Exam - Cloud Computing

1h30 - documents authorisés

1er mars 2020

1 Questions

Q.1) - Quel est l'intéret d'un réseau overlay ?

Q.2) - Quel est le rôle du champs `environment` dans un `docker-compose.yml`

2 Exercice

A Image Mongo

Vous devez mettre en place un service de chat basé sur rocketchat. Ce dernier sera composé de 3 conteneurs :

— un conteneur mondoDB pour le stockage ;
— un conteneur mongo express pour explorer et administrer cette base de données ;
— un conteneur rocketchat le service lui même.

Les documentations des 3 conteneurs sont données dans la suite.

Attention, malgrès les exemple fourni dans les documentations, *vous ne devez pas utiliser le système de* `link` de docker qui n'est plus maintenu. Pour mettre en relation les conteneurs, vous utiliserez un réseau interne.

Pour fonctionner, votre ensemble de conteneur doit comporter :

— Un réseau nommé `resrocket` et utilisant la plage d'adresse `172.17.17.0/24`. Tous les conteneurs utiliseront ce réseau.
— Un docker `mongoDB` :
  — dont le nom est `mongo` ;
  — dont l'alias réseau est `baserocket` ;
  — qui utilise l'authentification avec un administrateur de login `chef` et de mot de passe `zorg` ;
  — qui est initialisé par un script `./init_base.js`. Ce script est fourni et met un place la base pour rocketchat et un utilisateur pour ce dernier dont le login est `userrocket` et le mot de passe `passw0r2`.
— Un docker `mongo express` :
  — dont le nom est express ;
  — qui s'authentifie en tant qu'administrateur sur la base mongo ;
  — qui expose le service mongoexpress sur le port 8080 de la machine hôte.
— Un docker `rocket.chat` :
  — dont le nom est `rocket` ;
  — qui expose le service sur le port 8888 de la machine hôte ;
  — qui utilise le serveur mongoDB du docker `mongo` avec l'utilisateur `userrocket` créé par le script d'initialisation.

Q.3) - Donnez les commandes docket ou le fichier `docker-compose.yml` qui réalise cela.

A Docker MongoDB

A.1 Les Tags

— 3.4.19-jessie, 3.4-jessie (3.4/Dockerfile)

— 3.4.19-windowsservercore-ltsc2016, 3.4-windowsservercore-ltsc2016 (3.4/windows/windowsservercore-ltsc2016/Dockerfile)
— ...

A.2 Comment utiliser l'image

Lancer un serveur mongo

```
$ docker run --name some-mongo -d mongo:tag
```

... où `some-mongo` est le nom que vous voulez donner à votre conteneur et `tag` spécifie la version de MongoDB que vous voulez.

A.3 Vous connecter au serveur mongo depuis un autre conteneur

Le serveur MongoDB de l'image écoute sur le port stadart 27017, donc se connecter via les *links* ou le réseau de docker est identique à la connection à un serveur distant. L'exemple suivant lance un autre conteneur mongoDB et lance l'utilitaire mongo en ligne de commande qui se connecte au premier ce qui permet de lancer des commandes mongo sur votre instance de base de données :

```
$ docker run -it --link some-mongo:mongo --rm mongo mongo --host mongo test
```

... où `some-mongo` est le nom de votre premier docker mongo.

A.4 Variables d'environement

Lorsque vous lancez l'image mongo, vous pouver adapter l'instance en passant une ou plusieurs variables d'environement lors de la ligne de commande docker. Attention, aucune des variables présentées n'aura d'effet si vous lancez le conteneur avec un répertoire de données qui contient déjà une base de données initialisée. Toute base préexistant sera conservée intacte lors du lancement du conteneur.

`MONGO_INITDB_ROOT_USERNAME`, `MONGO_INITDB_ROOT_PASSWORD`

Ces variables, utilisées ensembles, créent un nouvel utilisateur et fixe son mot de passe. Cet utilisateur est créé dans la base d'authentification admin et obtient le rôle `root` qui est un rôle de super utilisateur.

Les deux variables sont exigées pour la création de l'utilisateur. Si les deux existent, alors `MongoDB` démarerra avec l'authentification active (`mongodb --auth`)

L'authentification dans MongoDB est assez compliquée, donc les configurations plus complexes peuvent uniquement être créés via `/docker-entrypoint-initdb.d/` (voir le chapitre Initialisation d'une nouvelle instance)

`MONGO_INITDB_DATABASE`

Cette variable vous permet de spécifier le nom d'une base de données qui peut être utilisée pour les scripts de création dans `/docker-entrypoint-initdb.d/*.js` (Voir le chapitre suivant). MongoDB est est fondamentalement conçu pour « créer à la première utilisation », donc si vous n'insérez pas de données via vos scripts Javascript, aucune base de données n'est réellement créée.

A.5 Initialisation d'une nouvelle instance

Quand un conteneur est lancé la première fois, il exécutera les fichiers `.sh` et `.js` qui se trouvent dans `/docker-entrypoint-initdb.d`. Les fichiers seront exécutés en ordre alphabétique. Les fichiers `.js` seront exécutés par mongo utilisant la base spécifiée par la variable `MONGO_INITDB_DATABASE`, si elle est présente ou la base `test` sinon. Vous pouvez aussi changer la base de données dans les scripts eux même.

A.6 Où stocker les données

La documentation de Docker est un bon point de départ pour connaître les différentes options de stockage et de modification, il y a de multiples blog ou post de forum qui en discute et donne des avis en ce domaine. Nous montrons juste la procédure basique ici :

Créez un répertoire de données dans un volume de votre système hôte, i.e. `/my/own/datadir`.

Lancez votre conteneur mongo comme cela :

```
$ docker run --name some-mongo -v /my/own/datadir:/data/db -d mongo
```

B Image Mongo-express

B.1 Tags supporté

— 0.49.0, 0.49, latest (Dockerfile)

B.2 Qu'est-ce que mongo-express ?

mongo-express est une interface administration de mongoDB en Node.js, Express.js, et Bootstrap3.

github.com/mongo-express/mongo-express

B.3 Comment utiliser cette image ?

```
$ docker run --link some_mongo_container:mongo -p 8081:8081 mongo-express
```

Alors vous pouvez taper `http://localhost:8081` ou `http://host-ip:8081` dans votre navigateur.

B.4 Configuration

Les variables d'environnement sont passées en ligne de commande pour configurer les conteneur mongo-express.

Name	Default	Description
ME_CONFIG_BASICAUTH_USERNAME	”	mongo-express web username
ME_CONFIG_BASICAUTH_PASSWORD	”	mongo-express web password
ME_CONFIG_MONGODB_ADMINUSERNAME	”	MongoDB admin username
ME_CONFIG_MONGODB_ADMINPASSWORD	”	MongoDB admin password
ME_CONFIG_MONGODB_PORT	27017	MongoDB port
ME_CONFIG_MONGODB_SERVER	'mongo'	MongoDB container name. Use comma delimited list of host names for replica sets.
ME_CONFIG_SITE_BASEURL	'/'	Set the baseUrl to ease mounting at a subdirectory. Remember to include a leading and trailing slash.

C Image RocketChat

C.1 Tags suportés

— 2.4.9, 2.4, 2, latest

C.2 Description

Rocket.Chat est un serveur web de chat, développé en JavaScript utilisant le framework fullstack Meteor.

C'est une bonne solution pour les communautés et les compagnies qui veulent privatiser leur propres services de chat pour développeur, ou pour les développeurs qui cherchent un point de départ pour créer et faire évoluer leur propre plateforme de chat.

C.3 Comment utiliser l'image

Premièrement, lancer une instance mongo et initialisez les replicaSet :

```
$ docker run --name db -d mongo:4.0 \
         --smallfiles --replSet rs0 --oplogSize 128

$ docker exec -ti db mongo --eval "printjson(rs.initiate())"
```

Ensuite lancer `Rocket.Chat` lié à cette instance :

```
$ docker run --name rocketchat --link db \
    --env MONGO_OPLOG_URL=mongodb://db:27017/local -d rocket.chat
```

Cela lancera l'instance `Rocket.Chat` écoutant sur le port part défaut de Meteor (le port 3000 du conteneur).

Si vous voulez accéder directement à l'instance sur un port de la machine hôte :

```
$ docker run --name rocketchat -p 80:3000 \
          --link db --env ROOT_URL=http://localhost \
          --env MONGO_OPLOG_URL=mongodb://db:27017/local -d rocket.chat
```

Vous pouvez ainsi accéder à l'instance via l'adresse `http://localhost` de votre navigateur. Remplacer `ROUT_URL` par votre propre nom de domain.

Si vous utilisez un serveur Mongo externe, ou via Kubernetes, vous devez remplacer la variable `MONGO_URL` :

```
$ docker run --name rocketchat \
       --env MONGO_URL=mongodb://myuser:mypass@mymongourl/mydb  \
       --env MONGO_OPLOG_URL=mongodb://myuser:mypass@mymongourl:27017/local \
       -d rocket.chat
```