Chapitre 3

Les hyperviseurs

By D.E. MENACER

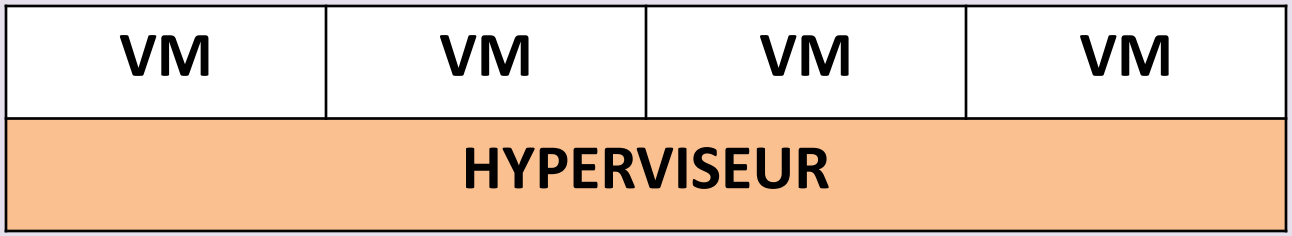
Sommaire

1. Introduction

- Un **hyperviseur** est une plate-forme de virtualisation qui permet à plusieurs systèmes d'exploitation de travailler sur une même machine physique en même temps
- Le terme **hyperviseur** prend sa source dans la ré-implémentation par **IBM** de **CP-67** pour le système d'exploitation **System/370** sorti en 1972 sous le nom **VM/370**.
- Désigné aussi par:
  - **VMM**: Virtual machine Monitor
  - **VMM**: Virtual machine Manager

2. Définitions

- Formellement, un **hyperviseur** est une **couche logicielle légère** (par rapport à un système d'exploitation) qui permet d'allouer un *maximum* de ressources physiques aux machines virtuelles.



Rappel: types de virtualisation

- **La Virtualisation complète**
- Cette stratégie de virtualisation consiste à créer des environnements virtuels (VM) qui sont une copie d'une machine physique (mémoire, disques,...).
- Cette méthode se base sur deux principes :
  - la **traduction binaire des instructions** que le noyau du système virtualisé souhaite exécuter (par ex. traduire une instruction privilégiée en une instruction non privilégiée);
  - **l'exécution directe** des instructions relatives aux **applications utilisateurs** (non privilégiées).

Rappel: types de virtualisation

- **Avantages :**
  - permet de faire fonctionner plusieurs systèmes différents sur la même machine physique ;
  - bonnes performances et stabilité.
- **Inconvénients :**
  - **consommation importantes des ressources (la consommation est fonction** du nombre de machines virtuelles).
- **Exemples**: Microsoft Virtual PC, Microsoft Virtual Server, VirtualBox, VMware Workstation

Rappel: types de virtualisation

- **La para-virtualisation**
  - Il s'agit d'une méthode qui implique une **modification du noyau** du système d'exploitation virtualisé afin de remplacer les instructions non virtualisables par des hyper-appels (hypercalls) qui vont communiquer directement avec la couche virtuelle de l'hyperviseur.

Rappel: types de virtualisation

- **Avantages :**
  - performances accrues et stabilité
- **Inconvénients :**
  - Nécessite une adaptation du noyau des systèmes invités ;
  - Usage limité aux systèmes libres (conséquence du point précédent).
- **Exemples :** Xen, Microsoft Hyper-V

Rappel: types de virtualisation

- **Virtualisation basée sur le matériel**
- Il s'agit d'une méthode qui consiste à installer *directement* sur le matériel une **couche logicielle minimale** (bare-metal) permettant de créer et configurer des VM.
- **Avantage**: performances optimales
- **Inconvénient**: nécessite une VM ou une machine physique pour la gestion complète de l'environnement virtuel.
- **Exemples**: vSphere Hypervisor (ESXi), PowerVM.

Rappel: types de virtualisation

- **Virtualisation au niveau du système d'exploitation**
- Il s'agit d'une stratégie qui consiste à réaliser le processus de virtualisation au niveau du noyau du système hôte: donc ne fait pas intervenir de couche virtuelle supplémentaire.
- Cette forme de virtualisation consiste à créer des environnements virtuels, les **conteneurs**, qui vont chacun dupliquer certains composants du système d'exploitation hôte.
- Les capacités de virtualisation font partie intégrante du système hôte et non plus de l'hyperviseur.

Rappel: types de virtualisation

- **Comparaison VM-Conteneurs**

	VM	**Conteneurs**
Traductions virtuelles	Chaque VM communique avec le matériel au travers d'une couche de virtualisation: plus le nombre de VM est important et plus le système est ralenti (plus de traductions virtuelles)	Autorise les conteneurs à communiquer directement avec le système d'exploitation hôte, comme s'il n'existait aucune couche de virtualisation, ce qui améliore les performances.
Nombre d'environnements virtuels	Nombre de VM Limité	les conteneurs sont généralement bien plus petits que les VM: il est possible d'exécuter un grand nombre de conteneurs en parallèle.

Rappel: types de virtualisation

- **Avantages :**
  - permet de faire fonctionner plusieurs instances du même système de façon étanche.
  - performances proches du mode natif.
- **Inconvénients :**
  - ne permet de faire fonctionner qu'un seul type de système, celui du système hôte.
- **Exemples** : Linux VServer, LXC, OpenVZ, Virtuozzo, Dockers…

3. Les types d'hyperviseur

- **Type 1 : Natif**
- Un hyperviseur de Type 1 ou « *bare metal* », est un logiciel qui s'exécute directement sur une plateforme matérielle.
- L'hyperviseur type 1 est un noyau hôte allégé et optimisé.
- Sur des processeurs ayant les instructions de virtualisation matérielle (AMD-V et Intel VT), l'hyperviseur n'a plus à émuler les **anneaux de protection** et le fonctionnement s'en trouve accéléré.
- Un hyperviseur de type 1 classique est **CP**, développé par IBM dans les années 60 et ancêtre de z/VM.
- **Exemples**: **Xen**, Oracle VM, **ESXI** Server de Vmware, **Hyper-V** de Microsoft…
- Les machines virtuelles utilisant un noyau Linux **KVM**, qui transforment un noyau Linux complet en hyperviseur, sont également considérées comme hyperviseurs de type 1.

Les types

- **Type 2 : Hosted**
- Un hyperviseur de Type 2 est un logiciel qui s'exécute à l'intérieur d'un autre système d'exploitation.
- Un système d'exploitation invité s'exécutera donc en troisième niveau au-dessus du matériel.
- Les systèmes d'exploitation invités n'ayant pas conscience d'être virtualisés, ils n'ont pas besoin d'être adaptés.
- **Exemples**: VMware Workstation, VMware Fusion, l'hyperviseur open source QEMU, les produits Microsoft Virtual PC, Virtual Server, VirtualBox d'Oracle, Parallels Workstation de Swsoft et Parallels Desktop.

4. Exemples d'hyperviseurs

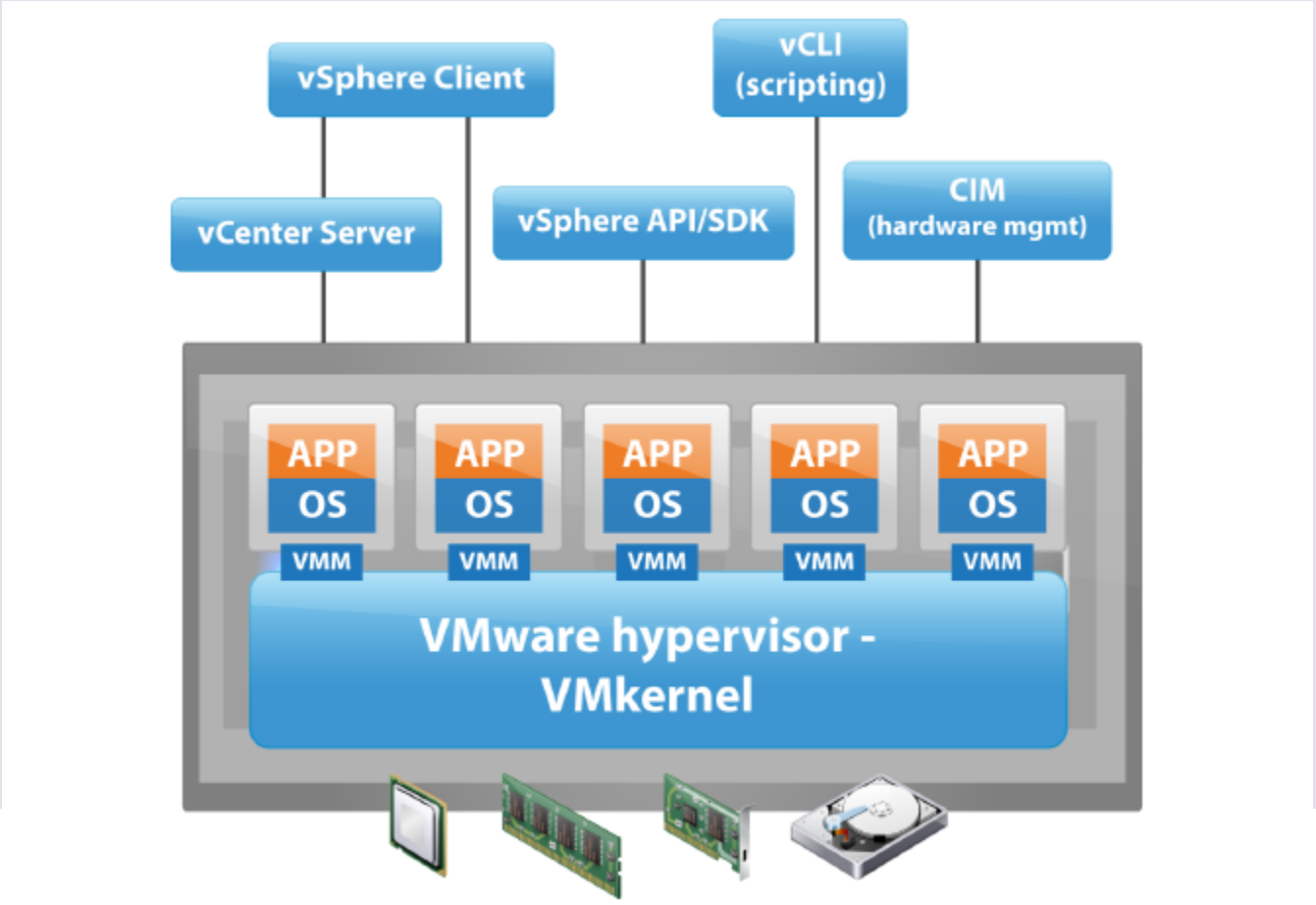
- **Vsphere Hypervisor** (VMWARE)
- **HYPER-V** (MICROSOFT)
- **XEN** (CITRIX)
- **KVM** (RED HAT)

Vsphere Hypervisor

- **Type 1 vrai: bare-metal**
- Utilise la virtualisation **basée sur le matériel: VT-x ou AMD-V**
- **Apparition: 2001 (GSX)**
- Successeur de **ESXi**
- **Leader mondial: 60%** du marché des hyperviseurs

Vsphere Hypervisor

- **Architecture**

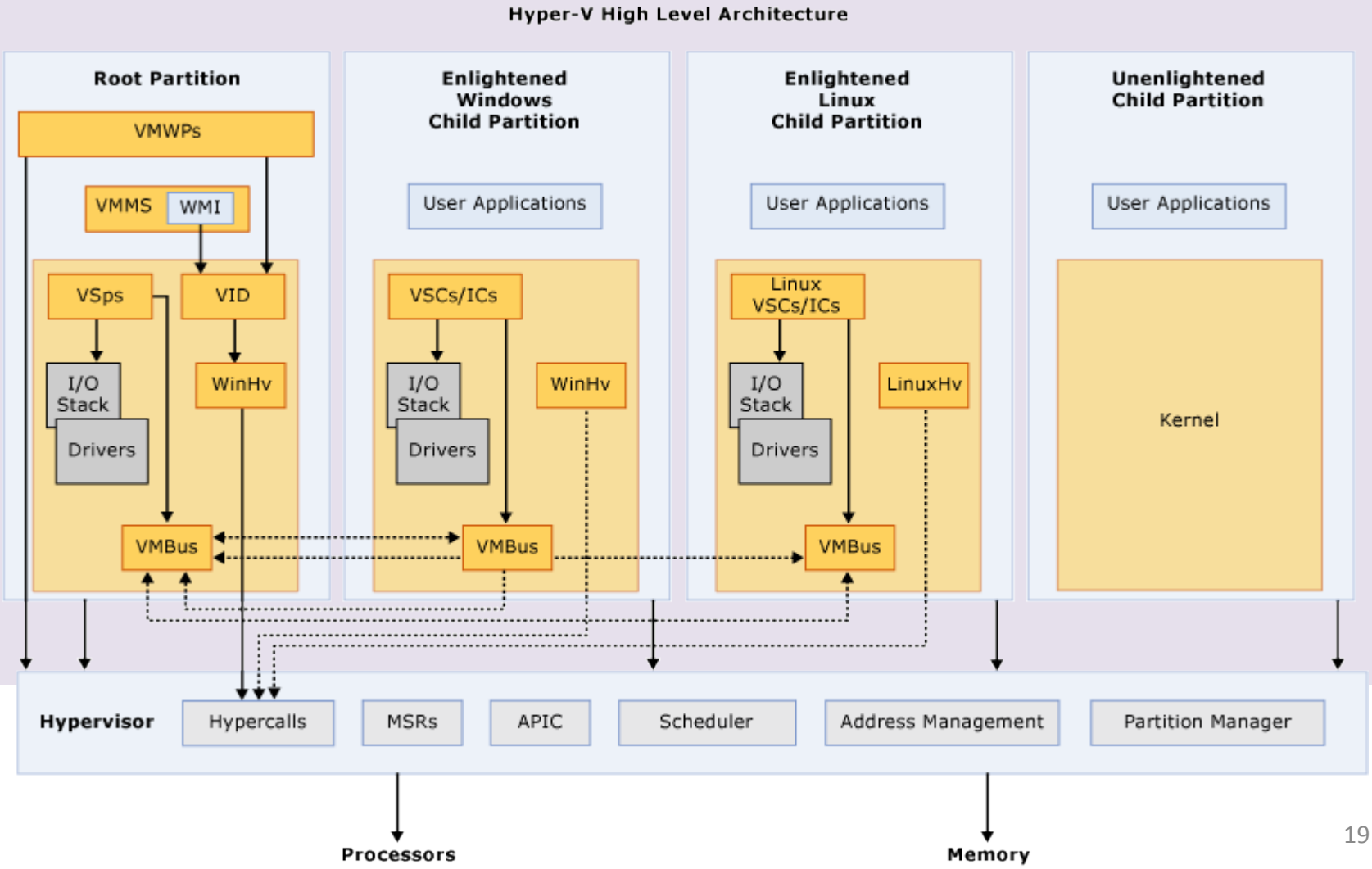


Hyper-V

- **Architecture**: **Type 2** (ajout du rôle HYPER-V dans Windows Server 2016)
- Considéré comme **Type 1** par ses fonctionnalités
- Apparition: **2008**
- Peut être installé en mode **bare-metal** par **Microsoft Hyper-V Server 2016**
- Utilise la virtualisation **basée sur le matériel: VT-x ou AMD-V**

Hyper-V

- **Architecture**

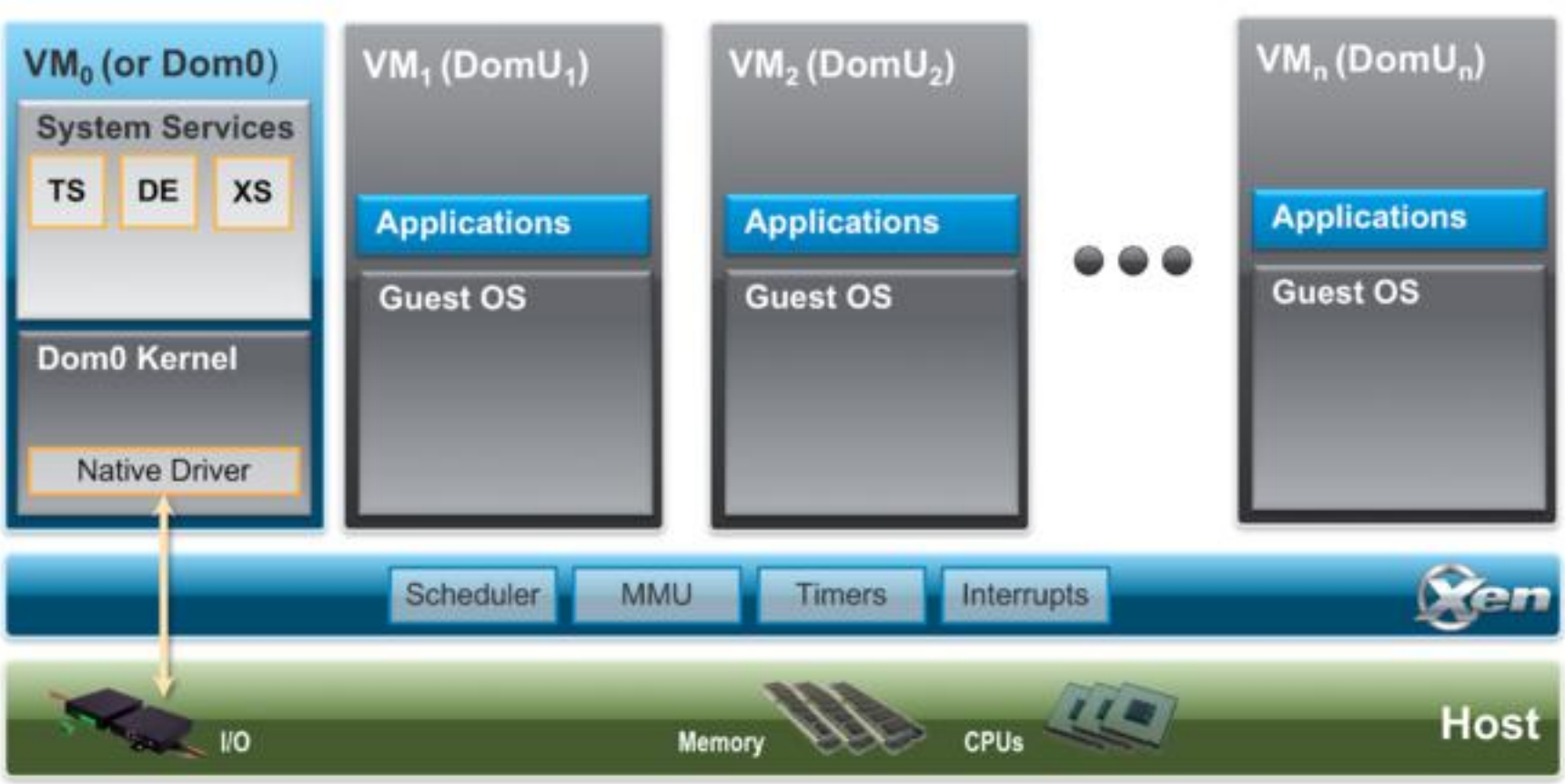


XEN

- **Architecture**: **Type 1 bare metal**
- Utilise la **para-virtualisation**
- **Apparition: 1997**

XEN

- **Architecture**

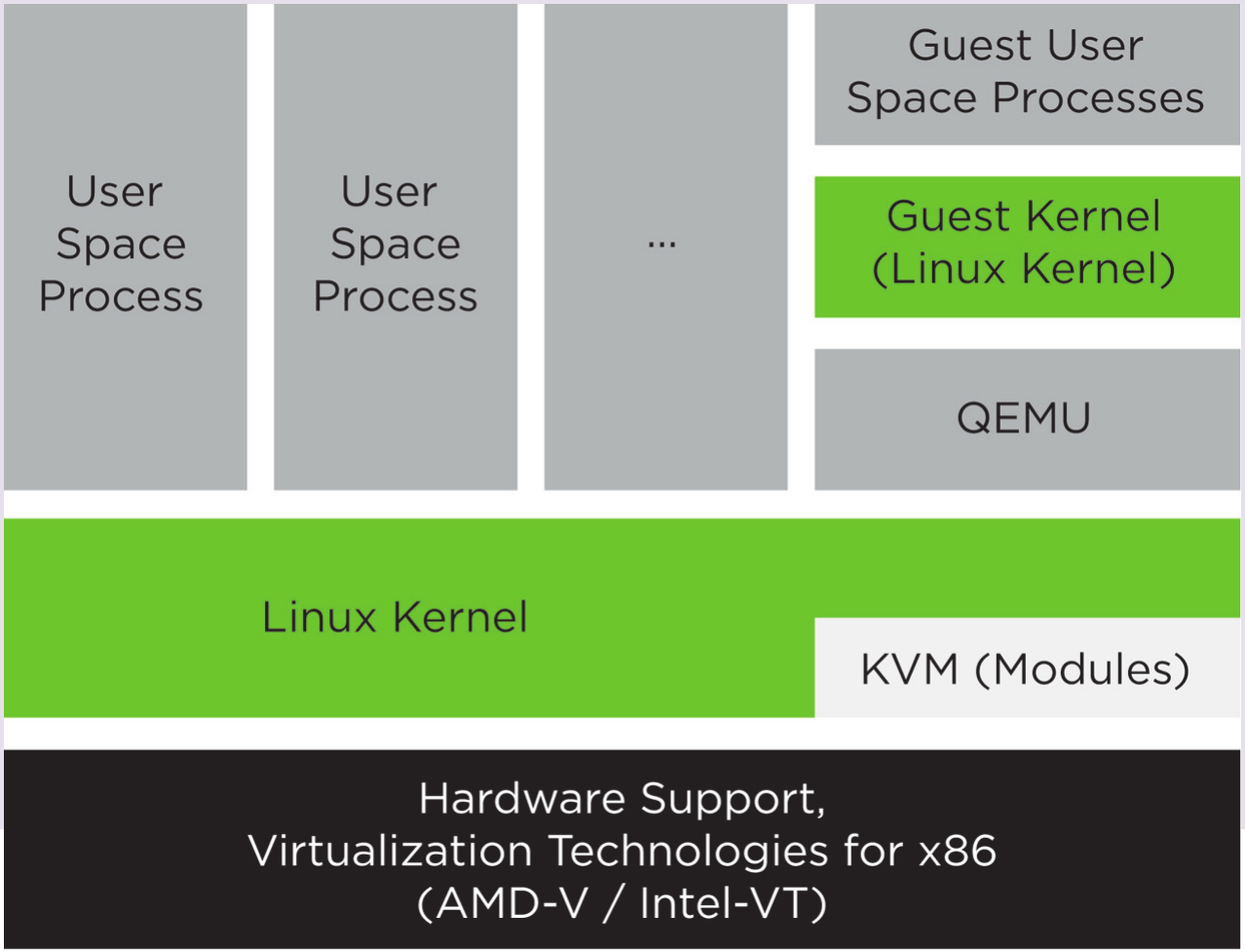


KVM

- **Architecture**: **Type 2**
- Considéré comme **Type 1** par ses performances
- Apparition: **2006** (Qumranet)**,** racheté par RH en 2008
- **Module** dans le noyau Linux
- Nécessite un émulateur I/0: **QEMU**
- Porté sur**: Intel x86, IA-64, PowerPC, ARM**

KVM

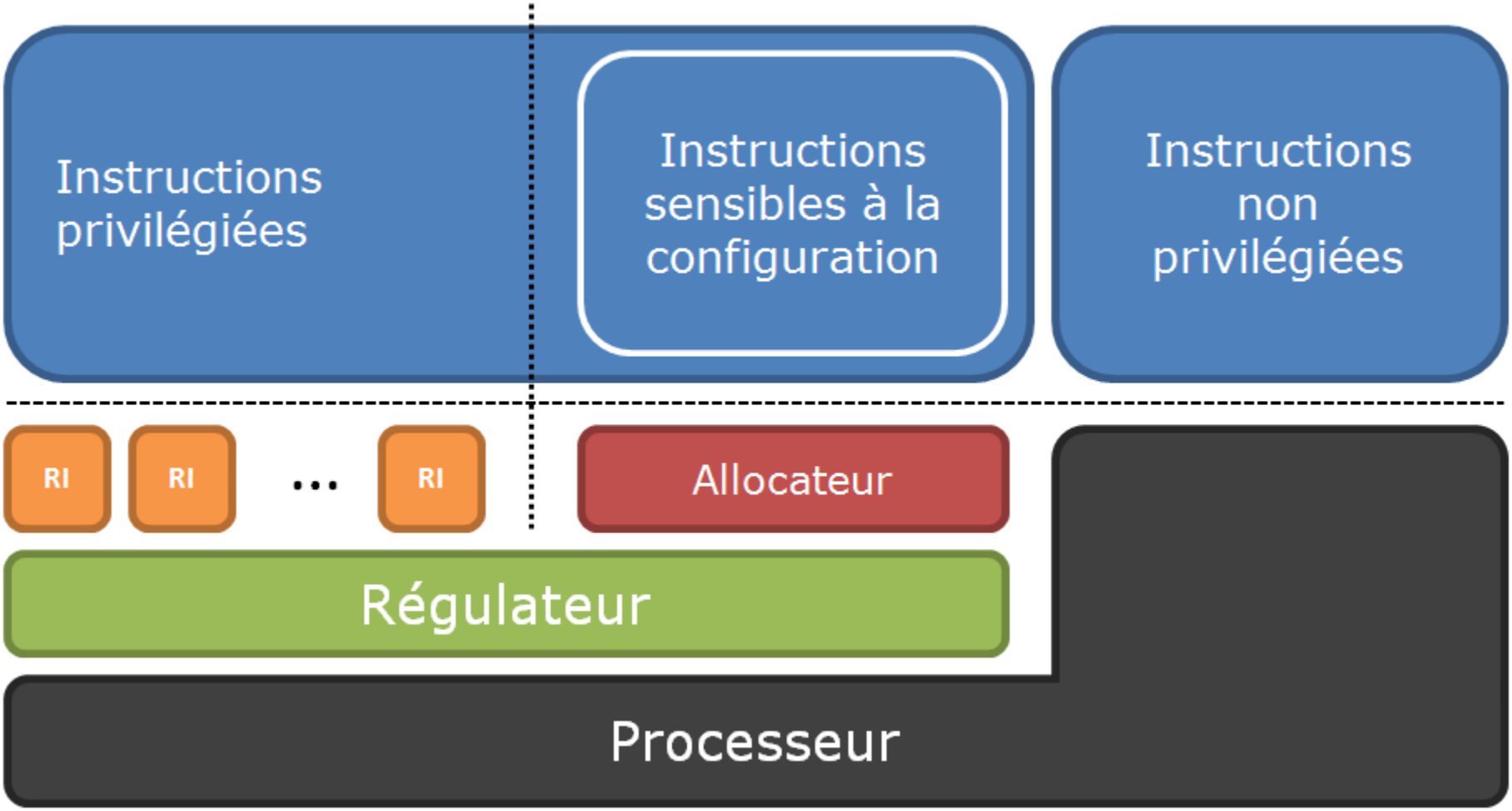
- **Architecture**



5. Composants d'un hyperviseur

- L'hyperviseur, est composé d'un ensemble de modules regroupés en **trois catégories** :
  - Le **régulateur** (dispatcher) : il peut être considéré comme le module de contrôle de plus haut niveau de l'hyperviseur. Son rôle est de donner le contrôle à un des modules de la deuxième ou de la troisième catégorie.
  - **L'allocateur** : son rôle est de déterminer quelle(s) ressource(s) doivent être allouées aux applications virtualisées. L'allocateur ne donne pas une même ressource simultanément à deux environnements virtuels distincts. Le régulateur fera appel à l'allocateur chaque fois qu'un environnement virtuel tentera d'exécuter une instruction privilégiée qui aurait comme répercussion de modifier les ressources allouées à cet environnement virtuel.
  - Des **interpréteurs** : à chacune des instructions privilégiées (à l'exception de celles qui sont prises en charge par l'allocateur), on va associer une routine d'interprétation (RI). Le rôle de ces routines est de simuler le résultat des instructions privilégiées qui sont piégées (trapped).

Composants d'un hyperviseur



6. Sémantique des hyperviseurs

- **Terminologie**
  - **Une machine virtuelle** (Virtual Machine, VM) : il s'agit d'une reproduction isolée et efficace d'une machine physique.
  - **Un hyperviseur** (VMM): logiciel qui va créer et gérer l'exécution d'un certain nombre de VM
- Popek et Goldberg [1] ont établi **trois contraintes** que doivent respecter les VMM pour pouvoir être considérés en tant que tels :

Sémantique des hyperviseurs

- **Critère d'équivalence**:
  - Tout programme doit montrer un comportement similaire, qu'il soit exécuté au travers d'un système de virtualisation ou bien directement sur la machine physique
  - *Any program run under the VMM should exhibit an effect identical with that demonstrated if the program had been run on the original machine directly, with the possible exception of differences caused by the availability of system resources and differences caused by timing dependencies.*

Sémantique des hyperviseurs

- **Critère d'efficacité**
  - Cette contrainte exige qu'une majorité des instructions du processeur virtuel soient directement exécutées par le processeur physique et ce, sans intervention du VMM.
  - *The second characteristic of a virtual machine monitor is efficiency. It demands that a statistically dominant subset of the virtual processor's instructions be executed directly by the real processor, with no software intervention by the VMM.*
  - La virtualisation matérielle permet de mieux répondre à cette contrainte. Ce critère exclut les systèmes purement logiciels (simulations) ainsi que l'émulation.

Sémantique des hyperviseurs

- **Critère de Contrôle des ressources**.
  - Ce critère impose au VMM d'avoir un contrôle total des ressources: Un programme virtualisé ne doit pas avoir la possibilité d'accéder à une ressource autrement qu'au travers du VMM et ce dernier se réserve le droit de reprendre le contrôle d'une ressource à n'importe quel moment.
  - Le terme **ressource** se réfère à la mémoire et aux autres périphériques, pas uniquement au processeur.
  - *The VMM is said to have complete control of the resources if it is not possible for a program running under it in the created environment to access any resource not explicitly allocated and it is possible under certain circumstances for the VMM to regain control of resources already allocated.*

Prérequis

- Les pré requis énoncés par Popek et Goldberg sont un ensemble de conditions nécessaires **assurant à une architecture matérielle de supporter efficacement le processus de virtualisation.**
- Le problème est de déterminer les caractéristiques que doit posséder le jeu d'instructions d'un processeur afin de répondre de manière efficace aux 3 contraintes évoquées.
- Le jeu d'instructions (Instruction Set Architecture, ISA) est l'ensemble des instructions qu'un processeur est en mesure d'exécuter.
- **Pour énoncer leurs théorèmes, Popek et Goldberg ont classé les instructions de l'ISA en 3 groupes**

Les 3 groupes d'instructions

- **Les instructions privilégiées** :
  - pour les exécuter, le processeur doit être dans un mode particulier, le mode privilégié.
  - Si le processeur ne se trouve pas dans cet état, une interruption est déclenchée et l'instruction est piégée (trapped).
  - L'interruption consiste en un changement de contexte au cours duquel le système (que ce soit l'OS hôte ou l'hyperviseur) décide de la suite à donner à cette instruction.
- **Les instructions sensibles à la configuration** :
  - ce sont les instructions qui ont la possibilité d'interroger ou de modifier la configuration du système comme le mode du processeur ou la quantité de mémoire allouée à un processus particulier.
- **Les instructions sensibles au comportement** :
  - il s'agit des instructions dont le comportement ou le résultat va dépendre de l'état ou la disponibilité des ressources (si le processeur se trouve en mode privilégié ou non,...).

Théorèmes

- **Théorème 1**
  - *For any conventional third generation computer, a VMM may be constructed if the set of sensitive instructions for that computer is a subset of the set of privileged instructions.*
  - Pour construire un hyperviseur pour une architecture particulière, il est nécessaire que les instructions qui pourraient compromettre le bon fonctionnement de cet hyperviseur (instructions sensibles à la configuration) soient piégées et que le contrôle soit donné à l'hyperviseur.
  - **Ceci garantit le critère de contrôle des ressources**.
  - Les **instructions non privilégiées** doivent être exécutées sans intervention de l'hyperviseur afin de garantir le critère d'efficacité.

Théorèmes

- **Cas particulier**:
  - si des instructions sensibles à la configurations ne font pas partie des instructions privilégiées, alors, en vertu du critère d'efficacité, elles seront exécutées sans intervention de l'hyperviseur et par conséquent, le critère de contrôle des ressources n'est pas respecté.

Théorèmes

- **Théorème 2**
  - *A conventional third generation computer is recursively virtualizable if 1) it is virtualizable and 2) a VMM [...] can be constructed for it.*
  - Ce théorème porte sur la possibilité de faire de la **virtualisation récursive**, c'est à dire d'exécuter une ou plusieurs machines virtuelles au sein d'une machine virtuelle.
  - Ce théorème affirme que ce processus est réalisable pour autant que l'architecture matérielle réponde au premier théorème et qu'un hyperviseur puisse être développé pour cette architecture.

Exercice et Démo

- Expliquez pourquoi Microsoft Hyper-V ne peut pas fonctionner en mode récursif (nested) au dessus de Vmware Workstation.
- Quelle modification doit-on effectuer dans la configuration d'une VM Hyper-V pour qu'elle fonctionne en tant qu'hyperviseur?

TP2: KVM

- Voir TP2

Références

- G. Popek & R. Goldberg; *Formal requirements for virtualizable third generation architectures*; Communications of the ACM, Volume 17 Issue 7, July 1974; EditorRobert L. Ashenhurst Univ. of Chicago, Chicago, IL; Pages 412-421