🐳 Cours sur les Conteneurs : De la Virtualisation à Docker et Buildah

---

1. 🕰️ Introduction & Historique

1.1. Virtualisation classique : Les Machines Virtuelles (VM)

Avant l'apparition des conteneurs, **la virtualisation avec hyperviseur** était la méthode principale pour exécuter plusieurs systèmes d'exploitation sur une même machine physique.

👉 Exemple :

Sur un **serveur physique de 100 Go de RAM et 100 CPU**, on installe un **hyperviseur** (comme **Xen** ou **VMware ESXi**), puis on crée plusieurs **machines virtuelles (VM)**. Chaque VM a :

- Son propre système d'exploitation (OS)
- Ses propres bibliothèques et dépendances
- Et son application

Problème :

- Chaque VM consomme **beaucoup de ressources** (par exemple 10 Go RAM, 6 CPU par application)
- Il y a une **perte importante de mémoire et de CPU** (surtout si on a des milliers d'instances comme chez AWS avec EC2)
- **Risque de saturation** rapide (Out of Memory, CPU trop utilisé)

---

2. 🚀 Pourquoi passer aux conteneurs ?

Les **conteneurs** sont venus résoudre les limites des machines virtuelles.

2.1. Architecture classique des VM :

```
Matériel
   |
Hyperviseur (ex : Xen, KVM)
   |
Système d'exploitation (OS)
   |
Application
```

2.2. Architecture des conteneurs :

```
Matériel
   |
Système d’exploitation
   |
Docker Engine
   |
Conteneurs
```

☑ Chaque **conteneur partage le même noyau du système** hôte, ce qui réduit l’empreinte mémoire et CPU.
☑ Les conteneurs sont **plus légers, rapides à démarrer** et **portables**.

---

3. 🐳 Qu’est-ce qu’un conteneur ?

Un **conteneur** est un **paquet léger** contenant :

- Une **application**
- Ses **dépendances** (librairies, outils)
- Une **base d’image minimale** du système (ex : Ubuntu, Alpine)

Mais **il ne contient pas un OS complet** : il se repose sur le noyau du système hôte.

---

4. 🏗 Docker : L’outil qui a tout changé

4.1. Pourquoi Docker est devenu populaire ?

Docker a simplifié la gestion des conteneurs :

- Facile à créer avec un simple fichier `Dockerfile`
- Portable : fonctionne partout (Windows, Mac, Linux, Cloud)
- Intégration facile avec DevOps et CI/CD
- Isolation forte entre les applications

4.2. Fonctionnement de Docker

1. Créer un `Dockerfile` :

```
FROM python:3.10
COPY . /app
WORKDIR /app
RUN pip install -r requirements.txt
CMD ["python", "app.py"]
```

2. Construire l'image :

```
docker build -t mon-app .
```

3. Exécuter le conteneur :

```
docker run -p 5000:5000 mon-app
```

4.3. Composants Docker :

- **Dockerfile** : Script de construction de l'image
- **Image Docker** : Modèle statique de votre application
- **Conteneur Docker** : Instance vivante de l'image
- **Docker Engine** : Moteur qui exécute les conteneurs

---

5. ⚠ Limites de Docker

- **Dépendance forte au Docker Engine**
  👉 Si le **Docker Engine tombe**, **tous les conteneurs tombent** aussi.
- **Couche unique de gestion** : Toutes les actions (build, run, pull, push) passent par Docker Engine.

---

6. Buildah : Une alternative moderne

Pour résoudre la dépendance au moteur Docker, des outils comme **Buildah** ont vu le jour.

6.1. Buildah, c'est quoi ?

- Un outil pour **créer des images de conteneurs** sans avoir besoin du Docker Engine.
- Fonctionne de manière **rootless** (sans accès super utilisateur)
- Compatible avec les **standards OCI** (Open Container Initiative)

6.2. Avantages de Buildah :

- Plus **léger** et **sécurisé**
- Intégré facilement avec **Podman** (autre outil de gestion des conteneurs)
- **Moins de dépendances système** que Docker

---

7. 🤔 Résumé des différences : VM vs Docker vs Buildah

Caractéristique	VM	Docker	Buildah
OS complet	Oui	Non	Non
Poids	Lourd	Léger	Très léger
Temps de démarrage	Long	Rapide	Rapide
Isolation	Forte	Forte	Forte
Utilise un moteur	Hyperviseur	Docker Engine	Non (rootless)
Création d’image	Snapshot	Dockerfile	Commandes Buildah

---

8. 🛠 Cas réel : AWS et perte de ressources

Chez AWS, on utilise **Xen comme hyperviseur** pour créer les **EC2**.

Imaginons :

- 1 EC2 utilise 10 Go RAM et 6 CPU
- Une entreprise a **1 000 000 d'EC2**
- Des pertes énormes de ressources inutilisées (RAM, CPU)

👉 **Passer aux conteneurs** permet de :

- **Réduire les coûts**
- **Optimiser l’utilisation des ressources**
- **Déployer plus vite**
- **Gérer plus facilement les mises à jour**

---

9. 🔥 Ce qu’il faut retenir

- Les conteneurs sont **l’évolution moderne** des VM
- Docker a **démocratisé** leur utilisation
- Buildah est une **alternative légère et sécurisée**
- Les conteneurs sont **idéals pour le cloud**, les microservices, et le DevOps