Advanced Docker

IMT3003 kyrre.Begnum@ntnu.no

Bootstrapping images

How to pass configuration parameters

- As container images grow more complex, the need to pass on more parameters grows
- Previously, having a service as entrypoint was sufficient, but now we would rather have a bootstrap script

Environment variables

- Every Linux system has a wealth of environment variables that processes can check
  - One can check the current environment variables with the command "set"
  - In Docker, we can pass environment variables to a container when starting it and "catch" them inside

Example: Alternative config.php

```
<?php
$dbhost = getenv("BF_DB_HOST");
$dbport = "3306";
$db = getenv("BF_DB");
$dbuser = getenv("BF_USER");
$dbpassw = getenv("BF_PASSWORD");
$webhost = getenv("BF_ENTRY_IP");
$weburl = 'http://' . $webhost ;
$memcache_enabled = 0;
if ( getenv("BF_MEMCACHE_SERVER")){
   $memcache_enabled_pictures = 1;
   $memcache_server = getenv("BF_MEMCACHE_SERVER");
   $memcache_enabled = 1;
}
?>
```

We can now run the container like:

```
docker run -e BF_DB_HOST=10.10.0.X -e BF_DB=bf -e BF_USER=bfuser ... mybookface_image:tag
```

Docker Swarm

IMT3003 kyrre.Begnum@ntnu.no

Docker Swarm Mode

- A docker swarm means multiple servers are connected to collaborate to run containers
- The swarm has one or more managers and multiple nodes
- Relatively easy to set up a swarm

Creating a swarm

- Install docker on all the involved servers
- On the server intended to be the master run:
  `docker swarm init`
- Node down the token and the example join command
- Run the join command on all the other nodes in the cluster

Checking status

- You can see if all nodes are connected:
  `docker node ls`

- To change the availability of a node:

  `docker node update --availability active <node>`

  For example, to remove all containers from node-2

  `docker node update --availability drain node-2`

Creating services from containers

- A service in docker consists of one or more instances from the same image
- The benefits of services are:
  - Load balancing accross all images
  - You can adjust the number of replicas
  - Fail-over

Creating a service

- To create a service, run:
  docker service create [options] [image]
- Many options can be used, like:
  --replicas=X
  --name=xyz
- For example:

```
docker service create -d --replicas=3 --name=hello
-p 3000:80 tutum/hello-world
```

Checking status

- List all services
  `docker service ls`
- How is the service doing?
  `docker service ps service-name`
- Increase/decrease the replicas (instances)
  `docker service scale name=replicas`

Docker Compose

Writing templates for containers

- Starting a single container from the command line can become impractical
- How would you start multiple containers that belong together?

Compose

- Docker compose files allow you to scpecify a docker instance
- Simple, declarative syntax with space as blocks
- Requires installation of the tool docker-compose:

```
apt-get install docker-compose
```

Example:

Create a new file in a folder docker-compose.yml with the following content:

```
version: '2'

services:
  bookface_web:

    image: bookface_web:v3

    ports:
      - 80
```

Start the containers using: `docker-compose up [-d]`

Check status with: `docker ps`

Stop the containers (in the same folder) docker-compose down

Elaborate example

```
version: '2'


services:
 bookface_web:
  image: docker.cs.hioa.no/kyrre/dookface:latest

  ports:
     - 80
  environment:
        BF_USER: bfuser
        BF_DB_HOST: bookface_db
        BF_DB: bf
        BF_PASSWORD: bfpassword
  links:
        - bookface_db

 bookface_db:
   image: docker.cs.hioa.no/kyrre/dookfacedb:latest

   environment:
    MYSQL_ROOT_PASSWORD: my-secret-pw
    MYSQL_DATABASE: bf
    MYSQL_USER: bfuser
    MYSQL_PASSWORD: bfpassword

 bookface_user:
   image: docker.cs.hioa.no/kyrre/simplewebuser:latest

   environment:
     INTERVAL: 10
     ENTRYPOINT: bookface_web

   links:
     - bookface_web
```

Even more advanced example

- Download more elaborate examples:

  `git clone https://git.cs.hioa.no/kyrre.begnum/bookface_docker_examples`

- The most advanced example contains a working bookface deployment as a stack:

```
cd bookface_docker_examples/bookface_web_db_user_memcache_loadbalanced
docker stack deploy -c docker-compose.yml bf
```

- Check the status

```
docker stack ls
docker stack ps bf
```

Docker Registry

Docker registries

- hub.docker.com is the most popular registry for docker images
- For private projects, it may be more appropriate to have a local, private registry
- There are many alternatives, some og them free
  port.us.org, Harbor (VMware), Artifactory and Docer Private Registry

Add insecure harbor registry

- Run these commands as root

```
curl http://10.212.136.140/harbor.crt > /usr/local/share/ca-certificates/harbor.crt
update-ca-certificates
service docker restart
```

- Now, login to the registry

```
docker login 10.212.136.160
```

Moving an image into the registry

- In Harbor, create a project. For example "myproject"

- Take a working local image you are happy with and give it an additional tag:
  `docker tag myimage:tag harbor-ip/myproject/myimage:tag`

- Upload the image into the registry
  `docker push harbor-ip/myproject/myimage:tag`

Using the image

- From another docker host, install the certificate and log in
- The new image is now available:

```
docker run harbor-ip/myproject/myimage:tag
```

Swarms and private registries

- When using a private registry with authentication, all swarm nodes need access too
- Add the option "--with-registry-auth" to service create to send along registration details from the master