

## Practical-8

**Aim:** Study of Docker swarm and Deployment of ML project in swarm network.

The objective of this lab is to understand Docker Swarm, a container orchestration tool, and deploy a machine learning project in a Swarm network using three Linux host systems.

### Prerequisites:

- Three Linux host systems (e.g., Ubuntu)
- Docker installed on each host (follow the installation steps in the previous response)
- Basic understanding of Docker and machine learning concepts

### Lab Setup:

- Assign unique hostnames and IP addresses to each Linux host.
- Ensure that the hosts can communicate with each other over the network.

### Lab Steps:

#### Step 1: Initialize Docker Swarm

On the first host, initialize Docker Swarm to create a Swarm manager node.

```
docker swarm init--advertise-addr <MANAGER-IP>
```

```
PS C:\Users\Nakul\Downloads\MLOPs\Practicals\Practical 8> docker swarm init --advertise-addr 192.168.65.3
Swarm initialized: current node (ojcz2cyxj8qnhafqmqra85duw) is now a manager.

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

    docker swarm join --token SWMTKN-1-1y13geamtvy5gv9kyrqhx0da9mho4ukrex1ra6qxe0rac1veqo-dq5bg72hyqam5q02sj3zave5m 192.168.65.3:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.
```

The command will generate a token; save it as you'll need it to join worker nodes.

#### Step 2: Join Worker Nodes

On the other two hosts, join them as worker nodes to the Swarm using the token generated in earlier Step 1.

```
docker swarm join--token <TOKEN> <MANAGER-IP>:2377
```

```
PS C:\Users\Nakul\Downloads\MLOPs\Practicals\Practical 8> docker swarm join-token manager
To add a manager to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-1y13geamtvy5gv9kyrqhx0da9mho4ukrex1ra6qxe0rac1veqo-37aan5xj5y05scxh8ba23vgai 192.168.65.3:2377
```

You should see a message confirming that the nodes have joined the Swarm.

**Step 3: Deploy a ML Model as a Service**

Now, let's deploy a simple ML model as a Docker service. Create a Docker Compose file (e.g.,

`ml\_app.yml`) with the following content:

```
version: '3'
```

```
services:
```

```
  ml_app:
```

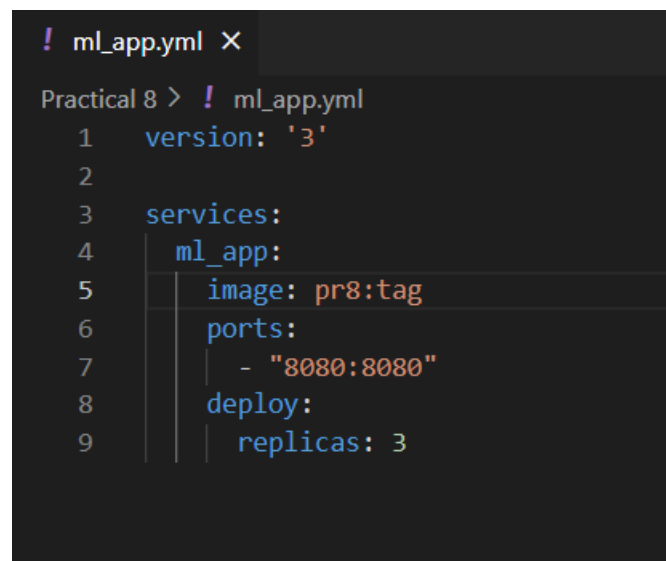
```
    image: your-ml-image:tag
```

```
    ports:
```

```
      - "8080:8080"
```

```
    deploy:
```

```
      replicas: 3
```

A screenshot of a code editor window titled 'ml\_app.yml'. The editor shows the following YAML content:

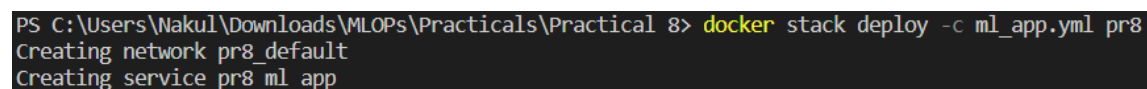
```
version: '3'

services:
  ml_app:
    image: pr8:tag
    ports:
      - "8080:8080"
    deploy:
      replicas: 3
```

**Step 4: Deploy the Service**

On the manager node, deploy the ML project as a service:

```
docker stack deploy -c ml_app.yml pr8
```

A screenshot of a terminal window showing the execution of the command 'docker stack deploy -c ml\_app.yml pr8'. The output shows the creation of a network and a service.

```
PS C:\Users\Nakul\Downloads\MLOPs\Practicals\Practical 8> docker stack deploy -c ml_app.yml pr8
Creating network pr8_default
Creating service pr8_ml_app
```

**Step 5: Verify the Deployment**

Check the status of the deployed service:

```
docker service ls
```

```
PS C:\Users\Nakul\Downloads\MLOPs\Practicals\Practical 8> docker service ls
ID                NAME          MODE          REPLICAS  IMAGE          PORTS
vgc43zk3mux3     pr8_ml_app    replicated    0/3       pr8:tag        *:8080->8080/tcp
```

You should see the service running on three replicas.

**Step 6: Access the ML Project**

You can access the ML project on any of the worker nodes using their IP addresses and port 8080 (the port we exposed in the Compose file).

<http://192.168.65.3:8000/>

**Step 7: Scaling**

Experiment with scaling the service up or down to see how Docker Swarm manages the replicas.

```
docker service scale pr8_ml_app=5
```

```
PS C:\Users\Nakul\Downloads\MLOPs\Practicals\Practical 8> docker service scale pr8_ml_app=5
pr8_ml_app scaled to 5
overall progress: 0 out of 5 tasks
1/5: preparing [=====]
2/5: preparing [=====]
3/5: preparing [=====]
4/5: preparing [=====]
5/5: preparing [=====]
```

**Step 8: Removing the Stack**

When done, you can remove the stack:

```
docker stack rm pr8
```

```
PS C:\Users\Nakul\Downloads\MLOPs\Practicals\Practical 8> docker stack rm pr8
Removing service pr8_ml_app
Removing network pr8_default
```

**Step 9: Leave Swarm**

On worker nodes, leave the Swarm when you're finished:

```
docker swarm leave
```

```
PS C:\Users\Nakul\Downloads\MLOPs\Practicals\Practical 8> docker swarm leave --force
Node left the swarm.
```

**Step 10: Shut Down**

Shut down all host systems.