Swarm stack introduction

Let’s deploy the voting app stack on a swarm.

Purpose

The purpose of this lab is to illustrate how to deploy a stack (multi services application) against a Swarm using a docker compose file.

The application

The voting app is a very handy multi containers application often used for demo purposes during meetup and conferences.

It basically allow users to vote between cat and dog (but could be “space” or “tab” too if you feel like it).

This application is available on Github and updated very frequently when new features are developed.

Init your swarm

There should be two terminals displayed. The first accesses the swarm `manager` node and the second accesses the swarm `worker` node once the swarm is created.

Let’s create a Docker Swarm first. Copy the below command into the first terminal.

```
docker swarm init --advertise-addr $(hostname -i)
```

From the output above, copy the join command (*watch out for newlines*) and paste it in the other terminal.

Show members of swarm

From the first terminal, check the number of nodes in the swarm (running this command from the second terminal `worker` will fail as swarm related commands need to be issued against a swarm manager).

```
docker node ls
```

The above command should output 2 nodes, the first one being the `manager`, and the second one a `worker`.

Clone the voting-app

Let’s retrieve the voting app code from Github and go into the application folder.

Ensure you are in the first terminal and do the below:

```
git clone https://github.com/docker/example-voting-app
cd example-voting-app
```

Deploy a stack

A stack is a group of services that are deployed together. The docker-stack.yml in the current folder will be used to deploy the voting app as a stack.

Ensure you are in the first terminal and do the below:

```
docker stack deploy --compose-file=docker-stack.yml voting_stack
```

Note: being able to create a stack from a docker compose file is a great feature added in Docker 1.13.

Check the stack deployed from the first terminal

```
docker stack ls
```

The output should be the following one. It indicates the 6 services of the voting app’s stack (named voting_stack) have been deployed.

```
NAME          SERVICES
voting_stack  6
```

Let’s check the service within the stack

```
docker stack services voting_stack
```

The output should be like the following one (your ID should be different though).

```
ID            NAME                      MODE        REPLICAS  IMAGE
10rt1wczotze  voting_stack_visualizer   replicated  1/1       dockersample
s/visualizer:stable
8lqj31k3q5ek  voting_stack_redis        replicated  2/2       redis:alpine
nhb4igkkyg4y  voting_stack_result       replicated  2/2       dockersample
s/examplevotingapp_result:before
nv8d2z2qhlx4  voting_stack_db           replicated  1/1       postgres:9.4
ou47zdyf6cd0  voting_stack_vote         replicated  2/2       dockersample
s/examplevotingapp_vote:before
rpnxwmoipagq  voting_stack_worker       replicated  1/1       dockersample
s/examplevotingapp_worker:latest
```

Let’s list the tasks of the vote service.

```
docker service ps voting_stack_vote
```

You should get an output like the following one where the 2 tasks (replicas) of the service are listed.

```
ID            NAME                  IMAGE
      NODE   DESIRED STATE   CURRENT STATE           ERROR   PORTS
my7jqgze7pgg  voting_stack_vote.1  dockersamples/examplevotingapp_vote:be
fore  node1  Running         Running 56 seconds ago
3jzgk39dyr6d  voting_stack_vote.2  dockersamples/examplevotingapp_vote:be
fore  node2  Running         Running 58 seconds ago
```

From the NODE column, we can see one task is running on each node.

Finally, we can check that our APP is running, the RESULT page is also available as well as SWARM VISUALIZER