UPES

UNIVERSITY WITH A PURPOSE

Lab Experiment – 7

NAME -Divyaansh Jain

ROLL NO – R171218040

SAP ID – 500067134

COURSE – B. Tech CSE- DevOps Batch 1

SUBJECT – Application Containerization

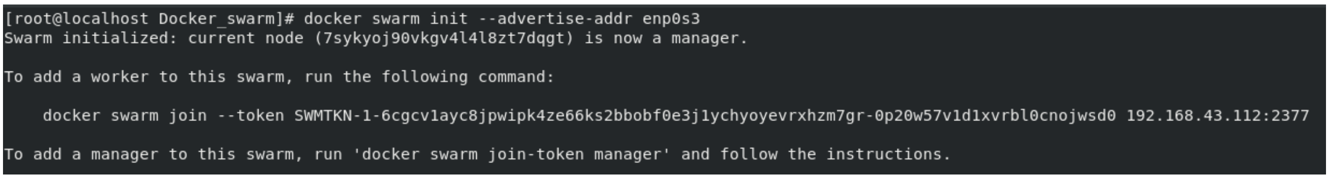
SEMESTER – 6th semester

Submitted To:

Dr. Hitesh Kumar Sir

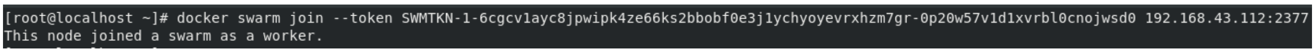
Docker Swarm

- Use command **docker swarm init** on a node to initialize it as master node, here --advertise-addr enp0s3 is used to give the init command the idea of which network card to use for the cluster. As soon as you execute the command, a token will be generated for another node to join as a worker in the cluster.



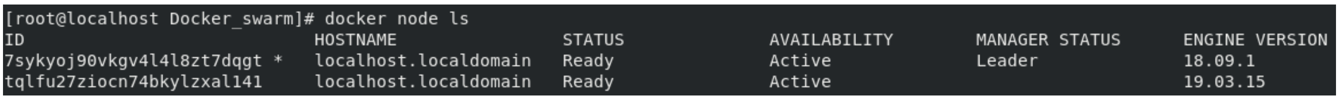
```
[root@localhost Docker_swarm]# docker swarm init --advertise-addr enp0s3
Swarm initialized: current node (7sykyoj90vkgv4l4l8zt7dqgt) is now a manager.

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

    docker swarm join --token SWMTKN-1-6cgcv1ayc8jpwipk4ze66ks2bbobf0e3j1ychyoyevrxhzm7gr-0p20w57v1d1xvrbl0cnojwsd0 192.168.43.112:2377

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

- Using the command “docker swarm join” any node can join the cluster and --token is used to tell that the particular cluster. Token is generated in the previous command.



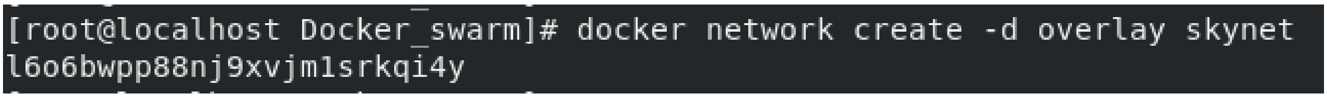
```
[root@localhost ~]# docker swarm join --token SWMTKN-1-6cgcv1ayc8jpwipk4ze66ks2bbobf0e3j1ychyoyevrxhzm7gr-0p20w57v1d1xvrbl0cnojwsd0 192.168.43.112:2377
This node joined a swarm as a worker.
```

- Use the command “docker node ls” to check how many master and worker nodes are there in your cluster. Here, Leader keyword indicated the master node.



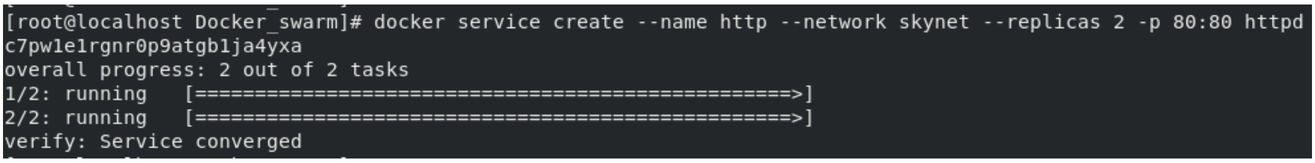
```
[root@localhost Docker_swarm]# docker node ls
ID                            HOSTNAME                STATUS              AVAILABILITY        MANAGER STATUS      ENGINE VERSION
7sykyoj90vkgv4l4l8zt7dqgt *   localhost.localdomain   Ready               Active              Leader              18.09.1
tqlfu27ziocn74bkylzxal141     localhost.localdomain   Ready               Active                                  19.03.15
```

- Use the command “docker network create -d overlay skynet” to create an overlay network over which containers will be launched.



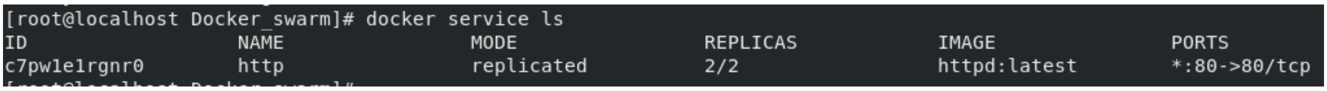
```
[root@localhost Docker_swarm]# docker network create -d overlay skynet
l6o6bwpp88nj9xvjm1srkqi4y
```

- Use the command “docker service create”, you can create a service and launch multiple containers using the same image as well as configurations, in this case, we use --replicas 2 to launch two containers.



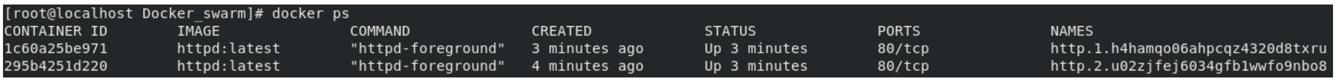
```
[root@localhost Docker_swarm]# docker service create --name http --network skynet --replicas 2 -p 80:80 httpd
c7pw1e1rgnr0p9atgb1ja4yxa
overall progress: 2 out of 2 tasks
1/2: running   [==================================================>]
2/2: running   [==================================================>]
verify: Service converged
```

- Use the “docker service ls” command, you can list the services in the cluster.



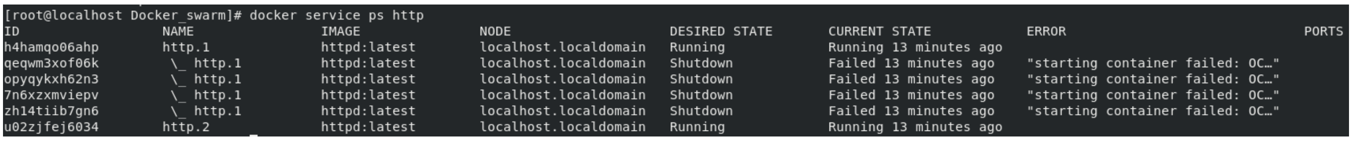
```
[root@localhost Docker_swarm]# docker service ls
ID                  NAME                MODE                REPLICAS            IMAGE               PORTS
c7pw1e1rgnr0        http                replicated          2/2                 httpd:latest        *:80->80/tcp
```

- Check the running containers using the “docker ps” command, you can see that name of the container starts with the service name.



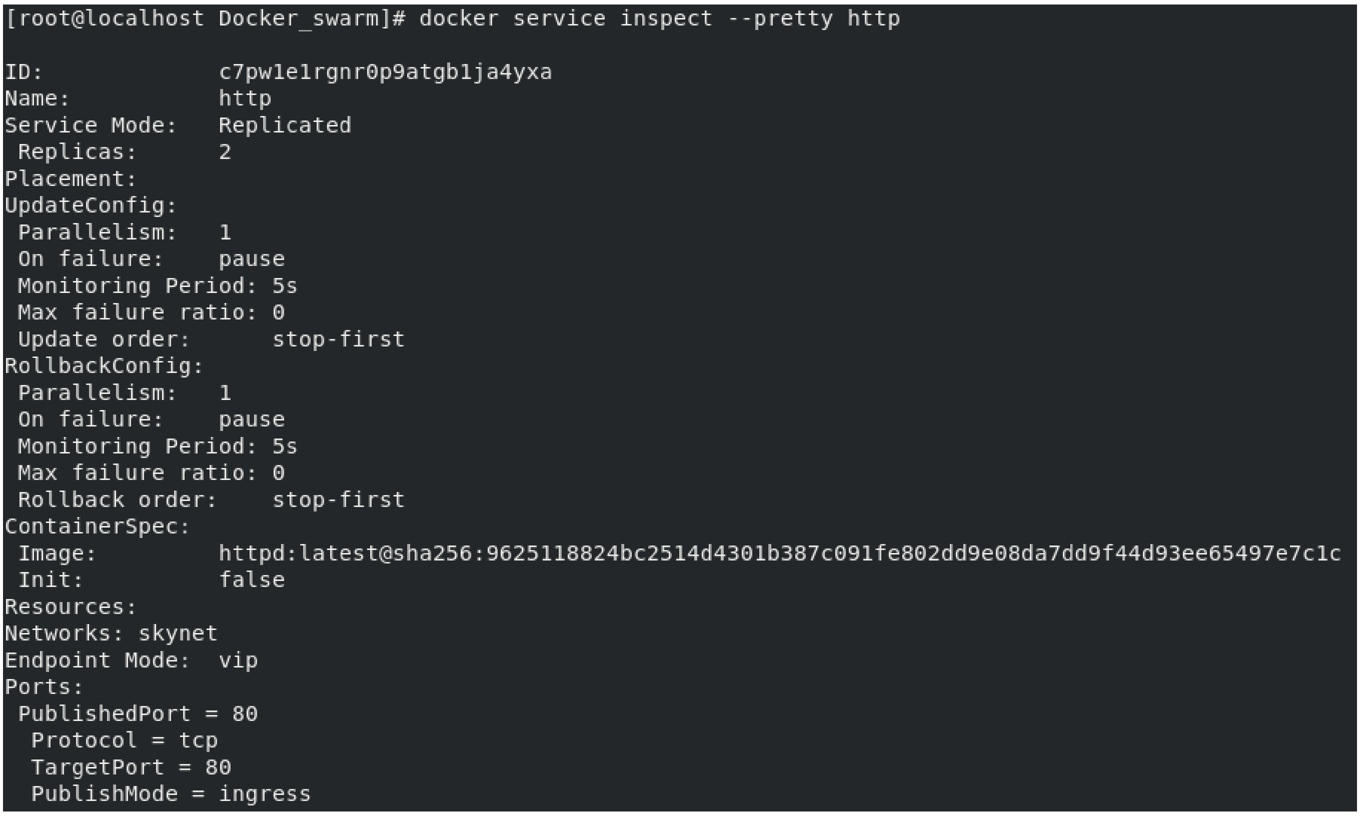
```
[root@localhost Docker_swarm]# docker ps
CONTAINER ID        IMAGE               COMMAND              CREATED             STATUS              PORTS               NAMES
1c60a25be971        httpd:latest        "httpd-foreground"   3 minutes ago       Up 3 minutes        80/tcp              http.1.h4hamqo06ahpcqz4320d8txru
295b4251d220        httpd:latest        "httpd-foreground"   4 minutes ago       Up 3 minutes        80/tcp              http.2.u02zjfej6034gfb1wwfo9nbo8
```

- Check the running service using “docker service ps service_name”, in this case http.



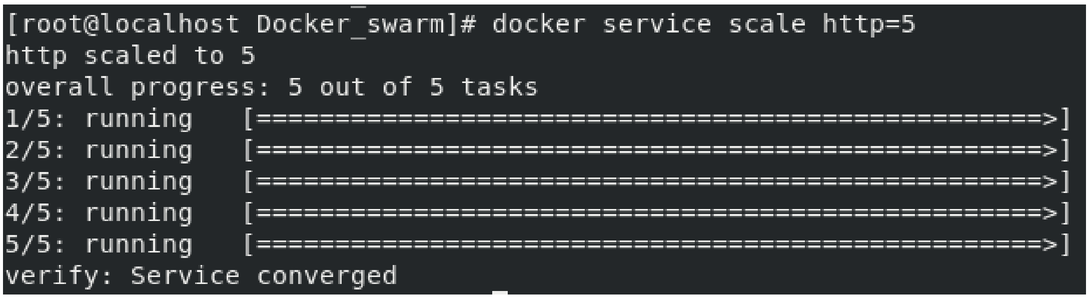
```
[root@localhost Docker_swarm]# docker service ps http
ID                  NAME                IMAGE               NODE                    DESIRED STATE       CURRENT STATE            ERROR                              PORTS
h4hamqo06ahp        http.1              httpd:latest        localhost.localdomain   Running             Running 13 minutes ago
qeqwm3xof06k         \_ http.1          httpd:latest        localhost.localdomain   Shutdown            Failed 13 minutes ago    "starting container failed: OC…"
opyqykxh62n3         \_ http.1          httpd:latest        localhost.localdomain   Shutdown            Failed 13 minutes ago    "starting container failed: OC…"
7n6xzxmviepv         \_ http.1          httpd:latest        localhost.localdomain   Shutdown            Failed 13 minutes ago    "starting container failed: OC…"
zh14tiib7gn6         \_ http.1          httpd:latest        localhost.localdomain   Shutdown            Failed 13 minutes ago    "starting container failed: OC…"
u02zjfej6034        http.2              httpd:latest        localhost.localdomain   Running             Running 13 minutes ago
```

- Check the complete details of the service using “docker service inspect --pretty http”.



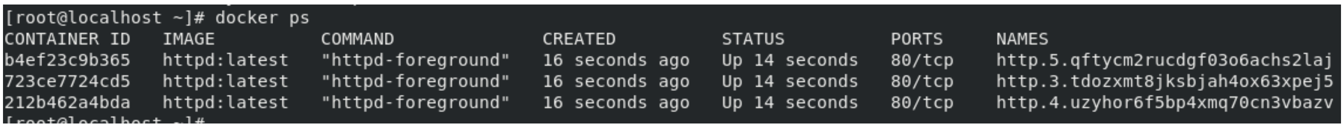
```
[root@localhost Docker_swarm]# docker service inspect --pretty http

ID:             c7pw1e1rgnr0p9atgb1ja4yxa
Name:           http
Service Mode:   Replicated
 Replicas:      2
Placement:
UpdateConfig:
 Parallelism:   1
 On failure:    pause
 Monitoring Period: 5s
 Max failure ratio: 0
 Update order:      stop-first
RollbackConfig:
 Parallelism:   1
 On failure:    pause
 Monitoring Period: 5s
 Max failure ratio: 0
 Rollback order:    stop-first
ContainerSpec:
 Image:         httpd:latest@sha256:9625118824bc2514d4301b387c091fe802dd9e08da7dd9f44d93ee65497e7c1c
 Init:          false
Resources:
Networks: skynet
Endpoint Mode:  vip
Ports:
 PublishedPort = 80
  Protocol = tcp
  TargetPort = 80
  PublishMode = ingress
```

- Increasing the number of running containers using “docker service scale http=5”, in this case we want to have total 5 containers.

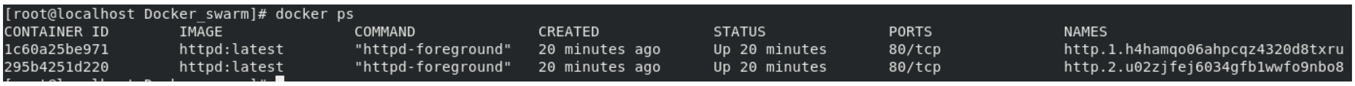


```
[root@localhost Docker_swarm]# docker service scale http=5
http scaled to 5
overall progress: 5 out of 5 tasks
1/5: running   [==================================================>]
2/5: running   [==================================================>]
3/5: running   [==================================================>]
4/5: running   [==================================================>]
5/5: running   [==================================================>]
verify: Service converged
```

- Check both the nodes for containers using “docker ps” command.

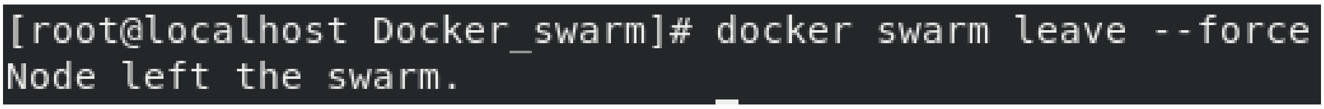


```
[root@localhost ~]# docker ps
CONTAINER ID   IMAGE          COMMAND              CREATED          STATUS          PORTS     NAMES
b4ef23c9b365   httpd:latest   "httpd-foreground"   16 seconds ago   Up 14 seconds   80/tcp    http.5.qftycm2rucdgf03o6achs2laj
723ce7724cd5   httpd:latest   "httpd-foreground"   16 seconds ago   Up 14 seconds   80/tcp    http.3.tdozxmt8jksbjah4ox63xpej5
212b462a4bda   httpd:latest   "httpd-foreground"   16 seconds ago   Up 14 seconds   80/tcp    http.4.uzyhor6f5bp4xmq70cn3vbazv
```



```
[root@localhost Docker_swarm]# docker ps
CONTAINER ID        IMAGE               COMMAND              CREATED             STATUS              PORTS               NAMES
1c60a25be971        httpd:latest        "httpd-foreground"   20 minutes ago      Up 20 minutes       80/tcp              http.1.h4hamqo06ahpcqz4320d8txru
295b4251d220        httpd:latest        "httpd-foreground"   20 minutes ago      Up 20 minutes       80/tcp              http.2.u02zjfej6034gfb1wwfo9nbo8
```

- Use the command “docker swarm leave --force” in any node to leave the swarm cluster. Note that using this command on master node terminates the whole cluster if only there is one master node.



```
[root@localhost Docker_swarm]# docker swarm leave --force
Node left the swarm.
```