Virtualisation et Conteneurisation

Guillaume Urvoy-Keller - Novembre 2022
Amaury Boller - Révision Octobre 2025 AIRFRANCE

Partie I — Virtualisation des machines

UNIVERSITÉ CÔTE D'AZUR

1. Introduction : la virtualisation dans les infrastructures modernes

La virtualisation est aujourd'hui **le fondement de toute infrastructure réseau et cloud**.
Elle permet de découpler les ressources matérielles physiques (serveurs, stockage, réseau) de leurs usages logiques (VM, routeurs virtuels, services).

Objectifs principaux :

- **Mutualisation** des ressources (optimisation de la charge des serveurs).
- **Isolation** des environnements (sécurité, stabilité).
- **Portabilité et automatisation** des déploiements (VM templates, snapshots).
- **Économie d'énergie** (moins de matériel physique).

💡 **Contexte R&T :** La virtualisation permet de déployer des routeurs, firewalls, serveurs DHCP/DNS, etc. sans matériel dédié.

1. Introduction : la virtualisation dans les infrastructures modernes

Type	OS invité modifié ?	Isolation	Performance	Exemples
Virtualisation complète	Non	🔒 Forte	Moyenne	VMware, KVM
Para-virtualisation	Oui	🔒 Forte	⚡Bonne	Xen,virtio
Conteneurisation	N/A	Moyenne	🚀Excellente	Docker, LXC

2. L'hyperviseur

2.1 Rôle

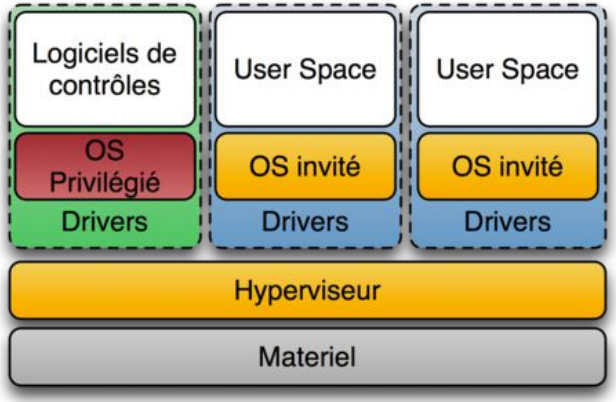
L'hyperviseur est une couche logicielle qui :

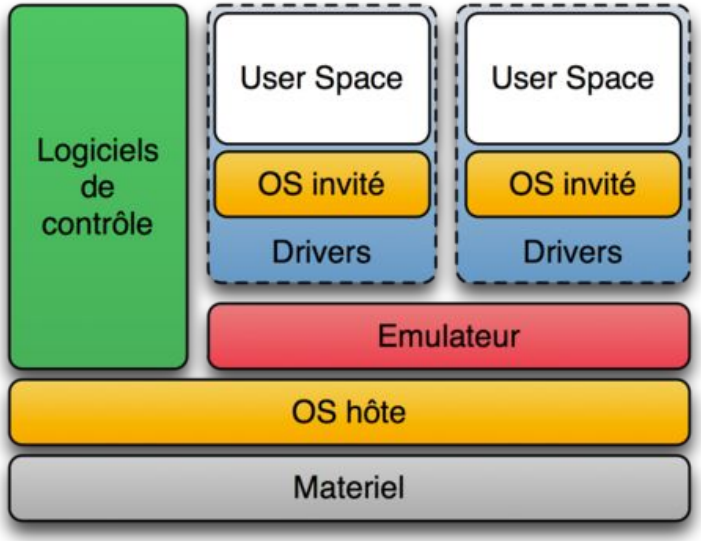
- crée et exécute les **machines virtuelles (VM)**,
- alloue et arbitre les **ressources matérielles** (CPU, mémoire, disques, interfaces réseau).

Type	Description	Exemples
Type 1 (bare metal)	Directement sur le matériel (OS dédié pour l'hyperviseur)	VMware ESXi, KVM, Hyper-V, Proxmox
Type 2 (hébergé)	Sur un OS hote (Windows, Linux, Mac …)	VirtualBox, VMware Workstation

2. L'hyperviseur

2.2 Types





Source: https://fr.wikipedia.org/wiki/Hyperviseur

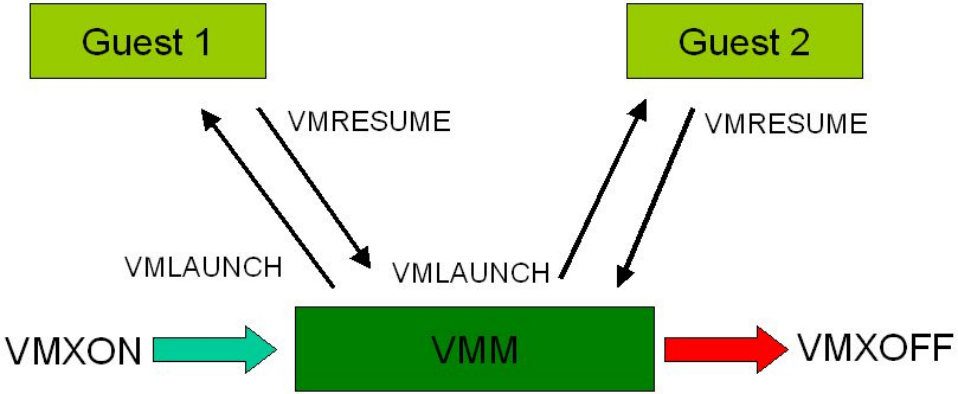
⚙️ En pratique : en R&T, on utilise souvent **VirtualBox** pour les TP

3. Virtualisation CPU et mémoire

3.1 CPU : modes et isolation

Les processeurs modernes (Intel VT-x, AMD-V) offrent :

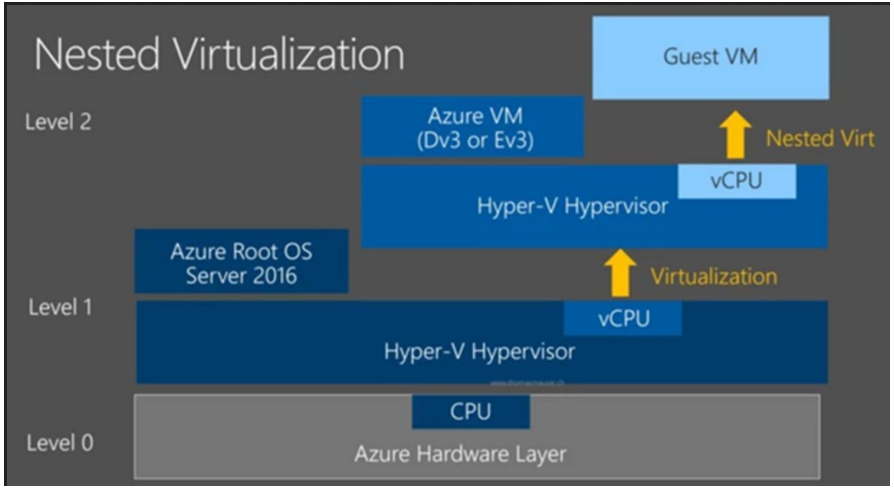
- des instructions spécifiques pour **passer du mode invité au mode hôte** sans overhead excessif,
- la gestion de **contextes multiples** (VMCS : Virtual Machine Control Structure).



3. Virtualisation CPU et mémoire

Fonctionnalités moderne :

- **Nested Virtualization** : exécution de VM dans une VM (clouds).
- **AMD SEV / Intel TDX** : chiffrement mémoire par VM.



Source: https://azure.microsoft.com/en-us/blog/nested-virtualization-in-azure/

3. Virtualisation CPU et mémoire

3.2 Mémoire : traduction double

Deux niveaux :

1. Programme invité → pages virtuelles de l'OS invité
2. OS invité → pages physiques gérées par l'hyperviseur

Technos associées :

- **EPT (Intel)** / **RVI (AMD)** : second-level address translation
- **MMU et TLB** : accélération de la traduction d'adresses.

🧠 **À retenir :** Ces technologies réduisent le coût de la virtualisation mémoire et augmentent la densité de VM par hôte.

Overcommit Memory

L’**overcommit** consiste à allouer aux machines virtuelles **plus de mémoire que la RAM réellement disponible sur l’hôte**.

Pourquoi ?

- Toutes les VMs n’utilisent pas leur mémoire à 100% simultanément.
- Optimise la densité de VMs par serveur.

Risques :

- Saturation de la RAM → swapping → fortes dégradations de performance.
- Doit être combiné avec des mécanismes de régulation (ballooning / TPS).

Ballooning

Le **balloon driver** (dans la VM) permet à l’hyperviseur de **reprendre de la mémoire à une VM** qui en utilise peu.

Principe :

1. L’hyperviseur “gonfle” un pilote dans la VM → le pilote consomme volontairement de la RAM.
2. La VM libère alors de la mémoire “inutile”.
3. L’hyperviseur récupère cette mémoire et peut l’attribuer à une autre VM.

Avantages :

- Ajustement dynamique et non intrusif.
- Permet l’overcommit sans trop de risque.

Limites :

- Dépend du comportement de l’OS invité.
- Peut dégrader les performances s'il fait pression sur la VM.

TPS – Transparent Page Sharing

Le **Transparent Page Sharing** consiste à **dédupliquer automatiquement** les pages mémoire *identiques* entre VMs.

Exemple :
Plusieurs VMs tournent avec la même installation Linux → de nombreux blocs mémoire sont identiques → une seule copie est réellement stockée.

Avantages :

- Réduit la consommation RAM globale.
- Très efficace dans les environnements homogènes (VDI, clusters).

Limites / évolutions :

- Désactivé par défaut sur certains hyperviseurs (VMware ESXi) pour des raisons de **sécurité** (atténuation des attaques par canal auxiliaire).
- Peut être réactivé dans des environnements maîtrisés.

4. Virtualisation des E/S et du réseau

4.1 E/S physiques et virtuelles

Les périphériques réels (carte réseau, disque, USB) doivent être **multiplexés** entre VMs.

Solutions :

- **virtio** : pilotes virtuels optimisés pour les VMs KVM.
- **SR-IOV (Single Root I/O Virtualization)** : plusieurs fonctions logiques sur une seule carte réseau.
- **IOMMU** : mappe les adresses physiques des périphériques vers celles des VMs.

4. Virtualisation des E/S et du réseau

4.2 Réseau virtuel

L’hyperviseur crée des **switchs virtuels** permettant la communication :

- entre VMs du même hôte,
- entre hôtes physiques (via ponts ou VLANs).

Outils :

- **vSwitch**, **Open vSwitch (OVS)** : standard du SDN (Software Defined Networking).
- **Bridging, NAT, VLANs internes** : VirtualBox, KVM, VMware.

Exemple R&T : création d’un routeur Linux virtuel interconnectant plusieurs réseaux VLAN.

5. Gestion et déploiement

- **Snapshots** : retour rapide à un état précédent.
- **Templates** : déploiement rapide de VMs préconfigurées.
- **Migration à chaud (live migration)** : déplacement d'une VM entre deux hôtes sans interruption.

💡 Utilisé dans les datacenters et les clusters de virtualisation.

6. Synthèse - Virtualisation système

Aspect	Avantage	Technologie clé
CPU	Isolation / scheduling	Intel VT-x, AMD-V
RAM	Efficacité / sécurité	EPT, RVI, SEV
I/Os	Partage matériel performant	SR-IOV, virtio, IOMMU
Réseau	Interconnexion virtuelle	vSwitch, OVS
Gestion	Automatisation	Proxmox, VMware vCenter