Université Constantine 2 – Abdelhamid Mehri
Faculté des nouvelles technologies de l'information et de la communication
Département de l'informatique fondamentale et ses applications

Spécialité : SDIA
Niveau : M2
Module : SDCC
Enseignant : R. MENNOUR

TP 6 : Introduction à Docker

1. Créer une instance EC2 avec le nom Docker Server et permettre l'accès par SSH à cette instance.
2. Installer Docker sur l'instance Docker Server.
3. Essayer les commandes suivantes sur l'instance :
    a. docker run hello-world
    b. docker images
    c. docker ps
    d. docker ps –a
4. Créer une image Docker basée sur une application Node simple.
    a. Créer un dossier test et basculer dedans.
    b. Créer un Dockerfile qui contient les commandes suivantes :

```
FROM node:6
WORKDIR /app
ADD . /app
EXPOSE 80
CMD ["node", "app.js"]
```

Ce fichier indique au daemon Docker comment créer votre image.

- La première ligne indique l'image parent de base, qui correspond ici à l'image Docker officielle de la version 6 de Node.
- Dans la deuxième ligne, nous définissons le répertoire de travail (actuel) du conteneur.
- Dans la troisième ligne, nous ajoutons le contenu du répertoire actuel (indiqué par le ".") dans le conteneur.
- Nous indiquons ensuite le port du conteneur afin d'autoriser les connexions sur ce port, puis nous terminons en exécutant la commande du nœud pour démarrer l'application.

5. Créer l'application, pour créer l'image ensuite.
    a. Créer un fichier app.js et mettez-y le code suivant :

```
const http = require('http');
const hostname = '0.0.0.0';
const port = 80;
```

```
const server = http.createServer((req, res) => {
   res.statusCode = 200;
     res.setHeader('Content-Type', 'text/plain');
       res.end('Congratulation RSD, you created your first container app\n');
});
server.listen(port, hostname, () => {
   console.log('Server running at http://%s:%s/', hostname, port);
});
process.on('SIGINT', function() {
   console.log('Caught interrupt signal and will exit');
   process.exit();
});
```

6. Créer l'image docker avec la commande suivante :
   docker build –t node-app :0.1 .

   Vous remarquerez là encore la présence du ".", qui désigne le répertoire actuel.

   Notez comment, pour chaque ligne du Dockerfile ci-dessus, des couches de conteneur intermédiaires sont ajoutées au fur et à mesure que l'image se crée.

7. Examiner les images que vous avez créé :
   docker images

8. Exécuter des conteneurs basés sur l'image que vous avez créée
   docker run –p 4000 :80 –name my-app node-app:0.1

9. Ouvrez un autre terminal, puis accéder au serveur que vous venez de créer à l'aide de curl

   curl http://<adresse du serveur>

Le conteneur s'exécute à condition que le terminal initial soit en cours d'exécution. Si vous souhaitez que le conteneur s'exécute en arrière-plan (il n'est alors pas associé à la session du terminal), vous devez définir l'indicateur –d.

10. Fermez le terminal initial, puis exécutez la commande suivante pour arrêter l'exécution du conteneur et le supprimer :
    docker stop my-app && docker rm my-app

11. Lancer de nouveau le conteneur pour qu'il s'exécute en arrière-plan.
    docker run -p 4000:80 --name my-app -d node-app:0.1
    docker ps

12. Consulter les logs, exécuter la commande :
    docker logs [id_conteneur]

Challenge :

Lancer un deuxième serveur qui s'exécute en même temps que le premier, à partir de l'image node-app :0.2 et du conteneur my-app-2, qui affiche le message « Welcom to Cloud Computing Lab 6 », et qui écoute les requêtes sur le port 8080.

Commande pour installer docker :

sudo yum update -y
sudo yum -y install docker

Lancer Docker

sudo service docker start

Acceder aux commandes Docker dans EC2-user

sudo usermod -a -G docker ec2-user
sudo chmod 666 /var/run/docker.sock

Liste des commandes Docker:

https://docs.docker.com/engine/reference/commandline/docker/

Liste des commandes Dockerfile

https://docs.docker.com/engine/reference/builder/