Présentation de Docker

Dans le cadre de nos travaux pratiques sur Kafka et les systèmes de streaming, **Docker** sera notre outil principal pour créer un environnement de travail cohérent et reproductible. Grâce à lui, nous pourrons exécuter l'ensemble des composants nécessaires (Kafka, ksqlDB, producteurs, consommateurs…) sans avoir à les installer ou configurer manuellement sur chaque machine. L'objectif est de se concentrer sur la logique des flux de données, la production et la consommation des messages, plutôt que sur les détails techniques d'installation. Docker permet ainsi de garantir que chaque étudiant dispose d'un environnement identique et fonctionnel dès le démarrage des TP.

Comme nous serons nombreux à récupérer sur le hub de Docker, depuis une seule @IP de sortie de l'ISIMA, les images de nos containers il nous sera rapidement nécessaire de nous authentifier chez Docker afin qu'on ne soit pas limités dans nos capacités à télécharger ces images.

Qu'est-ce que Docker dans le cadre de nos TP

Nous survolerons les commandes les plus courantes. Nous aborderons l'installation de l'outil jusque dans la partie post-installation.

Docker Compose (orchestration des services)

Lancer tous les service du docker-compose.yml placé sur le chemin courant

```
docker compose up
```

Regarder l'état des services :

```
docker compose ps
```

Les colonnes CREATED et STATUS vont vous permettre de comprendre l'état réel de vos services.

Arrêter tous les service du docker-compose.yml placé sur le chemin courant

```
docker compose down -v
```

Voir les logs du service "kafka-1" :

```
docker compose logs -f kafka-1
```

Relancer un service par exemple ksqldb :

```
docker compose restart ksqldb
```

Ouvrir un shell sur un container

Par exemple pour ouvrir un shell sur le container kafka-1 :

`docker exec -it kafka-1 bash`

Pour en ressortir il suffira de taper `exit`

Gestion du stockage Docker

Images Les images servant à construire les containers sont stockées en locale. On peut avoir la nécessité de les identifier voir de les supprimer dans certaines situations.

Lister les images téléchargées :

`docker image ls`

Télécharger une image particulière :

`docker pull <nom_image>:<tag-si-necessaire>`

Supprimer une image :

`docker image rm <nom_image>`

Supprimer toutes les images :

`docker image prune --all`

Volumes Liste les volumes :

`docker volume ls`

Supprimer un volume :

`docker volume rm <nom_volume>`

Supprimer tous les volumes :

`docker volume prune`

Quels sont les ports réseaux mis en oeuvres ?

Dans le dossier où on l'a le fichier docker-compose.yml pour lister les ports utilisés par l'application :

`docker compose ps`

On va avoir une liste de container

`0.0.0.0:8086->8086/tcp`

Le séparateur important est le `->`. `A gauche c'est l'hôte`, c'est à dire votre machine. `A droite c'est votre service s'exécutant dans le container.`

Par ailleurs le `:` désigne le fait qu'on précise une adresse IP à gauche du `:`. En l'occurence le `0.0.0.0` désigne le fait d'écouter sur toutes les adresses locales

IPv4 de votre machine. Si vous observez `[::]`cela désigne tous les IPv6 de votre machine.

cas de l'authentification chez Docker

1) Créer un compte Docker Hub
   - aller chez Docker Hub et créer un Docker ID (gratuit)
2) S'authentifier en local (avant le `docker compose up`)

```
docker login    #  entrer Docker ID + mot de passe
```

Ensuite les commandes de pull des images devraient bien se passer :

```
docker compose pull      # récupère les images avec vos droits authentifiés
docker compose up -d
```

Installation de Docker

Si votre machine n'a pas docker : l'URL suivante est le point d'entrée de l'installation de Docker -> https://docs.docker.com/engine/install/

Pour nous c'est le plus souvent Debia (12)

Cette installation sauf nouveauté chez Docker serait :

Installer les dépôts Docker

```
# Add Docker's official GPG key:
sudo apt-get update
sudo apt-get install ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/debian/gpg -o /etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc

# Add the repository to Apt sources:
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://do
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
```

Installer la dernière version :

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compo
```

Normalement à l'issue de la commande précédente votre premier service docker devrait fonctionner :

```
sudo docker run hello-world
```

la POST-INSTALLATION de docker

DANS TOUT LES TP JE CONSIDERE QUE VOUS N'AVEZ PAS BESOIN DE TAPER L'INSTRUCTION `sudo` CAR VOUS AVEZ FAIT LA POST-INSTALLATION DE DOCKER. Si ce n'est pas le cas vous ferez précédé `sudo`à toute vos commandes docker.

La post-installation :

```
sudo groupadd docker
sudo usermod -aG docker $USER
newgrp docker
```

Normalement ça devrait vous permettre de lancer

```
docker run hello-world
```

Si ce n'est pas le cas rebootez votre machine.