**DOCKER SWARM**

A swarm is a group of machines that are running Docker and joined into a cluster. Docker swarm is a tool for Container Orchestration.

Orchestration – managing and controlling multiple docker containers as a single service.

Tools available – Docker Swarm, Kubernetes, Apache Mesos.

**Prerequisites**

1. Docker 1.3 or higher
2. Docker Machine (pre-installed for Docker for Windows and Docker for MAC)

**Step 1:** Create Docker machines (to act as nodes for Docker swarm). Create one machine as manager and others as workers.

**docker-machine create –driver hyperv machine\_name**

**Step 2:** Check machine created successfully.

**docker-machine ls**

To get the IP of the docker machine created,

**docker-machine ip machine\_name**

**Step 3:** SSH connect to docker-machines.

**docker-machine ssh machine\_name**

Do the above command for all the machines created in separate command line.

**Step 4:** Initialize Docker swarm (should be done on the manager machine)

**docker swarm init –advertise-addr MANAGER\_IP**

**docker node ls 🡪** will list the swarm manager 🡪 will work only on manager and can’t run this command on worker machines.

**Step 5:** Join workers in the swarm. Get command for joining as worker. In manager node, run the command

**docker swarm join-token worker 🡪** will give the command to join the swarm as worker.

**docker swarm join-token manager 🡪** will give the command to join the swarm as manager.

Go to the worker command lined and run command to join swarm as worker.

In manager command line, give **docker node ls** to verify whether the worker is registered and is ready.

Do the above for all the worker machines.

**Step 6:** On manager run standard docker commands

**docker info 🡪** Check the swarm section, number of managers, nodes etc.

Check docker swarm command options.

**docker swarm**

**Step 7:** Run containers on Docker swarm (on manager node).

**docker service create –replicas 3 -p 80:80 –name service\_Name nginx**

**--replicas 3🡪** will run the service on 3 nodes

**-p 🡪** port number

**nginx 🡪** image name used for web server 🡪 we can run this service and see the result in Browser

Check the status.

**docker service ls 🡪** list all the services (should be run on the manager)

**docker service ps serviceName 🡪** to see the running status of the service

**Step 8:** Scale service up and down.

On manager node,

**docker service scale serviceName = 2**

If we reduce the count to less than the number of nodes running, some nodes be set as idle. If we increase the count to greater than the number of nodes, the same node can run more than once.

**Note:** The nodes are randomly selected for increasing and decreasing the count.

Inspecting nodes (can only run on the manager)

**docker node inspect nodeName**

**nodeName 🡪** can be any worker node

To inspect a node by itself,

**docker node inspect self 🡪** can’t be run on worker nodes

**Step 9:** Update service

**docker service update –image imageName: version serviceName**

eg: docker service update –image nginx:1.14.0 web1

**Step 10:** Shutdown node

**docker node update –availability drain machineName**

**Step 11:** Remove service

**docker service rm serviceName**

To leave the swarm,

**docker swarm leave 🡪** can be run on any node

To stop the machine, run the below command from outside the machine (use a separate command line)

**docker-machine stop machineName**

To remove the machine,

**docker-machine rm machineName**