

Formation OpenStack



Arnaud Morin

Orange Labs




Concernant ces supports de cours


Auteurs initiaux :

- Adrien Cunin <adrien.cunin@osones.com> 
- Pierre Freund <pierre.freund@osones.com> 

Modifiés et adaptés par :


- Arnaud Morin <arnaud1.morin@orange.com> 

Licence

- Copyright ©2014 Osones
- Creative Commons BY-SA 4.0 

<https://creativecommons.org/licenses/by-sa/4.0/deed.fr>

Sources

- <https://github.com/Osones/OpenStack-Formations/>
- <https://github.com/arnaudmorinol/OpenStack-Formations/> 

Objectifs de la formation

- Virtualisation
- Cloud
- Focus sur OpenStack

Objectifs de la formation : Virtualisation

- Comprendre les principes de la virtualisation et son intérêt
- Connaitre le vocabulaire inhérent à la virtualisation
- Avoir une vue d'ensemble sur les solutions existantes de virtualisation

Objectifs de la formation : Cloud

- Comprendre les principes du cloud et son intérêt
- Connaitre le vocabulaire inhérent au cloud
- Avoir une vue d'ensemble sur les solutions existantes en cloud public et privé
- Posséder les clés pour tirer partie au mieux de l'IaaS
- Pouvoir déterminer ce qui est compatible avec la philosophie cloud ou pas
- Adapter ses méthodes d'administration système à un environnement cloud



Objectifs de la formation : OpenStack

- Connaitre le fonctionnement du projet OpenStack et ses possibilités
- Comprendre le fonctionnement de chacun des composants d'OpenStack
- Pouvoir faire les bons choix de configuration
- Savoir déployer manuellement un cloud OpenStack pour fournir de l'IaaS
- Connaitre les bonnes pratiques de déploiement d'OpenStack
- Être capable de déterminer l'origine d'une erreur dans OpenStack
- Savoir réagir face à un bug

Plan

1 Virtualisation

Plan

1 Virtualisation

- Histoire de la virtualisation
- Principes généraux
- Comprendre la virtualisation

Définition

Selon Wikipedia :

« *La virtualisation consiste à faire fonctionner un ou plusieurs systèmes d'exploitation comme un simple logiciel, sur un ou plusieurs ordinateurs (serveurs), au lieu de ne pouvoir en installer qu'un seul par machine.* »

Plan

1 Virtualisation

- Histoire de la virtualisation
- Principes généraux
- Comprendre la virtualisation

Historique - 1

- 1946 Premiers ordinateurs « Turing-complet » (ex : ENIAC)
- 1958 Ordinateurs multitaches (Gamma 60 de Bull) : faire tourner plusieurs programmes en même temps, concept proche de la virtualisation.
- 196x Time-sharing systems, premières notions de Virtual Machine avec le projet IBM M44/44X
- 1972 IBM Mainframe Virtual Machine Facility/370 : **premier système de « full virtualisation » !**
- 1988 IBM sort son Hyperviseur PR/SM de type 1

Historique - 2

- 1990 Emulation de processeurs x86, mac sur Amiga (pionnier du genre)
- 1990 Architecture NUMA (Cloisonnement et partitionnement de la mémoire via des bus)
- 1999 VMWare Workstation : hyperviseur de type 2 intelligent : émulation seulement quand nécessaire
- 2000+ Développements de projet libres (QEMU, KVM, Bochs, Xen, etc.)
- 2004 Intel VT-x : Les VM ont directement accès au CPU. Les hyperviseurs ne font plus d'émulation mais contrôlent qui a accès au CPU

Plan

1 Virtualisation

- Histoire de la virtualisation
- Principes généraux
- Comprendre la virtualisation

Principes généraux de Popek et Goldberg

Popek et Goldberg sont deux chercheurs qui ont introduits des conditions pour qu'un système supporte la virtualisation :

- équivalence** fonctionnement identique dans une VM comme sur une machine physique
- efficacité** une part majoritaire d'instructions doit être exécutée directement sans intervention de l'hyperviseur
- contrôle** l'hyperviseur garde le contrôle des ressources et les partage entre les VM

Ces principes datent de 1974 !

Seulement à partir de 2004 (avec Intel VT) qu'ils sont respectés dans l'architecture x86.

Intérêt de la virtualisation

Trois avantages principaux

- Sécurité
- Coût
- Criticité et performances

Intérêt de la virtualisation - Sécurité

- Sécurité
 - ▶ Isolation et cloisonnement
 - ★ Ignorance de la présence d'autres environnements
 - ★ Utilisation des protocoles conventionnels
 - ▶ Etudes de sécurité
 - ★ Contrôle et étude d'environnements infectés
 - ★ Répétition de scénarios

Intérêt de la virtualisation - Coût

- Coût

- ▶ Mutualisation de ressources physiques
 - ★ Sous utilisation actuelle des serveurs
 - ★ Coût de l'énergie électrique
 - ★ Coût de l'espace en centre de données
 - ★ Coût opérationnel
- ▶ Meilleure gestion des ressources physiques
 - ★ Allocation exclusive
 - ★ Allocation temporelle

Intérêt de la virtualisation - Criticité et performances

- Criticité et performances
 - ▶ Possibilité de mettre en pause et de copier un environnement logiciel complet
 - ★ Sauvegarde
 - ★ Clonage
 - ▶ Migration d'environnements logiciels
 - ★ Transfert d'un environnement logiciel vers une autre machine physique
 - ▶ Allocation dynamique de ressources
 - ★ Flexibilité de l'offre
 - ★ Adaptabilité en cas de montée en charge

Plan

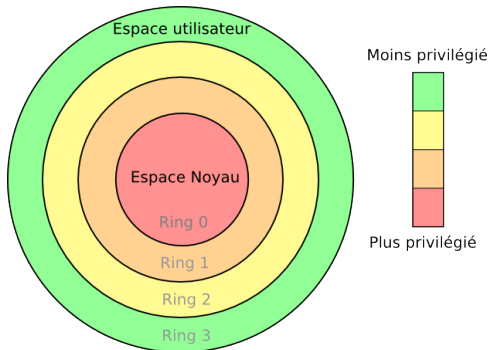


Virtualisation

- Histoire de la virtualisation
- Principes généraux
- Comprendre la virtualisation

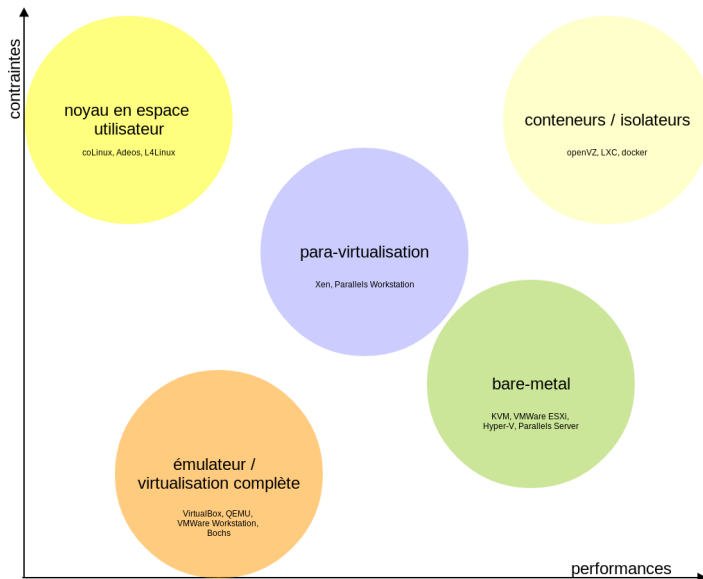
Les anneaux de protection

- 4 niveaux de privilèges dans l'architecture x86
- Ring 0 : Espace Noyau
- Ring 1 et Ring 2 : généralement pas utilisés
- Ring 3 : Espace Utilisateur



Crédit image : « Hertzprung » sur wikipedia, modifié pour le cours

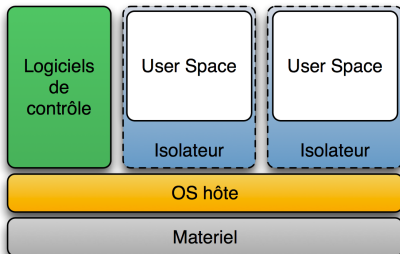
Les modèles de virtualisation en un schéma



Les conteneurs / isolateurs

Principe : isoler l'exécution d'applications dans un contexte (aussi appelé zone d'exécution)

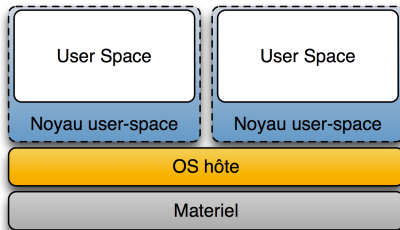
- Pros :
 - ▶ Très performant car peu d'overhead
 - ▶ Permet de faire tourner la même application en mode multi-instance (ex : serveur web)
- Cons :
 - ▶ Même noyau pour toutes les applications
 - ▶ Seulement linux
 - ▶ Pas vraiment de la virtualisation



Noyau en espace utilisateur

Principe : faire tourner un noyau Linux dans l'espace utilisateur

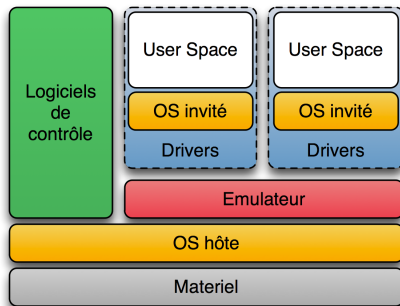
- Pros :
 - ▶ Utile pour tester un noyau linux ou faire du développement de noyau
- Cons :
 - ▶ Pas très performant (empilement de deux noyaux)
 - ▶ Pas très sécurisé (pas d'isolation entre les noyaux)
 - ▶ Nécessite une modification du noyau invité (possible avec linux seulement)



Émulation / Full virtualisation (Hyperviseur de type 2)

Principe : la machine hôte émule le matériel pour la machine invité

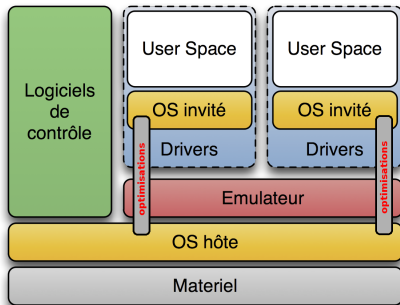
- Pros :
 - ▶ Bonne isolation entre les OS invités
 - ▶ Cohabitation d'architecture CPU et OS hétérogènes
- Cons :
 - ▶ Pas très performant (émulation provoque beaucoup d'overhead)



Para-virtualisation

Principe : évolution de l'émulation par modification des OS invités

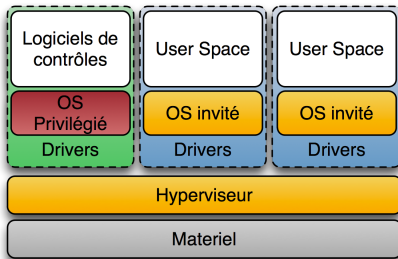
- Pros :
 - ▶ Bonne isolation entre les OS invités
 - ▶ Cohabitation d'architecture CPU et OS hétérogènes
 - ▶ Performance correctes
- Cons :
 - ▶ Nécessite la modification du noyau de l'OS invité (possible sur Linux seulement)



Bare metal (Hyperviseur de type 1)

Principe : noyau système léger qui partage l'accès aux ressources matérielles avec les OS invités

- Pros :
 - ▶ Bonne isolation entre les OS invités
 - ▶ Cohabitation d'architecture CPU et OS hétérogènes
 - ▶ Très performant (couche d'abstraction minimale)
- Cons :
 - ▶ Nécessite la virtualisation matérielle (Intel VT-x ou AMD-V)



À partir de 2004, Intel et AMD ont ajouté à leurs processeurs des instructions CPU supplémentaires pour aider à la virtualisation :

- Intel VT-x, VT-c, VT-d
- AMD-V

« To assist virtualization, VT and Pacifica insert a new privilege level beneath Ring 0. Both add nine new machine code instructions that only work at "Ring -1," intended to be used by the hypervisor. »

Exemples de création de machines virtuelles