

# Travaux Pratiques Docker

Découverte des commandes essentielles

IUT - Module R509

15 octobre 2025

## Objectifs du TP

À la fin de ce TP, vous serez capable de :

- Télécharger et gérer des images Docker
- Créer et manipuler des conteneurs
- Comprendre et utiliser le mapping de ports
- Configurer et utiliser les registres Docker
- Diagnostiquer des problèmes réseau Docker

## Prérequis

- Docker installé sur votre machine
- Accès à un terminal
- Connexion Internet (pour télécharger les images)

# Table des matières

# 1 Téléchargement d'images Docker

## 1.1 Exercice 1 : Première image

### Énoncé

Téléchargez l'image officielle `hello-world` depuis Docker Hub.

**Question 1.1 :** Quelle commande utilisez-vous ?

**Question 1.2 :** Que signifie le message "Using default tag : latest" ?

## 1.2 Exercice 2 : Erreur courante - Nom d'image incorrect

### Énoncé

Un étudiant essaie de télécharger BusyBox avec la commande suivante :

```
docker pull busy-box
```

Il obtient une erreur : `repository does not exist`.

**Question 2.1 :** Quelle est l'erreur commise ?

**Question 2.2 :** Corrigez la commande pour télécharger BusyBox correctement.

### 1.3 Exercice 3 : Télécharger Grafana

#### Énoncé

Téléchargez l'image officielle de Grafana. Attention : l'image ne s'appelle pas simplement "grafana".

**Question 3.1 :** Trouvez le nom complet de l'image Grafana et téléchargez-la.

**Question 3.2 :** Listez toutes les images présentes sur votre système.

## 2 Gestion des conteneurs

### 2.1 Exercice 4 : Lancer un premier conteneur

#### Énoncé

Lancez le conteneur `hello-world` et observez le résultat.

**Question 4.1 :** Quelle commande utilisez-vous ?

**Question 4.2 :** Listez tous les conteneurs (y compris ceux arrêtés). Quel est le statut du conteneur `hello-world` ?

**Question 4.3 :** Pourquoi le conteneur s'est-il arrêté immédiatement ?

### 2.2 Exercice 5 : Comprendre le problème de BusyBox

#### Énoncé

Lancez un conteneur BusyBox avec la commande :

```
docker run busybox
```

Vérifiez ensuite les conteneurs en cours d'exécution.

**Question 5.1 :** Le conteneur apparaît-il dans `docker ps` ?

**Question 5.2 :** Utilisez `docker ps -a`. Quel est le statut du conteneur ?

**Question 5.3 :** Expliquez pourquoi le conteneur s'est arrêté.

**Question 5.4 :** Comment lancer BusyBox de manière interactive pour obtenir un shell ?

## 3 Mapping de ports

### 3.1 Exercice 6 : Lancer Grafana avec mapping de ports

#### Énoncé

Lancez un conteneur Grafana en mode détaché avec le mapping de ports approprié. Grafana écoute sur le port 3000 par défaut.

**Question 6.1 :** Complétez la commande suivante :

```
docker run ____ -p _____ --name=grafana _____
```

**Question 6.2 :** Expliquez chaque option utilisée dans votre commande :

-d : \_\_\_\_\_

-p 3000:3000 : \_\_\_\_\_

--name : \_\_\_\_\_

**Question 6.3 :** Schématisez le mapping de ports entre votre machine hôte et le conteneur.

### 3.2 Exercice 7 : Erreur de conflit de nom

#### Énoncé

Vous essayez de lancer à nouveau Grafana avec la même commande et obtenez :

```
Error response from daemon: Conflict. The container name "/grafana" is already in use by container "bb6ecf43440f..."
```

**Question 7.1 :** Quelle est la cause de cette erreur ?

**Question 7.2 :** Donnez deux solutions possibles pour résoudre ce problème.



### 3.3 Exercice 8 : Mapping de ports avancé

#### Énoncé

Vous devez lancer Grafana, mais le port 3000 est déjà utilisé par une autre application.

**Question 8.1 :** Modifiez la commande pour que Grafana soit accessible sur le port 8080 de votre machine hôte, tout en gardant le port 3000 dans le conteneur.

**Question 8.2 :** Quelle URL utiliserez-vous pour accéder à Grafana depuis votre navigateur ?

### 3.4 Exercice 9 : Docker Registry

#### Énoncé

Lancez un registre Docker local qui écoute sur le port 5001 de votre machine (car le port 5000 est déjà utilisé).

**Question 9.1 :** Complétez la commande :

```
docker run -d -p ----- --restart=always --name=registry  
-----
```

**Question 9.2 :** Comment vérifier que le registre fonctionne avec curl ?

## 4 Exploration de conteneurs

### 4.1 Exercice 10 : Accéder à un conteneur en cours d'exécution

#### Énoncé

Le conteneur Grafana est en cours d'exécution. Vous devez explorer son système de fichiers.

**Question 10.1 :** Quelle commande permet d'exécuter une commande dans un conteneur en cours d'exécution ?

**Question 10.2 :** Exécutez la commande `ls -la /` dans le conteneur Grafana. Écrivez la commande complète.

**Question 10.3 :** Comment obtenir un shell interactif dans le conteneur Grafana ? (Indice : Grafana utilise Alpine Linux)

### 4.2 Exercice 11 : Consulter les logs

#### Énoncé

Vous devez consulter les logs du conteneur Grafana pour diagnostiquer un problème.

**Question 11.1 :** Quelle commande permet d'afficher les logs d'un conteneur ?

**Question 11.2 :** Comment suivre les logs en temps réel (mode "follow") ?

## 5 Configuration réseau Docker

### 5.1 Exercice 12 : Identifier la plage réseau

#### Énoncé

Vous devez identifier la plage réseau utilisée par le réseau bridge Docker par défaut.

**Question 12.1 :** Listez tous les réseaux Docker sur votre système.

**Question 12.2 :** Inspectez le réseau `bridge` et trouvez la plage d'adresses IP (subnet) utilisée.

### 5.2 Exercice 13 : Conflit de subnet

#### Énoncé

Le réseau Docker bridge utilise `172.17.0.0/16`, mais cette plage est déjà utilisée par votre réseau d'entreprise. Vous devez changer la configuration pour utiliser `10.10.0.0/16` à la place.

**Question 13.1 :** Quel fichier devez-vous modifier ?

**Question 13.2 :** Écrivez la configuration JSON nécessaire pour utiliser la plage `10.10.0.0/16`.

**Question 13.3 :** Quelle commande permet de redémarrer le service Docker après modification ?

## 6 Registres Docker

### 6.1 Exercice 14 : Comprendre les registres

#### Énoncé

Répondez aux questions suivantes sur les registres Docker.

**Question 14.1 :** Quel est le registre utilisé par défaut lorsque vous faites `docker pull nginx` ?

**Question 14.2 :** Réécrivez la commande `docker pull hello-world` avec le nom complet incluant le registre.

**Question 14.3 :** Citez trois autres registres publics de conteneurs (hors Docker Hub).

### 6.2 Exercice 15 : Utiliser un registre local

#### Énoncé

Vous avez un registre local qui tourne sur `localhost:5001`. Vous voulez y pousser l'image BusyBox.

**Question 15.1 :** Taguez l'image `busybox` pour votre registre local.

**Question 15.2 :** Poussez l'image vers votre registre local.

**Question 15.3 :** Vérifiez que l'image est bien dans le registre avec `curl`.

## 7 Diagnostic et résolution de problèmes

### 7.1 Exercice 16 : Problème de permissions

#### Énoncé

Un étudiant obtient l'erreur suivante :

```
Got permission denied while trying to connect to the Docker daemon  
socket at unix:///var/run/docker.sock
```

**Question 16.1 :** Quelle est la cause de ce problème ?

**Question 16.2 :** Donnez la solution permanente pour résoudre ce problème (sans utiliser sudo).

### 7.2 Exercice 17 : Timeout réseau

#### Énoncé

Lors d'un docker pull, vous obtenez :

```
Error response from daemon: Get "https://registry-1.docker.io/v2/":  
net/http: request canceled while waiting for connection
```

**Question 17.1 :** Listez trois diagnostics à effectuer pour identifier la cause.

**Question 17.2 :** Si le problème vient de DNS, comment configurer Docker pour utiliser les DNS de Google (8.8.8.8) ?

## 8 Commandes de gestion

### 8.1 Exercice 18 : Nettoyage

#### Énoncé

Vous avez plusieurs conteneurs arrêtés qui prennent de l'espace disque.

**Question 18.1 :** Comment supprimer un conteneur spécifique (exemple : ID = abc123) ?

**Question 18.2 :** Comment supprimer tous les conteneurs arrêtés en une seule commande ?

**Question 18.3 :** Comment supprimer une image Docker ?

### 8.2 Exercice 19 : Monitoring

#### Énoncé

Vous devez surveiller l'utilisation des ressources de vos conteneurs.

**Question 19.1 :** Quelle commande permet de voir en temps réel l'utilisation CPU/-RAM de vos conteneurs ?

**Question 19.2 :** Comment voir les processus en cours d'exécution dans un conteneur nommé "grafana" ?



## 9 Synthèse pratique

### 9.1 Exercice 20 : Mini-projet complet

#### Projet Final

Mettez en place l'infrastructure suivante :

1. Un registre Docker local sur le port 5001
2. Un conteneur Grafana sur le port 3000
3. Un conteneur Nginx sur le port 8080
4. Tous les conteneurs doivent redémarrer automatiquement
5. Poussez l'image Nginx vers votre registre local

**Question 20.1 :** Écrivez toutes les commandes nécessaires dans l'ordre :

**Question 20.2 :** Comment vérifier que tout fonctionne correctement ?

## 10 Annexes

### 10.1 Aide-mémoire des commandes essentielles

#### Gestion des images

```
docker images           # Lister les images
docker pull <image>     # Télécharger une image
docker rmi <image>      # Supprimer une image
docker search <terme>    # Rechercher une image
```

#### Gestion des conteneurs

```
docker ps               # Conteneurs actifs
docker ps -a            # Tous les conteneurs
docker run <image>       # Lancer un conteneur
docker stop <container>  # Arrêter un conteneur
docker start <container> # Démarrer un conteneur
docker rm <container>    # Supprimer un conteneur
docker logs <container>  # Voir les logs
docker exec <container> <cmd> # Exécuter une commande
```

#### Options de docker run

```
-d           # Mode détaché (arrière-plan)
-p <host>:<container> # Mapping de ports
--name <nom>      # Nommer le conteneur
-v <volume>:<path> # Monter un volume
-e <VAR=value>     # Variable d'environnement
--restart always   # Redémarrage automatique
-it              # Mode interactif avec terminal
--rm            # Supprimer après arrêt
```

#### Réseau et diagnostic

```
docker network ls      # Lister les réseaux
docker network inspect <network> # Inspecter un réseau
docker info            # Informations système
docker stats           # Statistiques en temps réel
docker port <container> # Voir les ports mappés
```

### 10.2 Format des noms d'images

Structure : [registre/]organisation/dépôt[:tag]

Exemples :

- nginx → docker.io/library/nginx:latest
- grafana/grafana → docker.io/grafana/grafana:latest

- `gcr.io/google-containers/busybox:1.27`
- `localhost:5001/mon-image:v1`