Introduction

Docker et Kubernetes

CCI Strasbourg

Chapitre 1 : Introduction

Objectifs

En suivant ce chapitre, nous allons aborder les modules suivants :

- 1.1 Historique et contexte de la conteneurisation
- 1.2 Avantages de la conteneurisation par rapport à la virtualisation traditionnelle

Chapitre 1 : Introduction

Module 1.1
Historique et contexte de la conteneurisation

Chapitre 1 : Introduction

Module 1.1 - Historique et contexte de la conteneurisation

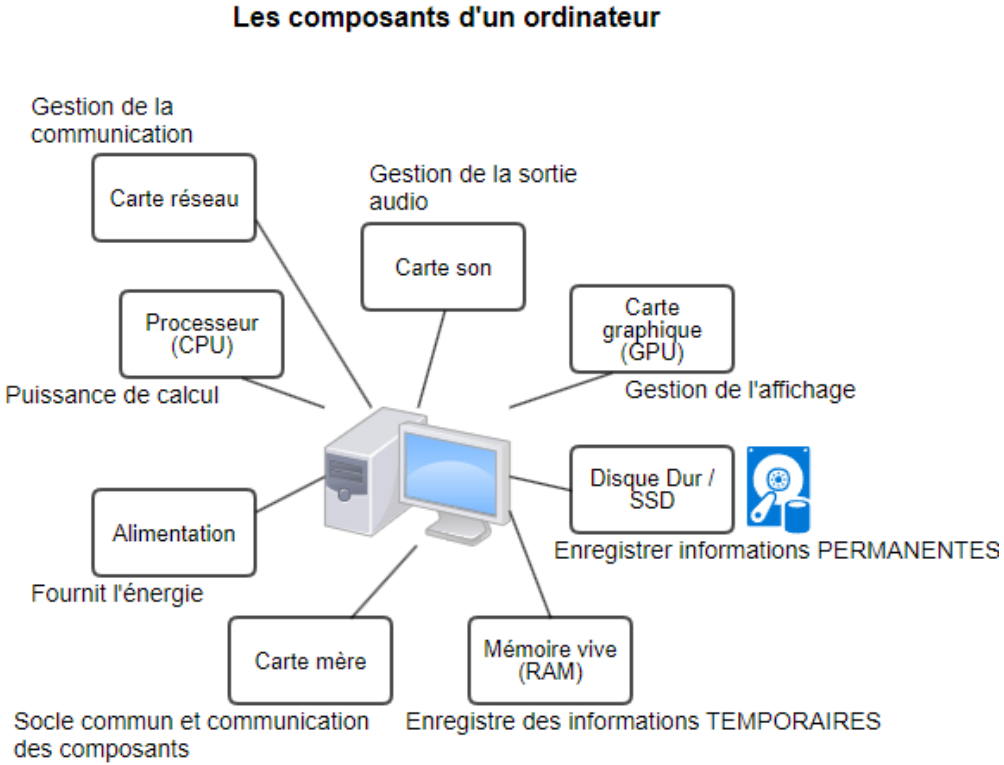
Sommaire :

- Présentation de la virtualisation
- Présentation de la conteneurisation
- Un peu d’histoire

Chapitre 1 : Introduction

Module 1.1 - Rappel des composants d'un ordinateur physique

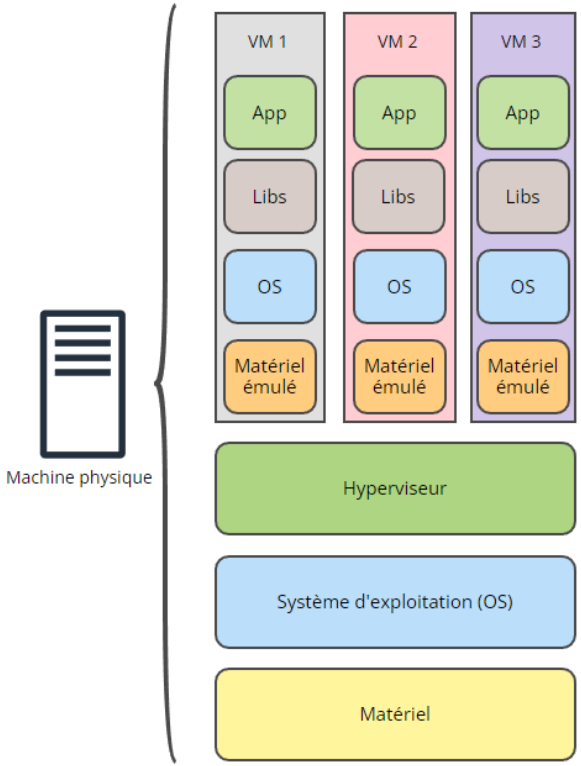
Un ordinateur physique est constitué d'un ensemble cohérent de composants qui permet de réaliser différentes tâches.



Chapitre 1 : Introduction

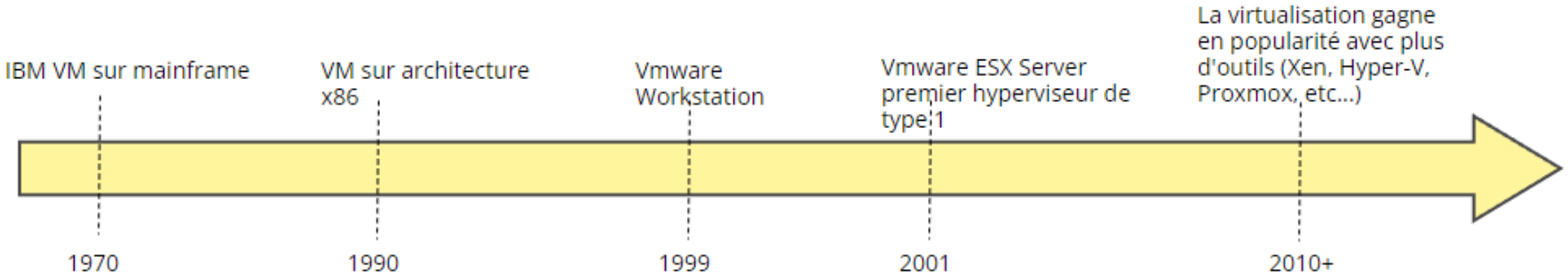
Module 1.1 - La virtualisation, le concept

La virtualisation est une technique informatique qui vise à réaliser une émulation complète d'une machine ou d'un serveur. Cette virtualisation est possible via un hyperviseur.



Chapitre 1 : Introduction

Module 1.1 - La virtualisation, un peu d'histoire



Chapitre 1 : Introduction

Module 1.1 - La virtualisation, les hyperviseurs

Un hyperviseur est une solution de virtualisation qui permet à plusieurs systèmes d'exploitation de fonctionner sur une même machine physique.

Il existe deux types d'hyperviseur :

- Le type 1 (ou hyperviseur natif) : il est directement installé en tant que système d'exploitation, il n'y aura donc pas de système d'exploitation intermédiaire et il sera possible de créer des machines virtuelles (VM) directement.

- Le type 2 (ou hyperviseur hébergé) : il est plus facile à mettre en œuvre, tout comme un programme lambda, l'hyperviseur va s'installer sur un système hôte pour pouvoir fonctionner. Une fois son installation terminée, il sera possible de créer des machine virtuelles (VM).

Chapitre 1 : Introduction

Module 1.1 - La virtualisation, hyperviseurs de type 1

Un hyperviseur de type 1 est une solution privilégiée en entreprise car il est accompagné de fonctionnalités plus avancés (gestion de cluster, permissions, etc...), les performances sont bien meilleures et la totalité des ressources est dédiée aux VM sans passer par un système intermédiaire.

Chapitre 1 : Introduction

Module 1.1 - La virtualisation, hyperviseurs de type 2

Un hyperviseur de type 2 est une solution privilégiée pour réaliser des tests, que cela soit :

- Simuler un réseau sur son poste,
- Tester une application sur un système d'exploitation différent de son système hôte,
- Installer une application sans perturber son propre système.

Parallels®

Chapitre 1 : Introduction

Module 1.1 - La virtualisation, quels avantages ?

La virtualisation permet de réaliser des tâches qui facilite le travail dans le monde de l'informatique, on peut citer par exemple :

- **Réaliser des snapshots** : permet de sauvegarder l'état d'une machine et de pouvoir y revenir
- **Optimiser les ressources** : en fonction des besoins, il est possible d'augmenter ou de réduire les ressources (gain en scalabilité)
- **Améliorer la sécurité** : l'isolation de la machine virtuelle permet de réduire l'impact d'une défaillance système
- **Réduire les coûts** : au lieu d'investir sur un serveur physique pour chaque application, un serveur physique va pouvoir héberger plusieurs serveurs virtuels (et donc plusieurs applications)
- **Simplifier les sauvegardes** : les hyperviseurs proposent souvent des fonctionnalités pour faciliter la sauvegarde des machines virtuelles.

Chapitre 1 : Introduction

Module 1.1 - La conteneurisation, le concept

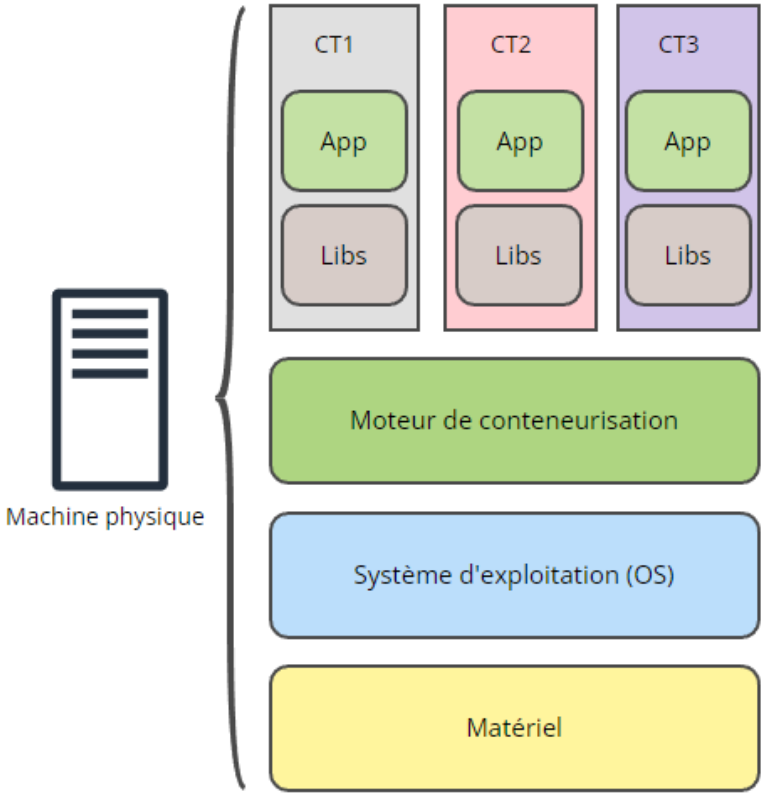
Contrairement à la virtualisation qui émule une machine complète, la conteneurisation (technologie plus récente) s'appuie sur le principe de la para-virtualisation (virtualisation d'un système sans émulation de la partie matérielle).

L'intérêt majeure de la conteneurisation va donc être un gain important de performances, car la « machine virtuelle conteneurisée » va utiliser directement les ressources de la machine physique hôte sans l'émuler.

Chapitre 1 : Introduction

Module 1.1 - La conteneurisation, le concept

Le schéma ci-dessous illustre le concept de la conteneurisation.

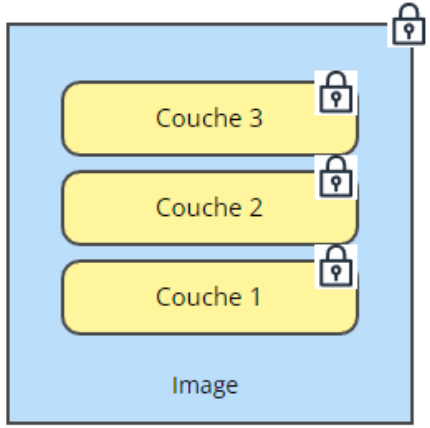


Chapitre 1 : Introduction

Module 1.1 - La conteneurisation, les images

Une image est un package autonome qui contient tous les éléments (code, dépendances, système d'exploitation de base, etc...) pour exécuter une application. Les images sont la base sur laquelle les conteneurs sont créés et exécutés.

Une image est composée de plusieurs couches. Chaque couche représente une modification apportée au système de fichiers. Ces couches sont empilées les unes sur les autres pour former l'image complète. Chaque couche est immuable, ce qui signifie qu'elle ne peut pas être modifiée une fois créée. Cela facilite la gestion des versions et des mises à jour.

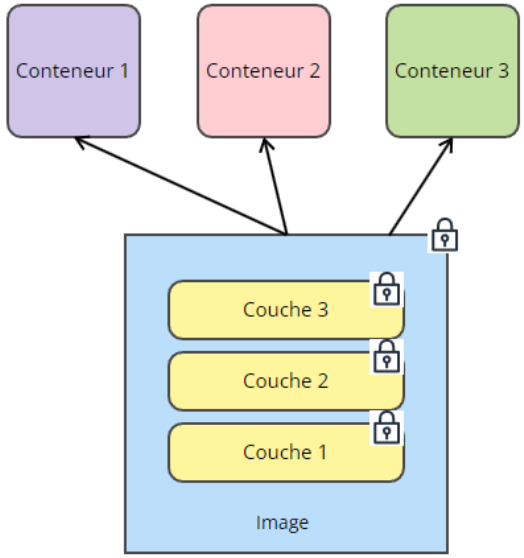


Chapitre 1 : Introduction

Module 1.1 - La conteneurisation, les conteneurs

Un conteneur est une instance d'une image. Il s'agit d'un environnement isolé qui utilise l'image comme base, avec son propre système de fichiers, ses processus, son espace mémoire, et son réseau.

Les conteneurs sont créés à partir d'une image spécifique. Lorsqu'un conteneur est lancé, il hérite des caractéristiques de l'image, mais il devient une entité distincte et indépendante. Les conteneurs peuvent être démarrés, arrêtés, supprimés et mis à l'échelle dynamiquement en fonction des besoins de l'application.



Chapitre 1 : Introduction

Module 1.1 - La conteneurisation, l'objectif principal

L'objectif principal de la conteneurisation va s'inspirer du domaine de la logistique, c'est à dire :

- Empaqueter un ensemble de ressources / produits → une application informatique et ses dépendances
- Livrer le tout à une destination → un serveur informatique
- Sans devoir analyser / prendre en compte le contenu

L'image ci-dessous illustre notre conteneurisation :

Chapitre 1 : Introduction

Module 1.1 - La conteneurisation, les moteurs d’exécutions

Les runtimes de conteneurs sont des composants logiciels responsables de l'exécution des conteneurs. Ils fournissent l'environnement d'exécution nécessaire pour que les conteneurs fonctionnent de manière isolée. Voici quelques exemples de runtimes de conteneurs populaires :

- **Docker** : runtime le plus utilisé sur le marché, il a joué un rôle important dans la popularisation de la conteneurisation.
- **Containerd** : runtime de conteneurs qui a évolué à partir du moteur de conteneurs Docker, il est conçu pour être plus modulaire, offrant un sous-ensemble de fonctionnalités de Docker.
- **CRI-O** : runtime de conteneurs léger et optimisé pour les environnements Kubernetes.

Chapitre 1 : Introduction

Module 1.1 - La conteneurisation, Open Container Initiative (OCI)

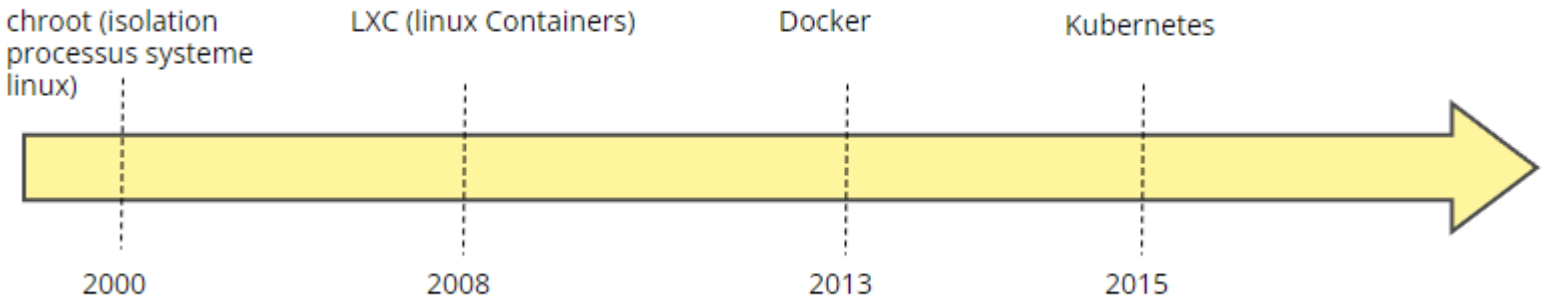
L'Open Container Initiative (OCI) est une initiative open source dont la mission principale est de créer des standards ouverts pour les conteneurs d'applications. Elle vise à assurer l'interopérabilité entre différentes solutions de conteneurisation, permettant aux développeurs et aux entreprises d'utiliser des conteneurs de manière cohérente et portable à travers divers environnements et plateformes.

Les missions clés de l’OCI :

- Standardiser les formats d’images
- Définir les spécifications de runtimes
- Encourager la collaboration et la contribution pour des normes communes
- Promouvoir l’interopérabilité entre les outils

Chapitre 1 : Introduction

Module 1.1 - La conteneurisation, un peu d'histoire



Chapitre 1 : Introduction

Module 1.1 - La conteneurisation, les avantages

La conteneurisation offre des avantages importants dans nos infrastructures informatiques, on peut notamment cité :

- **L'isolation légère** : les conteneurs partagent le même noyau et des ressources mais ils sont isolés en terme de processus, systèmes de fichiers et réseaux → offre une sécurité en isolant les systèmes
- **Portabilité** : les conteneurs embarquent l'ensemble des librairies et dépendances pour qu'une application puisse fonctionner de manière portable sur différents systèmes hôtes.
- **Déploiement rapide :** les conteneurs peuvent être lancés en quelques secondes, cette fonctionnalité permet d'accélérer les déploiements.
- **Optimisation des performances** : Les conteneurs partagent le même noyau et donc utilisent moins de ressources que les machines virtuelles
- **Orchestration** : des solutions comme Kubernetes permettent une gestion automatisée et l'évolutivité des conteneurs à plus grande échelle

Chapitre 1 : Introduction

Module 1.2
Avantages de la conteneurisation par rapport à la virtualisation traditionnelle

Chapitre 1 : Introduction

Module 1.2 – virtualisation vs conteneurisation

La virtualisation et la conteneurisation sont deux approches distinctes pour fournir des environnements d'exécution isolés, mais elles présentent des différences fondamentales.

- Le tableau ci-dessous exprime une comparaison entre ces deux concepts.

Aspect	Virtualisation	Conteneurisation
Isolation / sécurité	Isolation complète (car une VM execute un OS distinct de l'hôte). et une meilleure sécurité	Isolation légère, car utilise les mêmes ressources que l'hôte
Performance	Moins performant car chaque VM nécessite une émulation complète	Très performant, car les conteneurs partagent les ressources de l'hôte directement
Portabilité	Cela dépend des hyperviseurs, donc la portabilité est plus complexe	Très portable, les conteneurs peuvent facilement être executés sur différents systèmes
Espace disque / Taille	Beaucoup plus volumineux (car on doit installer un système)	Très léger

Chapitre 1 : Introduction

Module 1.2 – virtualisation vs conteneurisation

En conclusion, il est impossible d’indiquer qu’une solution (virtualisation ou conteneurisation) est meilleure qu’une autre, il faudrait plutôt indiquer qu’une solution est plus adaptée qu’une autre dans un besoin précis.

D’un autre point de vue, on pourrait même indiquer que ces solutions sont complémentaires. En effet, rien n’interdit de placer une solution de conteneurisation sur un machine virtuelle…

Chapitre 1 : Introduction

Module 1.1 - Questions ?