Virtualisation

Vincent Le Gallic

Séminaire Technique du Cr@ns

02 avril 2013

La virtualisation, c'est quoi ?

Définition de Wikipédia

La virtualisation consiste à faire fonctionner un ou plusieurs systèmes d'exploitation/applications, sur un serveur/système d'exploitation, au lieu d'en installer un seul par machine.

Le principe est de faire tourner plusieurs `DomU` peu gourmands en ressources sur un même `Dom0` (une brute).

La virtualisation, pourquoi ?

- Un serveur, ça coûte cher.
- La plupart du temps, ça se tourne les pouces.

→ On répartit les ressources (temps CPU, mémoire, . . .)

- On peut mettre en place un nouveau serveur facilement
- On peut allumer/éteindre un serveur plus rapidement
- On peut changer facilement les caractéristiques du serveur (coeurs, RAM alloués. . .)
- . . .

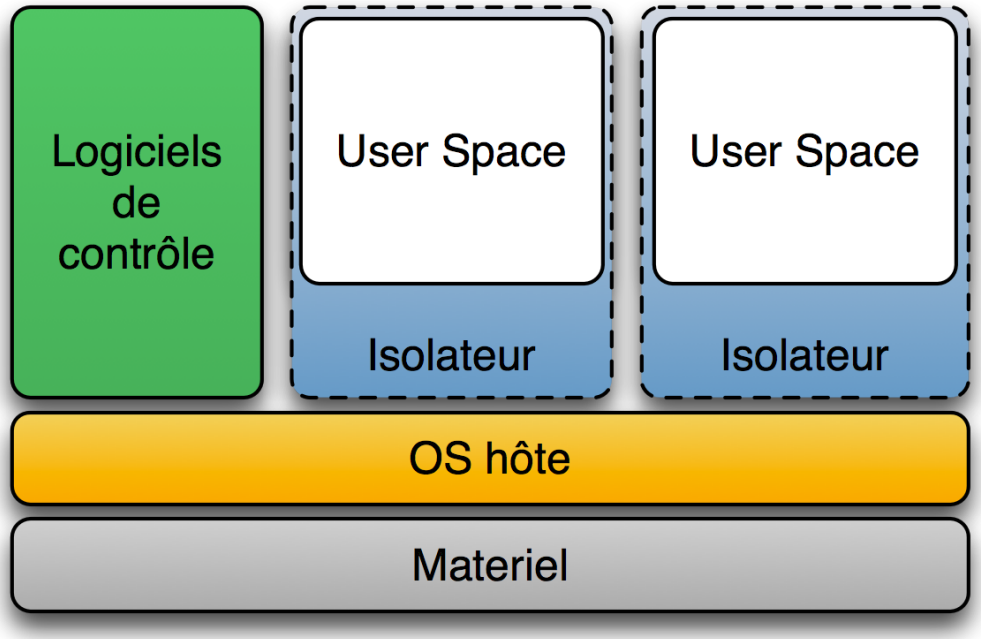
On fait ça comment ?

- Différentes méthodes sont possibles en fonction de l'utilisation qu'on veut en faire.
- En fonction de la méthode utilisée, deux systèmes virtualisés n'auront pas les mêmes choses en commun.

Isolateur

Définition

Un isolateur est un logiciel permettant d'isoler l'exécution des applications dans ce que l'on appelle des contextes ou bien zones d'exécution. L'isolateur permet ainsi de faire tourner plusieurs fois la même application dans un mode multi-instance (plusieurs instances d'exécution) même si elle n'était pas conçue pour ça.



Isolateur

Exemples :

- `chroot` : isolation de la racine. Le système de fichier «ignore» qu'il existe un monde extérieur.
- `Linux-VServer` : isolation des processus en espace utilisateur
- `OpenVZ` : partitionnement au niveau noyau sous Linux

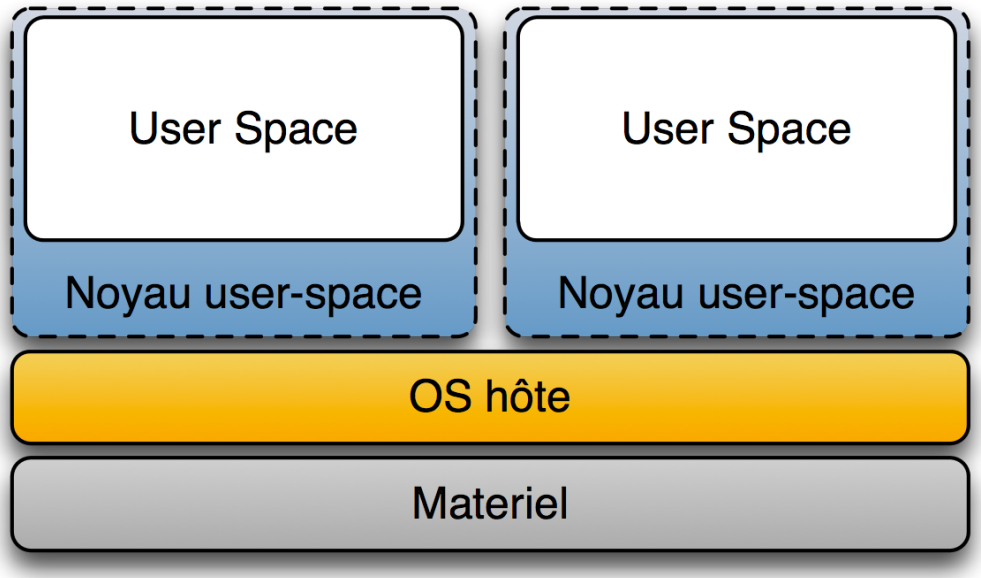
$\pm$:

- $\oplus$ Peu d'overhead (= temps où le système «se tourne les pouces»)
- $\ominus$ Seulement sous Linux.
- $\ominus$ Les environnements virtualisés ne sont pas complètement isolés.

Noyau en userspace

Définition

Un noyau en espace utilisateur (user-space) tourne comme n'importe quelle application en espace utilisateur de l'OS hôte.
Le noyau user-space a donc son propre espace utilisateur dans lequel il contrôle à son tour ses applications.



Noyau en userspace

Exemples :

- `User Mode Linux` : noyau tournant en espace utilisateur
- `coLinux` : noyau coopératif avec un hôte Windows

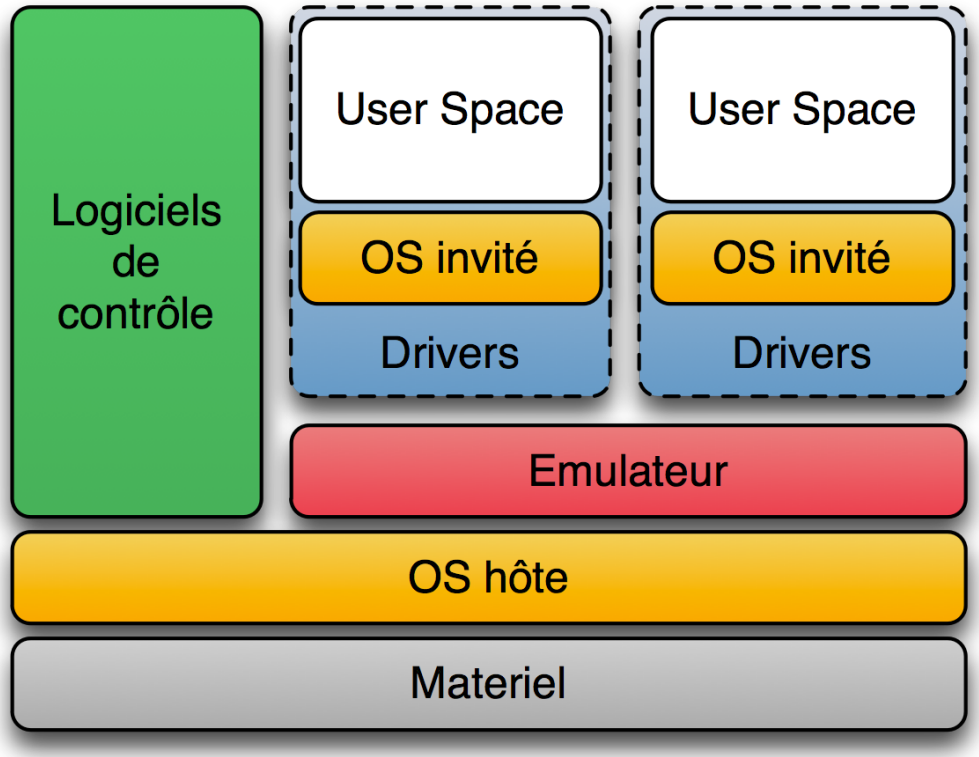
$\pm$:

- $\oplus$ Utile pour le développement du noyau
- $\ominus$ Pas d'indépendance par rapport au système hôte
- $\ominus$ Isolation des environnements non gérés
- $\ominus$ Deux noyaux empilés $\rightarrow$ très peu performant

Hyperviseur de type 2

Définition

Un hyperviseur de type 2 est un logiciel (lourd) qui tourne sur l'OS hôte. Ce logiciel permet de lancer un ou plusieurs OS invités. La machine émule le matériel pour les OS invités, ces derniers croient dialoguer directement avec ledit matériel.
C'est aussi ce qu'on appelle une **Machine Virtuelle**.



Hyperviseur de type 2

Exemples :

- `Microsoft VirtualPC / Virtual Server`
- `Parallels Desktop`
- `VirtualBox` (libre)
- `VMware`
- `QEMU, bochs` : émulateurs de plateforme x86

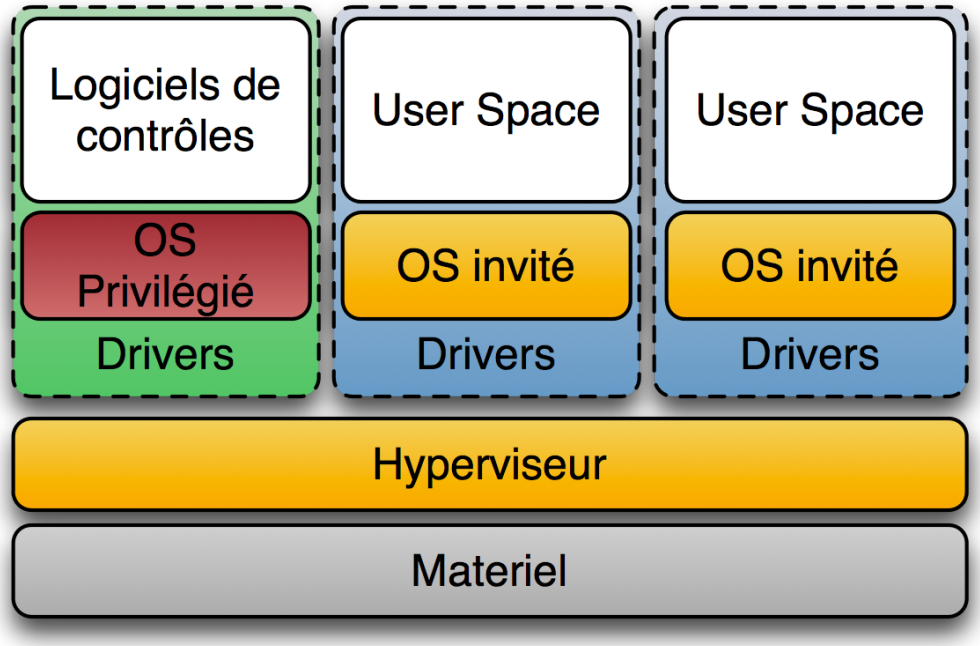
± :

- ⊕ Isolation complète des OS invités.
- ⊕ Cohabitation possible de plusieurs OS hétérogènes.
- ⊖ Gros coût en performance, notamment si il faut émuler le processeur.

Hyperviseur de type 1

Définition

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.



Hyperviseur de type 1

Exemples :

- `Microsoft Hyper-V Server`
- `VMware vSphere`
- `KVM` (libre)
- **`Citrix Xen Server`** (libre)

± :

- ⊖ Méthode contraignante (le noyaux hôte n'est plus classique)
- ⊕ Plus de flexibilité
- ⊕ Meilleure performances, surtout quand on augmente le nombre de système virtualisés
- ⊕ Actuellement la méthode de virtualisation d'infrastructure la plus performante

Les virtualiseurs au Cr@ns

`fy` et `fz`, au 0B (avant, `fx` aussi, mais il est devenu `zamok`)

- 8Go et 16Go de RAM
- 8×3 GHz et 16×2.4 GHz de processeurs
- accessibles par leur interface `ILO` en cas de besoin

xm : gestion des DomU

L'utilisation de la commande `xm` nécessite les droits root sur le `Dom0` (Le «Système privilégié»).

- `list` : voir la liste des `DomU` hébergés et leur état
- `info` : obtenir les caractéristiques du `Dom0`
- `create <DomU>` : booter le `DomU`
- `shutdown <DomU>` : équivalent à `shutdown` sur le `DomU`
- `destroy <DomU>` : éteindre sans condition le `DomU` (agressif)
- `migrate <DomU> <AutreDom0>` : envoyer «à chaud» le `DomU` sur un autre `Dom0`
- `dmesg <DomU>` : obtenir la sortie de `dmesg` du `DomU`
- `console <DomU>` : basculer sur un tty sur le `DomU`

Il a besoin de place

```
fz# cd /usr/scripts/gestion/iscsi
fz# ipython
ipython> import nolslib
ipython> conn = nolslib.Nols()
ipython> conn.create_volume("toto_slash", 2)       # La taille est en
ipython> conn.create_volume("toto_var", 2)         #  Go par défaut
ipython> conn.create_volume("toto_swap", 512, "MB")

fz# /usr/scripts/gestion/iscsi/update.sh

fz# mkfs.ext4 /dev/iscsi_toto_slash
fz# mkfs.ext4 /dev/iscsi_toto_var
fz# mkswap /dev/iscsi_toto_swap
```

Il a besoin d'un OS

Debootstrap :

```
fz# mkdir /mnt/toto
fz# mount /dev/iscsi_toto_slash /mnt/toto
fz# mkdir /mnt/toto/var
fz# mount /dev/iscsi_toto_var /mnt/toto/var
fz# debootstrap squeeze /mnt/toto ftp://mirror.adm.crans.org/debian
```

Dans `/etc/fstab.local` :

```
/dev/xvda       /       ext4    defaults        0       1
/dev/xvdb       /var    ext4    defaults        0       2
/dev/xvdc       none    swap    defaults        0       0
```

Dans `/etc/inittab` :

```
vc:2345:respawn:/sbin/getty 38400 hvc0
```

Mot de passe root du `DomU` :

```
fz$ sudo chroot /mnt/toto
root@fz# passwd
root@fz# exit
```

ensuite

RTFM :
`https://wiki.crans.org/CransTechnique/Virtualisation/CreerUnDomu`

Les virtualisés au Cr@ns

- `irc` : IRC
- `asterisk` : SIP
- `bcfg2` : Bcfg2 (cf séminaire idoine)
- `cas` : Central Authentication Service
- `kenobi` : serveur obby
- `nat64` : Nat des IPv6 derrière une IPv4
- `news` : serveur de newsgroups
- `niomniom` : wiki
- `o2` : intranet2
- `tracker` : Todolist
- `whatsupdoc` : ex-bugtracker

Les virtualisés au Cr@ns

- `dhcp` : DHCP
- `eap` : authentification wifi
- `vert` : LDAP
- `owl` : IMAP, Horde, Roundcube
- `radius` : authentification radius sur les switchs
- `redisdead` : SMTP, Mailman
- `routeur` : route les différents réseaux internes
- `sogo` : SOGo (webmail)
- `titanic` : routeur (entre autres) en connexion de secours
- `xmpp` : jabber
- `listes` : *ask Ping*
- `bob` : victime