Docker Swarm

Objectives

- Docker Swarm capabilities and terminology
- Demonstrate docker swarm

Docker Swarm

- Current versions of Docker include *swarm mode*
  - Support for natively managing a cluster of Docker Engines called a swarm.

- What is a swarm?
  - Multiple Docker hosts running in swarm mode
  - They can be …
    - Managers (to manage membership and task delegation)
    - Workers (to run swarm services)
    - Or both.

Docker Swarm

- You create *services* on a swarm
  - Define optimal state – number of replicas, network and storage resources, ports exposed etc.
- Docker works to maintain that desired state
  - If a worker node becomes unavailable, Docker schedules that node's tasks on other nodes.
- A *task* is a running container which is part of a swarm service and managed by a swarm manager, as opposed to a standalone container.

Advantage and Features

- Modify a service’s configuration without needing to (manually) restart the service
  - Including modifying networks and volumes it is connected to.
  - Docker updates the configuration and does the stopping and starting needed on the tasks.

- While Docker is running in swarm mode, you can still run standalone containers on any of the Docker hosts in the swarm.
  - But only swarm managers can manage a swarm.

- You can define and run Swarm Service stacks in the same way you use Docker Compose.

Key Concepts

- Nodes (or Docker Node)
  - Instance of the Docker Engine participating in the swarm.
  - Typically nodes are distributed across multiple physical and cloud machines.
- Manager Node
  - A service definition is submitted to a manager node, which dispatches work units (tasks) to Worker Nodes.

Key Concepts

- Services and Tasks
  - A Service is the definition of the tasks that the swarm executes.
  - It is the central structure of the swarm system & root of user interaction
  - A Task carries a Docker container and the commands to run in that container.
  - Replicated Services: manager distributes a specific number of tasks among the nodes.
  - Global Services: manager runs one task for the service on every available node.

- Load balancing
  - The swarm manager uses Ingress Load Balancing to make services available externally to the swarm.
  - The swarm manager uses internal load balancing to distributed requests among nodes in the cluster.

Getting Started with Swarm

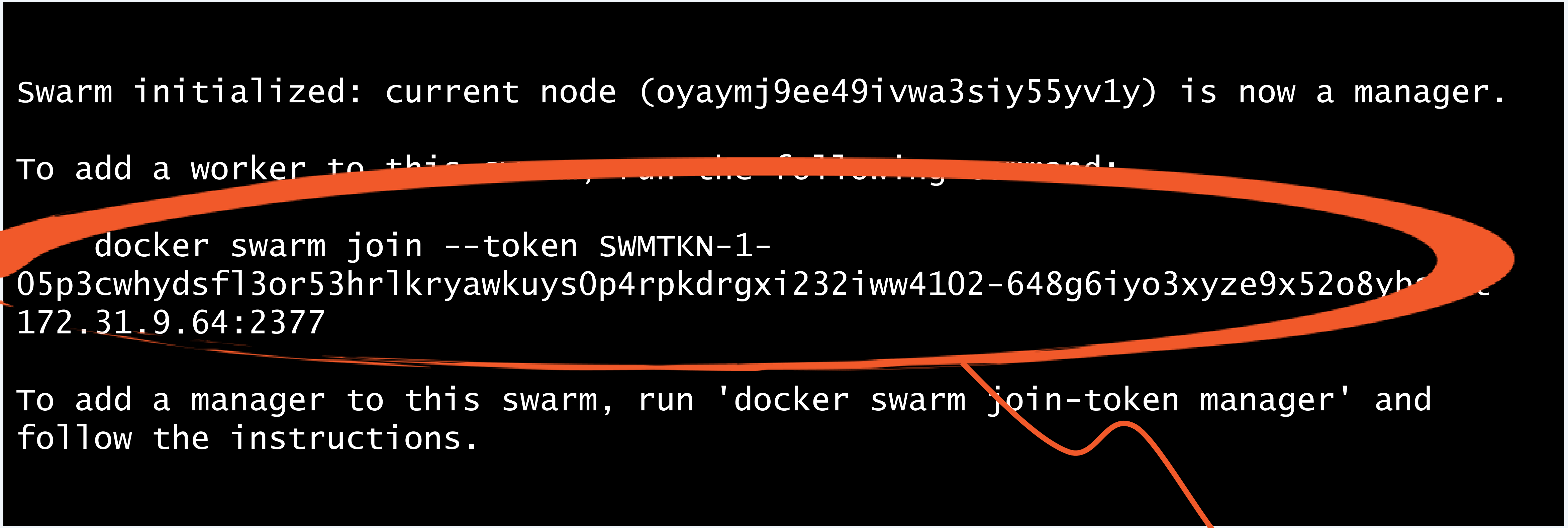
- You need at least 3 linux machines with docker installed.
- On the command line in the manager machine type

```
$ docker swarm init --advertise-addr 172.31.9.64
```

Your IP address in here

- This brings up a very long command you need to copy to join the swarm:

Getting started with a swarm



```
Swarm initialized: current node (oyaymj9ee49ivwa3siy55yv1y) is now a manager.

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

    docker swarm join --token SWMTKN-1-
05p3cwhydsfl3or53hrlkryawkuys0p4rpkdrgxi232iww4102-648g6iyo3xyze9x52o8yb
172.31.9.64:2377

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

Copy this!

Starting a swarm

- You can see the current state of the swarm using `$docker info`:

```
$ docker info
Containers: 11
 Running: 5
 Paused: 0
 Stopped: 6
Images: 22
… snip …
Swarm: active
 NodeID: oyaymj9ee49ivwa3siy55yv1y
 Is Manager: true
 ClusterID: lvxz99ibp9ltvgl9aqs6t9u65
 Managers: 1
 Nodes: 1
```

Starting a swarm

- Run `$docker node ls` to view information about nodes:

```
$ docker node ls
ID                             HOSTNAME            STATUS    AVAILABILITY   MANAGER STATUS    ENGINE VERSION
oyaymj9ee49ivwa3siy55yv1y *    <long name here>     Ready       Active         Leader            18.06.1-ce
```

- Now we are ready to add more nodes to our swarm…

Joining a swarm

- On the command line of your worker machine, enter the join command from the manager:

```
$docker swarm join --token SWMTKN-1-
05p3cwhydsfl3or53hrlkryawkuys0p4rpkdrgxi232iww4102-648g6iyo3xyze9x52o8ybszut
172.31.9.64:2377
```

- If you don’t have the join token, you can just type the following in the manager to retrieve it
  - `$docker swarm join-token worker`

- Use `$docker node ls` in the manager to see the worker nodes

Owerya RESOURCING

Deploy a service

- On the command line of your manager machine, run this command

```
$docker service create --replicas 1 --name helloworld alpine ping docker.com
```

  - The `$docker service create` command creates the service.
  - The `--name` flag names the service `helloworld`.
  - The `--replicas` flag specifies the desired state of 1 running instance.
  - The arguments `alpine ping docker.com` define the service as an Alpine Linux container that executes the command ping docker.com.

- Use `$docker service ls` to see the running service:

```
ID            NAME        MODE        REPLICAS   IMAGE          PORTS
viwvf7u8ceti  helloworld  replicated  1/1        alpine:latest
```

Now what?

- Inspect the service on the manager:

```
$docker service inspect --pretty helloworld
```

- Scale the service in the swarm

```
$docker service scale helloworld=5
```

- To see the updated task list:

```
$docker service ps helloworld
```

- `$docker ps` will show the containers running on each of the workers and the manager

Delete a service

- Deleting a service is simple:

```
$docker service rm helloworld
```

- Use `$docker service inspect helloworld` and `$docker ps` to verify that it has been removed.

What else can you do?

- You can apply rolling updates to your service.
- You can drain a single node (for example for maintenance) without stopping the service
  - Docker just pushes the tasks onto other nodes until you are ready to bring that node up again.

Final notes:

- The only command that was executed on the worker nodes was to join the swarm – no further information about services was required.

- There is some configuration needed to enable a swarm on a given set of machines
  - For example, in AWS you need to modify the security group to allow the swarm to communicate.
  - You need to have certain ports available.

Summary

- Docker Swarm capabilities and terminology
- Demonstrate docker swarm

Lab

Using play-with-docker create your own docker swarm:

Labs/07-docker-swarm-lab.md

Questions or Comments?

Owerya RESOURCING