

# Installation de XEN dans un environnement SAN

par Cédric TINTANET ([cedric-tintanet.developpez.com](http://cedric-tintanet.developpez.com))

Date de publication : 20 juillet 2008

Dernière mise à jour :

Cet article a pour but de décrire la mise en place de machines virtuelles sous XEN dans un environnement SAN.  
Il recense les différentes étapes nécessaires et les problèmes rencontrés lors du déploiement de nos machines virtuelles.

I - Introduction.....	3
I-A - Les principales technologies de stockage.....	3
I-A-1 - SCSI.....	3
I-A-2 - Le NAS (Network Attached System).....	3
I-A-3 - Le SAN (Stockage Area Network).....	3
I-B - Les hyperviseurs.....	4
I-B-1 - Différences entre les hyperviseurs de type 1 et de type 2.....	5
I-C - Quelques définitions.....	6
II - Description du système d'information initial.....	6
III - Attribution d'un disque provenant du réseau de stockage à un serveur sous LINUX.....	7
IV - Notion de multichemins ou multipath sous linux.....	9
IV-A - Le mapping de périphérique :.....	11
V - L'hyperviseur XEN.....	12
V-A - Virtualisation et paravirtualisation.....	12
V-B - Limites de la virtualisation totale.....	12
V-C - Le choix entre la virtualisation totale et la paravirtualisation.....	13
V-D - Notion de domaine 0, domaine utilisateur (domU) :.....	13
V-E - Notion de réseau Naté et réseau bridgé :.....	13
V-F - Installer un système paravirtualisé.....	14
V-G - Fichier de configuration d'un système paravirtualisé :.....	15
V-G-1 - Commentaires sur le fichier de configuration ci-dessus.....	15
V-H - Démarrer votre machine virtuelle :.....	16
V-I - Quelques commandes sympathiques.....	16
VI - La migration à chaud de machines virtuelles.....	18
VI-A - Le grand retour du multipath.....	20
VI - Conclusion.....	20
VIII - Remerciements.....	20

## I - Introduction

Avant de rentrer dans le vif du sujet je vais faire un rappel sur les principales technologies de stockage passées et présentes et tenter de faire un état des lieux des nouvelles fonctionnalités apportées par la virtualisation et les différents types d'hyperviseurs.

Ensuite je donnerais quelques définitions que le lecteur se devra de connaître avant d'aborder l'installation de XEN dans un réseau de stockage de type SAN.

Bonne lecture.

## I-A - Les principales technologies de stockage

### I-A-1 - SCSI

L'interface SCSI (*Small Computer System Interface*) est un bus permettant de gérer plusieurs périphériques. Un seul bus SCSI peut accepter de 8 à 15 unités physiques (disques durs, scanners graphiques, lecteur de bande, etc...), sachant que la carte SCSI que vous utiliserez sera comptée parmi ces unités physiques.

Il y a eu 3 normes SCSI et parmi elles on trouve plusieurs déclinaisons d'interfaces (Wide SCSI, Fast SCSI, Ultra SCSI, Ultra 160 SCSI, Ultra 320 SCSI, Ultra 640 SCSI, etc...)

Pour plus d'informations sur cette technologie je vous conseille de visiter [cette page](#)

### I-A-2 - Le NAS (Network Attached System)

Le système de stockage NAS est comme son nom l'indique un système de stockage en réseau.

Sauf que contrairement au SAN, il n'utilise pas de réseau dédié au stockage, mais un réseau ethernet classique.

On parle aussi de **serveur NAS**, qui est un serveur dédié au stockage de données.

Accessible par le réseau il permettra aux autres machines d'y stocker leurs données.

### I-A-3 - Le SAN (Stockage Area Network)

Le SAN est un réseau dédié au stockage. Le type de média utilisé est la fibre optique.

On peut trouver sur un SAN différents éléments tels que des robots de sauvegarde, des baies de disques, des commutateurs (équivalents des commutateurs ethernet mais dédiés à la technologie SAN) sur lesquels les différents éléments du réseau vont venir se brancher.

Chaque élément d'un SAN est identifié par un WWN (World Wide Name).

Il s'agit d'une sorte de plaque d'immatriculation pour chaque périphérique un peu à la manière des MAC adresses pour les cartes réseau, sauf que le WWN est composé de 8 octets (6 octets pour les MAC adresses).

### Les avantages du SAN sont les suivants:

- La fibre optique permet de très bons débits et la bande passante est uniquement dédiée aux opérations de lecture/écriture sur les différents éléments du réseau
- **La flexibilité:** On peut décider par une technique de zoning de faire voir ou de cacher différents éléments du réseau de stockage aux serveurs qui y sont reliés.
- **La flexibilité (le retour) :** Les possibilités d'évolution du réseau de stockage sont énormes. Tant que vous avez des ports sur votre commutateur SAN vous pouvez ajouter de la capacité de stockage.
- **Redondance:** Plusieurs cartes adaptatrices SAN permettront à un serveur d'avoir plusieurs chemins d'accès vers un ou plusieurs éléments du réseau de stockage. Un bon logiciel de multipathing et/ou de répartition de charge sauront gérer ça à merveille.
- **Partage de données:** Si plusieurs serveurs peuvent avoir accès au même contenu (technique de zoning) alors il nous sera facile de pouvoir monter des solutions de cluster haute disponibilité.

### Les désavantages d'un SAN

- Technologie chère à mettre en oeuvre.

- Requière une formation de base avant de se lancer dans le déploiement d'un SAN.

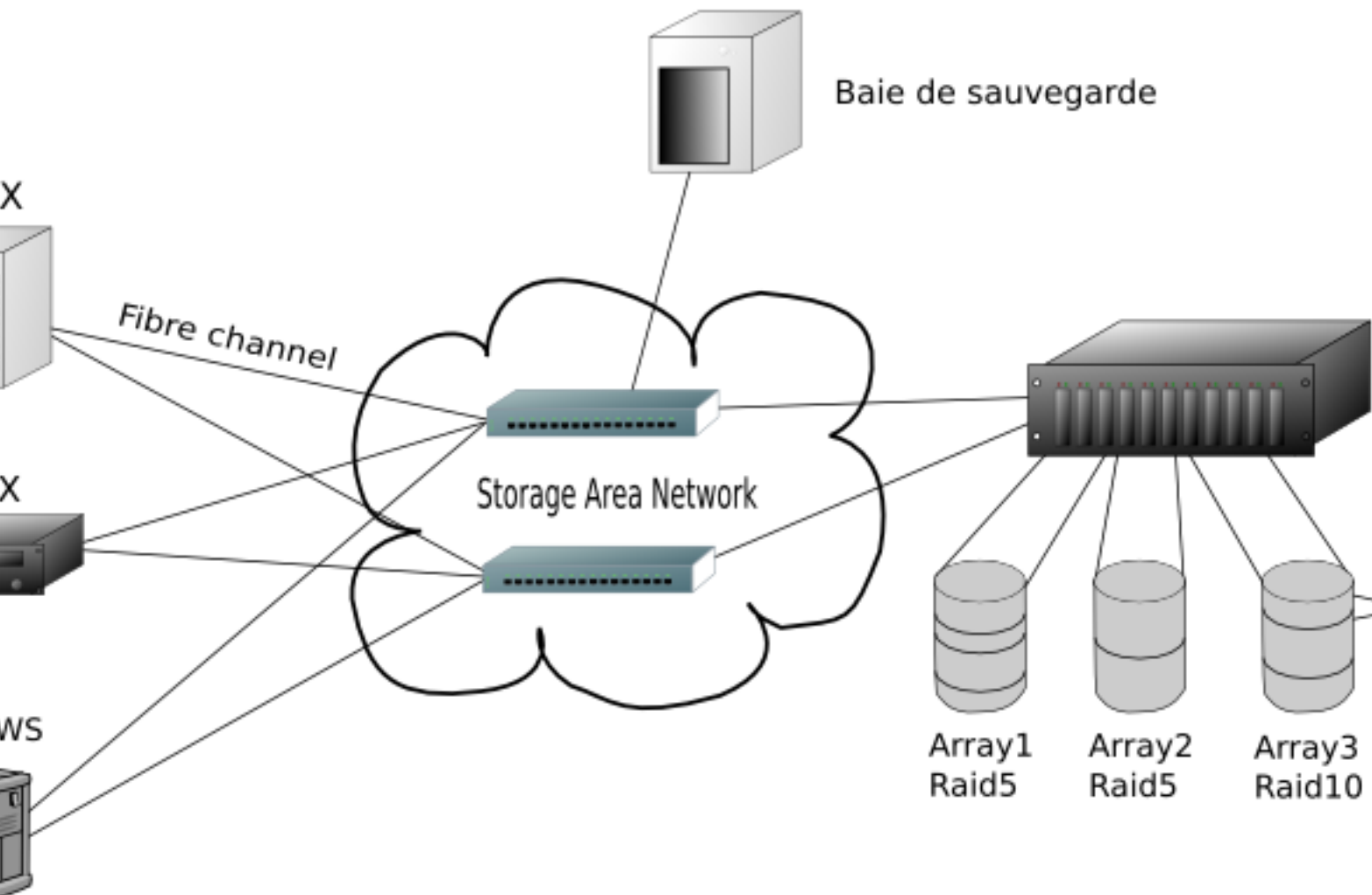


Schéma d'un réseau de stockage de type SAN ([http://fr.wikipedia.org/wiki/Storage\\_Area\\_Network](http://fr.wikipedia.org/wiki/Storage_Area_Network))

## I-B - Les hyperviseurs

Un hyperviseur est un système léger et adapté pour faire tourner des noyaux de systèmes invités. On peut dire aussi des hyperviseurs qu'ils sont des minis OS légers conçus pour faire tourner plusieurs autres systèmes d'exploitation **en même temps sur une même plateforme matérielle**.

Aujourd'hui la virtualisation est une tendance de plus en plus à la mode voir même en plein boom.

### L'intérêt de faire tourner plusieurs systèmes d'exploitation sur la même machine:

- **Réduction des coûts:** On évite de multiplier les machines physiques, on y gagne en place, en dépenses d'électricité et la puissance de calcul des dernières machines à la mode n'est plus sous exploitée.
- **Flexibilité:** Les derniers hyperviseurs ont des fonctionnalités intéressantes tel que l'ajout/retrait à chaud de ressources tel que la RAM, le stockage, le nombre de coeurs CPU, etc...  
Imaginez que vous avez trois machines virtuelles fonctionnant sur votre hyperviseur, et que vous savez qu'à une certaine période de la journée vous avez un pic d'activité sur la machine 1 alors que les deux autres machines ne sont pas beaucoup sollicitées.

Vous pouvez alors choisir grâce à votre hyperviseur d'enlever de la puissance de calcul à vos deux machines non sollicitées pour l'ajouter à la machine 1 et tout ça à chaud bien sûr.

Une fois le pic d'activité passé vous pouvez décider de répartir les ressources équitablement entre les trois machines.

- **Haute disponibilité/Redondance:** Bien sûr vous êtes en droit de vous demander, si ce n'est déjà fait, ce qui adviendrait de vos machines virtuelles si votre hyperviseur venait à tomber en panne ou encore plus simplement si vous deviez arrêter celui-ci pour maintenance. Et je vous répondrais immédiatement que vos machines virtuelles seraient autant indisponibles que le serait votre hyperviseur. Mais les formidables ingénieurs concepteurs d'hyperviseurs ont pensé à tout et ont ajouté à ceux ci une fonctionnalité qui permet à un hyperviseur A de confier la gestion de ses machines virtuelles à un hyperviseur B et tout cela une fois de plus sans interruption de services. Cependant un espace de stockage partagé est recommandé entre les deux hyperviseurs.

On s'abstrait des notions de serveur physique, aujourd'hui une machine virtuelle tourne sur un des hyperviseurs du système d'information. On a bien sûr moyen de savoir de quel hyperviseur il s'agit mais ce n'est pas essentiel tant que les services associés à cette machine virtuelle tournent.

Au cours de sa vie une machine virtuelle, pour peu que vous ayez programmé des maintenances automatiques sur vos hyperviseurs, sera amené à se déplacer d'un hyperviseur à un autre, mais une fois de plus tant qu'il n'y a pas d'interruption de services ce n'est pas un problème pour vous. A partir d'un seul poste vous pouvez avoir accès à de multiples informations sur vos serveurs virtuels (status du serveur, taux d'occupation de la ram, des cpus, etc...) et vous pouvez même agir sur ceux ci.

Pratiquement finis les déplacements en salle machine pour aller accéder à la console de la machine puisque vous y avez accès depuis votre poste.

Voilà ce qu'apporte la virtualisation et les hyperviseurs et j'en oublie certainement.

## I-B-1 - Différences entre les hyperviseurs de type 1 et de type 2

Les hyperviseurs de type 1 font ce qu'on appelle de **la paravirtualisation**.

Ils donnent aux systèmes invités l'accès au hardware sans émulation de celui-ci.

Les systèmes invités sont généralement modifiés pour que leur noyau puisse discuter avec le noyau de l'hyperviseur (noyaux dom0 et domU sous Xen Linux) et ainsi avoir accès directement au hardware.

**XEN, VMware ESX server, Hyper V (de microsoft)** peuvent être rangés dans la catégorie d'hyperviseurs de type 1.

Les hyperviseurs de type 2 font ce qu'on appelle **de la virtualisation** (et oui sans le « para » devant), en fait à mon sens ce sont plus des **virtualiseurs** que des hyperviseurs.

Ce sont généralement des logiciels que l'on fait tourner sur un système d'exploitation existant.

Ce logiciel va nous permettre de créer des machines virtuelles.

Il émuler du matériel à la machine virtuelle comme par exemple une carte réseau, un contrôleur IDE/SCSI, une carte graphique etc ...

Et bien sûr tout ce travail d'émulation de périphériques à un coût supplémentaire en terme de ressources CPU et les performances de votre système virtuel vont s'en ressentir.

Les performances d'un système paravirtualisé seront beaucoup plus proches de celles d'un système réel que les performances d'un système virtualisé.

C'est pour cette même raison que l'on a vu apparaître ces derniers temps des processeurs avec des instructions spécifiques dédiées à la virtualisation (VT chez Intel et pacifia chez AMD).

C'est un peu comme si vous ajoutiez un deuxième processeur (mais ce n'est pas le cas!) à votre processeur central, qui aura en charge uniquement l'émulation de matériel pour machine virtuelle et qui libèrera votre processeur central de cette charge.

Les machines virtuelles y gagneront donc en performances avec ce type de processeur.

**KVM (Kernel Virtual Machine), VMware Server, Virtualbox, qemu, bochs** peuvent être rangés dans la catégorie d'hyperviseur de type 2.

## I-C - Quelques définitions

**Définition d'un SAN :** Le SAN repose sur la fibre optique, il est un réseau à part optimisé pour le transfert à haute vitesse de blocs de données vers et en provenance d'un disque. Les baies de disques peuvent être connectées au SAN, n'encombrant pas le réseau local (LAN) dédié aux utilisateurs.

**Définition d'un HBA (Host Bus Adapter) :** HBA (Host Bus Adapter) se présente généralement sous forme d'une carte PCI (PCI, PCI-X, PCI-e) et permet de connecter un serveur à un réseau de stockage (notamment un réseau SAN).

**Définition d'un hyperviseur (wikipedia) :** Un hyperviseur est un noyau hôte allégé et optimisé pour ne faire tourner que des noyaux d'OS invités, adaptés et optimisés pour tourner sur cette architecture spécifique. Les applications en espace utilisateur des OS invités tournent ainsi sur une pile de deux noyaux optimisés, les OS invités ayant conscience d'être virtualisés.

On dit aussi que c'est une couche d'abstraction qui s'installe entre le système d'exploitation et le matériel.

**Logical Unit Number** Dans ce document nous appellerons LUN tout disque du réseau de stockage attribué à un serveur.

**Définition d'un volume physique :** Les disques durs provenant d'un SAN et pris en charge par LVM (Logical Volume Management) sont appelés volumes physiques.

**Définition d'un Volume Group :** Un Volume Group est un ensemble de volumes physiques mis côte à côte.

**Définition d'un logical Volume (ou volume logique) :** Un volume logique est un espace découpé dans un Volume Group. Cet espace pourra alors être utilisé par le système pour y stocker un système de fichiers et des données. L'avantage du volume logique c'est qu'il peut être agrandi/réduit à chaud selon les limites de place du volume Group auquel il appartient.

**Définition d'un snapshot :** Faire un snapshot d'un volume logique, c'est faire une photo de celui-ci à un instant T. A l'instant T le snapshot ne contient que des pointeurs sur les données du volume logique original. A l'instant T+1 lorsque le contenu du volume logique original aura changé, le snapshot stockera alors les différences entre la photo prise à l'instant T et le contenu du volume original à T+1.

**Système invité :** Dans cet article on désignera généralement par "système invité" une machine virtuelle/paravirtuelle.

**Système hôte :** Dans cet article on désignera généralement par "système hôte" l'OS chargé de créer/gérer les systèmes.

## II - Description du système d'information initial

Le système d'information du laboratoire dans lequel je travaille est constitué de nombreux serveurs sous Linux. Ces serveurs sont bien sûr reliés par un LAN Ethernet mais ils font également partie (par l'intermédiaire de deux adaptateurs HBA) d'un SAN (Stockage Area Network) sur lequel on trouvera une baie de disques et une librairie de sauvegarde.

Après délibérations, nous avons choisi d'utiliser les disques de la baie pour y stocker nos machines virtuelles.

### Plusieurs avantages:

- Grâce à notre baie de disques configurée en RAID 5 les données sont sécurisées et un crash de disque sur la baie n'empêchera pas votre système virtuel de tourner.
- Les mêmes disques provenant d'un SAN peuvent être vus par **plusieurs serveurs/hyperviseurs** ce qui nous permettra de pouvoir faire de la **migration de machine virtuelle à chaud entre hyperviseurs**.
- **Le multipath :** comme la plupart de nos serveurs possèdent deux adaptateurs HBA SAN, un disque sur le SAN attribué à un serveur sera vu comme deux périphériques distincts. Le démon multipath sous linux

(développé par C Varoqui) saura reconnaître en ces deux périphériques un seul et même disque (grâce à son identifiant SCSI) et nous permettra de faire de la répartition de charge au niveau des entrées sorties à partir des deux contrôleurs HBA. De plus si un des deux contrôleurs venait à tomber en panne, votre système ne sera pas corrompu, car le démon multipath permettra de faire passer les données par le contrôleur sain.

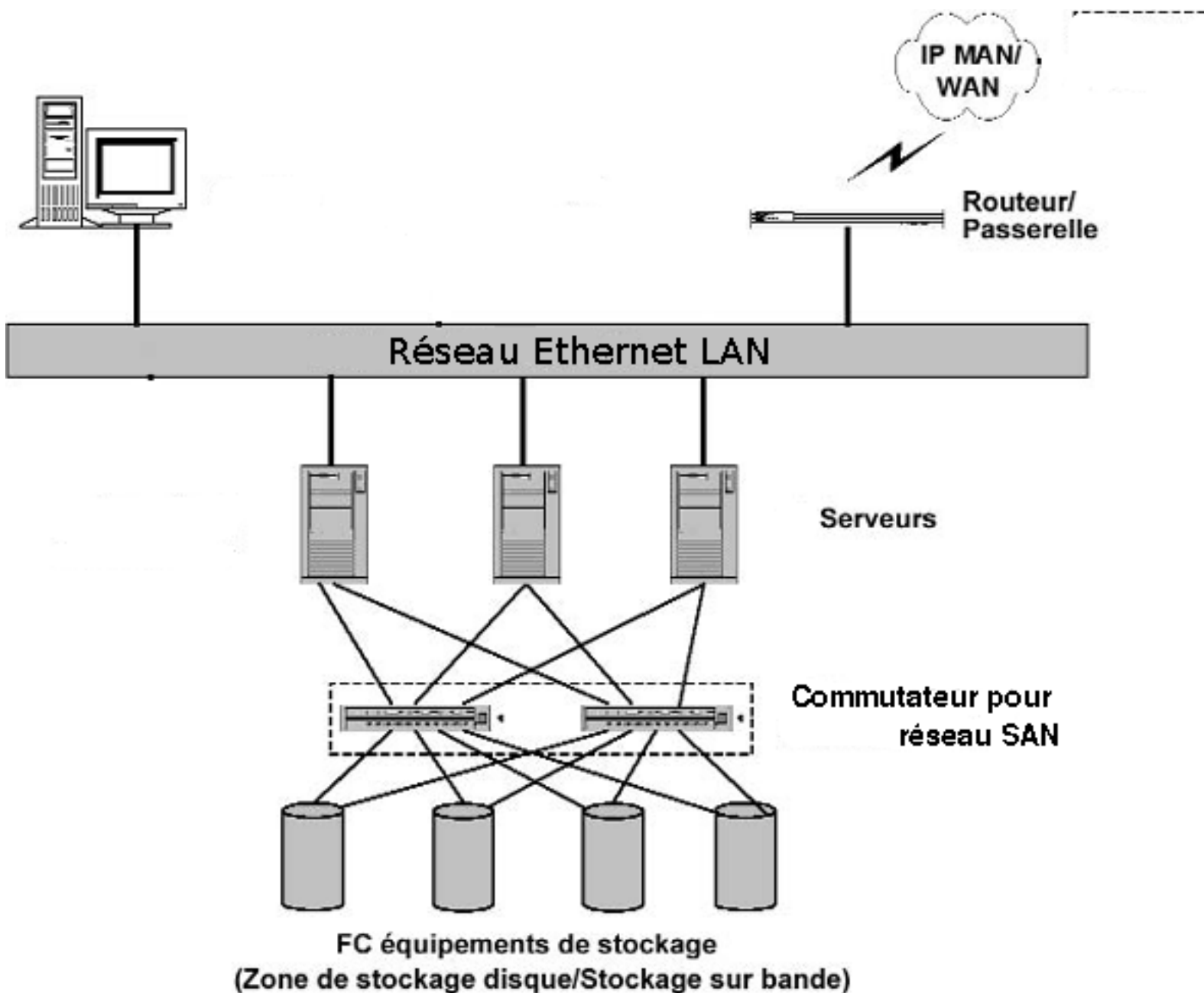


Schéma simplifié du système d'information ([www.siemon.com](http://www.siemon.com))

### III - Attribution d'un disque provenant du réseau de stockage à un serveur sous LINUX

Notre baie de disques, grâce à un logiciel propriétaire, peut rendre visible ses disques par l'intermédiaire du SAN à un ou plusieurs serveurs linux. On dit souvent plus classiquement que l'on **attribue une LUN à un serveur physique**. Lorsque vous attribuez une LUN à un serveur sous linux, la détection de celle-ci n'est pas automatique.

Sous le kernel 2.6 vous trouverez la plupart des informations connues du système sur votre adaptateur HBA dans ce répertoire :

**/sys/class/fc\_host/hostX** (où X est le numéro de votre adaptateur HBA)

Pour forcer votre système à scanner les périphériques qui lui sont attribués tapez la commande suivante :

```
echo 1 > /sys/class/fc_host/hostX/issue_lip
```

(donc quand vous disposez de deux adaptateurs HBA vous tapez la commande pour chacun d'eux) suivi d'un petit dmesg qui devrait vous indiquer si le disque en question a été vu par le système.

La commande **cat /proc/scsi/scsi** vous indiquera également quels sont les périphériques scsi/ san vus par votre système.

```
Attached devices :
Host: scsi0 Channel: 00 Id: 00 Lun: 00
Vendor: HITACHI Model: DF600F Rev: 0000
Type: Direct-Access ANSI SCSI revision: 03
Host: scsi0 Channel: 00 Id: 00 Lun: 01
Vendor: HITACHI Model: DF600F Rev: 0000
Type: Direct-Access ANSI SCSI revision: 03
Host: scsil Channel: 00 Id: 00 Lun: 00
Vendor: HITACHI Model: DF600F Rev: 0000
Type: Direct-Access ANSI SCSI revision: 03
Host: scsil Channel: 00 Id: 00 Lun: 01
Vendor: HITACHI Model: DF600F Rev: 0000
Type: Direct-Access ANSI SCSI revision: 03
```

Ici le système nous indique qu'il voit deux disques à partir de l'adaptateur scsi0 et deux disques à partir de l'adaptateur scsi1.

Il s'agit en fait des deux mêmes disques vus par deux adaptateurs différents.

Sous debian il existe un utilitaire qui s'appelle scsiadd qui permet comme son nom l'indique d'ajouter des disques un par un.

**scsiadd 1 0 0 1** va rajouter la ligne:

```
Host: scsil Channel: 00 Id: 00 Lun: 01
Vendor: HITACHI Model: DF600F Rev: 0000
Type: Direct-Access ANSI SCSI revision: 03
```

dans **/proc/scsi/scsi**

Une fois la LUN vue par votre système vous allez vouloir installer votre machine virtuelle sur celle-ci...

**Deux choix alors s'offrent à vous:**

Soit vous prenez la LUN telle qu'elle est et vous la partitionnez pour un système linux.

Soit vous transformez votre LUN en volume physique LVM,

```
pvcreate /dev/sdX
```

vous intégrez celui-ci à un volume group nommé vgvirtualdebian par exemple

```
vgcreate vgvirtualdebian /dev/sdX
```

Et enfin vous créez un volume logique d'environ 70 % de la capacité de votre LUN.

```
lvcreate -n lvsysteme -L <taille> vgvirtualdebian
```

Comme son nom l'indique ce sera le volume logique lvsysteme qui sera présenté en tant que disque système pour votre machine virtuelle.

L'avantage de cette solution est qu'elle permet de pouvoir faire des snapshots de votre système virtuel.

Une opération risquée à tester sur votre système linux virtuel ?

Un petit snapshot de votre système virtuel avec la commande :

Et on démarre une nouvelle instance de machine virtuelle à partir de lvsnap :-)

```
lvcreate --snapshot -L <taille> #name lvsnap /dev/vgvirtualdebian/lvsysteme
```

Besoin d'une sauvegarde de votre machine virtuelle mais pour des raisons de disponibilité vous ne pouvez pas vous permettre d'arrêter les services qui y tournent ?

Idem, créez un snapshot de votre système virtuel et effectuez la sauvegarde à partir du snapshot

## IV - Notion de multichemins ou multipath sous linux

Comme je l'ai spécifié plus haut, les serveurs de notre infrastructure, pour la plupart sont pourvus de deux cartes fibre (HBA).

Une lun sera donc vue par nos serveurs linux comme deux périphériques distincts.

Si ce n'est pas évident pour vous, imaginez que vous achetez un disque pourvus de deux connecteurs IDE.

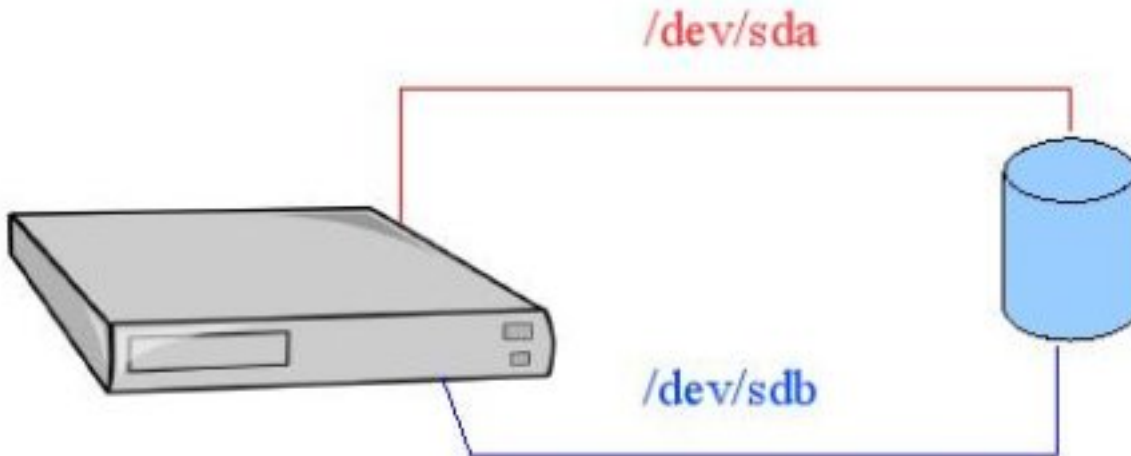
Chaque contrôleur IDE de votre carte mère sera relié au disque par l'intermédiaire d'une nappe.

Le fait que une de vos deux nappes IDE vienne à être défaillante ne sera pas gênant car vous pourrez toujours accéder à vos données par l'intermédiaire de la nappe IDE fonctionnelle (par contre pas de bol si les deux nappes ou les deux contrôleurs IDE viennent à être défaillants).

Or rappelez vous, tout périphérique sous UNIX/LINUX est défini par un fichier dans le répertoire /dev

Pour accéder alors au disque par l'intermédiaire de la première nappe le système va utiliser /dev/sda.

Et pour y accéder par l'intermédiaire de la deuxième nappe le système utilisera /dev/sdb.

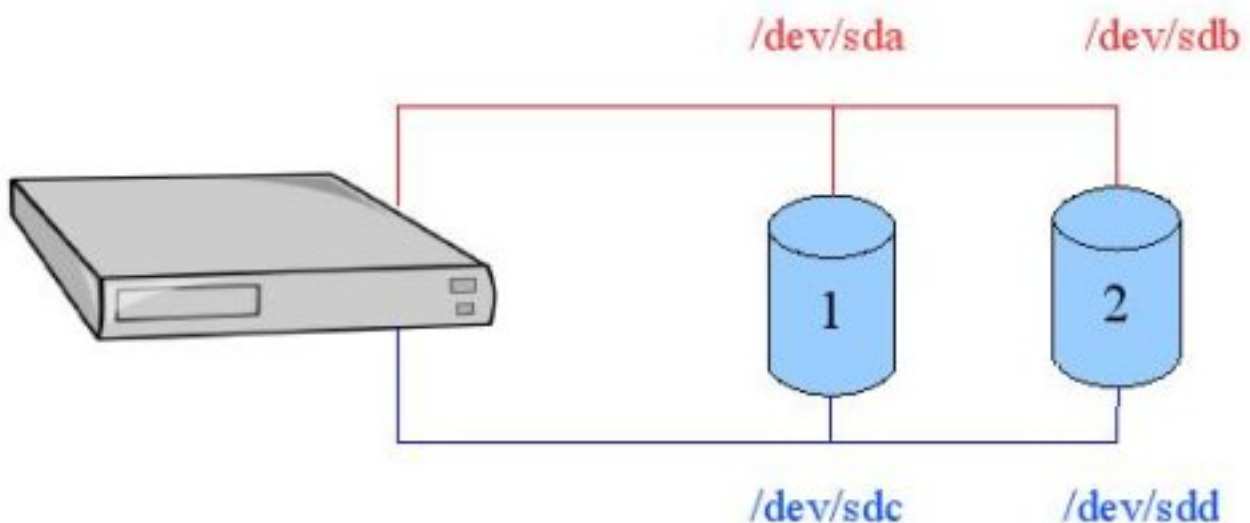


*Illustration simplifiée du multipath*

Si on pousse un peu plus loin la réflexion, on peut se dire qu'on pourra configurer notre système de façon à ce qu'il utilise /dev/sda par défaut donc de façon à ce qu'il utilise la première nappe IDE. En cas de défaillance de celle-ci le système commutera automatiquement sur la deuxième nappe. C'est le principe du **failover**.

Maintenant imaginez que vous possédez deux disques durs pourvus chacun de deux connecteurs cette fois non plus IDE mais SCSI. Imaginez également que vous disposez sur votre carte mère de deux contrôleurs SCSI et que sur chacun de ces contrôleurs SCSI vous ayez branché une nappe capable de brancher deux périphériques (ça tombe bien, on possède deux disques durs). Vous pouvez donc connecter sur chaque contrôleur SCSI les deux disques durs.

Le premier contrôleur verra donc le premier disque dur comme /dev/sda et le deuxième disque dur comme /dev/sdb. Le deuxième contrôleur verra le premier disque dur comme /dev/sdc et le deuxième comme /dev/sdd.



*Illustration du multipath multi disques*

Admettons qu'une importante opération d'écriture de données ait lieu sur le premier disque. L'envoi de données au disque se fait par l'intermédiaire de la première nappe, donc le système envoie des données sur sda.

Pendant ce laps de temps le système a un besoin d'écriture sur le deuxième disque. Comme la première nappe est déjà occupée par les données envoyées au premier disque le système prendra le chemin le moins encombré pour aller écrire sur le deuxième disque. Il s'agira donc de la nappe 2.

Dans ce cas là ce n'est plus du failover mais plutôt de la **répartition de charge**.

#### IV-A - Le mapping de périphérique :

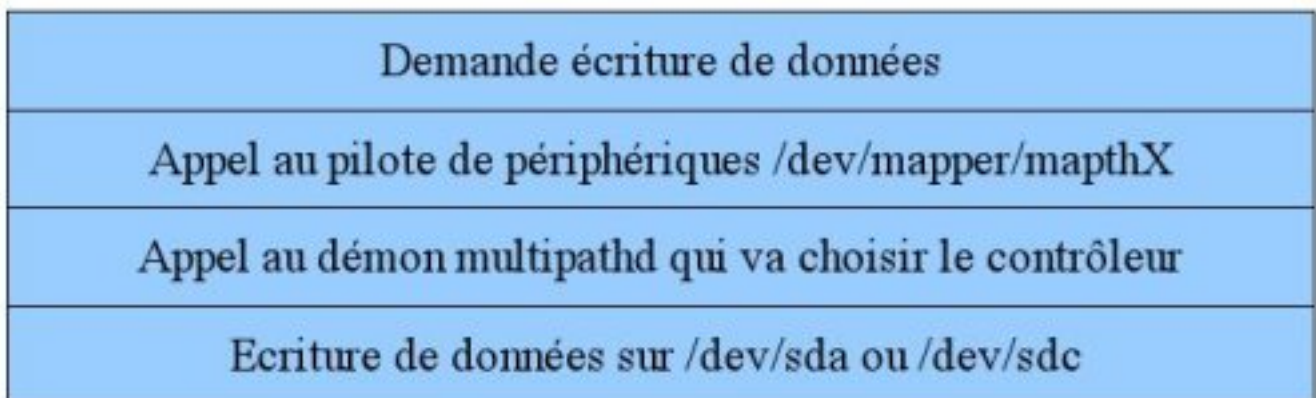
On l'a compris, le fait d'avoir deux contrôleurs fibre (HBA) sur nos serveurs nous permet de faire du multipathing (failover et/ou répartition de charge).

Mais concrètement comment ça va se passer au niveau de notre système linux ?

Prenons le cas précédent où le premier disque dur vu par le système (cf schéma ci-dessus) était accessible par /dev/sda et /dev/sdc.

Le démon multipath va faire appel au module dm-mod (device mapper) pour créer un mapping de périphérique, c'est-à-dire une couche intermédiaire, entre le disque tel qu'il est vu par le système et les données écrites réellement sur le disque.

Lorsque le système voudra écrire des données sur le disque, il adressera le nom de périphérique mappé.



*Illustration du fonctionnement du mapping de périphériques*

Pour la lecture des données le principe sera le même.

En fait il suffit d'adresser les données à /dev/mapper/mpathX.

Pour connaître la valeur de X il suffit de taper la commande **/sbin/multipath -ll** sous root.

```

/sbin/multipath -ll

mpath2 (1HITACHI_D60095920137) dm-6 HITACHI,DF600F
[size=100G][features=1 queue_if_no_path][hwhandler=0]
\_ round-robin 0 [prio=2][active]
\_ 0:0:1:1 sdb 8:16 [active][ready]
\_ 1:0:0:1 sdo 8:224 [active][ready]
```

Dans le cas ci-dessus il y a un mapping entre /dev/mapper/mpath2 et /dev/sdb, /dev/sdo.

Nous avons donc mis en place le multipathing sur les serveurs de notre système d'information.

Pour faire tourner un système d'exploitation sur un disque distant mieux vaut prendre le plus de précautions possibles.

#### **Avantages:**

- Sécurité accrue. Un des contrôleurs HBA tombe, la machine virtuelle aura toujours accès à son disque système.
- Performances : Un des chemins vers le disque système est encombré? Pas grave on passe par l'autre.

## Inconvénients:

- ça demande un peu plus de temps pour la mise en place de nos machines virtuelles, le temps de configurer correctement le démon multipath (/etc/multipath.conf)
- Au niveau de la migration à chaud de machines virtuelles mais nous en reparlerons dans le chapitre qui lui est consacré.

## V - L'hyperviseur XEN

### V-A - Virtualisation et paravirtualisation

#### Sous XEN il y a deux façons de faire

- Virtualisation totale.
- Paravirtualisation

Dans le premier cas le système invité n'a pas conscience d'être virtualisé puisque le système de virtualisation lui émule un vrai bios ainsi que du matériel comme une carte graphique, carte son, carte réseau, contrôleur IDE, etc... Ainsi on peut prendre son cd et installer sa distribution linux préférée comme s'il s'agissait d'une machine réelle.

Dans le deuxième cas, le système a conscience d'être un système invité puisque celui-ci a été adapté pour être paravirtualisé.

Le noyau utilisé pour faire tourner notre système paravirtualisé a été spécialement conçu pour communiquer avec le noyau de l'hyperviseur.

Cette fois pas question de faire voir au système une carte graphique, son etc ... On ne s'embarrassera pas avec ce genre de détail. Un système linux peut très bien fonctionner sans carte graphique.

Tout ce dont on a besoin après tout c'est d'un système pouvant communiquer par le réseau et qui ait assez de ressources pour faire tourner nos applications.

L'avantage de la paravirtualisation est que ses performances sont très proches d'un système réel (plus proches que dans le cas d'une virtualisation totale).

Autre avantage : **la flexibilité.**

### V-B - Limites de la virtualisation totale

Comme je le disais plus haut la virtualisation totale fait croire au système virtuel qu'il fonctionne sur une **vrai machine**. Au niveau stockage le système virtuel verra deux contrôleurs IDE (un primaire et un secondaire).

Malheureusement ces deux contrôleurs IDE nous limitent donc à quatre disques par machine virtuelle.

Ceci n'est malheureusement pas satisfaisant pour nos besoins de développement/production.

Chez nous, une machine qu'elle soit réelle ou virtuelle peut être amenée à faire tourner beaucoup d'applications au cours de sa vie.

Ces applications grandissent, requièrent plus d'espace disque et nous ne sommes pas capables par avance d'évaluer les besoins finaux en terme de stockage.

D'où l'intérêt de pouvoir attribuer de l'espace disque supplémentaire à la volée et sans limitation à une machine afin de pouvoir agrandir les espaces alloués aux applications à la demande.

Il se peut que à l'heure où j'écris ces lignes cette limitation soit maintenant levée.

Il me semble que sous KVM (Kernel Virtual Machine : la virtualisation native Linux) qui tout comme xen en mode virtualisation totale utilise une version modifiée de qemu, il est possible désormais d'ajouter des disques à chaud et que ceux-ci soient considérés comme disques SCSI par le système invité.

Par contre je ne sais pas ce qu'il en est pour xen.

## V-C - Le choix entre la virtualisation totale et la paravirtualisation

### En mode paravirtuel, il est possible :

- d'attribuer des disques supplémentaires à chaud au système invité,
- d'augmenter la mémoire RAM à chaud,
- d'attribuer de la cpu supplémentaire,
- d'attribuer une nouvelle interface réseau à la volée.

Bref la paravirtualisation représentait à nos yeux de meilleures performances et une flexibilité supplémentaire non négligeable.

Le choix n'aura donc pas été long, nos systèmes invités seront donc paravirtualisés.

## V-D - Notion de domaine 0, domaine utilisateur (domU) :

Pour transformer votre système linux en hyperviseur, vous devez installer un noyau spécifique patché XEN.

La plupart des distributions fournissent ce noyau dans le système de packages.

Lorsque ensuite vous bootez votre machine sur le noyau XEN, celui-ci va démarrer le système installé sur les disques durs de votre serveur physique et le considérer comme le premier système invité.

Il s'agit du domaine 0, le système invité à partir duquel vous allez pouvoir contrôler tous les autres.

Les systèmes invités créés à partir du domaine 0 seront les domaines utilisateurs ou plus communément appelés domU.

Dans la branche Xen 2.X il fallait un noyau spécifique pour le domaine 0 et un noyau pour domU.

Aujourd'hui vous pouvez (mais ça n'est pas une obligation) utiliser le noyau du domaine 0 pour démarrer votre domU.

Quoiqu'il en soit, il faudra toujours que le noyau que vous utilisez pour démarrer votre domU soit stocké sur le dom0.

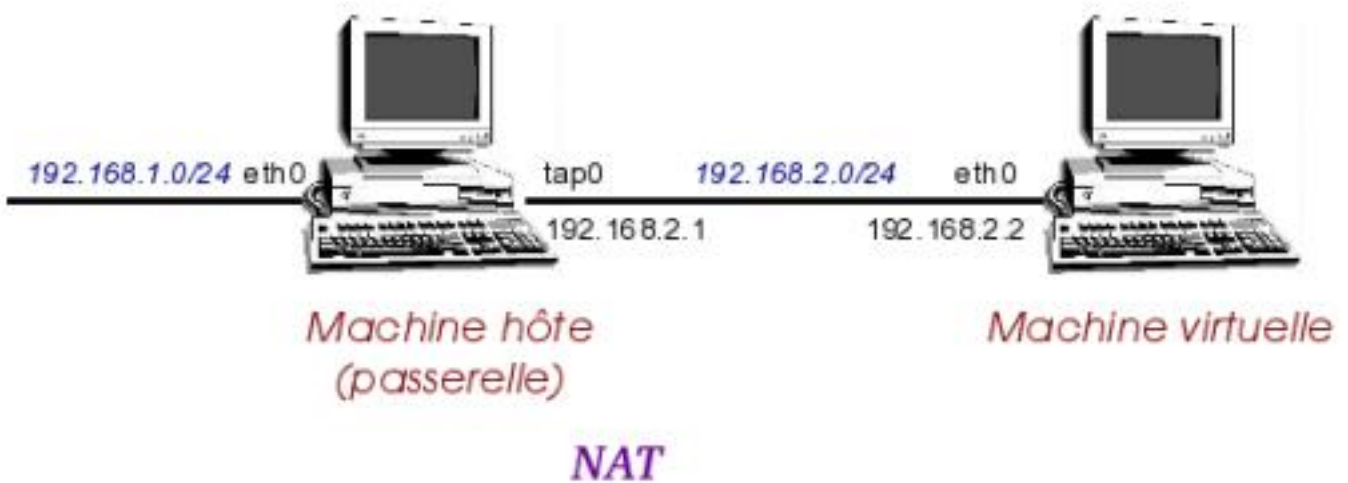
## V-E - Notion de réseau Naté et réseau bridgé :

C'est une notion que l'on retrouve dans la plupart des virtualiseurs/hyperviseurs que ce soit KVM, Vmware ou encore xen.

Ce qu'il faut retenir pour le mode NAT (Network Address Translation), **c'est que la machine/l'hyperviseur sur lequel tourne votre système invité fera office de passerelle.** Chaque fois que le système invité voudra envoyer des paquets sur le réseau, ceux-ci seront d'abord envoyés à la passerelle pour y subir une petite transformation.

Lorsque la passerelle recevra des paquets IP provenant du système invité et à destination du réseau elle va tout simplement remplacer l'adresse IP source de ce paquet (donc l'adresse du système invité) par la sienne (l'adresse IP de la passerelle) et ensuite envoyer le paquet sur le réseau.

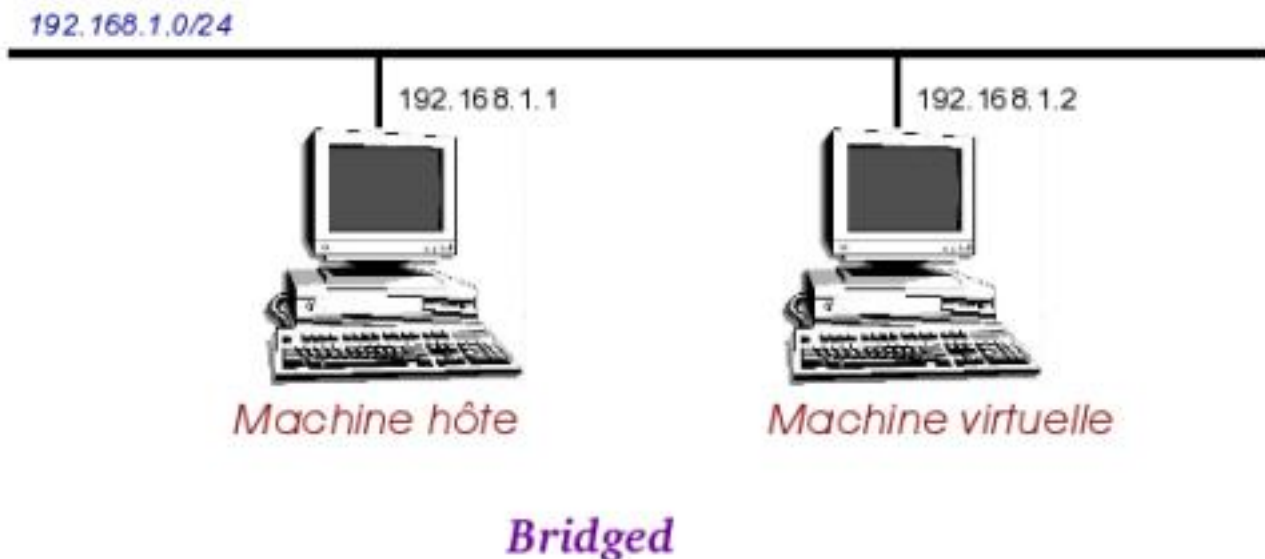
Ainsi le destinataire du paquet aura toujours l'impression que le paquet a été émis directement de la passerelle et n'aura donc aucune connaissance de la machine virtuelle.



*Illustration du principe de translation d'adresse (<http://www.adella.org/spip/QEMU-Configurer-le-reseau>)*

Ce qu'il faut retenir pour le mode bridgé c'est que la machine virtuelle sera sur le même réseau que sur le système hôte.

En fait en mode bridgé vous créez en quelque-sortes un hub/commutateur sur lequel votre système hôte et votre système invité seront connectés. Et ce commutateur sera lui-même connecté au réseau physique.



*Illustration du principe du mode bridgé (<http://www.adella.org/spip/QEMU-Configurer-le-reseau>)*

## V-F - Installer un système paravirtualisé

Comme je le disais plus haut, un système linux paravirtualisé est un système linux modifié.

La plupart des cdrom d'installation des distributions installent un système linux « normal » donc incompatible avec le système de paravirtualisation offert par xen.

La seule solution que nous ayons trouvée à ceci est d'installer le système linux invité en mode totalement virtuel.

(Toutefois s'il s'agit d'une distribution debian ou dérivée l'utilitaire **debootstrap** nous permet d'éviter d'avoir à installer le système en mode totalement virtualisé).

Une fois ceci fait, nous pourrions alors « Xenifier » le système c'est-à-dire installer sur le système le noyau xen pour système invité.

## V-G - Fichier de configuration d'un système paravirtualisé :

Création du fichier de configuration de notre machine virtuelle :

### Pas compliqué, il nous faut:

- un noyau et son image initrd sur lesquels démarrer (le noyau pour domaine Utilisateur),
- le nom de la machine virtuelle,
- le nombre de coeur CPU sur lesquels la machine va tourner,
- la quantité de RAM qui sera attribuée à la machine,
- dans le cas où vous décidez que votre machine virtuelle sera en mode bridgé (une adresse MAC, plus le nom du pont),
- le disque physique sur lequel devra démarrer la machine,
- quelques paramètres optionnels.

Voici un exemple de fichier de configuration basique :

```
#Je prends le noyau xen de l'hyperviseur
kernel = "/boot/vmlinuz-2.6.18-6-xen-686"
#Je prends l'initrd de l'hyperviseur
ramdisk = "/boot/initrd.img-2.6.18-6-xen-686"
# Mémoire par défaut de la machine virtuelle
memory = 2048
# on pourra monter à chaud la mémoire jusque 4 Go
maxmem = 4096
#Nom de l'instance de la VM
name = "virtualdebian"
#celle-ci tournera sur 2 coeurs de processeurs
vcpus = 2
#mode bridgé avec attribution d'adresse mac
#à l'interface réseau de la VM
vif = [ 'mac=00:16:3e:00:00:01, bridge=xenbr0' ]
#ici j'attribue le volume logique lvsysteme à la machine virtuelle
#et je le fais voir en tant que premier disque dur scsi (/dev/sda) en écriture (w)
disk = [ 'phy:/dev/vgvirtualdebian/lvsysteme,sda,w' ]
#ici j'indique au noyau que la racine de mon système virtuel se trouve
#sur le volume logique lvroot qui se trouve lui même dans le Volume Group VolGroup00
root = "/dev/mapper/VolGroup00-lvroot ro"
```

### V-G-1 - Commentaires sur le fichier de configuration ci-dessus

C'est un fichier de configuration assez dépouillé et c'est volontaire, mon but étant d'aller à l'essentiel.

Les lignes **kernel** et **ramdisk** font référence à un fichier présent sur l'hyperviseur, en effet on se sert du noyau de l'hyperviseur pour démarrer notre machine virtuelle ce qui implique que vous devez avoir installé ce noyau également sur la machine virtuelle ne serait-ce que pour avoir le répertoire `/lib/modules/2.6.18-6-xen-686` qui contient les modules associés au noyau.

**Memory=2048, maxmen=4096** : par défaut la machine virtuelle démarrera avec 2048 Mo de mémoire, on pourra à chaud augmenter la mémoire de cette machine jusqu'à 4096 Mo

**vif = [ 'mac=00:16:3e:00:00:01, bridge=xenbr0' ]** :ici on choisit le mode bridgé, on assigne à la machine une adresse MAC, et la machine virtuelle sera sur le même réseau que l'hyperviseur.

Je conseille vivement aux utilisateurs d'assigner une adresse MAC à leur machine virtuelle. Si vous ne le faites pas Xen va attribuer une adresse MAC aléatoire, et sur certains systèmes comme debian, ubuntu le démon udev crée des règles de nommage de périphérique réseau en fonction de leur adresse MAC.

Exemple de règle :

```
SUBSYSTEM=="net", DRIVERS=="?* ", ATTRS{address}=="00:16:3e:00:00:01",NAME="eth0"
```

Ce que l'on pourrait traduire par :

Je nomme eth0 toute interface réseau dont l'adresse mac est 00:16:3e:00:00:01

Si vous redémarrez ensuite votre machine virtuelle et que l'adresse mac de l'interface a changé une nouvelle règle udev va être générée et ainsi vous aurez donc deux règles comme suit :

```
# Xen virtual device (vif)
```

```
SUBSYSTEM=="net", DRIVERS=="?* ", ATTRS{address}=="00:16:3e:00:00:01",  
NAME="eth0"
```

```
# Xen virtual device (vif)
```

```
SUBSYSTEM=="net", DRIVERS=="?* ", ATTRS{address}=="00:16:3e:38:ef:cc",  
NAME="eth1"
```

et votre interface réseau se nommera eth1 (et comme vos fichiers de configuration d'interface réseau ont été créés pour eth0 du coup vous n'aurez plus de réseau).

### Explication:

Vous démarrez votre machine virtuelle une première fois, xen a attribué à votre interface réseau virtuelle l'adresse MAC 00:16:3e:00:00:01, une première règle udev a été générée (cf ci-dessus). Vous redémarrez votre machine une deuxième fois, xen a attribué à l'interface réseau une nouvelle adresse MAC 00:16:3e:38:ef:cc, une deuxième règle s'est créée, or le nom eth0 étant déjà pris par la première règle, udev nommera eth1 l'interface réseau d'adresse MAC 00:16:3e:38:ef:cc. Pour éviter donc la génération de ces règles udev, il faut donc attribuer une adresse MAC dans le fichier de configuration.

```
disk = [ 'phy:/dev/vgvirtualdebian/lvsysteme,sda,w']
```

Ici nous attribuons le volume logique lvsysteme appartenant au volume Group vgvirtualdebian, à la machine virtuelle et il sera présenté en tant que disque scsi sda.

On aurait pu partitionner directement la lun du san avec la commande fdisk et dans ce cas on aurait eu cette ligne dans le fichier de configuration.

```
disk = [ 'phy:/dev/mapper/mpathX,sda,w']
```

```
root = "/dev/mapper/VolGroup00-lvroot ro"
```

J'indique à mon noyau (défini dans le fichier de configuration par la ligne **kernel**) que la partition racine de mon système virtuel se trouve sur le volume logique lvroot du volume group VolGroup00

## V-H - Démarrer votre machine virtuelle :

Une fois votre fichier de configuration créé et sauvegardé il est grand temps de le tester.

Nous allons donc créer notre premier domU avec la commande suivante :

```
xm create -c <nom_du_fichier_de_configuration>
```

L'option **-c** nous permet de suivre à l'écran la séquence de boot de la machine virtuelle (comme si vous démarriez un serveur physique et que vous étiez derrière l'écran).

Une fois la séquence de boot terminée nous obtenons la bannière de login du système (tout comme un vrai système).

## V-I - Quelques commandes sympathiques

Sous xen une commande ultime à retenir :

**La commande xm.**

A partir d'un terminal avec cette commande vous pourrez contrôler de nombreux aspects de votre machine virtuelle/paravirtuelle (démarrage, arrêt, attribution mémoire, disque etc...)

### Démarrer une machine virtuelle :

```
xm create -c <nom_du_fichier_de_configuration>
```

le **-c** vous servira à suivre la séquence de boot de votre machine virtuelle en mode paravirtualisation mais sera totalement inutile en mode virtualisation (puisque le démarrage d'une machine virtuelle va provoquer le démarrage d'une instance de qemu).

### Lister les machines virtuelles en fonctionnement :

```
xm list
Retourne une liste de ce type:
Name      ID Mem VCPUs State
Domain-0  0  512   1  r-----
debvirt   20 2048   2  -b----
```

### Figurer une machine virtuelle :

```
xm pause <nom_de_machine_virtuelle>
ou
xm pause <id_machine_virtuelle>
```

### Sauvegarder l'état d'une machine virtuelle dans un fichier :

```
xm save <nom_machine_virtuelle> <file>
ou
xm save <id_machine_virtuelle> <file>
```

### Restaurer une machine virtuelle à partir d'un fichier :

```
xm restore <file>
```

### Affichage de l'état des machines virtuelles un peu comme avec la commande top :

```
xm top
```

### Accéder à la console d'une machine paravirtualisée :

```
xm console <nom_machine_virtuelle>
ou
xm console <id_machine_virtuelle>
```

### Attacher un disque à chaud de l'hyperviseur vers une machine paravirtuelle :

```
xm block-attach <domain> <nom_de_peripherique_disque_sur_hyperviseur>
<nom_voulu_du_peripherique_sur_VM> <mode:lecture,ecriture>
```

(ici domain = nom\_machine\_virtuelle ou id\_machine\_virtuelle)

Exemple :

**xm block-attach 5 phy:/dev/sdb /dev/sdc w**

(J'attache à la machine virtuelle dont l'ID est 5 le disque /dev/sdb et je lui fait voir en tant que disque /dev/sdc et j'autorise l'écriture).

Avec ce type de commande inutile de redémarrer la machine virtuelle, grâce à dbus et udev le disque sera vu tout de suite et un fichier /dev/sdc sera créé.

**Détacher un disque à chaud d'une machine virtuelle :**

Si le disque en question n'est pas utilisé par la machine virtuelle vous avez la possibilité de le détacher de la machine virtuelle.

```
xm block-detach <id_domaine> <nom_de_péripherique>
```

Exemple :

Détacher le disque que l'on avait alloué au dessus avec la commande xm block-attach

**xm block-detach 5 /dev/sdc**

**Redémarrer une machine virtuelle :**

```
xm reboot <nom_machine_virtuelle>
ou
xm reboot <id_machine_virtuelle>
```

(Dans le cas d'une machine paravirtualisée sous linux --> redémarrage linux)

**Éteindre une machine virtuelle :**

```
xm shutdown <nom_machine_virtuelle>
ou
xm shutdown <id_machine_virtuelle>
```

Provoquera un arrêt classique d'une machine virtuelle : arrêt des services + extinction machine virtuelle.

**Stopper immédiatement une machine virtuelle :**

Provoquera un arrêt brutal de votre machine virtuelle (un peu comme si vous débranchiez le câble d'alimentation).

```
xm destroy <nom_machine_virtuelle>
ou
xm destroy <id_machine_virtuelle>
```

**Confier la gestion d'une machine virtuelle à un autre hyperviseur :**

```
xm migrate [-l] <nom_ou_id_machine_virtuelle> <adresse ip ou nom réseau de l'hyperviseur cible>
```

Le -l permet de faire de la migration à chaud (sauvegarde des pages mémoires, figer la machine, restauration des pages mémoires sur l'hyperviseur cible, on défige la machine).

Sans le « -l » «vous éteignez la machine et vous la redémarrez sur l'hyperviseur cible.

## VI - La migration à chaud de machines virtuelles

Parmi les fonctionnalités proposées par xen nous avons « la migration de machines virtuelles à chaud d'un hyperviseur à un autre ».

En d'autres termes cette fonctionnalité permet de confier la gestion d'un domU à un autre hyperviseur sur le réseau. Une mise à jour matérielle nécessite le redémarrage d'un de vos hyperviseurs (on l'appellera l'hyperviseur A) ? Vous possédez un deuxième hyperviseur (l'hyperviseur B) sur le réseau ?

Pour éviter d'avoir à arrêter les domU gérés par votre hyperviseur A en même temps que celui-ci vous avez donc la possibilité de confier la gestion des domU de votre hyperviseur A à l'hyperviseur B.

La migration se fait à chaud (pas besoin d'arrêter le domU sur l'hyperviseur A pour ensuite avoir à le redémarrer sur l'hyperviseur B).

La continuité de services est donc assurée puisque les services de vos domU ne sont pas interrompus (ou pendant un laps de temps très bref de l'ordre de moins de 10 secondes).

Une fois l'intervention faite sur votre hyperviseur A et celui-ci redémarré, il ne vous reste plus qu'à faire l'opération inverse.

**Pour que ceci puisse se dérouler sans encombre il faut que certaines conditions soient réunies :**

- il faut en premier lieu que le disque sur lequel le domU fonctionne soit vu des deux hyperviseurs,
- il faut que les deux hyperviseurs soient de préférence sur le même LAN,
- il faut que l'hyperviseur B autorise l'hyperviseur A à dialoguer avec lui et inversement.

Et surtout il faut que le nom de périphérique correspondant au disque partagé (celui qui contient notre domU à migrer) entre les deux hyperviseurs **soit le même sur les deux systèmes physiques.**

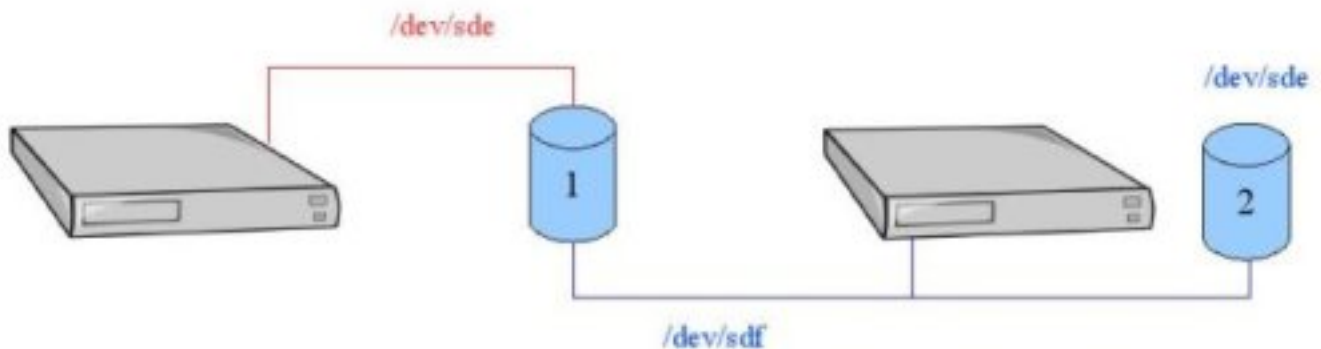
En effet la première chose que va faire Xen lors d'une migration à chaud de domU, c'est envoyer le contenu du fichier de configuration du domU à migrer à l'hyperviseur cible.

Imaginez que le disque qui contient les partitions système de votre domU soit vu par l'hyperviseur A comme étant **/dev/sde** et que nous décidons que celui-ci sera vu sur le domU comme étant le premier disque SCSI.

Nous aurons donc dans notre fichier de configuration une ligne de ce type :

**disk=['phy:/dev/sde,/dev/sda:w']**

Imaginez maintenant que sur l'hyperviseur B ce disque soit vu comme étant **/dev/sdf** et que **/dev/sde** existe. Sur l'hyperviseur B **/dev/sde** ne contient donc pas les partitions de notre domU à migrer.



*Même nom mais pas les mêmes disques référencés*

Lors de la migration, l'hyperviseur A va envoyer à notre hyperviseur B le fichier de configuration du domU. Une fois les pages mémoire du domU rechargées sur l'hyperviseur B, le domU va essayer de retrouver ses données sur /dev/sde (celui de l'hyperviseur B donc celui qui ne contient pas les partitions système de notre domU) et ça risque grandement de coincer.

Quand le domU est installé sur un volume logique (LVM), il n'y a à priori pas de problèmes.

Comme le disque contenant le domU est vu par les deux hyperviseurs, les mêmes groupes de volume et volume logique seront vus par les deux hyperviseurs.

## VI-A - Le grand retour du multipath

Rappelez-vous un périphérique multipathé est généralement **nommé /dev/mapper/mpathX** (où X est un entier). Donc si les partitions système de votre domU sont installées sur une lun multipathée nous aurons une ligne de ce type dans le fichier de configuration de notre domU  
**disk=['phy:/dev/mapper/mpath5,/dev/sda,w']**

Dans ce cas là aussi il y a un risque qu'une lun multipathée soit dénommée **/dev/mapper/mpath5** sur l'hyperviseur A et **/dev/mapper/mpath6** sur l'hyperviseur B.

N'étant pas intime avec le noyau linux j'ignore comment le système décide de numéroter ses périphériques multipath. Pour contourner le problème nous pouvons en fonction de **l'ID SCSI du disque partagé** (de la lun multipathée) **créer un alias**.

Ceci se passe au niveau du fichier de configuration du démon multipath.

Cela consiste donc à dire que pour tout disque dont l'ID SCSI est XYZ, nous voulons que celui-ci ne soit plus nommé **/dev/mapper/mpathX** mais plutôt **/dev/mapper/domU-systeme\_XYZ**.

Il faudra alors avoir configuré le démon multipath de manière identique sur les deux hyperviseurs, donc en d'autres termes avoir le même fichier /etc/multipath.conf sur les deux serveurs physiques.

## VI - Conclusion

Voilà nous arrivons à la fin de cet article.

J'espère qu'il aura pu vous donner un aperçu des capacités de XEN.

J'ai essayé ici tant bien que mal d'aborder des notions nécessaires à connaître avant de se lancer dans la virtualisation et également de vous faire un petit retour d'expérience sur la mise en place de systèmes invités dans un environnement dédié au stockage (réseau SAN) tel que nous l'avons dans le laboratoire où je travaille.

Je vous confie maintenant les manettes, à vous de vous faire votre propre expérience avec XEN et les machines virtuelles. A vos claviers :-)

CedrX

## VIII - Remerciements

Je tiens tout particulièrement à remercier **mlny84** et **Celira** pour leur relecture orthographique.

Je tiens également à remercier **gorgonite** et **ovh** qui m'auront guidé durant la création de mon article.

Enfin je dis un grand merci aux admins de developpez qui m'ont ouvert un espace afin que je puisse publier cet article.