Cloud - 3

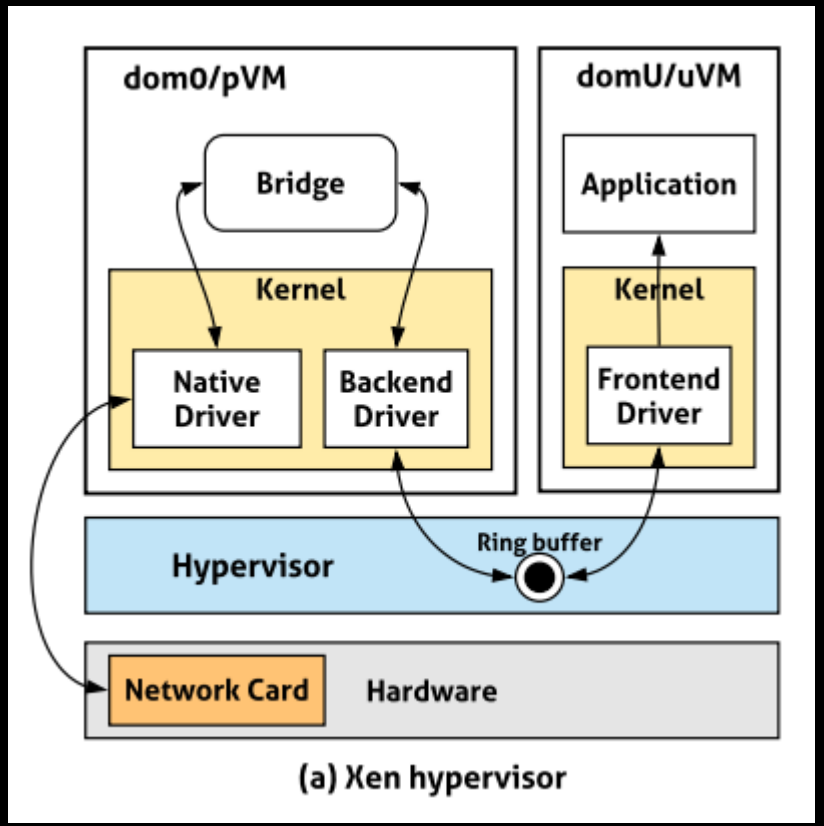
Djob Mvondo

Virtualisation : Concepts techniques

Xen : La gestion des périphériques

Architecture « split-driver »: similaire au « client-serveur »

- Exploite le dom0 qui contient les librairies pour accéder au matériel
- Chaque unité a un représentant qui communique avec le dom0 pour s'échanger les requêtes/réponses.

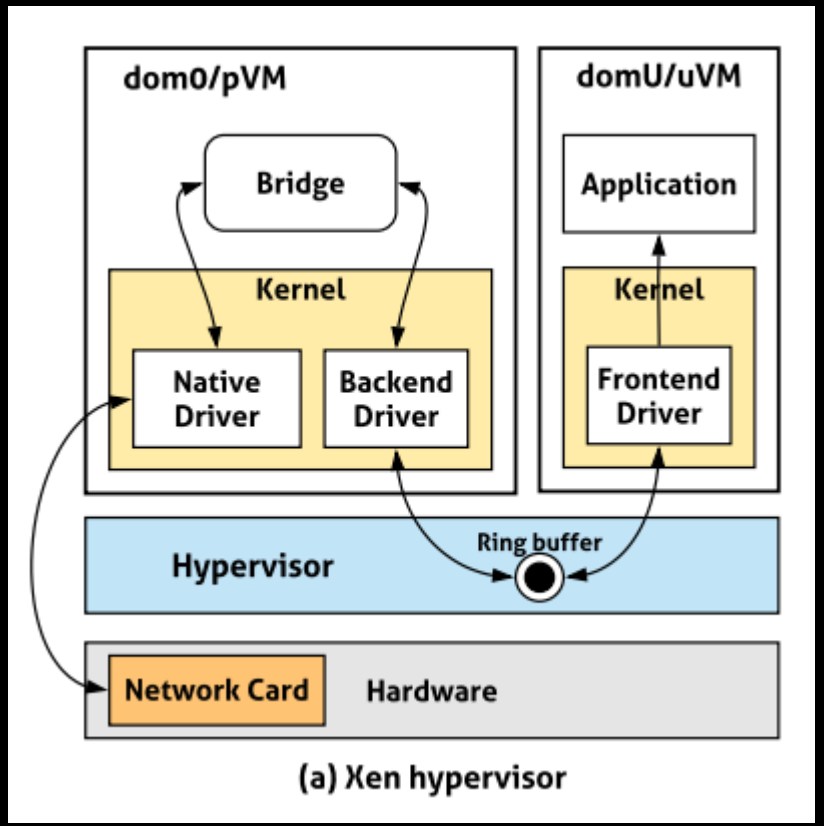


Virtualisation : Concepts techniques

Xen : La gestion spécifique du réseau

Architecture « split-driver »: similaire au « client-serveur »

- Plusieurs mode de réseau
  - Bridge (Pont)
  - NAT
  - Route

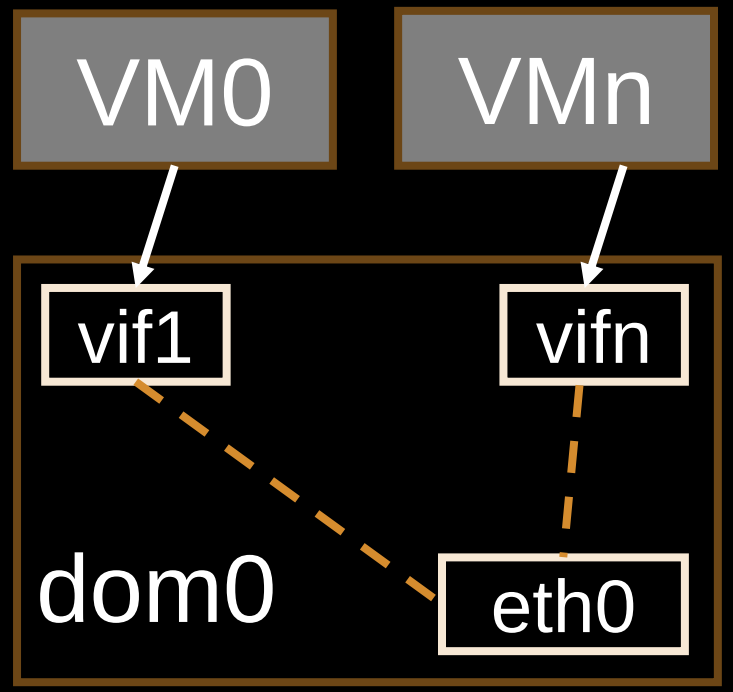


Virtualisation : Concepts techniques

Xen : La gestion spécifique du réseau

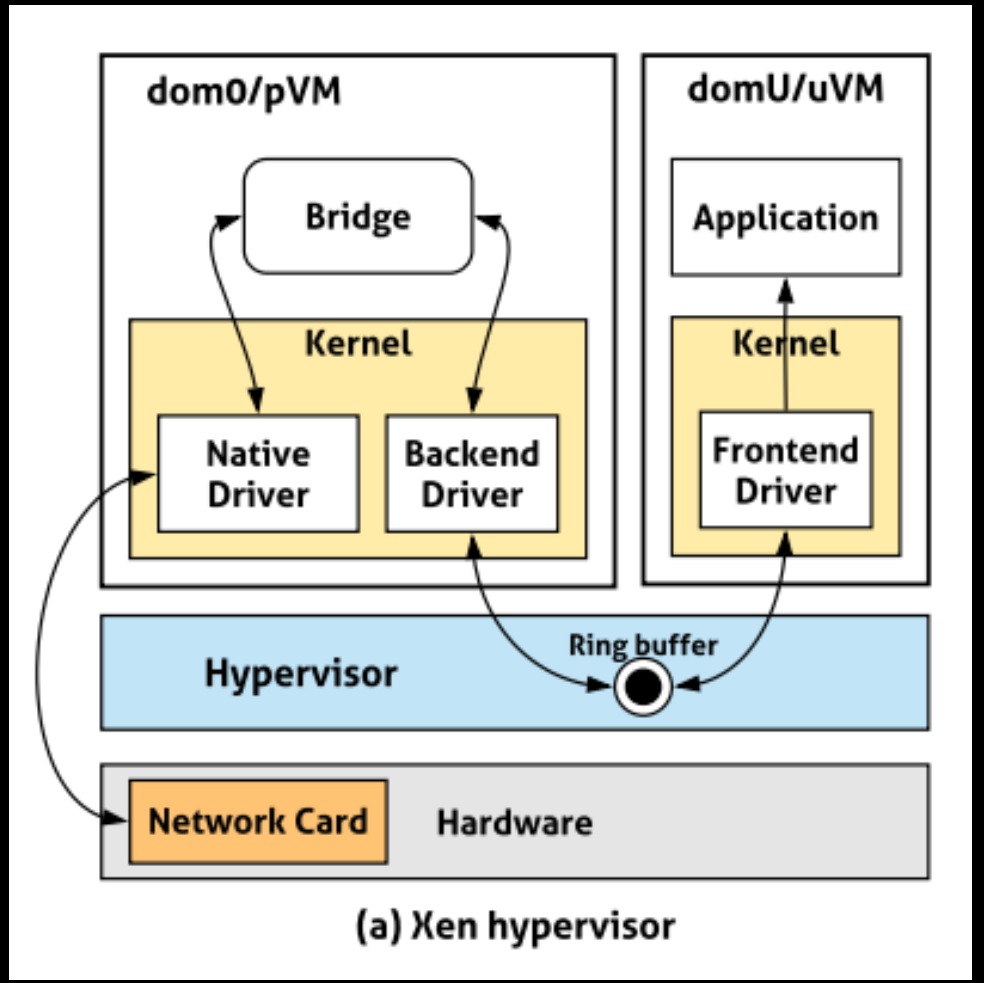
Architecture « split-driver »: similaire au « client-serveur »

- Bridge le plus utilisé, chaque VM
  - A une interface virtuelle (vif)
  - Est relié à l'interface réseau (ethx)
  - Est accessible de l'extérieur



Virtualization infrastructure

The **split driver model** is often used:
Frontend/Backend + Ring buffer idea



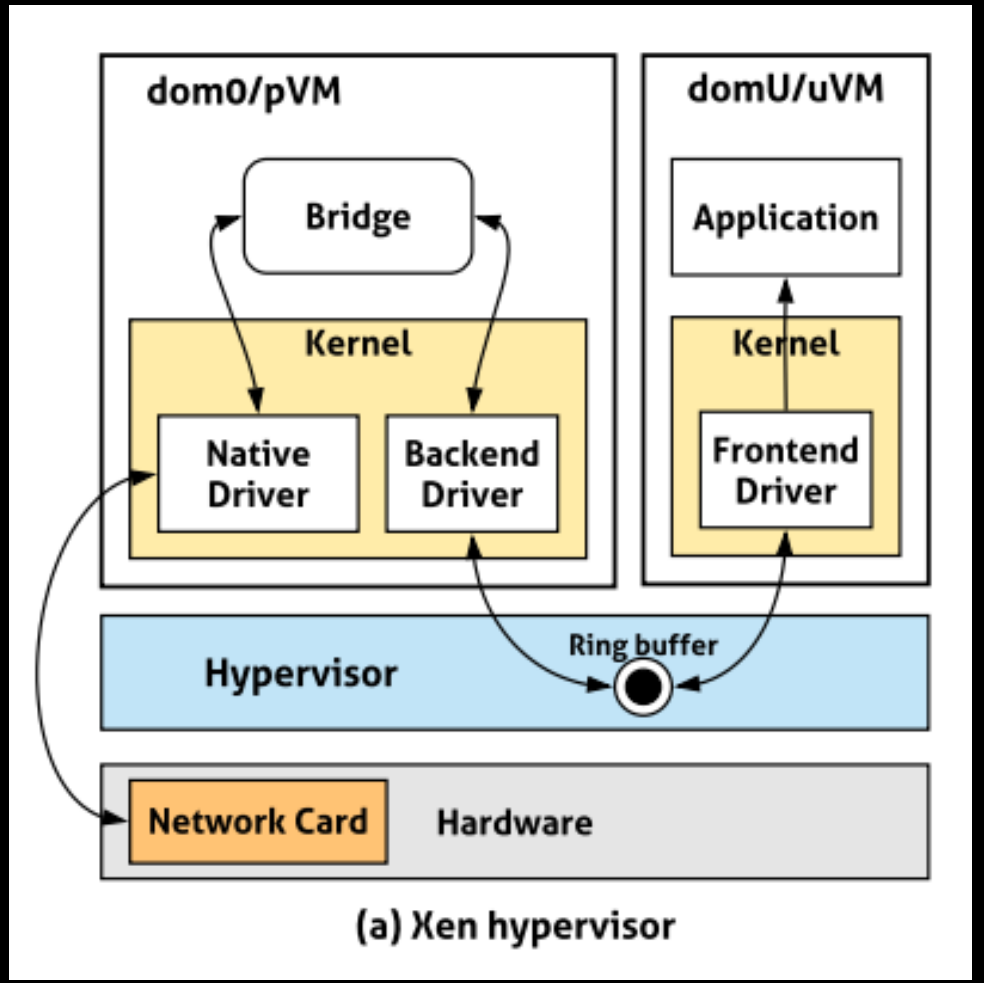
Virtualization infrastructure

The **split driver model** is often used:
Frontend/Backend + Ring buffer idea

Modularity

Performance

Existing code reuse



Virtualization infrastructure

The **split driver model** is often used:
Frontend/Backend + Ring buffer idea

Modularity

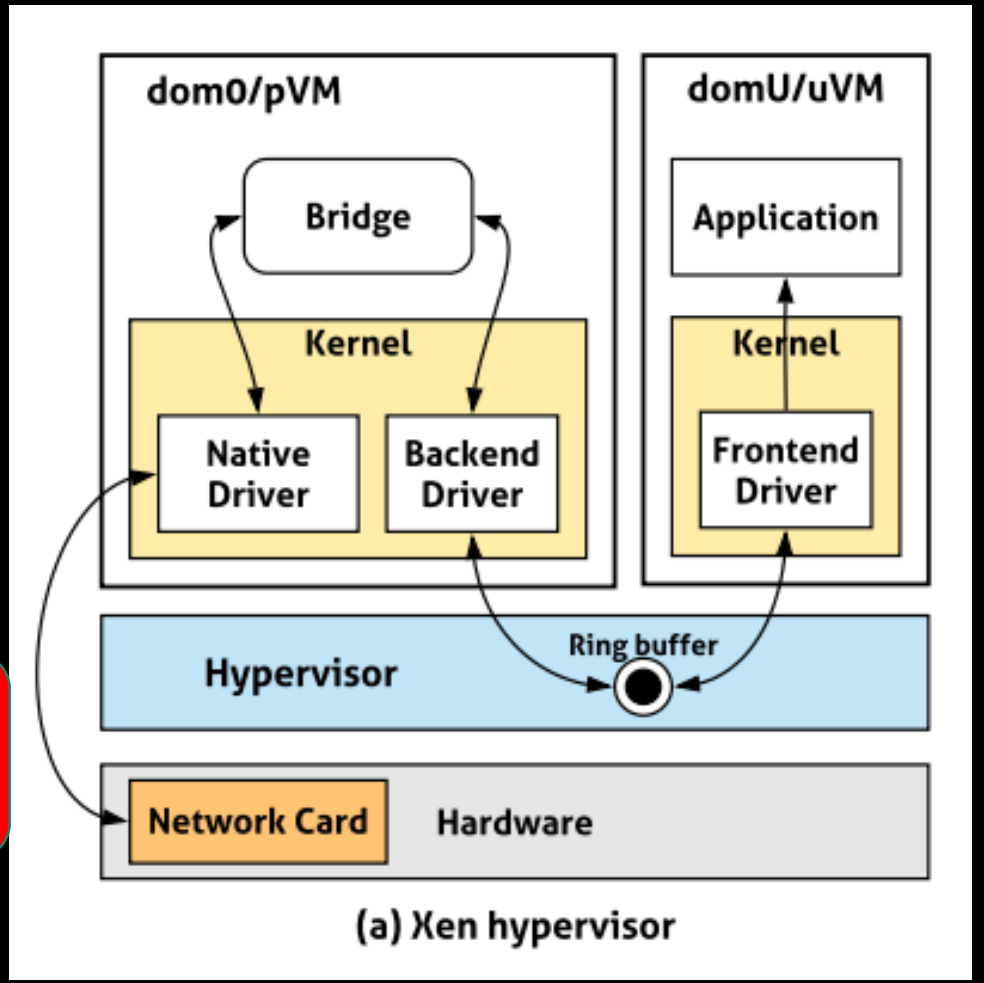
Performance

Existing code reuse

Single point of failure and bottleneck for the pVM

Bottleneck on the backend driver

Memory issues regarding ring buffers



Résilience et dimensionnement des environnement virtualisés. Djob Mvondo. Dec 2020

Single point of failure and bottleneck illustration

The **split driver model** is often used:
Frontend/Backend + Ring buffer idea

Modularity

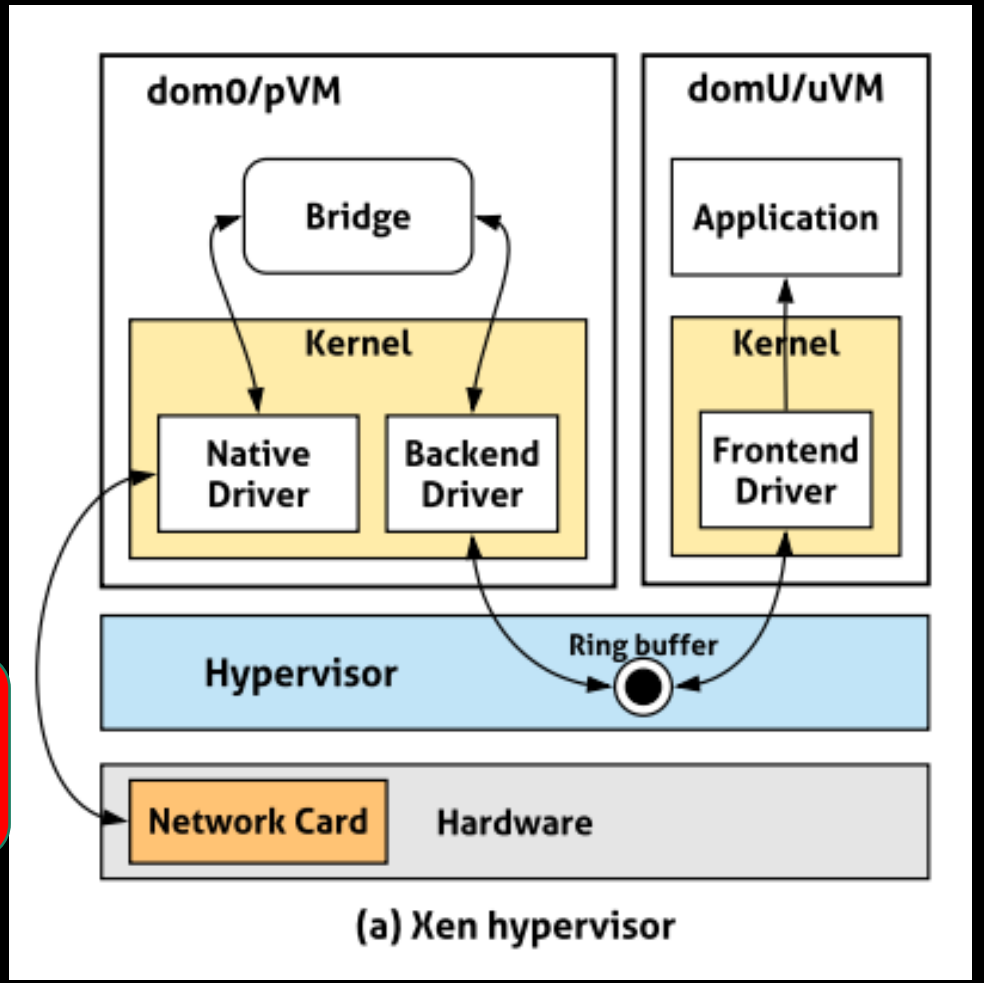
Performance

Existing code reuse

Single point of failure and bottleneck for the pVM

Bottleneck on the backend driver

Memory issues regarding ring buffers



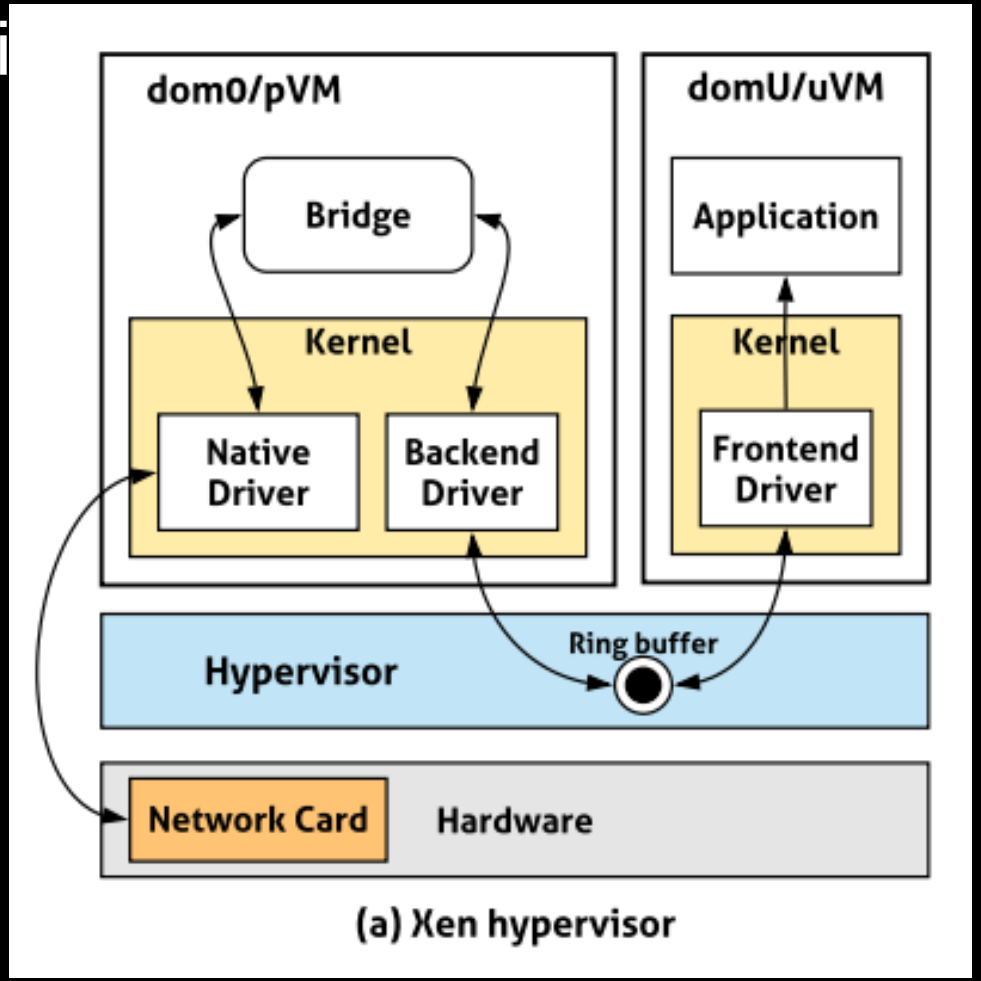
Résilience et dimensionnement des environnement virtualisés. Djob Mvondo. Dec 2020

Mitigating single point of failures

The key idea is to decompose the single poi
failure to reduce the **blast radius** in case of
problems.

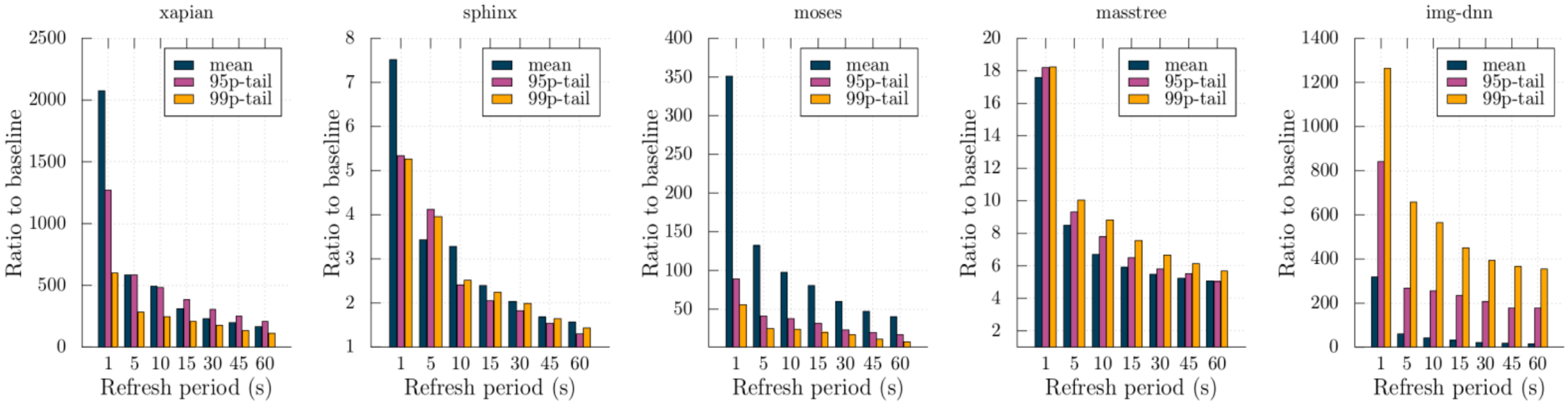
***Full replication[1]**: Replicate virtualized components across the data center*

- *Resource consuming*
- *Synchronization across the different replicas*



[1] https://nutanixbible.com

Mitigating single point of failures



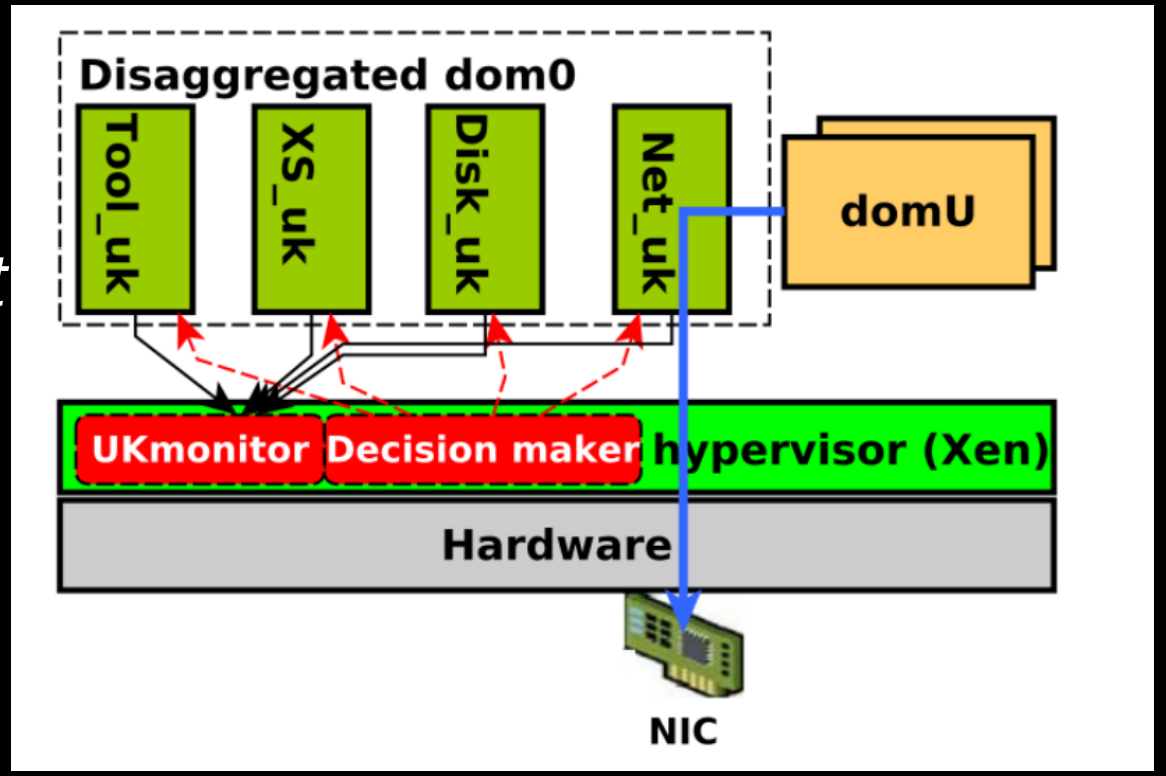
Djob Mvondo et al. Fine-Grained Fault Tolerance For Resilient pVM-based Virtual Machine Monitors. DSN'20

[2] Colp et al. Breaking Up is Hard to Do: Security and Functionality in a Commodity Hypervisor. SOSP'11

Mitigating single point of failures

The key idea is to decompose the single point of failure to reduce the **blast radius** in case of problems.

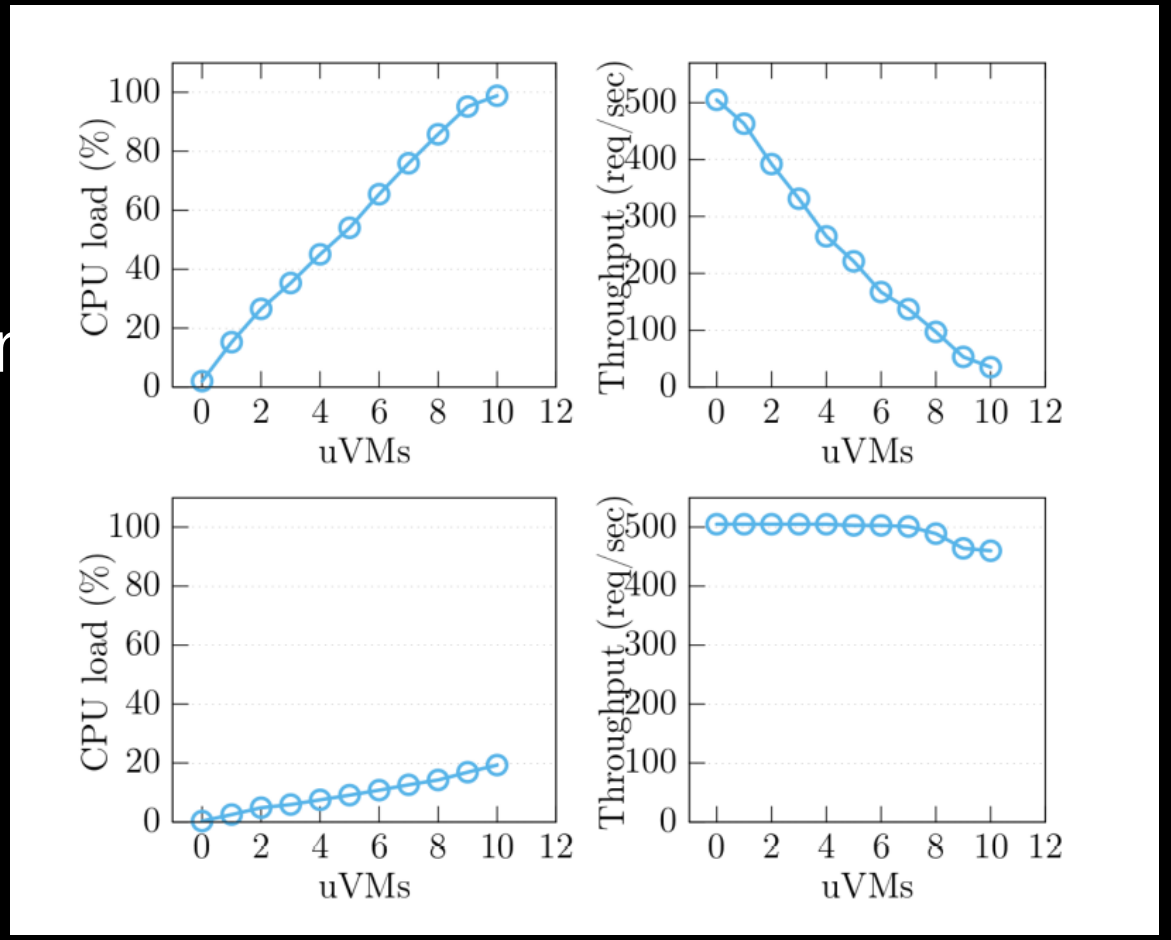
Disaggregation* + *Specialization* + *Pro-activity: *Reuse Xoar idea without the periodic reboot but introduce a tailored monitoring and recovery mechanism for each sub-component.*



[1] Mike Swift et al. Recovering Device Drivers. OSDI'04
[2] Djob Mvondo et al. Fine-Grained Fault Tolerance For Resilient pVM-based Virtual Machine Monitors. DSN'20

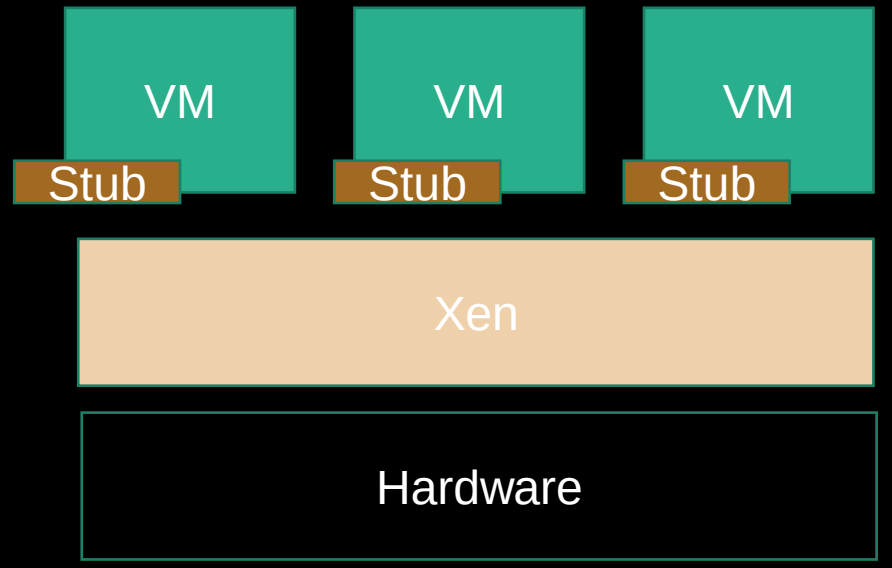
Mitigating bottlenecks

Bottlenecks can cause degradation or **application performance** and affect **response times**.



Djob Mvondo et al. Closer: A new design principle for the privileged virtual machine OS. MASCOTS 2019

Mitigating bottlenecks



Bottlenecks are mitigated by trying to **reduce the load** on the target component when **input load increases.**

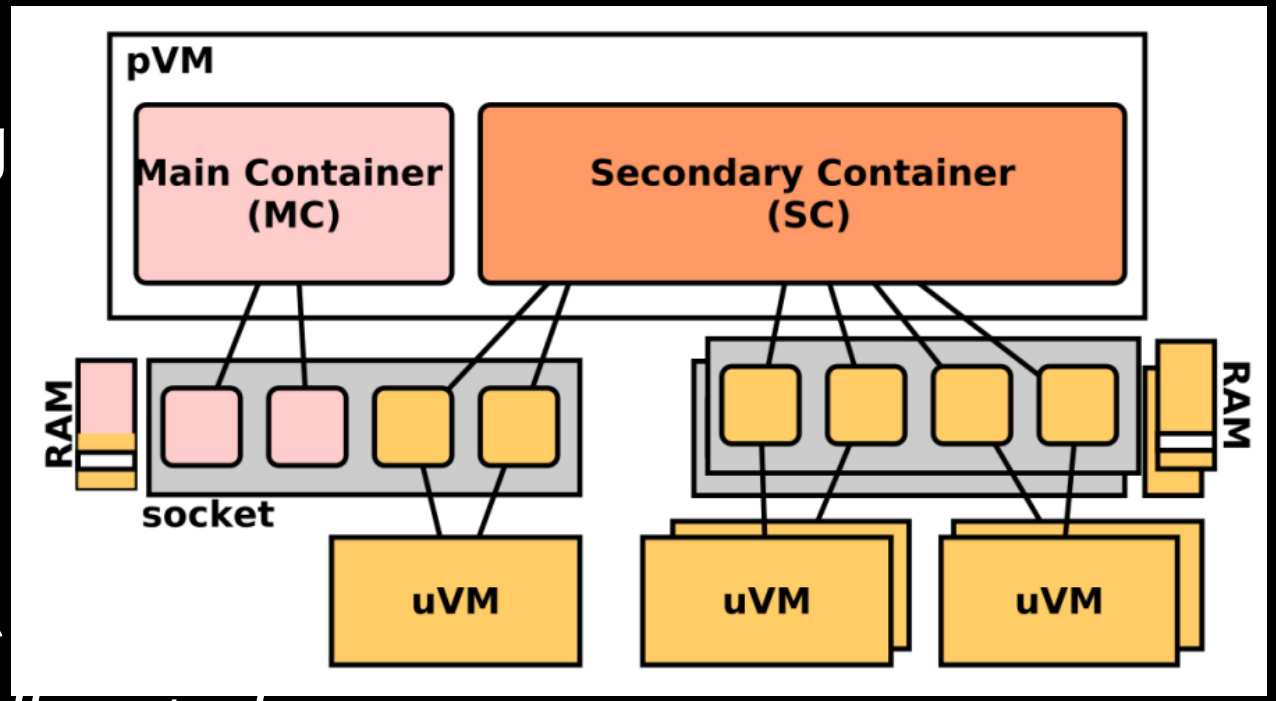
Stub-domains[1]: *Dedicate a specific component for each VM responsible to only help that VM.*

- *Quid of resource provisioning and positioning ?*

[1] Xen studdomains: https://xenproject.org

Mitigating bottlenecks

Bottlenecks are mitigated by trying **reduce the load** on the target component when **input load increases.**



Closer principle[1]: *Stubdomains provisioned automatically on VM allocated resources leaving out administration tasks.*

[1] Djob Mvondo et al. Closer: A new design principle for the privileged virtual machine OS. MASCOTS 2019

Virtualisation : Concepts techniques

Les caractéristiques des machine virtuelles

- Démarrage: **Couteux (mins)**
- Isolation: **Très forte**
- ABI : **Plusieurs OS disponible**
- Taille image : **Assez lourd**

Virtualisation : Concepts techniques

IOT : Ressources limités, isolation faible, réactivité

« On a besoin d'aller vite sans trop se soucier de l'isolation interne mais utiliser les ressources limités efficacement »

Virtualisation : Concepts techniques

Le choix des containers

Un container est un processus

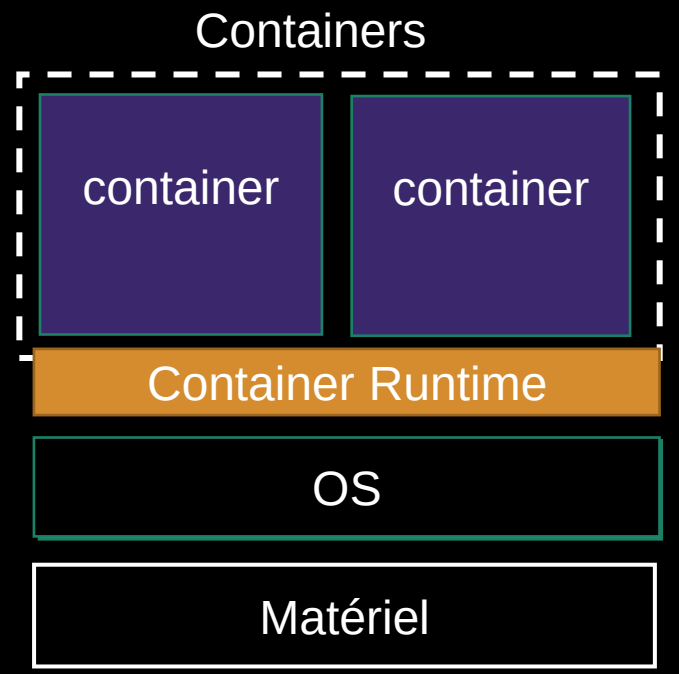
- Isolation de l'OS
- Namespaces, cgroups, …

Exploite les librairies existantes

- Pas de système d'exploitation
- Spécifie au démarrage ces besoins

Plus léger qu'une VM

- Moins **d'indirection** ➔ plus de réactivité



Virtualisation : Concepts techniques

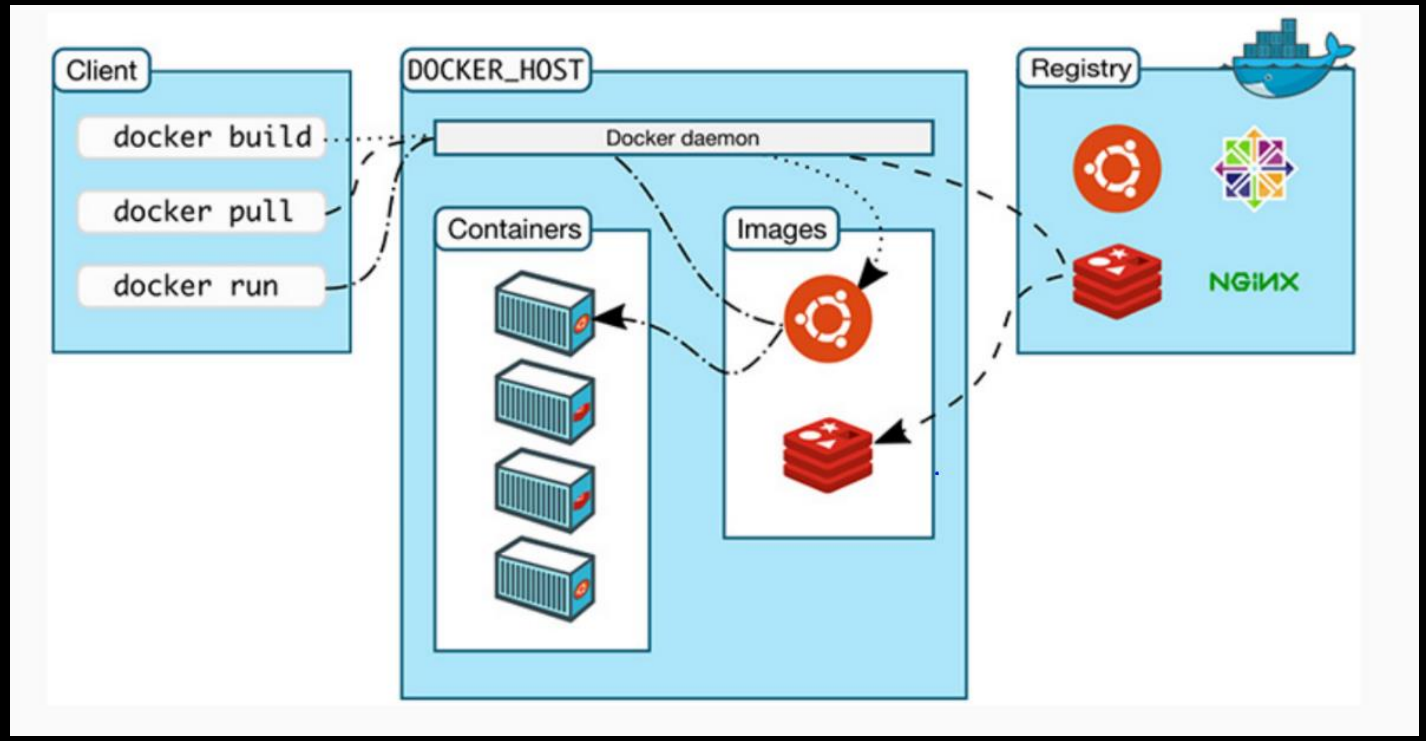
Le choix des containers : Focus sur Docker

Gestionnaire de containers

- Ecrit en Go, 2013

Basé sur containerd

- Open-source



Source: docker.com

Virtualisation : Concepts techniques

Le choix des containers : Focus sur Docker

Dockerfile

```
FROM openjdk:11
RUN apt-get -y upgrade
RUN apt-get -y update
ENV JAVA_HOME /usr/lib/jvm/java-8-oracle
WORKDIR /usr/src/myapp
```

CLI

```
docker build –t  java-en .
docker images list
```

Virtualisation : Concepts techniques

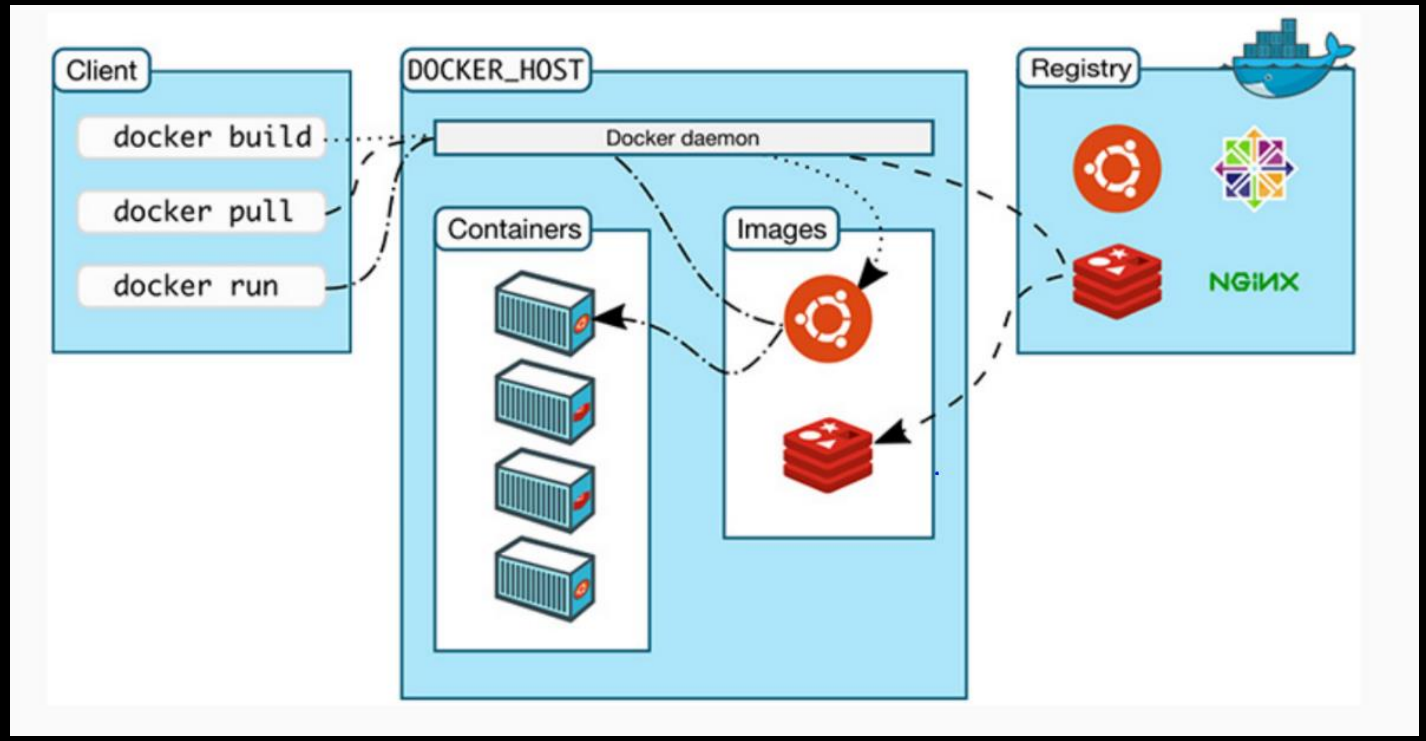
Le choix des containers : Focus sur Docker

Chaque couche est extraite d'un registre

- Local
- Distant (Docker Hub)

Communication via API Rest avec **dockerd**

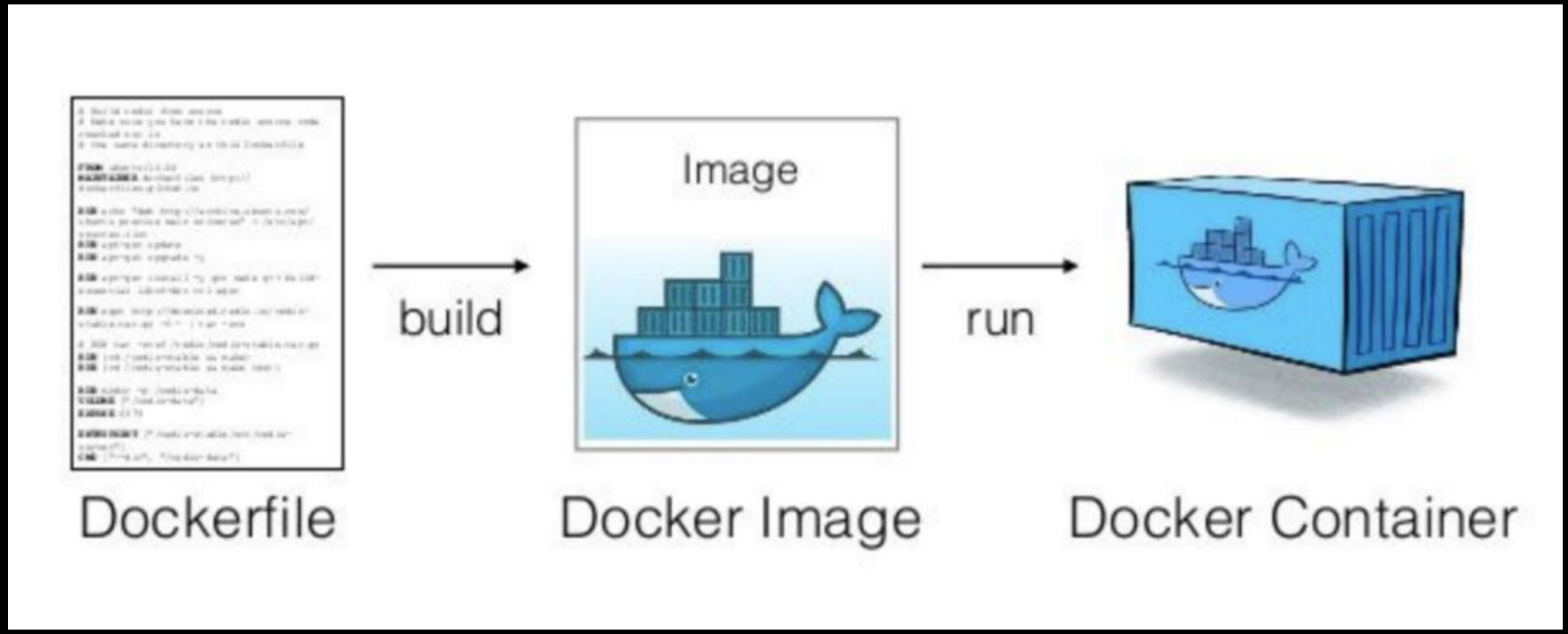
- Gère les images, volumes, etc…



Source: docker.com

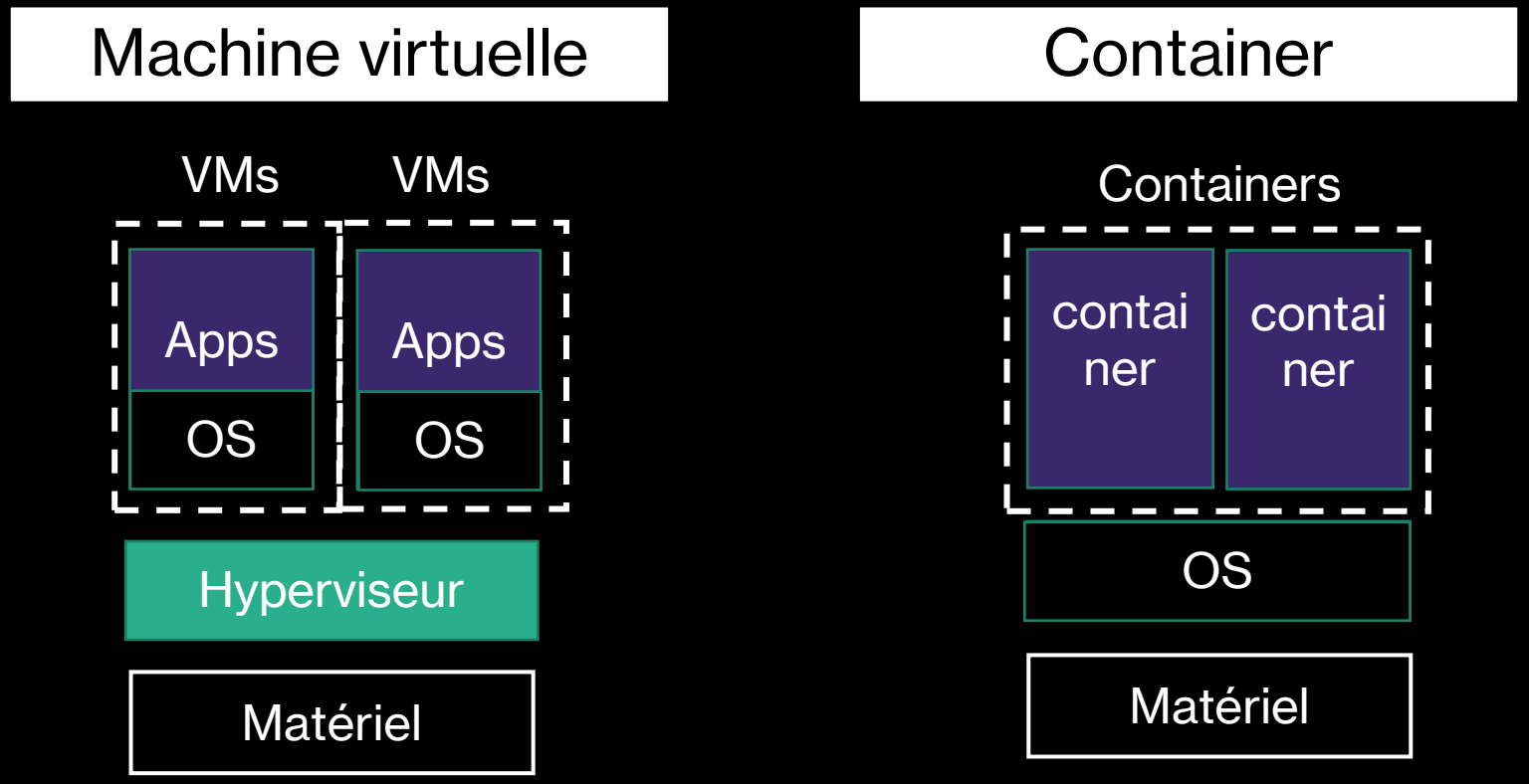
Virtualisation : Concepts techniques

Le choix des containers : Focus sur Docker



Virtualisation : Concepts techniques

Différences entre les containers et VMS



Virtualisation : Concepts techniques

« Impact sur la sécurité et architecture
des applications »