

**UNIVERSITY OF PETROLEUM & ENERGY STUDIES**

**Dehradun**

**Application Containerization**

**Experiment 7**

**Name: Muskaan Madan**

**Course: B-Tech CSE DevOps (2018-22)**

**Roll number: R171218063**

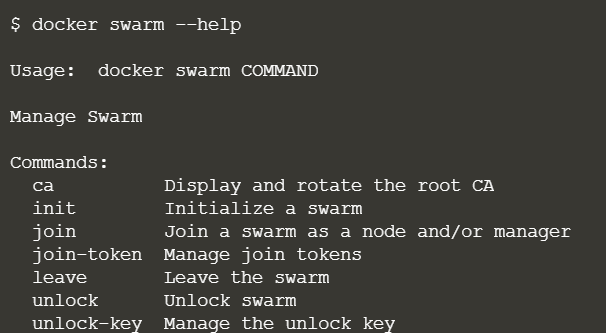
**Sap ID: 500068520**

**Docker Swarm**

**Creating swarm mode cluster**

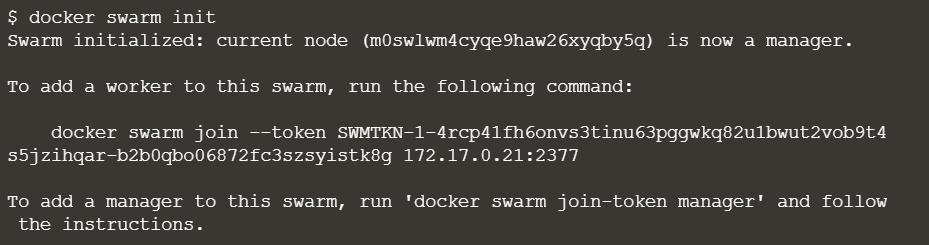
* To get help for any docker swarm use the following command:

docker swarm help



* Initialize the docker swarm mode using the following command:

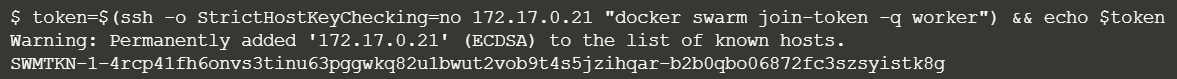
docker swarm init



**Join an existing cluster**

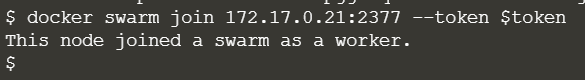
* Obtain the token required to add a worker to the cluster, using the following command:

token=$(ssh -o StrictHostKeyChecking=no 172.17.0.10 "docker swarm join-token -q worker") && echo $token



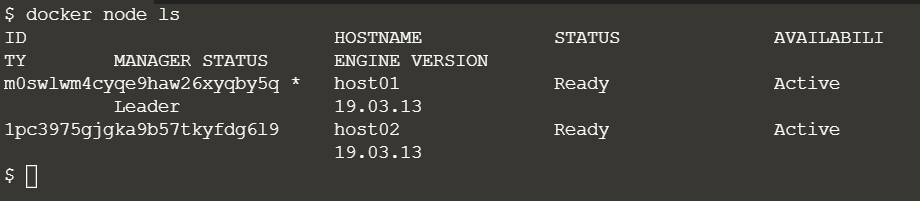
* Join the cluster by requesting access via the manager. The token is provided as an additional parameter as shown in the following command:

docker swarm join 172.17.0.10:2377 --token $token



* view all nodes in the cluster using the following command:

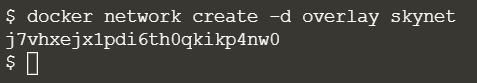
docker node ls



**Join an existing cluster**

* Create a new overlay network called “skynet”. All containers registered to this network can communicate with each other, regardless of which node they are deployed onto. It is done via the following command:

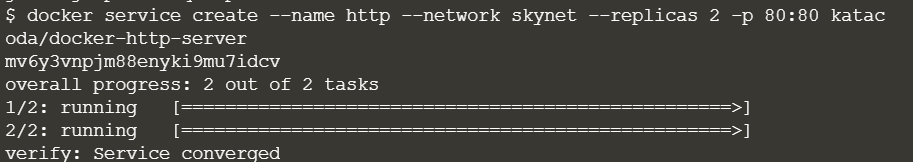
docker network create -d overlay Skynet



**Deployment of Services**

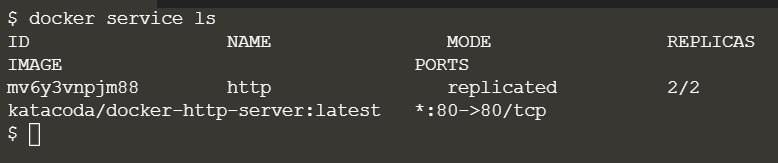
* Services is used to run containers across the cluster. Create a docker swarm service using the following command:

docker service create --name http --network skynet --replicas 2 -p 80:80 katacoda/docker-http-server



* You can view the services running on the cluster using the following command:

docker service ls



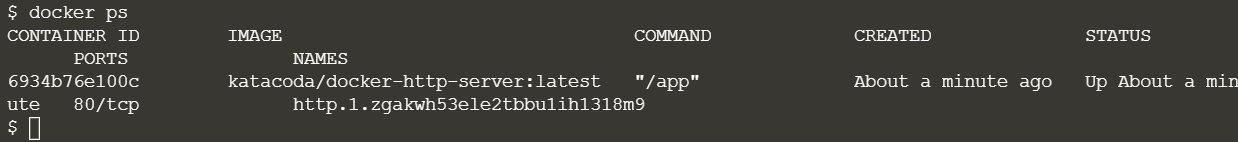
* List containers on the first host

docker ps



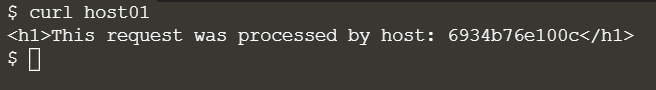
* List containers on the second host

docker ps



* If we issue an HTTP request to the public port, it will be processed by the two containers

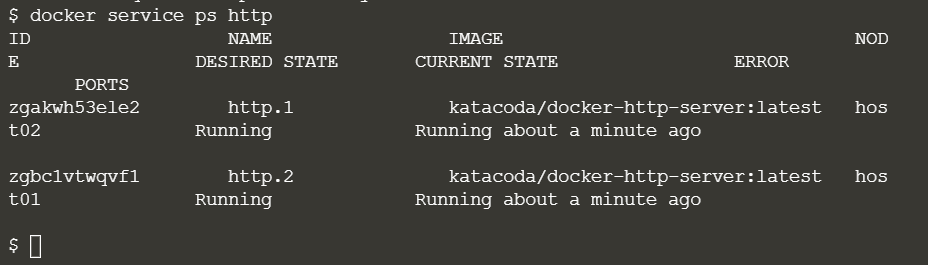
curl host01



**Deployment of Services**

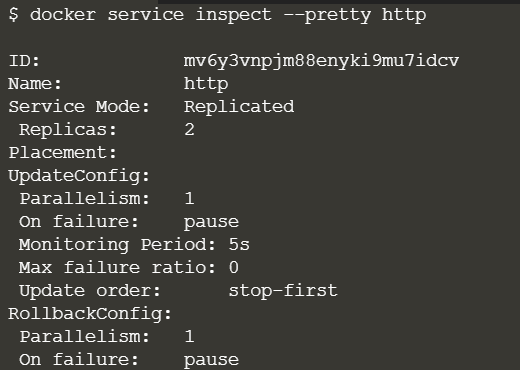
* You can view the list of all the tasks associated with a service across the cluster. In this case, each task is a container

docker service ps http



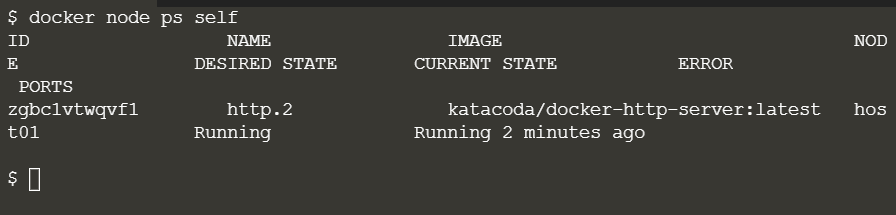
* You can view the details and configuration of a service via the following command:

docker service inspect --pretty http



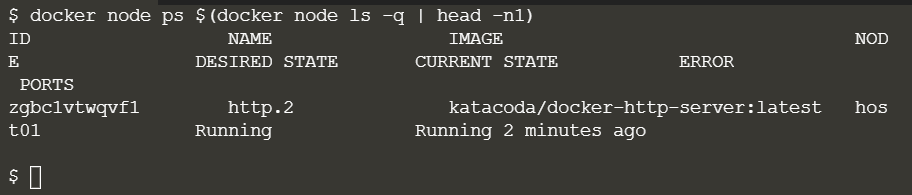
* On each node, you can ask what tasks it is currently running. Self refers to the manager node Leader:

docker node ps self



* Using the ID of a node you can query individual hosts:

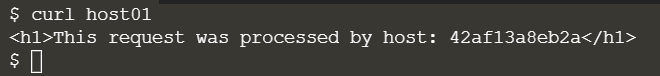
docker node ps $(docker node ls -q | head -n1)



**Scaling of Services**

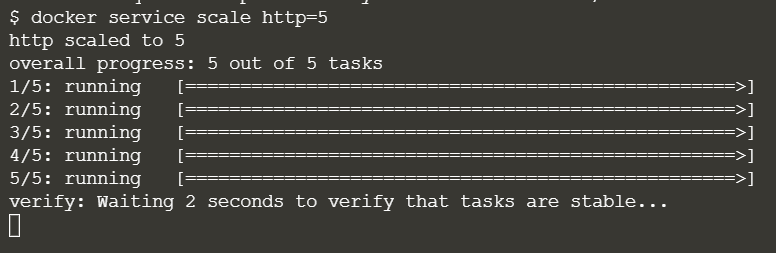
* We have two load-balanced containers running, which are processing our requests

curl host01



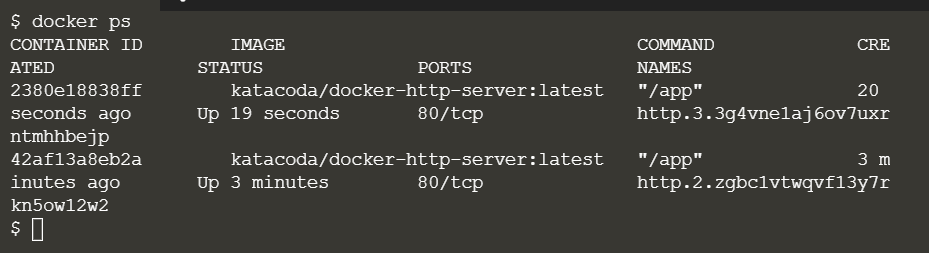
* The command below will scale our *http* service to be running across five containers:

docker service scale http=5



* On each host, you will see additional nodes being started

docker ps



* The load balancer will automatically be updated. Requests will now be processed across the new containers. Try issuing more commands via the following command:

curl host01

