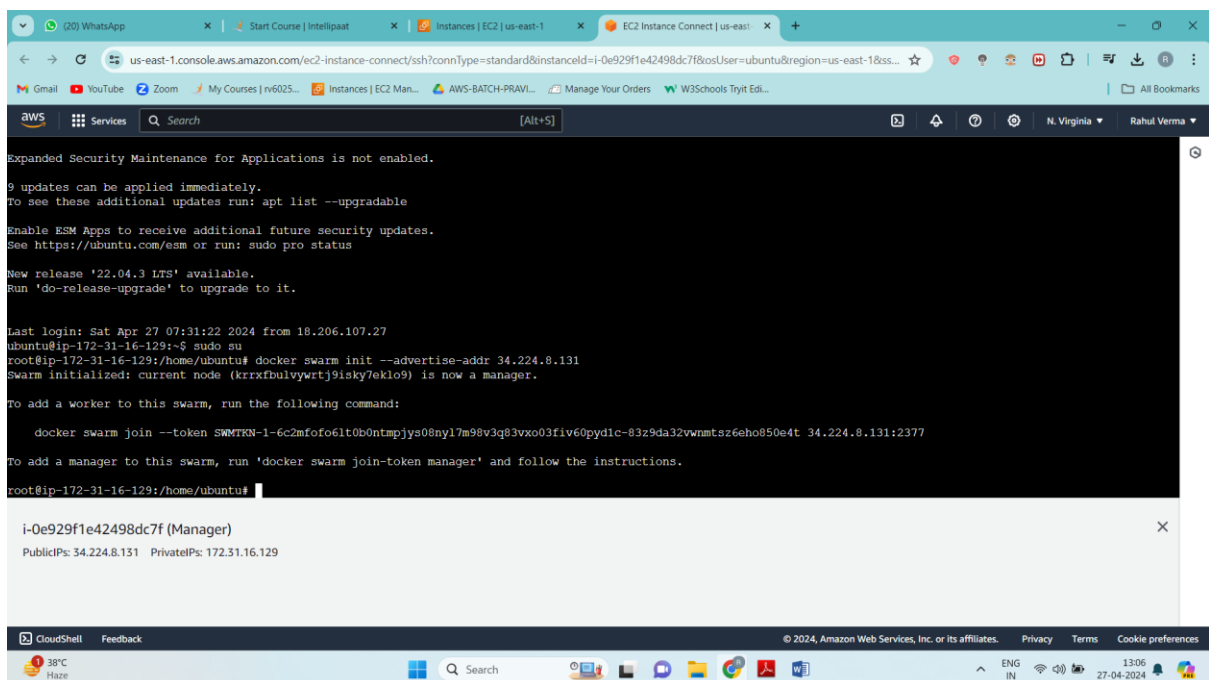


Initializing a Docker Swarm

1: First create the swarm on the Main Node (Instance) by using the below command. This command is used to initialize a swarm with a manager.

```
docker swarm init --advertise-addr <manager-ip>
```



The screenshot shows a terminal window with the following output:

```
Expanded Security Maintenance for Applications is not enabled.
9 updates can be applied immediately.
To see these additional updates run: apt list --upgradable
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status
New release '22.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Sat Apr 27 07:31:22 2024 from 18.206.107.27
ubuntu@ip-172-31-16-129:~$ sudo su
root@ip-172-31-16-129:/home/ubuntu# docker swarm init --advertise-addr 34.224.8.131
Swarm initialized: current node (krrxfbulvywrtj9isky7eklo9) is now a manager.

To add a worker to this swarm, run the following command:

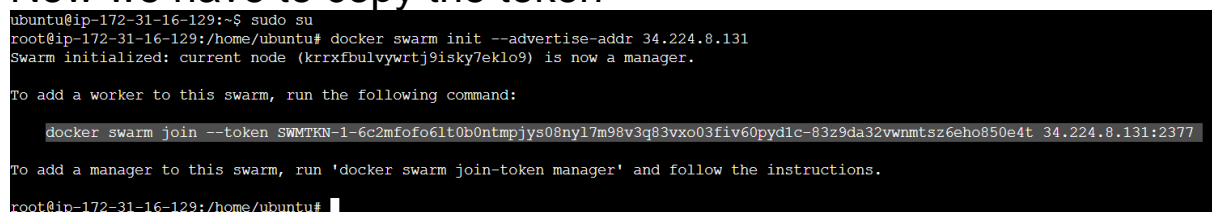
    docker swarm join --token SWMTKN-1-6c2mfofo6lt0b0ntmpjys08nyl7m98v3q83vxo03fiv60pydlc-83z9da32vwmmtsz6eho850e4t 34.224.8.131:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.
root@ip-172-31-16-129:/home/ubuntu#
```

A pop-up window displays the manager's token and IP addresses:

```
i-Oe929f1e42498dc7f (Manager)
PublicIPs: 34.224.8.131 PrivateIPs: 172.31.16.129
```

Now we have to copy the token



The screenshot shows a terminal window with the following output:

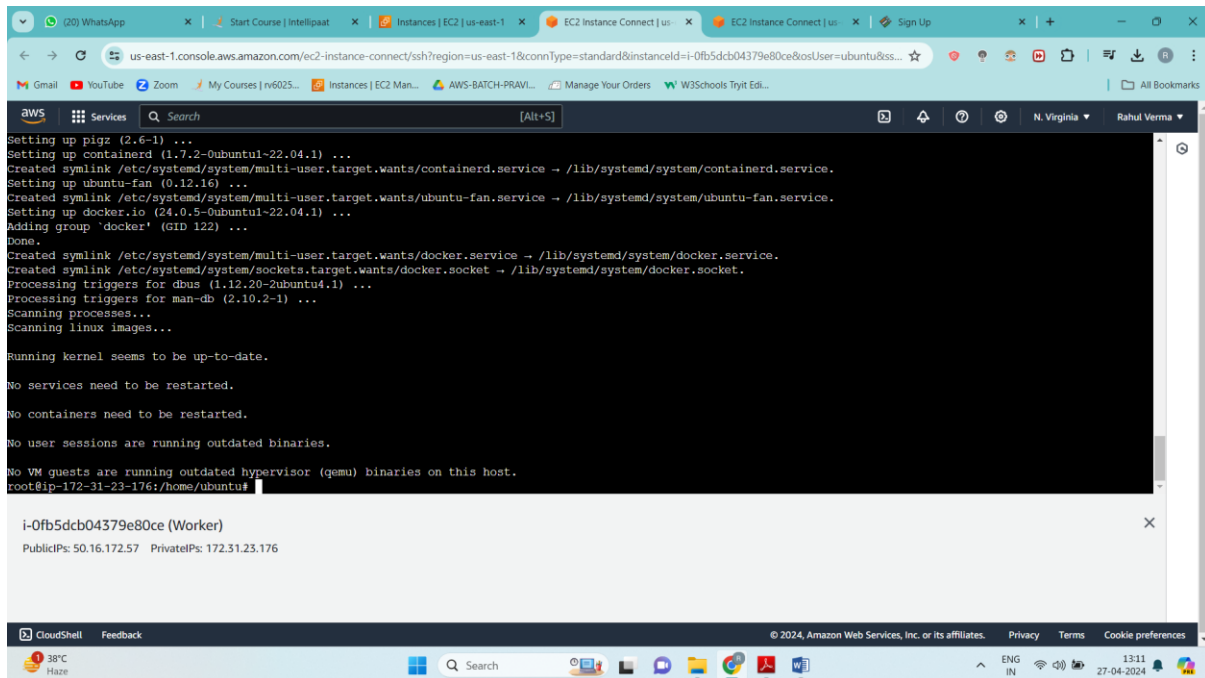
```
ubuntu@ip-172-31-16-129:~$ sudo su
root@ip-172-31-16-129:/home/ubuntu# docker swarm init --advertise-addr 34.224.8.131
Swarm initialized: current node (krrxfbulvywrtj9isky7eklo9) is now a manager.

To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-6c2mfofo6lt0b0ntmpjys08nyl7m98v3q83vxo03fiv60pydlc-83z9da32vwmmtsz6eho850e4t 34.224.8.131:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.
root@ip-172-31-16-129:/home/ubuntu#
```

2: Now create another instance so that you can join it as a worker node in the previously initialized swarm.
We have already installed docker in worker instance



The screenshot shows the AWS Management Console with a terminal window open. The terminal output displays the steps to install Docker on an Ubuntu 22.04.1 instance. It includes commands for updating the package list, installing Docker, creating the docker group, and setting up the systemd service. The output confirms that Docker is installed and ready to use.

```
Setting up pigz (2.6-1) ...
Setting up containerd (1.7.2-0ubuntu1-22.04.1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/containerd.service → /lib/systemd/system/containerd.service.
Setting up ubuntu-fan (0.12.16) ...
Created symlink /etc/systemd/system/multi-user.target.wants/ubuntu-fan.service → /lib/systemd/system/ubuntu-fan.service.
Setting up docker.io (24.0.5-0ubuntu1-22.04.1) ...
Adding group 'docker' (GID 122) ...
Done.
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /lib/systemd/system/docker.socket.
Processing triggers for dbus (1.12.20-2ubuntu4.1) ...
Processing triggers for man-db (2.10.2-1) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

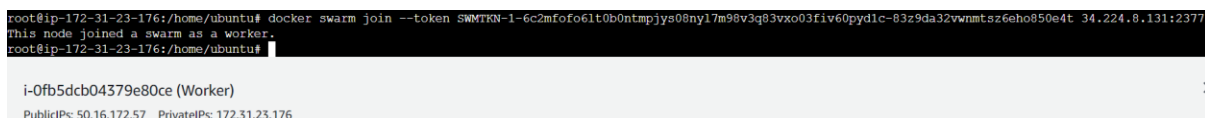
No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-23-176:/home/ubuntu#
```

i-0fb5dcb04379e80ce (Worker)
PublicIPs: 50.16.172.57 PrivateIPs: 172.31.23.176

Now copy the token from manager instance and paste it in worker instance



The screenshot shows a terminal window where the command to join a worker node to a Docker Swarm is executed. The output indicates that the node has successfully joined the swarm as a worker.

```
root@ip-172-31-23-176:/home/ubuntu# docker swarm join --token SWMTKN-1-6c2mfo6it0b0ntmpjys08nyl7m96v3q83vxo03fiv60pyd1c-83z9da32vwnmtaz6eho850e4t 34.224.8.131:2377
This node joined a swarm as a worker.
root@ip-172-31-23-176:/home/ubuntu#
```

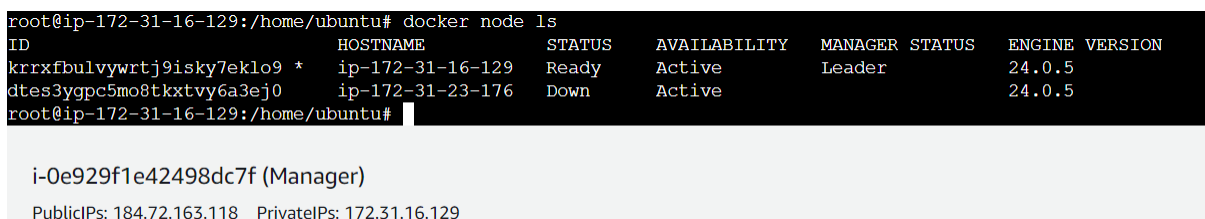
i-0fb5dcb04379e80ce (Worker)
PublicIPs: 50.16.172.57 PrivateIPs: 172.31.23.176

3: This command can be used to get information about the swarm

docker info

4: This command gives you the list of connected nodes.

docker node ls



The screenshot shows a terminal window where the command to list Docker nodes is executed. The output displays a table with columns for ID, HOSTNAME, STATUS, AVAILABILITY, MANAGER STATUS, and ENGINE VERSION. Two nodes are listed: one as a Leader and one as Down.

```
root@ip-172-31-16-129:/home/ubuntu# docker node ls
```

ID	HOSTNAME	STATUS	AVAILABILITY	MANAGER STATUS	ENGINE VERSION
krxxfbulvywrtj9isky7eklo9	* ip-172-31-16-129	Ready	Active	Leader	24.0.5
dtes3ygc5mo8tkxtvy6a3ej0	ip-172-31-23-176	Down	Active		24.0.5

root@ip-172-31-16-129:/home/ubuntu#

i-0e929f1e42498dc7f (Manager)
PublicIPs: 184.72.163.118 PrivateIPs: 172.31.16.129

5: This command can be used from the node terminal to leave a swarm.

Execute this on worker instance

`docker swarm leave --force`

```
ubuntu@ip-172-31-23-176:~$ sudo docker swarm leave --force
Node left the swarm.
ubuntu@ip-172-31-23-176:~$
```

i-0fb5dcb04379e80ce (Worker)

PublicIPs: 54.146.252.91 PrivateIPs: 172.31.23.176

And if you check nodes-

Worker node status is down

```
root@ip-172-31-16-129:/home/ubuntu# docker node ls
ID                                HOSTNAME                STATUS    AVAILABILITY    MANAGER STATUS    ENGINE VERSION
krrxfbulvywrtj9isky7eklo9 *    ip-172-31-16-129      Ready    Active           Leader            24.0.5
dtes3yggpc5mo8tkxtvy6a3ej0     ip-172-31-23-176      Down     Active           Leader            24.0.5
root@ip-172-31-16-129:/home/ubuntu#
```

i-0e929f1e42498dc7f (Manager)

PublicIPs: 184.72.163.118 PrivateIPs: 172.31.16.129

6: This command can be used in a manager node terminal to remove a node.

`docker node rm -f <node-id>`

```
root@ip-172-31-16-129:/home/ubuntu# docker node rm dtes3yggpc5mo8tkxtvy6a3ej0
dtes3yggpc5mo8tkxtvy6a3ej0
root@ip-172-31-16-129:/home/ubuntu# docker node ls
ID                                HOSTNAME                STATUS    AVAILABILITY    MANAGER STATUS    ENGINE VERSION
krrxfbulvywrtj9isky7eklo9 *    ip-172-31-16-129      Ready    Active           Leader            24.0.5
root@ip-172-31-16-129:/home/ubuntu#
```

i-0e929f1e42498dc7f (Manager)

PublicIPs: 184.72.163.118 PrivateIPs: 172.31.16.129

7. if you want to add some more nodes

`docker swarm join-token worker`

this will give you token