Docker

Contenu du chapitre:

2022-2023

- La virtualisation
- Les conteneurs
- Docker

Table des matières

2022-2023

Virtualiser

Virtualisation de systèmes d'exploitation: technique consistant à faire fonctionner en même temps, sur un seul ordinateur, plusieurs systèmes d'exploitation comme s'ils fonctionnaient sur des ordinateurs distincts. On parle alors de Virtual Machines (VM).

hyperviseur: processus qui crée et exécute des machines virtuelles en répartissant les ressources (mémoire, CPU) de l'ordi.

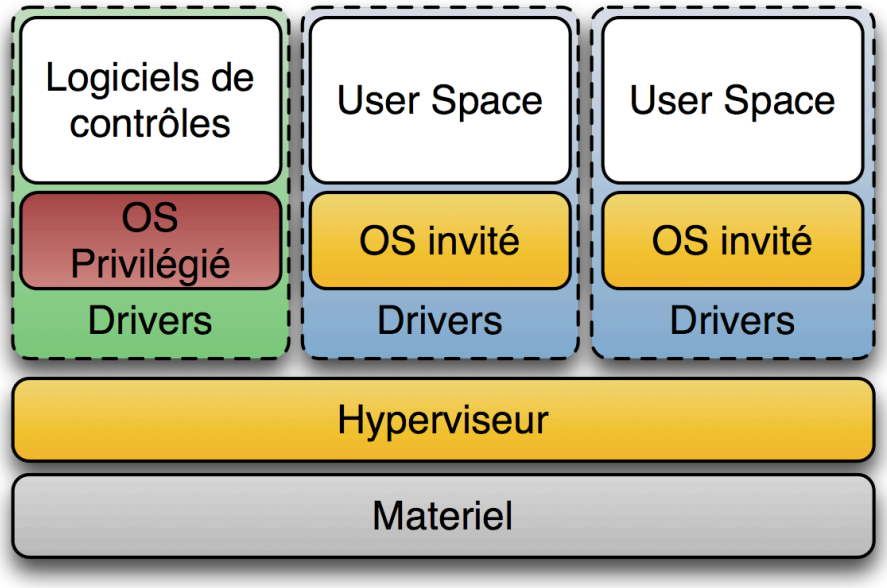
[https://doc.ubuntu-fr.org/virtualisation]

Pourquoi virtualiser?

- Minimiser les coûts matériels en utilisant mieux les machines
  - exécuter plusieurs OS/applications simultanément sur une même machine
- Faciliter l'automatisation/déploiement/la flexibilité/passage à l'échelle. Permet :
  - d'exploiter des logiciels et périphériques ne fonctionnant pas dans
  - de tester des logiciels dans des environnements contrôlés
  - de transporter application et OS d'un ordinateur à tout autre ordinateur ayant un hyperviseur compatible.
  - de sécuriser (isolation des invités entre eux, des invités par rapport à l'hôte)
  - de faciliter les redémarrages

Inconvénient: augmente la complexité, et la couche d'abstraction nuit aux performances/consomme des ressources.

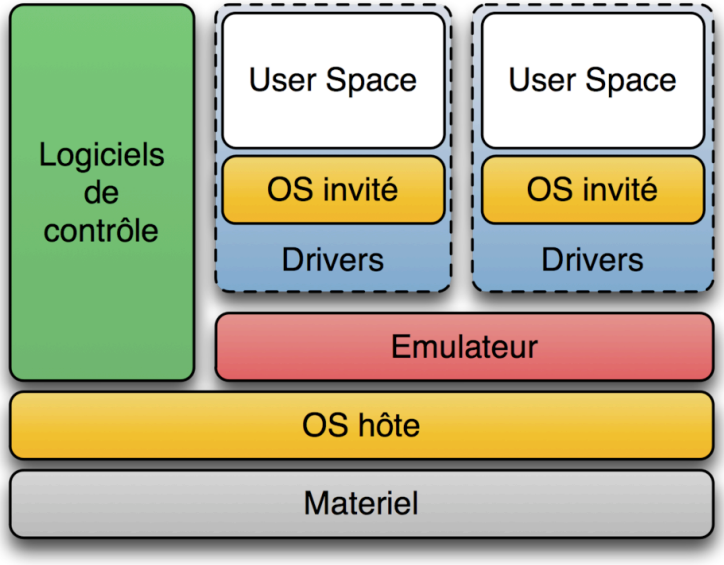
Paravirtualisation = type1



paravirtualisation (hyperviseur type 1 = bare metal): hyperviseur simple, s'exécute directement sur le matériel de lhôte, performance proche du matériel réel. Mais l'OS invité doit être adapté.

ex: VMware vSphere, MS Hyper-V Server

Virtualisation complète= type2

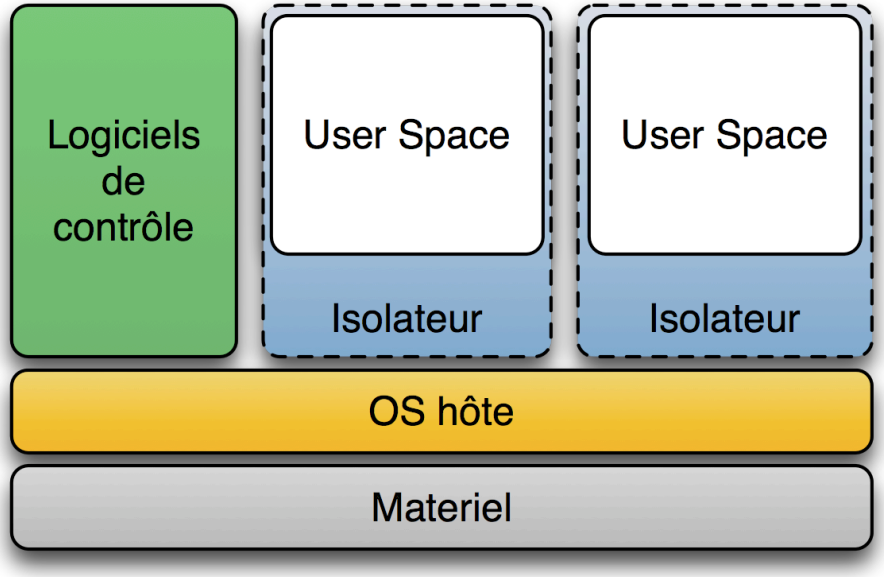


virtualisation complète (hyperviseur type 2): sexécute sur l'OS hôte en simulant complètement l'ordinateur (matériel). Par contre l'unité centrale (processeur, RAM et stockage dans fichier) sur l'hôte restent accessibles aux invités.
La solution la plus courante pour les particuliers. On ne peut virtualiser qu'un OS invité utilisant une archi matérielle similaire à l'hôte (ex: x86).

ex: MS VirtualPC, Oracle VM VirtualBox, VMWare Player, VMWare Workstation

Très similaire à:
émulation: comme virtualisation complète mais en plus on simule l'unité centrale. L'architecture matérielle invitée et hôte peut alors différer. Performance médiocre.

Conteneurs



environnement virtuel (isolateur): Chaque environnement utilise son propre espace mémoire isolé, mais ressources systèmes (pilote, noyau) partagées sur l'hôte. Conteneurs "légers", mais les environnements doivent pouvoir s'exécuter sur l'OS hôte.

Conteneurs linux, Docker, LXD

Les principales techniques de virtualisation

- *paravirtualisation (hyperviseur type 1 = bare metal):* hyperviseur simple, s'exécute directement sur le matériel de lhôte, performance proche du matériel réel. Mais l'OS invité doit être adapté. ex: VMware vSphere, MS Hyper-V Server
- *virtualisation complète (hyperviseur type 2):* sexécute sur l'OS hôte en simulant complètement l'ordinateur (matériel). Par contre l'unité centrale (processeur, RAM et stockage dans fichier) sur l'hôte restent accessibles aux invités.
  La solution la plus courante pour les particuliers. On ne peut virtualiser qu'un OS invité utilisant une archi matérielle similaire à l'hôte (ex: x86).
  ex: MS VirtualPC, Oracle VM VirtualBox, VMWare Player, VMWare Workstation
- *émulation:* comme virtualisation complète mais en plus on simule l'unité centrale. L'architecture matérielle invitée et hôte peut alors différer. Performance médiocre.
- *environnement virtuel (isolateur):* Chaque environnement utilise son propre espace mémoire isolé, mais ressources systèmes (pilote, noyau) partagées sur l'hôte. Conteneurs "légers", mais les environnements doivent pouvoir s'exécuter sur l'OS hôte. Conteneurs linux, Docker, LXD

Table des matières

2022-2023

Les conteneurs

Un conteneur est un silo léger et isolé qui permet d'exécuter une application sur l'OS hôte.

- isolation des conteneurs: ne peuvent accéder aux autres sans autorisation explicite.
- le conteneur contient ses propres fichiers et toutes les données dont il a besoin.
- quand le conteneur est détruit, les données ne sont pas conservées.
- tout conteneur est créé à partir d'une image.
- le conteneur peut être interrompu, redémarré, sauvegardé, transféré sur une autre machine.

Process vs Container vs VM

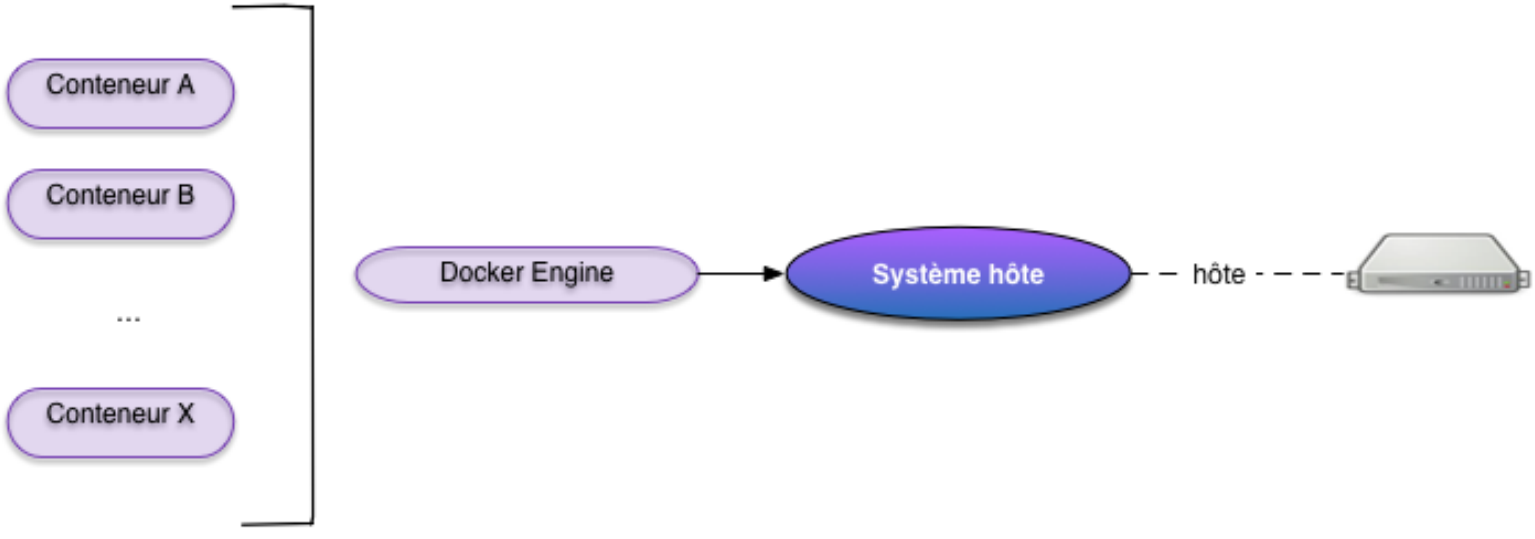
	Process	**Container**	**VM**
definition	running instance of a program	isolated group of processes managed by a shared kernel	an OS that shares host hardware through a supervisor
OS for virtualized apps	same OS	same kernel	multiple indepdt OS
Isolation	memory space and user privileges	namespaces and cgroups	full OS isolation
Security	-	improving	higher
Size	user application	image = some MBs + user app	image = some GBs + user app
Lifecycle	created by forking, generally short-lived	runs on kernel with hardly any boot (1s), generally short lived	needs boot process (>10s), generally long lived.
Computer overhead	-	<5%	>10%
Disk I/O overhead	-	generally 0	>50%
Communication	inter-process (IPC)	IPC	network devices
Support	all OS	mostly linux	all OS

Table des matières

2022-2023

Docker

Transparents tirés de :
`http://b3d.bdpedia.fr/docker.html` (qui est expliqué plus en détail).
Docker (vient de Dock worker, qui travailler avec les conteneurs maritimes).
Implémenté en Go.
Le docker engine est le programme qui gère les conteneurs. On lui transmet des instructions via le client docker sur le système hote (ex: instruction `docker` en ligne de commande).
Docker permet d'émuler un *système distribué* de serveurs: un ensemble de processus tournant chacun sur un serveur.
Un *serveur* étant une machine connectée en permanence au réseau via des ports; un serveur machine est identifiable sur le réseau par son adresse IP.



Docker: images

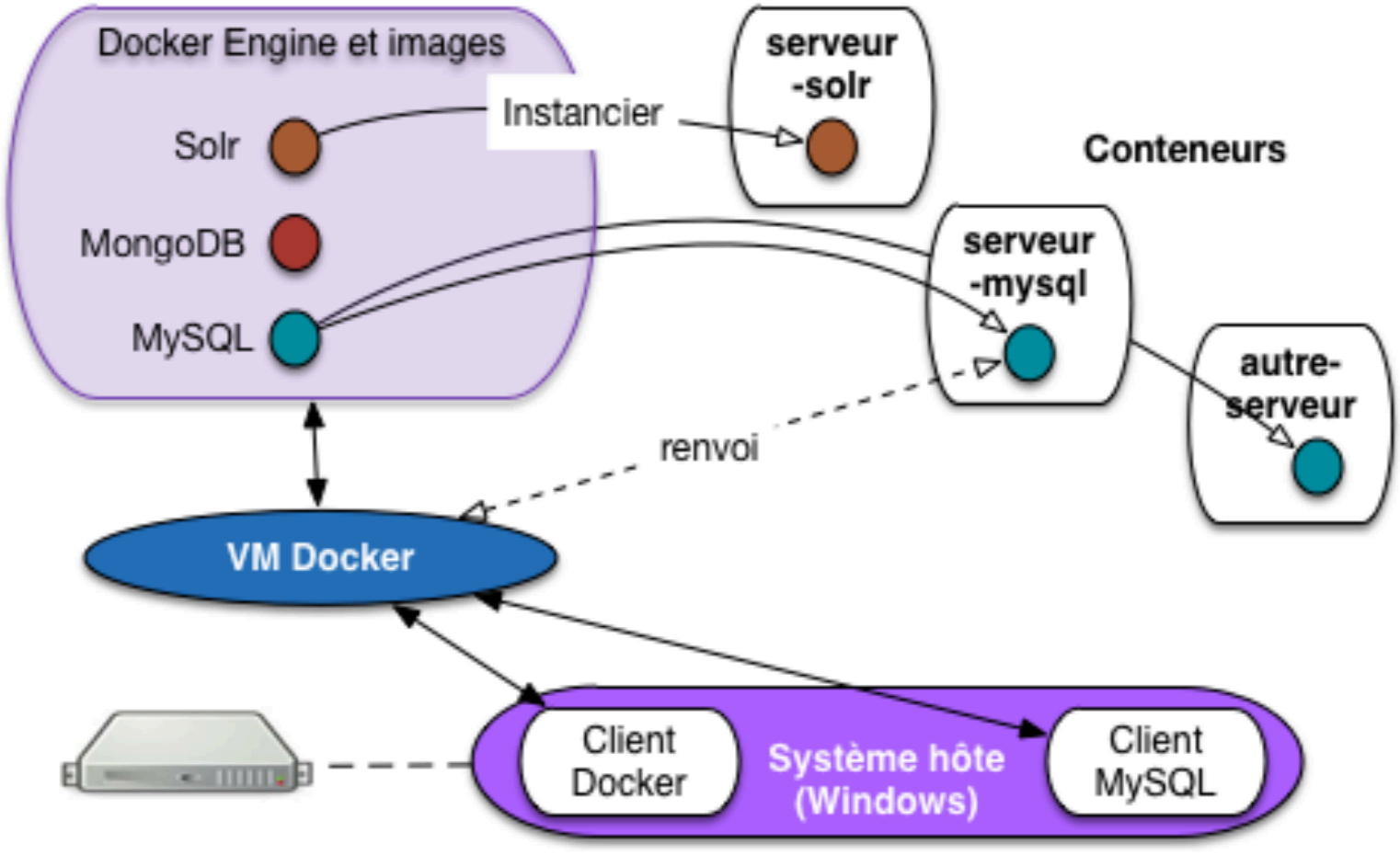


Fig. 2.3 Les images docker, constituant un pseudo système distribué

Les images sont téléchargées dans docker engine, depuis le répertoire en ligne.

Chaque image peut ensuite être instanciée dans un ou plusieurs conteneur.

Docker: instructions pour gérer les images

`docker pull <image>` télécharge une image docker (mais run les télécharge automatiquement si besoin).

`docker images` liste les images déjà téléchargées.

`docker rmi <image>` efface une image téléchargée.

Docker: instructions pour gérer les conteneurs

`docker run <image> [programme sur l'image]` pour créer et activer un conteneur.

Options de docker run:

`--help`

`-d` lance le conteneur en mode détaché (i.e., en arrière-plan).

`--name <conteneur>` permet de choisir le nom du conteneur

`--rm <conteneur>` efface le conteneur quand on le quitte

`-it <conteneur>` nécessaire pour lancer un programme interactif de type shell

`-v <source>:<chemin>` pour monter le volume source dans le répertoire chemin du conteneur

`docker exec -it <conteneur> <application>` lance l'application dans un conteneur en cours d'exécution.

Docker: instructions pour gérer les conteneurs (2)

`docker cp <fichier> <conteneur>:<fichier>` copie le fichier dans le conteneur, depuis l'hôte. Comportement similaire à cp -a sous Unix; si docker cp repertoire, copie récursivement le contenu.

`docker ps -a` liste les conteneurs, qu'ils soient actifs ou non.

`docker rm -f <conteneur>` efface un conteneur (`-f` (force) étend l'arrêt aux conteneurs actifs).

`docker restart <conteneur>` relancer un conteneur interrompu.

Exemples:

```
docker run -it --name mon-linux-shell ubuntu bash
```

nom du conteneur — image — programme dans l'image

```
docker exec -it mon-linux-shell sh
```

(2^{nd} shell sur même conteneur)

Docker: instructions pour faire le ménage

Lorsque vous en avez fini:

```
#!/bin/bash

# remove  containers:
docker ps -aq | xargs -r docker rm -f

# remove unused images (those without running containers):
docker images --no-trunc | awk '{ print $3 }' | xargs -r docker rmi

# remove unused volumes:
docker volume ls -qf dangling=true | xargs -r docker volume rm
```

Bibliographie

https://www.cse.wustl.edu/~jain/cse570-18/ftp/m_21cdk4.pdf

https://medium.com/@jessgreb01/
what-is-the-difference-between-a-process-a-container-and-a-vm-f36ba0f8a8f7

https://fr.wikipedia.org/wiki/Virtualisation

http://b3d.bdpedia.fr/docker.html