Introduction à la virtualisation

Jean-Patrick Gelas
UE Cloud Computing – M2 TI + DS
Université Claude Bernard – Lyon 1

Introduction

Virtualiser : proposer, par l'intermédiaire d'une couche d'abstraction proche du matériel, une vue multiple d'un matériel unique, en sérialisant les appels vus concurrents de l'extérieur.

2

La virtualisation recouvre l'ensemble des techniques matérielles et/ou logicielles qui permettent de faire fonctionner sur une seule machine plusieurs systèmes d'exploitation, plusieurs instances différentes et cloisonnées d'un même système ou plusieurs applications, séparément les uns des autres, comme s'ils fonctionnaient sur des machines physiques distinctes.
Chaque outil de virtualisation implémente une ou plusieurs de ces notions :
o couche d'abstraction matérielle et/ou logicielle,
o système d'exploitation hôte (installé directement sur le matériel),
o systèmes d'exploitations (ou applications, ou encore ensemble d'applications) « virtualisé(s) » ou « invité(s) »,
o partitionnement, isolation et/ou partage des ressources physiques et/ou logicielles,
o images manipulables : démarrage, arrêt, gel, clonage, sauvegarde et restauration, sauvegarde de contexte, migration d'une machine physique à une autre
O réseau virtuel : réseau purement logiciel, interne à la machine hôte, entre hôte et invités.

Analogie avec les processeurs

- Cadence max atteinte (~3.6GHz)
- Naissance du multi-cœurs suivit de l'
- Hyper-threading (1 cœur = 2 cœurs logiques)
  (Ex: Core i7 => 1 processeur, 4 cœurs physiques hyper-threadés => soit 2x4=8 cœurs logiques).

- La virtualisation consiste à « augmenter » le nombre de cœurs.

3

Prenons l'exemple d'un processeur, une fois que la cadence de celui-ci a atteint sa limite physique de fabrication et d'utilisation, comprenez là environs 3,6 Ghz, les fondeurs (Intel & AMD) ont élaboré un nouveau système : le multi-cœurs. À l'origine un processeur ne contient qu'un seul cœur cadencé à une certaine vitesse, mais du fait de l'amélioration des finesses de gravure il est possible de disposer de plusieurs cœurs physiques sur un même processeur. **Ici il est question non pas d'augmenter le nombre de processeurs dans une machine mais d'augmenter le nombre de cœurs dans un processeur**, cette notion de mise en abîme va encore plus loin avec le concept de l'hyper-threading qui pour un cœur d'un processeur crée 2 cœurs logiques. Exemple : un processeur Intel Core i7 a pour caractéristique 1 processeur, 4 cœurs physiques hyper-threadés, soit 2×4=8 coeurs logiques.

http://www.sebastien-han.fr/blog/2011/04/12/introduction-a-la-virtualisation/

Dans la logique, la virtualisation fonctionne sur le même principe, au lieu de multiplier les machines physiques avec un seul système d'exploitation, on utilise une machine physique pour virtualiser plusieurs systèmes d'exploitations.

Terminologie

- Le **système hôte (*host*)** est l'OS principal de l'ordinateur.
- Le **système invité (*guest*)** est l'OS installé à l'intérieur d'une machine virtuelle.
- Une **machine virtuelle (*VM*)** est un ordinateur virtuel qui utilise un système invité.
- Un ordinateur virtuel est aussi appelé **serveur privé virtuel** (*Virtual Private Server* ou VPS) ou environnement virtuel (*Virtual Environment* ou VE)

4

Intérêts

- Usage optimale des ressources
- Installation, déploiement et migration facile
- Économie sur le matériel
- Sécurisation
- Isolation
- Allocation dynamique
- Diminution des risques

5

➔ utilisation optimale des ressources d'un parc de machines (répartition des machines virtuelles sur les machines physiques en fonction des charges respectives),
➔ installation, déploiement et migration facile des machines virtuelles d'une machine physique à une autre, notamment dans le contexte d'une mise en production à partir d'un environnement de qualification ou de pré-production, livraison facilitée,
➔ économie sur le matériel par mutualisation (consommation électrique, entretien physique, monitoring, support, compatibilité matérielle, etc.),
➔ installation, tests, développements, réutilisation avec possibilité de recommencer arrêt du système hôte,
➔ sécurisation et/ou isolation d'un réseau (arrêt des systèmes d'exploitation virtuels, mais pas des systèmes d'exploitation hôtes qui sont invisibles pour l'attaquant, tests d'architectures applicatives et réseau),
➔ isolation des différents utilisateurs simultanés d'une même machine (utilisation de type site central),
➔ allocation dynamique de la puissance de calcul en fonction des besoins de chaque application à un instant donné,
➔ diminution des risques liés au dimensionnement des serveurs lors de la définition de l'architecture d'une application, l'ajout de puissance (nouveau serveur etc.) étant alors transparent.

Historique

- Idée développée au centre IBM de Cambridge et de Grenoble en 1972 (VM/CMS) (pseudo-machine.)
- Mi-90's émulateurs d'Atari, Amiga,NES,SNES,…
- Début des années 2000 : VMware
- Logiciels libre : Xen, Qemu, Bochs,...
- Propriétaire (mais gratuits) : VirtualPC,...

6

Une bonne part des travaux sur la virtualisation fut développée au centre de recherche IBM France de Grenoble (aujourd'hui disparu), qui développa le système expérimental CP/CMS, devenant ensuite le produit (alors nommé hyperviseur) VM/CMS, proposé au catalogue dès 1972.
Par la suite, les mainframes ont été capables de virtualiser leurs OS avec des technologies spécifiques et propriétaires, à la fois logicielles et matérielles.
Les grands Unix ont suivi avec les architectures NUMA des Superdome d'HP (PA-RISC et IA64) et des E10000/E15000 de Sun (UltraSparc).
Dans la seconde moitié des années 1990, les émulateurs sur x86 des vieilles machines des années 1980 ont connu un énorme succès, notamment les ordinateurs Atari, Amiga, Amstrad et les consoles NES, SNES, Neo Geo.
La société VMware développa et popularisa au début des années 2000 un système propriétaire de virtualisation logicielle pour les architectures de type x86. Les logiciels libres Xen, Qemu, Bochs, Linux-VServer et les logiciels propriétaires mais gratuits VirtualPC et VirtualServer ont achevé la popularisation de la virtualisation dans le monde x86.

https://aconit.inria.fr/omeka/exhibits/show/informatique-grenoble/consolidation/ibm

Différents domaines

- Virtualisation d'applications (du contexte d'exécution).
- Virtualisation de serveur
- Virtualisation du réseau (VLAN)*
- Virtualisation du stockage*

(*: non traité dans ce cours)

7

La virtualisation de stockage :
La virtualisation de stockage est un procédé qui va séparer la représentation logique et la réalité physique de l'espace de stockage. Son but est de faire abstraction des périphérique de stockage utilisés et des interface qui leur sont associés (SATA, SCSI, ...) afin de limiter l'impact des modifications structurelles de l'architecture de stockage.

Ce type de virtualisation fait appel à une application d'administration de volumes logiques (Logical Volume Manager, LVM). Il s'agit d'une couche logicielle qui va permettre de regrouper plusieurs espaces de stockage, appelés volumes physiques, pour ensuite découper cet espace global suivant la demande en partitions virtuelles appelées volumes logiques.
Ce processus de virtualisation peut être vu comme une extension du modèle de partitionnement
classique des disques dur.
La virtualisation de stockage permet :

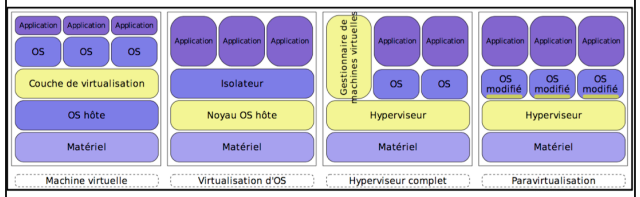
- d'adjoindre un périphérique de stockage supplémentaire sans interruption des services ;

- de regrouper des unités de disques durs de différentes vitesses, de différentes tailles et de différents constructeurs ;

- de réallouer dynamiquement de l'espace de stockage. Ainsi, un serveur nécessitant un espace de stockage supplémentaire pourra rechercher des ressources non allouées sur le disque logique. Inversement, un serveur nécessitant moins d'espace de stockage pourra libérer cet espace et le rendre disponible pour d'autres serveurs.

Différentes techniques de virtualisation

- Usage de couches logicielles intermédiaires
- 4 types de technologies



8

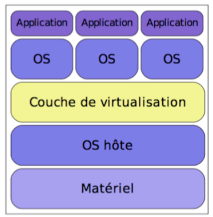
Très peu de systèmes démarrent directement, et sur plateforme PC tous les systèmes démarrent selon un processus (boot) en trois temps : le BIOS, le chargeur de démarrage (bootloader) et le système. Pour les systèmes embarqués et les anciens calculateurs, ces phases étaient aussi un choix à prendre en compte. Des moniteurs, OS minimaux emportant des fonctionnalités de débug, ont vu le jour. L'idée initiale était de se concentrer sur cette phase, puis ensuite d'en généraliser les concepts : un système d'exploitation reposant sur l'API d'un noyau réduit.

Les systèmes de virtualisation partent donc du principe de l'utilisation de couches logicielles intermédiaires. Afin d'avoir une idée théorique des performances des applications au sommet, il faut comparer verticalement l'empilage de couches. Il faut garder à l'esprit qu'il est possible d'élargir les schémas en rajoutant des environnements virtualisés consommant également des ressources de l'hôte, en mémoire puis en disque.
Fondamentalement, on trouvera deux idées principales: isolation par empilement ou par juxtaposition, dont les variations, dans des buts d'optimisation, donneront principalement quatre types de technologies :

Techno #1 : Machine Virtuelle

(Hyperviseur de type 2 ou Architecture hébergée)

- Application installée sur l'OS
- Performances réduites (Ex: E/S)
- Bonne étanchéité entre les OS invités.



Exemples : VirtualBox, QEMU, Vmware (workstation, fusion,player), Microsoft Virtual PC,…

9

Une machine virtuelle est un logiciel qui tourne sur l'OS hôte, ce logiciel permettant de lancer un ou plusieurs OS invités, c'est l'archétype de la solution de virtualisation par empilement de systèmes. La machine virtualise le matériel (ce qui passe généralement par un émulation partielle) pour les systèmes d'exploitation invités: les systèmes d'exploitation invités croient dialoguer directement avec le matériel. En pratique on a recours à une émulation logicielle des périphériques, et parfois aussi de tout ou partie de la machine.

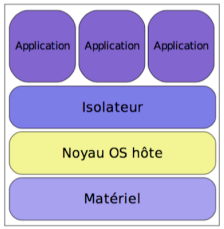
Cette solution isole bien les systèmes d'exploitation invités, mais elle a un coût, en premier lieu en **performance**, dont les principales victimes seront les entrées-sorties. Ce coût, assez important déjà s'il suffit de protéger les instructions privilégiées, peut être très élevé si le processeur doit être intégralement émulé. Cette solution n'est pas non plus économe en mémoire, puisque aucune économie d'échelle ne peut être réalisée en ce qui concerne les OS (kernels) chargés.
Exemple :
➔ Qemu:émulateurdeplateformesx86,PPC,Sparc,
➔ VMWare:propriétaire,émulateurdeplateformex86,
➔ MicrosoftVirtualPCetVirtualServer:propriétaire,émulateurdeplateformex86,
➔ VirtualBox:émulateurdeplateformex86.

Le talon d'Achille de cette solution est donc l'importante consommation en performances. Certains projets minimisent l'impact de ce coût en ressources en permettant des courts-circuits optionnels (KVM, KQemu, VirtualBox), disponibles uniquement hors du contexte d'émulation bien sûr.
Les deux technologies suivantes constituent elles aussi le résultat de considérations d'optimisation de cette solution, le principe étant de diminuer au mieux l'épaisseur des couches de virtualisation tout en conservant le même degré d'isolation.

L'hyperviseur de type 2 ou architecture hébergée est une application installée sur un système d'exploitation, elle est donc dépendante de celui-ci. Les performances sont réduites en comparaison des hyperviseurs de type 1 car l'accès au matériel (CPU, RAM…) se fait via une couche intermédiaire. Néanmoins il propose une parfaite étanchéité entre les systèmes d'exploitations installés.

Techno #2 : Virtualisation d'OS ou Isolateur

- Isole l'exécution des applications dans des contextes d'exécution.
- Généralisation de la notion de « contexte » Unix, plus isolation
  - des périphériques,
  - des systèmes de fichiers
- Solution très performante et économique en mémoire mais
- Partage du code noyau (donc mauvaise isolation).
- Ex: Linux Vserver, OpenVZ (Virtuozzo)



10

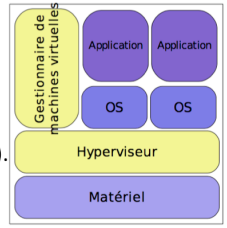
Un isolateur est un logiciel permettant d'isoler l'exécution des applications dans des contextes ou zones d'exécution, c'est l'archétype de la solution de virtualisation par "juxtaposition". L'isolateur permet ainsi de faire tourner plusieurs fois la même application (à base d'un ou plusieurs logiciels) prévue pour ne tourner qu'à une seule instance par machine.

Notons que cette technologie consiste en quelque sorte à généraliser la notion de "contexte" Unix : ce dernier isole les processus (mémoire, accès aux ressources), on ajoute alors : une isolation des périphériques (c'est le rôle de l'isolateur), voire leur partage, les systèmes de fichiers donc les fichiers eux-mêmes et leurs accès.
Cette solution est très performante, du fait du peu d'overhead (chute de performance conséquente de l'ajout des couches de virtualisation), mais les environnements virtualisés ne sont pas complètement isolés, ils partagent en particulier le code du noyau. Cette solution est aussi remarquablement économique en mémoire par conséquence de la dernière remarque. Ces environnements sont donc bien adaptés au déploiement de nombreux serveurs virtuels de test ou développement basés sur un même système.
Quelques isolateurs :
➔ Linux-VServer:isolationdesprocessusenuser-space,
➔ BSDJail:isolationenuser-space,
➔ OpenVZ:libre,partitionnementauniveaunoyausousLinuxetWindows2003.C'estla version open-source du logiciel Virtuozzo.
Les isolateurs tendent à isoler à un niveau de plus en plus proche du système, voire dans le système dans le cas de OpenVZ, qui peut être vu comme un Linux avec plusieurs tables de

Techno #3 : Hyperviseur complet

(dit de type-1 ou bare-metal)

- Noyau léger (micro-noyau) plus
- Outils de supervision
- Permet l'exécution d'OS natifs
- Usage d'instructions dédiées à la virtualisation (sinon émulation).



- Ex: XEN, KVM, Vmware vSphere,…

11

Partant du principe, exposé précédemment, qu'une approche pour une virtualisation efficace consiste à affiner les couches, une première approche consiste à proposer un noyau léger (de type micro-noyau par exemple), lequel est accompagné d'outils de supervision, et adapté pour
faire tourner des systèmes d'exploitation natifs. Pour réussir cette approche, soit on émule le matériel (et on revient aux performances de
la machine virtuelle pour les I/O), soit on dispose des instructions dédiées à la virtualisation. Dans ce dernier cas, les logiciels libres concernés se limiteront au monde x86 ou x86-64, munis des instructions ad-hoc.
Les principaux exemples de ce principe sont :

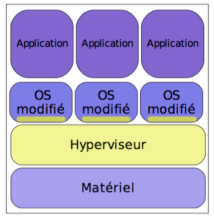
➔ KVM:IntégréaunoyauGNU/Linuxàpartirdelaversion2.6.20.
➔ Xen sur une machine offrantlesupportdesinstructions AMD-V(version3.03ouultérieure)
ou VTx (version 3.0 ou ultérieure).

L'hyperviseur de type 1 ou bare-metal est un outil qui s'interpose entre la couche matérielle et logicielle. Celui-ci a accès aux composants de la machine et possède son propre noyau. C'est donc par dessus ce noyau que les OS seront installés. Il pilote donc les OS à partir de la couche matérielle, il s'administre via une interface de gestion des machines virtuelles. Il est beaucoup plus puissant que les hyperviseurs de

Techno #4 : Paravirtualisation

(Hyperviseur de type 1 également)

- Noyau hôte allégé et optimisé
- Noyau invités adaptés et optimisés
- Utilisable sans les instructions spécifiques.
- Impraticables pour les systèmes non libres.



12

Un paravirtualiseur est un noyau hôte allégé et optimisé pour ne faire tourner que des noyaux de systèmes d'exploitation invités, **adaptés et optimisés**. Les applications en espace utilisateur des systèmes d'exploitation invités tournent ainsi sur une pile de deux noyaux optimisés, les systèmes d'exploitation invités ayant conscience d'être virtualisés. Cette approche offre l'avantage d'être utilisable en l'absence des instructions spécifiques, mais elle est impraticable pour des systèmes non libres pour lesquels l'éditeur ne fera pas l'effort d'adaptation.
Exemple :
Xen : noyau léger supportant des noyaux Linux, Plan9, NetBSD, etc,

Un hyperviseur de type 1 est comme un noyau système très léger et optimisé pour gérer les accès des noyaux d'OS invités à l'architecture matérielle sous-jacente. Si les OS invités fonctionnent en ayant conscience d'être virtualisés et sont optimisés pour ce fait, on parle alors de para-virtualisation (méthode indispensable sur Hyper-V de Microsoft et qui augmente les performances sur ESX de VMware par exemple).
Actuellement l'hyperviseur est la méthode de virtualisation d'infrastructure la plus performante mais elle a pour inconvénient d'être contraignante et onéreuse, bien que permettant plus de flexibilité dans le cas de la virtualisation d'un centre de traitement de données.

Une 5ème technologie…
Noyau dans l'espace utilisateur

- Un noyau exécuté comme une application dans le *user-space*.
- Très peu performant (empilement de deux noyaux !)
- Utile au développement noyau.

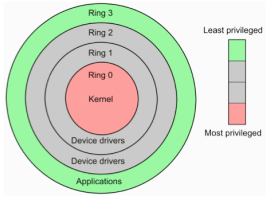
Ex : UML (User Mode Linux)
http://user-mode-linux.sourceforge.net/

13

Un noyau en espace utilisateur (user-space) tourne comme une application en espace utilisateur de l'OS hôte. Le noyau user-space a donc son propre espace utilisateur dans lequel il contrôle ses applications. Cette solution est très peu performante, car deux noyaux sont empilés et l'isolation des environnements n'est pas gérée et l'indépendance par rapport au système hôte est inexistante. **Elle sert surtout au développement du noyau.**

Solution x86

- 4 niveaux de privilèges mal exploité
  - Système : au niveau le plus privilégié (*ring 0*)
  - Applications : au niveau le plus faible (*ring 3*)
- Ajouts d'instruction dédiés par AMD et Intel pour une virtualisation matérielle (~2007)
  - Coexistence de plusieurs *ring 0* simultanée
  - On parle de *ring -1*



Cependant, les solutions envisageables sur architecture x86 sont longtemps restés bridés par les spécificités de l'architecture x86 et surtout par son utilisation. En effet, cette famille de processeurs (depuis le 80386) offre, pour l'écriture de systèmes, 4 niveaux de privilèges, mais à priori tous les systèmes d'exploitation ont pris le parti de placer le système au niveau le plus privilégié (ring 0), et les applications au niveau le plus faible (ring 3), sans penser à se rendre plus indépendant des niveaux d'exécution. Par conséquent, 2 niveaux de privilèges sont perdus, et écrire une couche de virtualisation pour embarquer des systèmes d'exploitation complets, nécessitant des privilèges supérieurs à ceux du ring 0, ne peut plus se faire que par émulation.

C'est pourquoi les fabricants de processeurs x86 AMD et Intel ont ajouté dans leurs gammes des instructions dédiées afin de proposer des solutions de virtualisation matérielle dans la seconde moitié des années 2000. Ces instructions permettent la coexistence de plusieurs "ring 0" simultanée, on parle parfois par extension à leur sujet de "ring -1".

A propos de AMD-V et Intel VT

- Jeu étendu d'instructions de virtualisation
- Virtualisation assistée par le processeur
  - Virtualisation des accès mémoire.
  - Protéger le processeurs physique des accès les plus bas niveau.
- Un « super Bios » fait l'interface avec la puce.
- Simplifie
  - la virtualisation logicielle
  - Réduit la dégradation de performances

15

Le support de la virtualisation peut être intégré au processeur ou assisté par celui-ci, le matériel se chargeant, par exemple, de virtualiser les accès mémoire ou de protéger le processeur physique des accès les plus bas niveaux. Cela permet de simplifier la virtualisation logicielle et de réduire la dégradation de performances.

Bit NX/XD

- NX (Non eXecutable) ou XD (eXecute Disable)
- Bit spécial qui permet de marquer des zones mémoires comme non exécutable.
- Améliore l'isolation des VM.

16

LAHF/SAHF

- *Load AH From flags/Save AH from Flags* autorise un contrôle direct du registre AH.
- Un hyperviseur utilise ces instructions pour assurer un contrôle plus direct du traitement des E/S et IRQ de chaque cœur de processeur.

17

EPT (Extended Page Table) RVI (AMD) / SLAT (Intel)

- Dans une VM la traduction d'adresses est faite deux fois.
- La traduction d'adresses est nécessaire car les processeurs doivent utiliser une table de pages ou un tampon de traduction (TLB) pour convertir les adresses relatives en adresse physique.

18

http://www.lemagit.fr/conseil/CPU-et-Virtualisation-les-caracteristiques-fondamentales-a-verifier

SLAT Second level Address Translation (INTEL) / RVI Rapid Virtualization Indexing (AMD)
EPT Extended Page Table

TLB Translation Lookaside buffer

Cgroups

- Fonctionnalité du noyau Linux pour limiter, compter et isoler l'utilisation des ressources (processeur, mémoire, utilisation disque, etc.).
  - Limitation des ressources
  - Priorisation
  - Comptabilité
  - Isolation
  - Contrôle
- Isolation par espace de nommage

19

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

L'un des buts de la conception de cgroups a été de fournir une interface unifiée à différents cas d'utilisation, en allant du contrôle de simples processus (comme nice) à la virtualisation au niveau du système d'exploitation (comme OpenVZ, Linux-VServer, LXC). Cgroups fournit :

Limitation des ressources : des groupes peuvent être mis en place afin de ne pas dépasser une limite de mémoire — cela inclut aussi le cache du système de fichier. L'article original a été présenté au Linux Symposium et s'intitule Containers: Challenges with the memory resource controller and its performance
Priorisation : certains groupes peuvent obtenir une plus grande part de ressources processeur ou de bande passante d'entrée-sortie.
Comptabilité : permet de mesurer la quantité de ressources consommées par certains systèmes en vue de leur facturation par exemple.
Isolation : séparation par espace de nommage pour les groupes, afin qu'ils ne puissent pas voir les processus des autres, leurs connexions réseaux ou leurs fichiers.
Contrôle : figer les groupes ou créer un point de sauvegarde et redémarrer.

Quelques solutions

(Open source)

- QEMU
- KVM
- XEN
- VirtualBox

- OpenVZ
- Docker

20

QEMU

- Machine virtuelle complète
- Techniquement très aboutie
- Émulation complète de machine (x86,ARM,MIPS,…)
- L'usage du module kQemu pour une virtualisation accélérée.
- Émulation par recompilation sur un modèle « just-in-time »
- Gourmand en mémoire
- Sans accéleration lent et charge l'hôte.
- (VirtualBox et KVM reposent sur Qemu)

21

Qemu ne nécessite pas de processeur spécifique, n'impose pas de modifier le système invité et marche sur un système hôte non modifié. Cependant, kQemu, approche d'accélération dédiée, permet d'accélérer notablement les performances (toujours sans contrainte sur la génération de processeur).

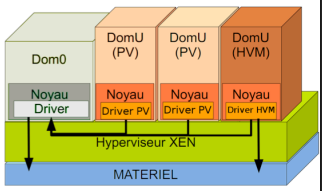
VirtualBox

- Machine virtuelle, émule un PC complet
- Support des instructions de virtualisation
- Solution de virtualisation efficace
- Repose sur Qemu (au boot au moins)
- Gourmand en mémoire
- Simple à utiliser

22

XEN

- Solution libre ancienne mais très utilisée.
- Vocabulaire :
  - OS privilégié : Dom0
  - OS invités : DomU
    - DomU standard (paravirt.)
    - DomU HVM (hardware assisted)
- Deux modes d'usage :
  - Paravirtualisation :
    - Noyau speçifique dans le DomU
    - Très bonnes performances
  - Virtualisation matérielle
    - Virtualisation transparente pour le système invité.
    - Besoin d'un support dans le processeur (AMD-V ou Intel VT)



23

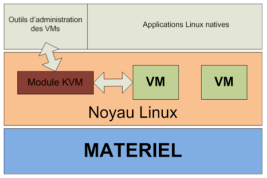
http://blog.octo.com/presentation-des-hyperviseurs-xen-et-kvm/

https://www.antoinebenkemoun.fr/2010/01/les-domaines-xen-les-domaines-non-privilegies/

Xen est capable de faire de la paravirtualisation mais aussi de la virtualisation matérielle assistée. A chacun de ces types de virtualisation correspond un type de domaine non privilégié. Les domU de paravirtualisation sont appelés « domU standards » et les domU de virtualisation matérielle assistée sont appelés « domU HVM ». L'acronyme HVM signifie « Hardware-Assisted Virtual Machine ».

KVM (*Kernel based Virtual Machine*)

- Projet plus récent que Xen mais très populaire.
- Basé en partie sur QEMU (pour le supports des périphériques)
- Entièrement intégré au noyau Linux -> Facile à utiliser.
- Support de la virtualisation dans les processeurs indispensable.
- Paravirtualisation (virtio) pour les performances.



24

http://blog.octo.com/presentation-des-hyperviseurs-xen-et-kvm/

OpenVZ

- PVS (Private Virtual Server)
- Virtualisation de niveau OS basée sur Linux
  - Un Linux avec plusieurs tables de processus
  - Chacune son contexte

25

Les isolateurs tendent à isoler à un niveau de plus en plus proche du système, voire dans le système dans le cas de OpenVZ, qui peut être vu comme un Linux avec plusieurs tables de processus, chacune dans le contexte d'une distribution propre.

Virtualisation : Les inconvénients

- Un point de défaillance unique
- Un recours à des machines puissantes
- Une dégradation des performances
- Une complexité accrue de l'analyse d'erreurs
- Parfois inadapté (Ex: bases de données (nombreux accès disques)).

26

Conclusions

- Ne pas virtualiser un serveur déjà beaucoup sollicité (la virtualisation rajouterai de *l'overhead* inutile)
- Réduit les coûts, facilite l'administration mais
- Il faut être capable de gérer un grand nombre de serveurs.
- Deux grandes familles de virtualisation :
  - Containers
  - Hyperviseurs

- Concept indispensable et étroitement lié à la réussite du Cloud.

27

Sources

- Livre blanc sur la Virtualisation, Groupe Linagora
- https://storify.com/selossej/la-virtualisation
- https://fr.wikipedia.org/wiki/Virtualisation

28