Table des matières

Table des matières 1
Introduction 2
I – La virtualisation 3
1. Les principes de la virtualisation 3
1.1. Les différents types de virtualisation *3*
A. Les isolateurs 4
B. La virtualisation totale 5
C. La virtualisation matérielle assistée 6
D. La paravirtualisation 7
1.2 – Les intérêts de la virtualisation *8*
2. XEN 11
2.1. L'outil Xen *11*
2.2. Installation de Xen *12*
2.3. Installation de machines virtuelles *13*
II - Etudes 14
1. Introduction 14
2. Etude du réseau 14
2.1. Étude et expérimentation des modes de configuration réseau *15*
A. Les ponts 16
a) Principe 16
b) Expérimentations 17
B. Routage 20
a) Principe 20
b) Expérimentations 20
C. NAT 21
a) Principe 21
b) Expérimentations 22
2.2. Utilisation des VLANs *24*
2.3. Bilan *26*
3. Etude du cloisonnement 27
3.1. Mise en place de l'architecture virtualisée *28*
3.2. Mise en place des machines virtuelles *29*
3.3. Protocole expérimental *30*
3.4. Résultats expérimentaux *34*
Conclusion 41
Bibliographie 43
Annexes 44
Tableau de données des résultats expérimentaux de l'étude sur le cloisonnement 44

Introduction

Nous avons réalisé ce rapport dans le cadre d'une TX réalisée au cours de notre formation à l'Université de technologie de Troyes, en Systèmes d'information et Télécommunications et encadrée par Guillaume Doyen. Cette UV se place dans la suite de l'AC qui avait été effectuée par Romain Hinfray et Antoine Benkemoun qui avait permis d'établir une base de connaissances théoriques sur la virtualisation.

La virtualisation est un sujet d'actualité et d'innovation en plein essor dans le milieu de l'informatique et nous avons estimé qu'il était important pour nous de travailler sur ce domaine pour notre future vie professionnelle.

Le but de ces travaux est d'acquérir des connaissances et des compétences dans le déploiement et l'utilisation de la virtualisation. Nous souhaitions pouvoir avoir une approche plus pratique de la virtualisation afin d'acquérir des compétences dans le déploiement d'une plateforme. Le support de cette acquisition de compétences a été l'étude de deux aspects de la virtualisation à travers un protocole expérimental réitérable et une analyse des résultats obtenus. Nous avons choisi d'utiliser la solution libre Xen car cette solution avait déjà été étudiée en profondeur lors de l'AC.

Les deux aspects que nous avons choisis sont la gestion du réseau et le cloisonnement des performances. Le premier aspect se justifie par les spécificités apportées par la virtualisation en termes de réseau ainsi que par la spécialisation de notre formation en Intégration Réseau. Le second aspect a pour objectif de démontrer les avantages de la virtualisation de manière quantitative par rapport à une plateforme classique. Cet aspect nous a également permis de pouvoir vérifier par nous même les arguments mis en avant par la solution de virtualisation.

Nous commencerons par rappeler les principes et les enjeux de la virtualisation ainsi que le fonctionnement de Xen. Puis, nous détaillerons le déroulement des deux études, à savoir, une explication théorique, un protocole expérimental et une analyse des résultats obtenus.

I – La virtualisation

1. Les principes de la virtualisation

La virtualisation est un procédé informatique qui consiste à exécuter plusieurs systèmes d'exploitation et/ou applications indépendants les uns des autres sur une seule machine, comme s'ils étaient sur des machines physiques différentes. Ainsi, la virtualisation permet de conserver un environnement cloisonné entre les différentes machines virtuelles de la même façon qu'entre des machines physiques.

Ce procédé partage les ressources matérielles d'une machine entre plusieurs systèmes d'exploitation en ordonnançant l'accès matériel de manière à ce que cette gestion soit transparente du point de vue des systèmes d'exploitation virtualisés. Ainsi les systèmes d'exploitation sont traités comme des processus par l'ordonnanceur qui leur alloue un espace virtuel d'adressage de façon à ce qu'ils croient être les seuls sur la machine physique.

1.1. Les différents types de virtualisation

La virtualisation est une extension du principe d'émulation qui consiste à simuler le fonctionnement d'un logiciel afin de l'exécuter sur n'importe quel type de matériel compatible ou non. Il existe plusieurs types de virtualisation qui ont chacun leurs caractéristiques spécifiques.

On trouve d'abord la virtualisation des processus. Le système d'exploitation crée un environnement cloisonné au niveau applicatif où les processus peuvent s'exécuter indépendamment les uns des autres. Ensuite, on trouve la virtualisation des systèmes modifiés ou non modifiés. Les systèmes non modifiés sont les plus connus. Il s'agit de virtualisation totale comme l'utilise l'outil VMware. Celle-ci permet d'exécuter facilement tout système d'exploitation comme nous l'expliquerons ci-après. La virtualisation matérielle assistée est similaire à la virtualisation totale, cependant elle diminue les pertes de performances dues à l'émulation du matériel en tirant parti des instructions processeur. La paravirtualisation nécessite que le système d'exploitation ait été modifié afin de supporter ce type de

virtualisation. La figure N°1 représente la classification de ces différents types de virtualisation.

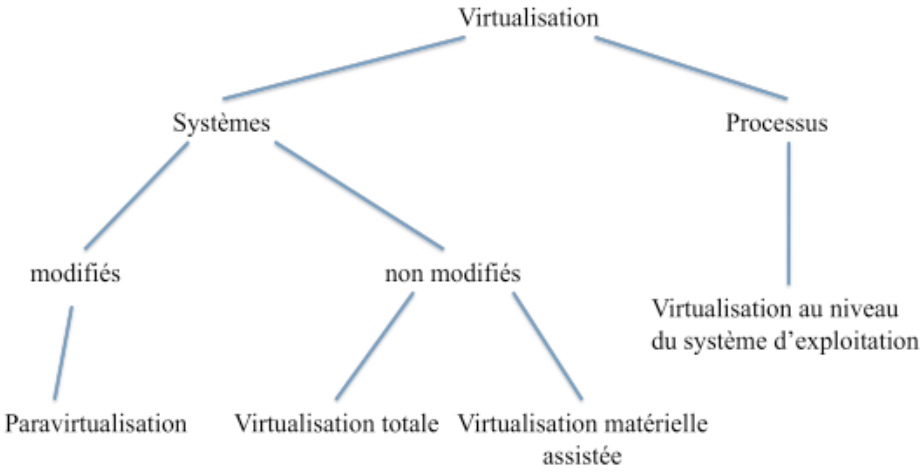

Figure 1 : Classification des différents types de virtualisation

A. Les isolateurs

Les isolateurs sont des logiciels qui permettent de créer un environnement utilisateur cloisonné au sein d'un système d'exploitation. Cet environnement peut alors exécuter des programmes sans que leur exécution ne perturbe le système d'exploitation de la machine en cas de dysfonctionnement. Ces environnements sont appelés des contextes ou bien des zones d'exécution. Cette technique est peu coûteuse en ressources et très performante car il n'y a pas besoin d'émuler le matériel.

Cependant elle est limitée puisque les environnements appartiennent tous au même système d'exploitation et ils ont tous accès directement au matériel. L'isolation n'est donc pas parfaite entre les différentes zones d'exécution. Il est tout de même possible d'effectuer de l'allocation de ressources entre les environnements notamment en attribuant des quotas sur des ressources telle que l'espace disque, la mémoire vive disponible et la bande passante réseau.

Ainsi, comme le montre la figure 2, le système d'exploitation avec le noyau a accès au matériel. Au-dessus, se situe une couche logicielle qui permet de créer et de gérer les environnements cloisonnés. C'est pourquoi ce type de virtualisation se situe au niveau du

système d'exploitation, seuls des processus sont isolés et non pas des systèmes d'exploitation à part entière. On peut citer en exemple Linux-VServer, OpenVZ, chroot, Zones Solaris et Jail de BSD.

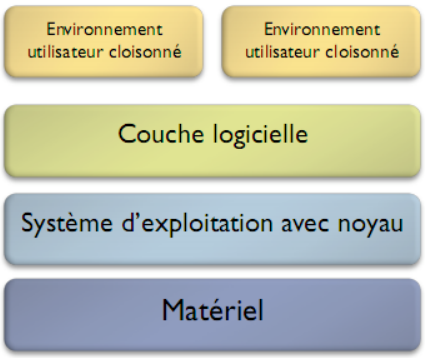

Figure 2 : Isolateur

B. La virtualisation totale

La virtualisation totale permet d'émuler totalement le matériel afin d'exécuter des systèmes d'exploitation entier qui n'ont absolument pas conscience qu'ils sont virtualisés. Ainsi le système invité croit qu'il est seul à s'exécuter sur une machine physique comme lors d'une installation classique.

Le système hôte exécute un ou plusieurs systèmes invités en fonction de ses ressources à travers un logiciel de virtualisation. Cette technique permet d'exécuter des systèmes d'exploitation indépendamment de la compatibilité ou non avec le matériel. Ainsi, quelque soit l'architecture ou le matériel, on pourra exécuter le système d'exploitation désiré.

Au niveau de l'architecture representé dans la figure 3, le système d'exploitation hôte gère l'accès au matériel. Un logiciel de virtualisation correspondant à la couche logicielle sur la figure, tel que VMware s'occupe de créer et gérer les systèmes invités. Pour chaque système invité, un couche de matériel émulé permet de faire croire au système qu'il se situe sur une machine physique.

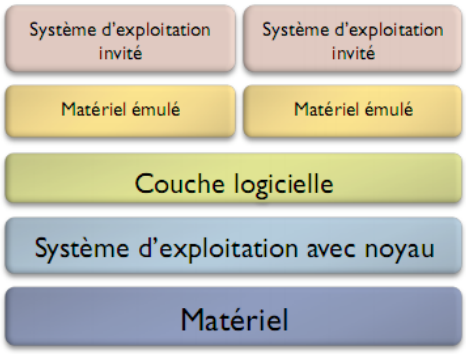

Figure 3 : Virtualisation totale

Cette technique est très coûteuse en ressources puisque qu'elle implémente une version complète du matériel pour chaque système invité. C'est là son inconvénient principal. En revanche, elle permet d'exécuter n'importe quel système d'exploitation quelque soit l'architecture matérielle supportée par celui-ci. Il existe beaucoup de logiciels de virtualisation totale. VMware est un des plus connu, mais on a aussi Qemu, VirtualBox et VirtualPC.

C. La virtualisation matérielle assistée

Ce type de virtualisation permet d'exécuter des systèmes d'exploitation différents mais en implémentant un support matériel plus performant qu'avec la virtualisation totale.

Le logiciel de virtualisation va permettre d'accéder à des fonctionnalités matérielles et en particulier du processeur afin de limiter au maximum les pertes de performance induites par les multiples couches. La virtualisation matérielle assistée est une évolution de la virtualisation totale comme on peut le voir dans la figure 4, cependant elle nécessite que le matériel dispose des fonctionnalités spécifiques qu'elle utilise. Cette technique a été implémentée dans les processeurs d'Intel (modèles Intel-VT) et AMD (modèle AMD-V).

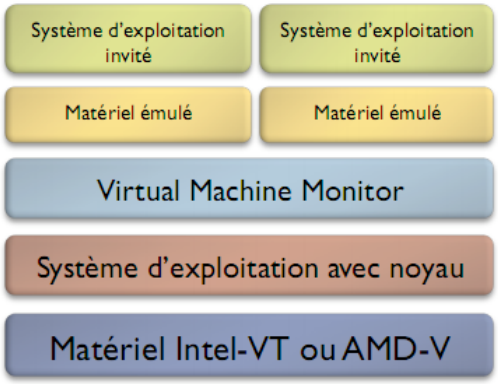

Figure 4 : Virtualisation matérielle assistée

D. La paravirtualisation

La paravirtualisation permet de substituer le système d'exploitation hôte par un hyperviseur. Il s'agit d'une couche logicielle qui va faire le lien entre les systèmes virtualisés et le matériel. Elle est représenté dans la figure 5. La différence la plus importante avec les autres techniques de virtualisation est que les systèmes invités ont conscience de leur état ainsi que de l'existence de l'hyperviseur. C'est pour cela que l'on parle de systèmes d'exploitation modifiés. En effet, les sources du système d'exploitation ont été modifiées afin de lui permettre d'avoir conscience de l'existence de l'hyperviseur et de communiquer avec lui. Cela permet de diminuer les besoins en ressources par rapport à la virtualisation totale en optimisant l'accès à ces mêmes ressources. Ils ont accès au matériel par l'intermédiaire de l'hyperviseur.

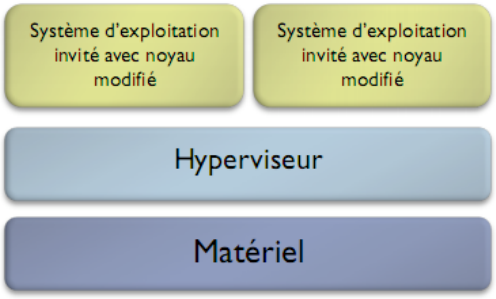

Figure 5 : Paravirtualisation

Xen est un outil de paravirtualisation ainsi que Vmware ESX.

L'hyperviseur est conçu pour qu'il soit le plus simple et léger possible afin d'économiser les ressources au maximum. Il est chargé de gérer l'accès au matériel. Les systèmes d'exploitation invités ont un noyau modifié qui leur permet de communiquer avec l'hyperviseur. Contrairement aux techniques précedentes, le système invité demande l'accès à une ressource directement à l'hyperviseur.

Avec cette technique, il est théoriquement impossible de contrôler l'hyperviseur, cependant comme il peut être nécessaire de le configurer, l'un des systèmes invités est privilégié. On parle donc de systèmes privilégié et de systèmes non privilégiés plutôt que de systèmes hôtes et invités. De plus, ces notions ne sont pas bonnes puisqu'il n'y a pas de systèmes hôtes comme dans les autres techniques.

1.2 – Les intérêts de la virtualisation

La virtualisation a été mise en œuvre pour répondre à différentes problématiques rencontrées par l'industrie de l'informatique. Des exemples de problématiques sont l'augmentation du coût de l'énergie électrique et le manque de place dans les centres de données à travers le monde. L'idée initiale était donc de mutualiser des équipements pour plusieurs services sans compromettre le principe d'isolation des services. Ceci a été rendu possible par l'augmentation de la puissance des processeurs génériques via la démultiplication des cœurs de calcul et la baisse des prix de la mémoire vive.

Dans le but de cadrer les études que nous avons fait nous avons souhaité détailler les avantages et les limites de la virtualisation. Nous allons tout d'abord énumérer et décrire rapidement les avantages de la virtualisation. Nous évoquerons les avantages en terme d'isolation, de coût et de criticité.

Tout d'abord, comme nous l'avons dit précédemment, la virtualisation a été mise en œuvre afin de mutualiser des équipements pour plusieurs services tout en respectant le principe d'isolation. Cette dernière est donc une condition nécessaire pour que la virtualisation soit viable pour une utilisation en entreprise. L'isolation peut se situer à plusieurs niveaux. Le premier type d'isolation est l'isolation au niveau de la communication entre les différents systèmes d'exploitation. Il s'agit donc d'éviter qu'un système d'exploitation n'accède à des

informations qui ne lui appartiennent pas. Cette isolation est gérée par l'hyperviseur au niveau de la mémoire vive et le processeur, et par le domaine 0 pour le reste des périphériques. Un autre type d'isolation est le cloisonnement au niveau de l'utilisation des ressources physiques. Il n'est pas envisageable qu'un système d'exploitation qui essaie de consommer toutes les ressources puisse saturer également les autres systèmes d'exploitation présents à ses cotés. Cette isolation est gérée par l'hyperviseur en ce qui concerne l'allocation des ressources en mémoire vive et en processeur. Un autre type d'isolation est l'isolation au niveau de l'accès au support réseau. Dans le cas d'un réseau classique, chaque équipement physique accède par le biais d'une interface à un équipement réseau de type commutateur ou routeur. Dans le cas de la virtualisation, ces interfaces sont virtuelles et aboutissent sur une même interface physique. Ceci peut être problématique dans l'intégration d'une plateforme de virtualisation à un réseau d'entreprise déjà existant.

Ensuite, comme nous l'avons vu précédemment, la principale motivation est de réduire les coûts d'infrastructure. Le coût est un gros avantage ce cette technique. Le coût initial de l'installation d'une plateforme basée sur la virtualisation est globalement plus faible que l'installation d'une plateforme sans virtualisation. Au lieu d'acheter un grand nombre d'équipements, il n'est plus nécessaire d'acheter qu'un petit nombre d'équipements plus puissants. Le coût de l'exploitation du système d'information sera également réduit au niveau de la consommation électrique car un équipement puissant consomme moins d'énergie rapporté au nombre de systèmes d'exploitation en exécution. Le coût de l'exploitation sera également réduit au niveau de l'espace à louer en centre de données. Les coûts de maintenance en production seront cependant plus élevés car il y a une couche supplémentaire à maintenir et à entretenir, à savoir la couche de virtualisation. Au final, les coûts de production de la plateforme basée sur la virtualisation seront inférieurs à une plateforme équivalente sans virtualisation.

Au final, la mutualisation des services sur un équipement augmente fortement la criticité des équipements en question. Ceci est lié au fait que l'indisponibilité d'un équipement rend indisponible tous les systèmes d'exploitation présents sur cette équipement. Ceci pourrait être un inconvénient majeur de la virtualisation or les divers éditeurs de logiciels ont su retourner cet inconvénient pour en faire un point fort. Tout d'abord, la virtualisation permet d'effectuer facilement des sauvegardes de l'intégralité d'un système. Dans le cas de sauvegardes classiques, il est possible de sauvegarder les fichiers de données, les fichiers de

configuration, les bases de données, ... Cependant, il est difficile de sauvegarder l'état complet de la machine sans retirer le disque dur physiquement et d'en faire une copie. La virtualisation permet de sauvegarder simplement les environnements logiciels et de les réexécuter par la suite réduisant ainsi fortement les délais d'intervention et les efforts de rétablissement.

Ensuite, il est possible de changer et donc de faire évoluer le matériel informatique sur lequel repose un système d'exploitation simplement. Il suffit de transférer l'environnement logiciel d'un équipement vers un autre sans avoir à se soucis de la compatibilité avec ces équipements. Finalement, il est possible de répondre à des surcharges périodiques ou exceptionnelles ou encore une panne matérielle en faisant transférer de manière automatique et quasi instantanée des systèmes d'exploitation d'un équipement vers un autre. Ceci permet de réduire les temps d'indisponibilité ou de ralentissement.

Finalement, la virtualisation est une solution est une technique qui permet de répondre aux évolutions des attentes de l'industrie informatique en déployant ces trois avantages.

2. XEN

2.1. L'outil Xen

Nous avons choisi d'utiliser l'outil Xen car c'est un outil *open source*, ce qui nous permet d'avoir accès à beaucoup d'information concernant son fonctionnement et son implémentation. Il s'agit aussi d'un outil puissant qui reste une référence en matière de virtualisation car le projet est soutenu par nombre de communautés et d'industriels tel que Novell, Cisco, Sun, AMD, ...

Xen a été le premier outil à implémenter la paravirtualisation même s'il est possible de faire de la virtualisation totale. Ce que l'on appelle Xen est en réalité l'hyperviseur. Il a pour but de traiter les instructions provenant des systèmes virtualisés à destination du matériel. Xen a pour but d'être minimaliste afin de limiter au maximum les risques d'erreur qui lui serait dû. L'hyperviseur reçoit des appels systèmes (les hypercalls) provenant des systèmes virtualisés lorsque ceux-ci ont besoin d'accéder au matériel.

Les systèmes d'exploitation virtualisés sont appelés domaines. Cela vient principalement du fait qu'il n'y a pas de système d'exploitation hôte comme c'est le cas avec beaucoup de solutions de virtualisation. En effet, l'hyperviseur prend la place du système d'exploitation d'origine lors du premier démarrage après son installation sur celui-ci. Ce système devient alors le domaine privilégié ou dom0. Pour cela, l'hyperviseur va démarrer en premier et lancer ensuite le dom0. Ce domaine tient une place particulière car il va permettre d'accéder directement à l'hyperviseur afin de configurer celui-ci au fil des besoins. Les autres machines virtuelles n'ont pas accès à l'hyperviseur.

L'hyperviseur va gérer l'accès aux ressources processeurs et mémoires tandis que les périphériques seront gérés par le dom0 avec ses propres pilotes. Il mettra à disposition des autres machines virtuelles, les périphériques au travers de pilotes génériques. Les machines virtuelles auront donc un accès indirect à ces périphériques. Ce principe est illustré dans la figure 6.

Le dom0 est aussi en charge du trafic réseau entre les machines virtuelles et avec l'extérieur. C'est lui qui redirige les paquets en fonction de sa configuration. Le dom0 a ainsi

accès à toutes les fonctionnalités critiques de la machine. Il est donc très important que celui-ci soit bien configuré et sécurisé.

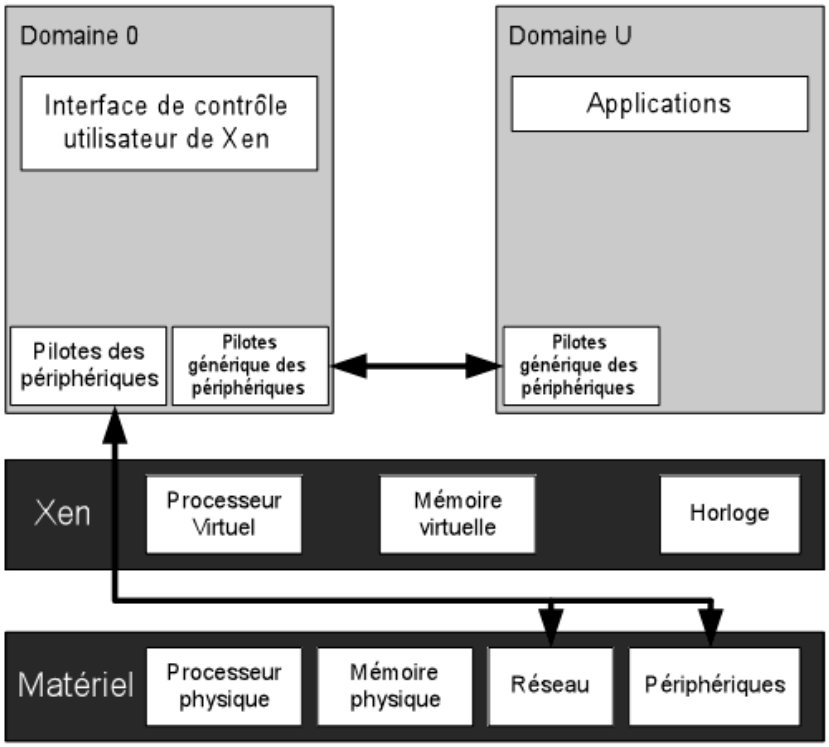

Figure 6 : Interaction entre l'hyperviseur, le dom0 et les domU

Les autres systèmes virtualisés sont les domaines non privilégiés appelés domU (Unprivileged). Ils sont lancés par le dom0 au démarrage ou bien par l'intermédiaire d'outils. Beaucoup de systèmes d'exploitation peuvent être exécutés en tant que domU grâce à la simplification des accès aux périphériques.

2.2. Installation de Xen

L'installation de Xen se réalise en ligne de commande. Nous avons choisi d'utiliser la version 3.3 de Xen qui est la version la plus aboutie actuellement. Nous avons aussi choisi d'utiliser la distribution Debian 5.0 Lenny qui implémente très bien le support de Xen et notamment la version 3.3 en tant que dom0. De plus, le noyau Xen existe dans les dépôts. Il devient alors très simple d'installer Xen sur ce système d'exploitation. Pour cela, il suffit d'installer les paquets nécessaires :

```
apt-get install xen-hypervisor-i386 linux-image-2.6-xen-686
            linux-headers-2.6-xen-686 xen-utils-3.2-1 xen-tools
            bridge-utils
```

Bien sûr, ces paquets supposent que le système d'exploitation a une architecture de type i368 (systèmes 32 bits).

Le redémarrage de la machine permet d'exécuter le noyau Xen nouvellement installé. Par défaut, le système démarre automatiquement sur le noyau Xen. Pour vérifier que celui-ci est bien installé et qu'il est toujours possible de démarrer sur le noyau debian, utiliser la commande suivante si votre boot-loader est grub.

```
Grub-update
```

2.3. Installation de machines virtuelles

L'installation des machines virtuelles est facilitée dans Xen par rapport à d'autres outils. En effet, il est possible à l'aide d'un script d'installer un système d'exploitation libre complet en tant que domU directement à partir des dépôts situés sur des miroirs sur Internet. Cette méthode fonctionne pour les systèmes d'exploitation Debian, Fedora et CentOS. Dans un soucis d'homogénéité et afin d'éviter les pertes de temps dues aux manipulations de ces différents systèmes d'exploitation, nous avons décidé d'installer des domU basé sur Debian.

L'installation et la gestion des machines virtuelles se fait majoritairement en ligne de commande en utilisant les outils de Xen soit le fichier xen-tools.conf et la commande xen-create-image.

Le fichier /etc/xen-tools/xen-tools.conf permet de configurer les propriétés de la future image (système d'exploitation, taille, mémoire, swap, …). Une fois la configuration du fichier terminée, l'éxécution de la commande suivante va créer l'image du domU.

```
xen-create-image --hostname nom_de_la_VM
```

II - Etudes

1. Introduction

Dans le cadre de cette TX, nous devions mettre en place deux études permettant d'approfondir deux aspects de la virtualisation qui pourraient potentiellement être adaptés en TP dans le cadre de l'UV RE12 enseignant les services réseaux. L'objectif de ces études est d'illustrer les intérêts de la virtualisation et de permettre une première prise en main de cette technique.

Le premier aspect que nous avons choisi d'étudier est le réseau. La virtualisation apporte des changements au niveau de l'implémentation du réseau du fait que plusieurs systèmes d'exploitation se partagent une même interface réseau. La première étude est donc placée sur cet axe. Les manipulations liées à cette étude sont également l'occasion de prendre en main les applications liées à Xen (Xen Manager et ConVirt) et les divers fichiers de configuration.

Le second aspect que nous avons choisi étudier est le cloisonnement. Parmi les différents types de cloisonnement que nous avons évoqué précédemment, nous avons choisi de travailler sur le cloisonnement au niveau des performances. Ce choix se justifie par le fait que nous souhaitions effectuer une étude quantitative des promesses faites par l'hyperviseur de virtualisation afin de pouvoir les confirmer.

2. Etude du réseau

La première manipulation a pour but d'étudier le fonctionnement du réseau à travers la virtualisation et comment une plateforme de virtualisation s’integre dans un réseau d’entreprise. Il existe plusieurs méthodes que nous allons expérimenter afin de comprendre comment elles peuvent être utilisées.

Nous voulions dans un premier temps permettre aux étudiants d'installer Xen ainsi qu'une machine virtuelle. Cependant, nous nous sommes aperçus que malgré quelques

améliorations, l'installation et la configuration des outils nécessaires à une bonne manipulation de Xen restent difficiles à installer et à configurer.

C'est pourquoi, nous avons finalement décidé de fournir une image déjà configurée avec tous les outils nécessaires au bon déroulement des manipulations. Ainsi, il ne sera pas nécessaire d'installer un quelconque logiciel. L'image sera opérationnelle en permanence et sera indépendante des mises à jour ou modifications inhérentes au développement des systèmes d'exploitation libres.

La séance de travaux pratiques se déroulera de la manière suivante : nous commencerons par manipuler l'interface Convirt afin de comprendre comment elle fonctionne. Ensuite, nous manipulerons les configurations réseau de Xen afin d'étudier les intérêts et les limites de chaque mode.

2.1. Étude et expérimentation des modes de configuration réseau

Xen a défini 3 modes standards en ce qui concerne la configuration du réseau. Ces 3 modes correspondent à des besoins différents qu'il est nécessaire de prendre en compte lorsque l'on défini le plan d'adressage d'un réseau d'entreprise.

Par défaut, à chaque fois qu'une machine virtuelle démarre, le dom0 lui attribue un identifiant de domaine. Ainsi, le premier domaine aura l'identifiant 1, le second l'identifiant 2 … Pour chaque nouveau domU, Xen crée une nouvelle paire d'interfaces réseau virtuelles. Ainsi, le domU numéro 1 aura une interface eth0 lié avec une interface virtuelle du dom0 nommée vif1.0.

Les cartes réseaux virtuelles des domU peuvent avoir des adresses mac. On peut soit les configurer soi même, ou bien le deamon xend sélectionne une adresse au hasard. Dans le cas où l'on choisi une adresse mac, il faut s'assurer qu'il s'agit bien d'une adresse unicast. Il est recommandé d'utiliser une adresse mac de la forme 00:16:3E:xx:xx:xx où x représente un chiffre hexadécimal. Ces adresses sont réservées à l'utilisation de Xen. Par défaut, lors de la création d'une image avec l'outil xen-create-image, une adresse mac appartenant à la plage d'adresse réservée à Xen est attribué de façon statique dans le fichier de configuration de la machine virtuelle.

Les 3 modes réseau de Xen sont le pont, le routage et le NAT. Les concepteurs de Xen ont choisi de gérer la mise en place de ces modes dans le dom0 et non dans les machines virtuelles comme c'est le cas avec de nombreux systèmes de virtualisation. Par exemple, dans VMWare, le réseau se configure au niveau de chaque machine virtuelle avant son démarrage. Il est ainsi possible d'avoir une machine en mode pont et une autre en mode NAT. Dans Xen, le mode est choisi dans la configuration du dom0 ce qui implique que toutes les machines virtuelles sont soit en mode pont, soit en mode routage ou soit en mode NAT.

A. Les ponts

a) Principe

Un pont est un équipement matériel qui permet d'interconnecter 2 réseaux physiques. Le pont se situe au niveau 2 du modèle OSI. Cela veut dire que le transfert des trames est effectué suivant l'adresse Ethernet (adresse MAC).

Dans le cas de Xen, il s'agit d'un pont logiciel. Cette stratégie est celle par défaut déployé par Xen. Le pont de Xen permet ainsi de connecter les cartes réseaux virtuelles des domU sur le même segment que la machine hôte. De cette façon, tous les domaines ayant une carte réseau apparaissent sur le réseau comme des machines individuelles. Elles peuvent ainsi avoir une adresse IP du réseau auquel appartient le dom0. Les machines virtuelles peuvent alors voir les machines autre que le dom0 et inversement.

La transmission des informations du dom0 au domU destinataire s'effectue en fonction de l'adresse mac et non de l'adresse IP. Le dom0 transmet donc les trames Ethernet au domU.

Figure 7 : Illustration du mode pont

b) Expérimentations

Le mode pont est le mode par défaut. Il est déjà configuré dans le dom0 et dans les machines virtuelles tel quel avec le DHCP. Dans nos expérimentations, nous avons créer un bridge que nous avons nommé xenbr0.

Durant les manipulations, il sera nécessaire d'utiliser l'interface Convirt. Cela permettra de manipuler facilement les machines virtuelles. Nous utiliserons principalement les images linux1 et linux2. A partir de Convirt, nous pouvons les démarrer ainsi qu'accéder à la console pour manipuler ces deux images. La visualisation de leur configuration réseau via la commande *ifconfig* et le fichier /etc/network/interfaces permet de se rendre compte que les deux machines fonctionnent en mode DHCP et possèdent bien une adresse ip appartenant au réseau local. Il est également possible de visualiser l'adresse mac des interfaces et de vérifier leur appartenance à la plage d'adresses réservée à Xen soit 00:16:3E:xx:xx:xx. Ces adresses MAC sont visibles sur le réseau par les autres machines, il est donc possible de savoir que ces machines sont des machines virtuelles.

Figure 8 : Application Convirt sur la plateforme d'expérimentation

La manipulation de la commande *ping* va permettre de vérifier l'accessibilité des machines entre elles mais aussi avec les autres machines du réseau local et même internet. D'autres commandes peuvent être utilisées afin de visualiser la configuration du réseau dans les domU ainsi que dans le dom0 notamment les commandes comme *traceroute, route* ...

Lorsque que l'on exécute la commande *ifconfig* sur le dom0, on obtient le résultat suivant :

Figure 9 : Résultat de la commande ifconfig sur le dom0

Le résultat de la commande nous permet de visualiser l'interface eth1 qui est l'interface réseau réelle, l'interface de loopback du dom0, le bridge créé pour le dom0 nommé xenbr0 et les deux bridges qui permettent aux machines virtuelles linux1 et linux2 d'accéder au réseau respectivement vif1.0 et vif2.0. Ces deux interfaces servent uniquement de lien avec les interfaces eth0 des machines virtuelles. C'est pourquoi elles n'ont ni adresses IP ni adresses MAC réelles.

Ainsi, le mode pont permet de connecter des machines virtuelles à un réseau local de la même façon que s'il s'agissait de machines physiques. Pour les autres machines du réseau, elles apparaissent au niveau 2 et au niveau 3 du modèle OSI comme des entités matérielles distinctes.

B. Routage

a) Principe

Le routage est un mécanisme permettant de sélectionner le bon chemin afin d'acheminer les données d'un expéditeur jusqu'à un ou plusieurs destinataires. Le routage est un mécanisme de niveau 3, soit la couche réseau du modèle OSI.

Le routage dans Xen permet de créer un lien point à point entre le dom0 et chacun des domU. Chaque route vers un domU est ajoutée à la table de routage du dom0. Le domU doit avoir une adresse IP statique afin que le routage des communications puissent fonctionner dans le dom0.

Lors du lancement d'une machine virtuelle, l'adresse IP de son interface virtuelle eth0 est copiée sur l'interface vif<id>.0 et la route statique vers cette interface est créée.

Figure 10 : Illustration du mode routage

b) Expérimentations

Ce mode nous a posé beaucoup de problème. Il est apparemment très peu utilisé et donc moins bien maintenu que les deux autres modes. Nous n'avons pas réussi à le mettre en œuvre. Les scripts de configuration génèrent plusieurs erreurs dont certaines ont pu être corrigées. Cependant, nous n'avons pas réussi à démarrer les machines virtuelles ce qui limite grandement l'expérimentation de ce mode.

De manière plus générale, ce mode permet de regrouper les machines virtuelles dans un même réseau local sans cacher leurs adresses IP au réseau extérieur.

C. NAT

a) Principe

Le NAT (Network Address Translation) est un mécanisme de niveau 3. Il permet de faire correspondre une adresse interne d'un intranet à une adresse unique externe. Cela permet notamment de faire correspondre les adresses IP d'un réseau privé d'une entreprise à l'adresse publique externe fournit par le FAI de cette même entreprise. Le NAT permet de pallier au nombre limité d'adresses IP publiques.

Le dom0 joue le rôle de passerelle pour les domU. Ainsi, il n'y a aucune visibilité du réseau que forment le dom0 et les domU depuis le réseau externe. Le dom0 crée un réseau interne aux machines virtuelles. Il est possible de configurer le dom0 pour qu'il joue le rôle de serveur DHCP pour les domU.

Figure 11 : Illustration du mode NAT

b) Expérimentations

Afin de changer le mode de configuration réseau, il faut modifier le fichier de configuration du serveur Xen. Pour cela, il faut éditer le fichier /etc/xen/xend-config.sxp et commenter les lignes suivantes :

```
(network-script network-dummy)
(vif-script vif-bridge)
```

Ensuite, il faut décommenter les lignes suivantes :

```
(network-script network-nat)
(vif-script vif-nat)
```

Pour terminer, le redémarrage du serveur Xen par l'exécution de la commande /etc/init.d/xend restart permet de lancer la nouvelle configuration.

Il est aussi nécessaire de reconfigurer les interfaces réseaux des machines virtuelles. Celle-ci se déroulent en 2 étapes. Il faut tout d'abord éditer le fichier de configuration de la machine virtuelle, par exemple /etc/xen/linux1.cfg et remplacer les lignes suivantes

```
dhcp='dhcp'
vif=['mac=XX:XX:XX:XX:XX:XX']
```

par celles-ci

```
#dhcp='dhcp'
vif=['ip=192.168.0.1,mac=XX:XX:XX:XX:XX:XX']
```

La seconde étape consiste à démarrer la machine virtuelle linux1 et à éditer le fichier /etc/network/interfaces afin de configurer statiquement l'adresse IP.

```
auto eth0
iface eth0 inet static
      address 192.168.0.1
      netmask 255.255.255.0
      gateway 192.168.0.128
```

L'adresse de la passerelle est l'adresse IP que Xen donne à l'interface vifX.0 dans le dom0. La commande ifconfig permet de la visualiser.

Vous pouvez faire de même avec la machine linux2 en prenant soin de choisir une autre adresse IP dans le même réseau.

A la fin de ces étapes, le réseau interne c'est à dire le réseau entre les machines virtuelles est fonctionnel. Il est possible de tester la connectivité entre les machines virtuelles via la commande ping.

En visualisant la table de routage du dom0, on se rend compte que pour atteindre la machine d'adresse IP 192.168.0.1, il faut passer par l'interface vif1.0. Cette interface correspond à l'interface eth0 sur la machine virtuelle linux1.

```
tpxen:~#route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.1     *               255.255.255.255 UH    0      0        0 vif1.0
192.168.0.2     *               255.255.255.255 UH    0      0        0 vif2.0
10.23.0.7       *               255.255.255.0   U     0      0        0 eth1
default         10.23.0.254     0.0.0.0         UG    0      0        0 eth1
```

Afin que les machines virtuelles aient accès à d'autres réseaux externes ainsi qu'à l'Internet, il faut configurer les règles de translation de la même manière que sur une passerelle. Le dom0 joue en effet ce rôle pour les machines virtuelles. Sur un système Debian, cette fonctionnalité est gérée par l'outil iptables. Cet outil permet de définir des règles de filtrage sur les ports de la machine mais aussi des règles de translation.

Elles sont les suivantes :

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

```
iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE
```

La première commande permet d'activer la translation au niveau du système d'exploitation. La seconde est la règle de translation que les paquets IP passant par l'interface eth1 vont suivre.

Lorsque l'on visualise les tables de filtrage avec la commande iptables -L, on obtient le résultat suivant.

```
tpxen:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
ACCEPT     0    --  10.0.0.1             anywhere            PHYSDEV match
--physdev-in vif1.0
ACCEPT     udp  --  anywhere             anywhere            PHYSDEV match
--physdev-in vif1.0 udp spt:bootpc dpt:bootps

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

Les machines virtuelles ont accès au réseau extérieur, par contre il n'est pas possible depuis l'extérieur d'accéder directement à une machine en particulier. Dans ce cas, on a installé de la translation dynamique permettant aux machines virtuelles d'accéder au réseau externe. Pour permettre l'accès en sens inverse, il faut mettre en place de la translation statique.

Le mode NAT permet de cacher les machines virtuelles derrière une seule adresse exactement comme le ferait un routeur. Toutes les machines sont dans un même réseau privé mais elles sont vu par l'extérieur comme une seule et même entité.

2.2. Utilisation des VLANs

L'utilisation des VLANs permet de segmenter un réseau au niveau 2 du modèle OSI indépendamment de la répartition géographique. Ainsi des machines situées dans différents lieux géographiques peuvent être regroupées dans le même segment.

Au niveau des machines virtuelles, ce principe peut être appliqué non pas pour regrouper des machines dans le même segment mais pour séparer logiquement les machines virtuelles en cours d'exécution au niveau du réseau. En effet, sans cette technique, les machines virtuelles sont condamnées à appartenir au même segment.

Les VLANs sont une configuration non standard du réseau dans Xen. Une des possibilités pour utiliser les VLANs repose sur le principe des ponts c'est à dire le mode par défaut de Xen. Nous étudions les VLANs dans cette configuration car c'est la seule configuration qui permet d'avoir une incidence sur le réseau extérieur. En effet, il nous paraît moins pertinent de manipuler des VLANs uniquement dans le réseau virtuel que forment le dom0 et les domU.

Pour cela, il faut utiliser des sous interfaces configuré en VLAN c'est à dire eth0.2, eth0.3 comme le montre la figure 11. Un pont par interface eth0.x est crée, vlanbr2 et vlanbr3 sur le schéma. Ces ponts représentent chacun un lien vers un domU. De cette façon, chaque domU est correctement isolé sur un réseau virtuel. Les domU sont aussi isolés du dom0 puisque le traffic est directement dirigé vers la sous interface correspondant au domU destinataire de la trame

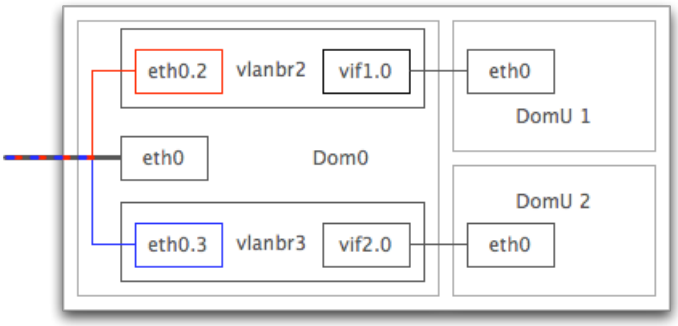

Figure 12 : Implémentation des VLANs dans Xen

L'utilisation des VLANs requiert le support du 802.1Q dans la carte réseau. Il faut donc activer ce mode dans le système d'exploitation et aussi dans le commutateur sur lequel est branché physiquement la machine.

Cette configuration va permettre d'offrir une flexibilité dans la conception de l'architecture du réseau. En effet, l'architecture logique peut ainsi être complétement différente de l'architecture physique. Avec cette technique, l'architecte réseau s'affranchit dans une certaine mesure des contraintes physiques et matérielles. Dans la figure 13, on peut voir que l'une des machines virtuelles se situe dans une DMZ et l'autre dans un réseau regroupant des serveurs internes à une entreprise.

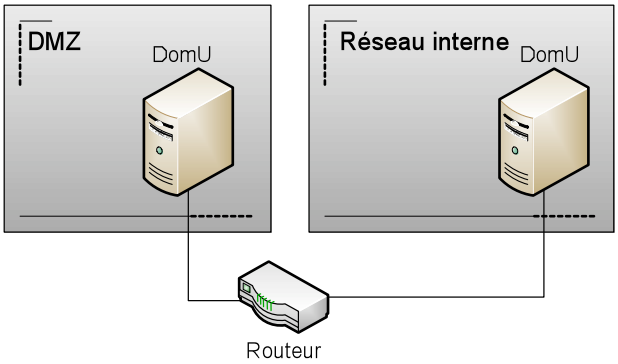

Figure 13 : Topologie logique possible

2.3. Bilan

Xen offre la possibilité de choisir différentes stratégies d'adressages au niveau du réseau. Le mode pont permet de rendre visible les machines virtuelles dans un réseau local. Le mode NAT permet au contraire de les cacher. Il est aussi possible, comme nous avons pu le voir avec les VLANs de manipuler Xen afin de mettre en place des configurations plus complexe de la même façon que dans un réseau classique. Malgré le fait que la manipulation de Xen nécessite une certaine prise en main, cet outil permet de mettre en place des stratégies réseau que l'on retrouve dans un réseau non virtualisé.

Enfin, la mise en place d'une plateforme de virtualisation dans un réseau d'entreprise ne va pas obliger l'administrateur à adapter la topologie de son réseau. Les outils que Xen a mis à disposition sont des outils de base permettant de mettre en place des outils plus complexe comme les VLANs afin d'adapter la plateforme de virtualisation à la topologie logique d'un réseau d'entreprise. Les contraintes physiques dues au fait que toutes les machines virtuelles se situent sur une même machine physique et donc en un seul lieu d'un point de vue physique, peuvent être contournées afin de profiter pleinement de la mutualisation sans en supporter les désagrements d'un point de vue architecture du réseau d'entreprise.

3. Etude du cloisonnement

Tout d'abord, la seconde manipulation s'appuiera sur les compétences acquises lors de la première manipulation au niveau de l'utilisation de l'hyperviseur Xen et des outils associés. Il sera donc supposé que la personne effectuant la manipulation sait utiliser les outils précédents.

Le but de cette seconde manipulation a été de démontrer le fonctionnement de la virtualisation au niveau du cloisonnement et ses avantages par rapport à une plateforme classique, non virtualisée. Le but était de montrer concrètement ce que la virtualisation pourrait apporter en termes de cloisonnement et de vérifier les promesses faites par le produit de virtualisation que nous avons choisi d'étudier.

Pour pouvoir mettre en œuvre l'impact du cloisonnement de performances, il a été nécessaire de trouver un moyen efficace de faire monter en charge les différents domaines tout en gardant la possibilité de pouvoir faire des tests. La montée en charge sera orientée sur un service réseau afin de pouvoir facilement mesurer un temps de réponse. Un service réseau relativement simple à faire monter en charge et à configurer est le service HTTP. Nous travaillerons donc avec le serveur HTTP Apache. Pour pouvoir mettre en œuvre une forte montée en charge, nous avons choisi d'utiliser d'utiliser une suite d'outils nommés "Isolation Benchmark Suite" conçus par des chercheurs de l'Université de Clarkson pour mettre à l'épreuve l'isolation de performances dans les solutions de virtualisation.

Nous avons ensuite établi un protocole expérimental permettant de mesurer l'impact d'une montée en charge d'un service sur un autre service. Ce protocole expérimental a consisté en l'évaluation d'un certain nombre de métriques qui permettront par la suite de comparer l'impact des différents tests. Au final, nous avons évalué l'impact des différentes manipulations sur une plateforme non virtualisée puis sur une plateforme virtualisée afin de pouvoir comparer les deux et de conclure sur l'intérêt de la virtualisation dans ce domaine.

Une métrique intéressante est le temps de réponse d'un service à une connexion TCP initiale. Cette métrique permet de mettre en avant la réactivité d'une application à une sollicitation. Par exemple, si un système d'exploitation a atteint son nombre de sessions TCP maximale ou alors que le serveur web a atteint son nombre de connexions maximales, le temps de réponse devrait fortement augmenter. Une autre métrique pourrait être le temps de

transfert d'un fichier volumineux. Cette métrique permet de mettre en avant le débit réseau disponible pour le service réseau en question.

3.1. Mise en place de l'architecture virtualisée

Lors de cette étude, nous avons utilisé une configuration réseau par défaut, à savoir le pont, car cette dernière est suffisante pour ce que nous souhaitons faire. Les outils graphiques de gestion de l'architecture tels que Convirt seront utilisés pour simplifier la prise en main de l'outil. De plus, les ordinateurs seront utilisés en binôme disposé systématiquement de la manière suivante : un nœud de calcul et un nœud d'exécution des tests. Les figures 14 et 15 montrent l'architecture mise en place pour cette étude.

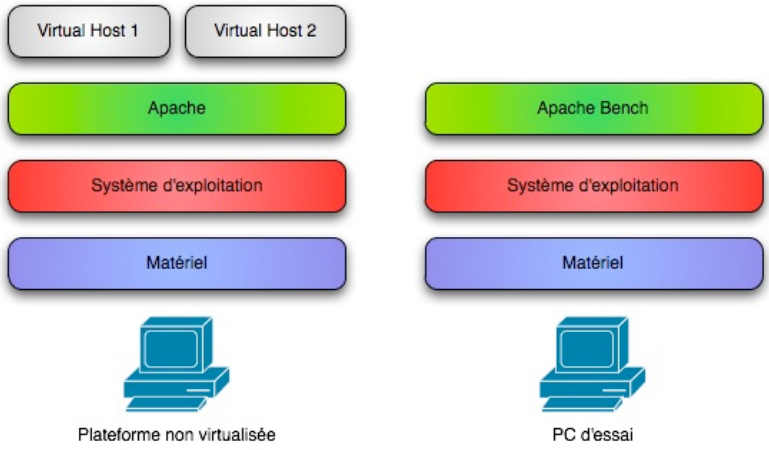

Figure 14 : Plateforme non virtualisée

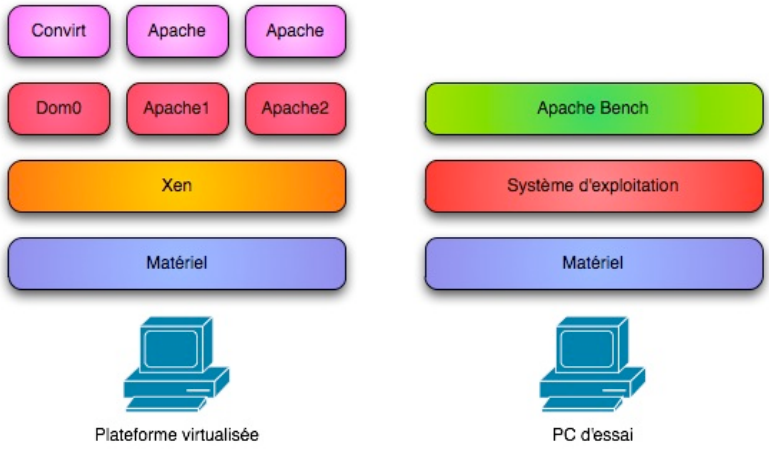

Figure 15 : Plateforme virtualisée

3.2. Mise en place des machines virtuelles

Ensuite, il va être nécessaire d'exécuter les machines virtuelles préconfigurées qui seront présentes avec l'image utilisée pour le TP. Nous avons choisi de fournir des machines virtuelles préconfigurées avant d'éviter de perdre du temps sur la prise en main et la configuration du serveur Apache. Les machines virtuelles seront donc intégralement préconfigurées à l'exception de la pile réseau car une configuration unique ne pourra pas convenir à toutes les manipulations.

Les machines virtuelles devront pouvoir répondre selon les critères que nous définirons ultérieurement de la manipulation que nous allons effectuer avec. Par exemple, la manipulation ne donnerait aucun résultat si la configuration Apache par défaut était utilisée car cette dernière ne permet l'utilisation que de très peu de threads et de mémoire vive. Ceci résulte donc à un faible potentiel de montée en charge et donc de saturation potentiel des ressources logicielles bien avant la saturation des ressources physiques. Au niveau de la configuration réseau, la configuration la plus simple sera mise en œuvre à savoir le pontage de toutes les interfaces. Ce choix va permettre de mettre en place les manipulations le plus rapidement possible en profitant du serveur DHCP déjà présent sur le réseau et d'éliminer un certain nombre d'inconnues potentielles.

Nous avons donc préparé et déployé les machines virtuelles suivantes :

- Apache-1 et Apache-2 : distribution standard avec un Apache dotée d'une configuration conçue pour une forte montée en charge.
- Linux-1 et Linux-2 : distribution standard dotée de l'utilitaire IBS (*Isolation Benchmark Suite*) et des utilitaires nécessaires à sa compilation.

Au final, l'architecture que nous utilisons pour ce TP se résumera au schéma suivant.

Figure 16 : Architecture complète de la plateforme de virtualisation

3.3. Protocole expérimental

Tout d'abord, il nous est nécessaire de présenter l'outil de mesure que nous allons utiliser et qui servira également à mettre en œuvre une montée en charge. Ensuite, nous allons présenter les différentes phases de manipulation. La phase préliminaire des expérimentations aura pour objectif d'obtenir les valeurs de référence sur une plateforme sans virtualisation avec et sans montée en charge. La première phase aura pour objectif de fournir des données de références sur une plateforme dotée de la virtualisation. La seconde phase aura pour objectif d'implémenter une première montée en charge modérée. La troisième phase aura pour objectif d'implémenter une très forte montée en charge. Les données de la seconde et de la troisième phase seront ensuite comparées aux données de la première phase. Cette comparaison nous permettra de tirer des conclusions sur l'isolation au niveau des performances dans un environnement de virtualisation.

3.3.1. Outils utilisés

L'outil de mesure et de montée en charge que nous avons utilisé est Apache Bench plus couramment appelé "ab". Cet outil permet de générer un nombre défini de requêtes HTTP en un nombre de threads définie sur une page ou un script. A la fin de son exécution, cet outil va afficher le temps de réponse du serveur HTTP interrogé ainsi que le taux de

transfert de données. Il fournit également une rapide analyse statistique sur les données qu'il a récolté. Nous pouvons donc faire usage de cet outil de deux façons, à savoir, la montée en charge du serveur HTTP et la mesure du temps des métriques. L'outil ping pourra également être utilisé pour mesurer la responsivité des machines virtuelles à de simples requêtes ICMP provenant du réseau. Cet outil donne nativement des données intéressantes telles que le taux de perte de paquets et le temps de réponse. L'inconvénient de l'utilitaire ping est la simplicité des requêtes qu'il fournit et donc le peu de temps de calcul nécessaire pour les traiter. Un impact sur la réponse aux requêtes ping indiquera donc que la machine est très largement surchargée. Cependant une absence d'impact n'indiquera pas forcément que la machine n'est pas surchargée.

3.3.2. Phase d'expérimentation sur la plateforme non virtualisée

La phase préliminaire a pour objectif de fournir des données de référence qui vont permettre de montrer l'intérêt de la virtualisation par rapport à une plateforme classique. Ces manipulations se feront sur le domaine 0. Il est préférable de faire ces manipulations en démarrant l'image sur le noyau sans Xen pour que la manipulation soit plus claire. Cette possibilité s'offre dans le menu *Grub* au démarrage. Les mesures dans le cadre de cette phase vont se faire sur une même instance d'Apache car il n'y a aucun intérêt à exécuter deux instances de ce dernier sur une même machine. La distinction entre différents sites ou différentes adresses IP se fait au niveau des *VirtualHosts*. De plus, cette phase préliminaire permettra de se familiariser avec l'outil de mesure. Le serveur HTTP contenu dans le domaine Apache-1 sera donc soumis à un nombre de requêtes HTTP générées par Apache Bench exécutées à partir d'un autre poste. Par exemple, il est envisageable de faire 50 requêtes HTTP à travers 5 threads. Il n'est pas souhaitable de générer les requêtes HTTP sur le poste exécutant le serveur HTTP. Ceci s'explique par le fait qu'une surcharge du serveur HTTP pourrait ralentir l'outil de mesure et ainsi fausser les mesures. Le test pourra être refait plusieurs fois pour minimiser l'écart type de la mesure qui sera retenue et ainsi éliminer de potentielles mesures aberrantes. Le poste distant pourra également obtenir une référence du temps de réponse du serveur HTTP secondaire en dehors la sollicitation et pendant la sollicitation. Nous allons distinguer trois manipulations dans le cadre cette phase préliminaire. La première phase consistera à obtenir des données de référence sans aucune montée en charge. La seconde manipulation consistera à obtenir des données dans le cas d'une montée en

charge un des deux sites hébergés par Apache. La dernière manipulation aura pour objectif de pousser plus loin les expérimentations effectuées dans la phase précédente en implémentant l'IBS.

Dans un premier temps, il va être nécessaire d'obtenir des valeurs de référence sans montée en charge particulière. Pour obtenir ces dernières, il est nécessaire d'appliquer une charge assez faible sur le serveur Apache tout en prenant un nombre suffisant de valeurs pour pouvoir détecter facilement les valeurs aberrantes. Nous avons donc choisi d'exécuter l'outil Apache Bench en envoyant 50 requêtes avec une concurrence de 5. Ensuite, l'objectif suivant était de quantifier la réactivité de la plateforme lorsqu'un site était fortement chargé et son impact sur un autre site ne subissant pas de charge particulière. Nous avons donc exécuté une première instance d'Apache Bench ayant pour but de faire monter la charge. Cette dernière enverra donc 500.000 requêtes avec une concurrence de 1000. Pour pouvoir tester l'impact de cette montée en charge, nous avons ensuite exécuté Apache Bench avec les mêmes paramètres que lors de la première manipulation. Nous avons ensuite implémenté une forte charge extérieure au service HTTP par le biais de l'application IBS. Il suffit d'exécuter une instance du test IO pour avoir des résultats significatifs sans perdre le contrôle sur la machine. L'exécution de plusieurs tests en même temps rend la machine inutilisable sans un redémarrage forcé. Ce dernier test permet de simuler une forte montée en charge qui, dans un contexte de production, aurait pu être causé par la compromission du système d'exploitation ou d'une utilisation abusive. Une fois toutes ces manipulations effectuées, il sera nécessaire de démarrer la machine de test sur la plateforme de virtualisation si cela n'a pas été effectué. Nous passons donc ensuite aux manipulations permettant d'étudier l'isolation de performance dans le cas d'une plateforme virtualisée.

3.3.3. Phase d'expérimentation sur la plateforme virtualisée

La première phase de manipulation ayant pour objectif de fournir des données de référence, il n'est pas souhaitable de soumettre la machine à une charge particulièrement forte. Pour permettre une comparaison fiable avec les résultats obtenus sur la plateforme sans virtualisation, nous utiliserons les mêmes paramètres d'exécution d'Apache Bench, à savoir 50 requêtes avec une concurrence de 5.

La seconde phase de manipulation permettra de faire une première comparaison avec les données de références. L'objectif est donc de charger le serveur HTTP secondaire le plus

fortement possible et de voir les conséquences que cela peut avoir sur le serveur HTTP primaire et sur la connectivité réseau. Pour charger le serveur HTTP secondaire le plus fortement possible, l'exécution de l'outil Apache Bench devra s'effectuer sur un poste autre que celui qui exécute l'environnement de virtualisation. Des paramètres d'exécution d'Apache Bench adaptés sont 500.000 de requêtes HTTP à travers 500 threads. La page qui sera ainsi demandée sera un script PHP effectuant des calculs de sorte à contribuer à la surcharge de la machine. Il est possible de modifier ces paramètres de test pour voir une évolution de la latence en fonction du la charge appliquée. La mesure des métriques se fera toujours sur le serveur HTTP primaire par le biais d'une exécution parallèle d'Apache Bench avec les mêmes paramètres que lors de la première phase. Le temps d'exécution de la première instance d'Apache Bench va permettre d'effectuer plusieurs mesures sur le serveur HTTP primaire et ainsi pouvoir lisser d'éventuels résultats aberrants.

Figure 17 : Architecture de virtualisation avec montée en charge du domaine Apache-2

La troisième et dernière phase de manipulation va permettre d'obtenir des valeurs extrémums par rapport à l'architecture mise en place. L'objectif est d'éprouver l'architecture afin de pouvoir ensuite tirer quelques conclusions sur l'utilisation de la virtualisation. Le but est de simuler une situation extrême de production qui correspondrait à une utilisation abusive de la machine par un utilisateur ou alors à la compromission d'un domaine. Pour implémenter cette montée en charge extrême, nous avons choisi d'exécuter plusieurs instances de l'IBS. Ce

procédé permet de surcharger plusieurs processeurs à la fois et de remplir fortement la mémoire. Un script sera fourni pour l'exécution des différentes instances afin de supprimer un niveau de complexité indésirable. Ce script sera exécuté dans les deux domaines Linux-1 et Linux-2. Lorsque les instances concurrentes auront été exécutées, il faudra ensuite exécuter les tests de référence sur le serveur HTTP contenu dans Apache-1. Il sera possible d'exécuter ces tests plusieurs fois afin d'éviter certaines valeurs aberrantes.

Figure 18 : Architecture de virtualisation avec montée en charge des domaines Linux1 et Linux2

3.4. Résultats expérimentaux

Afin d'illustrer le protocole expérimental, nous avons effectué divers relevés de performance en appliquant ce dernier. Les valeurs aberrantes ont été éliminées et ne seront pas présentées ci-dessous. Nous avons relevé très peu de valeurs aberrantes lors des manipulations et le peu que nous avons obtenu étaient liées à une utilisation de la machine hôte.

Nous avons également du établir différents seuils pour pouvoir tirer des conclusions des chiffres que nous avons obtenu. Nous avons considéré un temps de réponse de 100 ms comme un temps de réponse anormalement élevé et 1000 ms comme un temps de chargement inacceptable. Toutes les mesures ci-dessous ont pour unité la seconde. Nous avons découpé ces résultats expérimentaux en deux parties : l'impact de la montée en charge sur la plateforme non virtualisée et l'impact de la montée en charge sur la plateforme virtualisée.

3.4.1. Impact de la montée en charge sur la plateforme non virtualisée

Voici les résultats que nous avons obtenus lors des expérimentations sans virtualisation. Ces mesures ont servi de référence pour la suite. Nous avons envoyé 50 requêtes HTTP avec une concurrence de 5.

Figure 19 : Temps de réponse sans montée en charge

Nous voyons immédiatement que les temps de réponses restent tout à fait acceptables, à savoir en deca de 5 ms. Ces résultats sont identiques lors d'un essai sur un serveur Apache sans virtualisation et lors d'un essai sur un serveur Apache avec virtualisation lorsque tous les autres domaines n'effectuent aucune tache spécifique.

Voici les résultats que nous avons obtenu lors des expérimentations lorsque nous avons chargé le second "site" présent sur la machine avec de nombreuses requêtes Apache, à savoir, 500.000 avec une concurrence de 1000.

Figure 20 : Montée en charge d'un site

Nous voyons très rapidement que l'ordre de grandeur des mesures a radicalement changé. Nous avons intentionellement supprimé la valeur maximale du site "Client2" pour une meilleur lisibilité. A titre d'information, elle était au delà de 72 secondes. La moyenne du temps de réponse du site "Client1" sont largement au dessus de 100 ms est oscillent entre 300 et 400 ms. Nous pouvons donc conclure que l'impact d'un site sur un autre dans un même environnement logiciel est très important.

Voici les résultats que nous avons obtenu lorsque nous avons exécuté les tests de montée en charge des disques dur de l'IBS sur une machine, toujours sans virtualisation. Nous avons en même temps mesuré le temps de réponse du serveur Apache.

Figure 21: Montée en charge du système avec l'IBS

Nous remarquons, comme lors des précédents tests, que l'impact sur les autres sites est important. Ceci se remarque particulièrement sur les temps de réponse maximaux. La moyenne des temps de réponse est largement supérieure à 100 ms dans le premier cas (272 ms) et s'en approche dans les deux autres cas (81 et 71 ms). Nous pouvons conclure que l'impact d'un autre programme peut également affecter fortement un service en dégradant largement sa qualité.

3.4.2. Impact de la montée en charge sur la plateforme virtualisée

Nous sommes ensuite passés aux tests de performance avec virtualisation. Dans le cas de la prise des mesures de référence, nous avons remarqué que ces résultats étaient identiques aux mesures de référence sans virtualisation. Nous avons ensuite effectué des tests avec une montée en charge graduelle du second serveur Apache. Nous avons tout d'abord envoyé 100.000 requêtes avec une concurrence de 100, puis 500.000 requêtes avec une concurrence de 500 et finalement, 500.000 requêtes avec une concurrence de 1000. Les résultats que nous avons obtenus sont présentés dans les trois histogrammes suivants.

Figure 22 : Montée en charge d'un serveur HTTP

Figure 23 : Montée en charge d'un serveur HTTP

Figure 23 : Montée en charge d'un serveur HTTP

Tout d'abord, nous avons intentionnellement omis les valeurs des maximales dans ces graphiques car les valeurs dépassent largement les 5 secondes ce qui rendait le graphique illisible. Ensuite, à travers ces trois histogrammes, nous pouvons remarquer rapidement que malgré la forte dégradation des performances dans le domaine Apache2, le domaine Apache1 continue à répondre correctement. Si nous regardons les chiffres de plus près, nous pouvons nous rendre compte que les performances de Apache1 ne sont pas totalement inaffectées mais leur augmentation est de l'ordre de 7 ms au maximum. Cette dégradation reste relativement anecdotique par rapport aux dégradations constatées sans virtualisation.

Pour mettre plus fortement à l'épreuve la plateforme de virtualisation, nous avons ensuite utilisé les différents tests de l'IBS. Dans un premier temps, nous avons exécuté une instance du script "CpuBomb", dans un second temps, nous avons exécuté quatre instances du script "CpuBomb" et dans un troisième temps, nous avons exécuté simultanément quatre instances du script "CpuBomb" et une instance du script "IO". Tous ces tests ont été exécutés simultanément dans les deux domaines Linux1 et Linux2. Les résultats que nous avons observés sont présentés dans les histogrammes ci-dessous.

Figure 24 : Montée en charge de deux domaines avec l'IBS

Figure 25 : Montée en charge de deux domaines avec l'IBS

Figure 26 : Montée en charge de deux domaines avec l'IBS

Les mesures sont absentes des domaines Linux1 et Linux2 car ils n'exécutent pas d'instance d'Apache. La présence de ces deux domaines était intéressante dans la mesure où cela nous permet de différencier les trois histogrammes ci-dessus. Dans le premier histogramme, nous remarquons que l'impact des tests est inexistant dans la mesure où les valeurs sont quasiment identiques aux valeurs de référence. Dans le second histogramme, nous remarquons que l'impact est bien présent. Le premier test montre des temps de réponse qui vont au delà de 30 ms ce qui reste acceptable. Les deux autres tests montrent des temps de réponse en deca de 15 ms. L'impact est donc très limité et les temps de réponse d'Apache1 restent largement acceptables. Dans le troisième histogramme, les temps de réponse restent largement acceptables malgré la très forte sollicitation des ressources physiques dans les deux autres domaines.

3.4.3. Conclusion

Après observation de ces valeurs, nous pouvons tirer certaines conclusions quant aux apports de la virtualisation en terme d'isolation en terme de performance. Ces différents tests nous ont montré la faiblesse d'un système d'exploitation unique et mutualisé en terme d'isolation de performance. Ils ont cependant mis en avant la dextérité avec laquelle la virtualisation réussit à gérer une très forte demande en termes de performances tout en assurant une certaine qualité de service en termes de temps de réponse. Nous aurions pu étendre ces tests à d'autres solutions de virtualisation afin de les comparer mais le temps a manqué. Au final, nous pouvons conclure que la virtualisation apporte non seulement une isolation d'environnements logiciels mais assure également une répartition équitable des ressources afin d'assurer une qualité de service à travers la plateforme.

Conclusion

La première étude nous a permis, tout d'abord, de mettre en place les outils nécessaires à une plateforme de virtualisation basée sur Xen. Cette mise en place a été concrétisée par la mise en place d'une image de la plateforme de virtualisation sur le serveur d'images de la salle de travaux pratiques réseau D202. Ensuite, nous avons pu mettre en avant et tester les différents modes de fonctionnement réseau de Xen. Lors de cette étude, nous avons pu nous rendre compte de la complexité de la gestion du réseau sur la plateforme que nous avions mis en place. Cette problématique est liée au système d'exploitation Linux et non à l'hyperviseur Xen. Pour palier à cette problématique, Citrix commercialise une version de Xen qui dispose d'une interface de gestion largement simplifiée. Cependant Xen met à disposition des outils permettant de configurer les machines virtuelles selon une topologie logique indépendante de la topologie physique. Ainsi Xen offre de la souplesse et de la flexibilité dans la configuration réseau des machines virtuelles permettant d'intégrer une plateforme dans un réseau d'entreprise sans avoir besoin d'en redéfinir l'architecture.

La seconde étude nous a ensuite permis d'utiliser les outils mis en place lors de la première étude afin de comparer une plateforme non virtualisée à une plateforme virtualisée. Nous avons réussi à montrer de manière quantitative les avantages significatifs apportés par la virtualisation dans le cadre de la mutualisation des ressources physiques à travers plusieurs services. La virtualisation apporte un avantage considérable en terme d'isolation de performances qui n'est pas présent dans les systèmes d'exploitation. Ceci nous a permis de valider et d'obtenir des chiffres qui permettent de défendre la virtualisation vis à vis des systèmes conventionnels. La mise en place de la méthode de test a également été une expérience très intéressante dans la mesure où cela nous a obligé à présenter et défendre des résultats expérimentaux.

Au final, cette TX nous a permis de compléter nos connaissances sur la virtualisation par des études pratiques de la virtualisation ainsi que d'appliquer une rigueur particulière au niveau de la mise en place d'un protocole expérimentale et le commentaire des résultats en découlant. Nous avons acquis des compétences dans la mise en place d'un environnement de test et l'exécution d'un protocole adéquat sur ce dernier. Ces compétences seront directement

transposables dans un contexte professionnel que nous serons amenés à rencontrer prochainement.

Bibliographie

William Von Hagen, Professional *Xen Virtualisation*. Wiley Publishing

David Chisnall, *The Definitive Guide to the Xen Hypervisor*. Prentice Hall Open Source Developpement Series.

Todd Deshane, *Quantitative Comparison of Xen and KVM*. Xen Summit, 23 Juin 2008.

Xen Networking [En ligne]. Disponible sur : xensource.org : Xen Networking

Bridging domaines to tagged VLANs in Xen [En ligne]. Disponible sur : http://renial.net/weblog/2007/02/27/xen-vlan/

Xen Network Wiki [En ligne]. Disponible sur : http://wiki.kartbuilding.net/index.php/Xen_Networking

Annexes

Tableau de données des résultats expérimentaux de l'étude sur le cloisonnement

	Min	Moy	Ecart Type	Med	95%	Max
Apache1 - 50 / 5	1	2	0.4	2	3	3
Apache1 - 50 / 5	1	2	0.2	2	2	3
Apache1 - 50 / 5	1	2	0.6	2	2	5
Apache1 - 50 / 5	1	3	2.1	3	9	9
Apache1 - 50 / 5	2	2	0.3	2	2	3
Apache1 - 50 / 5	1	2	0.6	2	3	4
Apache2 - 500k / 500	1	126	602.3	11	226	21080
Apache1 - 50 / 5	1	3	1.6	3	7	8
Apache1 - 50 / 5	2	2	0.3	2	2	3
Apache1 - 50 / 5	1	2	0.5	2	2	4
Apache2 - 100k / 100	1	26	256.8	2	32	9031
Apache1 - 50 / 5	1	2	0.3	2	3	3
Apache1 - 50 / 5	2	3	2.1	2	9	9
Apache1 - 50 / 5	1	2	0.7	2	4	4
Apache2 - 500k / 1000	1	250	1080.5	23	704	45296
Apache1 - 50 / 5	2	3	0.2	3	7	7
Apache1 - 50 / 5	1	2	0.2	2	2	3
Apache1 - 50 / 5	2	3	1.5	3	7	7
Linux1 - 1x CPU Bomb						
Linux2 - 1x CPU Bomb						
Apache1 - 50 / 5	1	8	10.6	2	33	33
Apache1 - 50 / 5	2	2	0.3	2	2	3
Apache1 - 50 / 5	1	3	2.9	2	10	13
Linux1 - 4x CPU Bomb						
Linux2 - 4x CPU Bomb						
Apache1 - 50 / 5	1	3	2.6	2	10	11
Apache1 - 50 / 5	2	4	3.4	3	13	13
Apache1 - 50 / 5	1	2	0.3	2	2	3
Linux1 - 4x CPU + IO						
Linux2 - 4x CPU + IO						
TPXen Client1 - 50/5	21	408	1390.6	25	3148	6451
TPXen Client1 - 50/5	23	331	923.1	26	3029	3235
TPXen Client1 - 50/5	21	411	1541.4	25	6450	6451
TPXen Client2 - 500k/1000	13	196	1278.2	26	233	74128

TPXen Client1 - 50/5	1	272	818.3	2	2702	2703
TPXen Client1 - 50/5	1	81	209.8	2	698	698
TPXen Client1 - 50/5	1	71	211.6	2	700	700
TPXen Client2 - IO	TO	TO	TO	TO	TO	TO