Volume: 04 / N°: 01(2021), p 254-264

# *Introduction à la virtualisation*

# *Introduction to virtualization*

*Nouh TOUTI**
*Université el Djelfa*
*(Algérie)*

*Mourad noumri*
*Université el Djelfa*
*(Algérie)*
*noumri.mourad@yahoo.fr*

**Abstract:**
In this paper we will introduce the notion of virtualisation concepts and hypervisors types and the different variations in virtualisation.
The main functionalities and the different solutions which virtualisation provides
**Keywords:** virtualisation, complex systems, XEN, VMWare ESX/ESXi, Hyper-V, mxVM

**Résumé:**
Dans cet article, nous introduirons la notion de concepts de virtualisation et de types d'hyperviseurs et les différentes variations de virtualisation.
Les principales fonctionnalités et les différentes solutions apportées par la virtualisation
**Mots clés :** virtualisation, systèmes complexes, XEN, VMWare ESX / ESXi, Hyper-V, mxVM

**Introduction à la virtualisation :**

Un serveur est un ordinateur utilisé à distance depuis différents postes de travail, ou autres serveurs. Il possède des ressources matérielles, principalement CPU, mémoire, disques et interfaces réseau. Ces ressources sont utilisées par des applications, non pas de manière directe, mais en s'appuyant sur un système d'exploitation. La virtualisation de serveurs est un ensemble de techniques et d'outils permettant de faire tourner plusieurs systèmes d'exploitation sur un même serveur physique. Le principe de la virtualisation est donc un principe de partage : les différents systèmes d'exploitation se partagent les ressources du serveur. Pour être utile de manière opérationnelle, la virtualisation doit respecter deux principes fondamentaux :

Le cloisonnement : chaque système d'exploitation a un fonctionnement in-dépendant, et ne peut interférer avec les autres en aucune manière.

La transparence : le fait de fonctionner en mode virtualisé ne change rien au fonctionnement du système d'exploitation et a fortiori des applications.

La transparence implique la compatibilité : toutes les applications peuvent tourner sur un système virtualisé, et leur fonctionnement n'est en rien modifié.

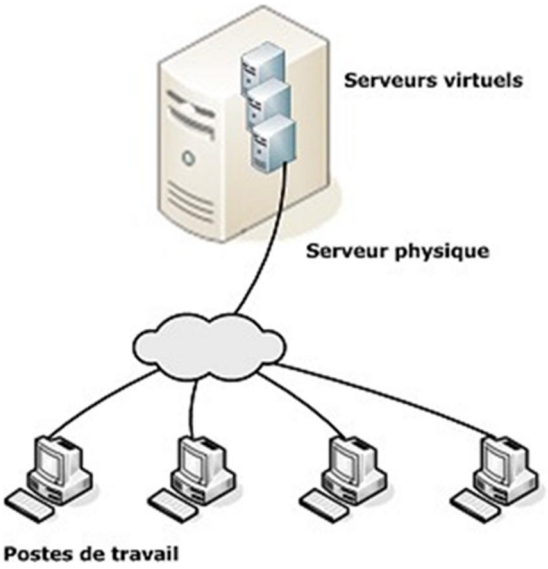


Figure 1.1 – Architecture Virtualisée

**Les hyperviseurs:**

Un hyperviseur est une plate-forme de virtualisation qui permet à plusieurs systèmes d'exploitation de travailler sur une même machine physique en même temps.

**Hyperviseur de type 1 :**

Un hyperviseur de Type 1, ou natif, voire "bare metal" (littéralement "métal nu"), est un logiciel qui s'exécute directement sur une plateforme matérielle ; cette plateforme est alors considérée comme outil de contrôle de système d'exploitation. Un système d'exploitation secondaire peut, de ce fait, être exécuté au-dessus du matériel. L'hyperviseur type 1 est un noyau hôte allégé et optimisé pour ne faire tourner initialement que des noyaux de systèmes d'exploitation invités adaptés et optimisés à cette architecture spécifique, ces systèmes invités ayant "conscience" d'être virtualisés. Sur des processeurs ayant les instructions de virtualisation matérielle (AMD-V et Intel VT), le système d'exploitation invité n'a

plus besoin d'être modifié pour pouvoir être exécuté dans un hyperviseur de type 1.Quelques exemples de tels hyperviseurs plus récents sont Xen, Oracle VM, ESX Server de VMware.

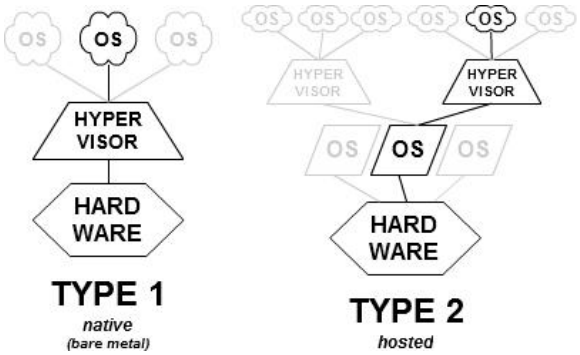


Figure 1.2 – Les types des hyperviseurs

**Hyperviseur de type 2 :**

Un hyperviseur de Type 2 est un logiciel qui s'exécute à l'intérieur d'un autre système d'exploitation. Un système d'exploitation invité s'exécutera donc en troisième niveau au-dessus du matériel. Les systèmes d'exploitation invités n'ayant pas conscience d'être virtualisés, ils n'ont pas besoin d'être adaptés. Quelques exemples de tels hyperviseurs sont VMware Workstation, VMware Fusion, l'hyperviseur open source QEMU, les produits Microsoft Virtual PC et Virtual Server, VirtualBox d'Oracle, de même que Parallèles Workstation de SWsoft et Parallels Desktop.

**Les différents types de la Virtualisation :**

**Virtualisation Complète :**

La virtualisation dite complète permet de faire fonctionner n'importe quel système d'exploitation en tant qu'invité dans une machine virtuelle. Pour l'utilisateur final, ce type de virtualisation est la plus simple à mettre en place et est la plus pratique.

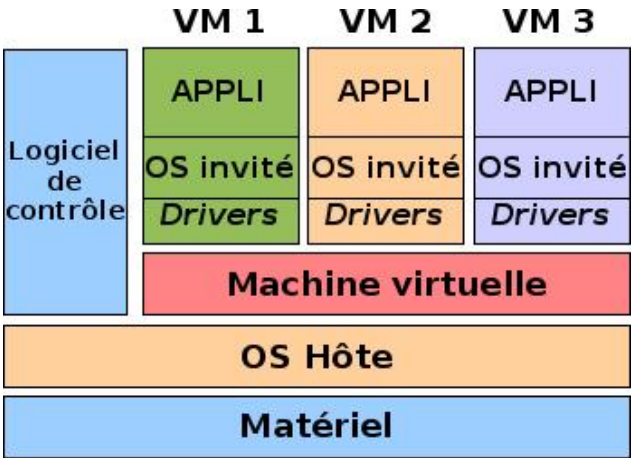


Figure 1.3 – Virtualisation Complète

Principe : L'hyperviseur crée un environnement virtuel complet simulant littéralement un nouvel ordinateur complet, avec du "faux matériel". À quelques rares exceptions, le système d'exploitation invité (installé dans la machine virtuelle) ne communique qu'avec ce faux matériel simulé, rendant étanche l'environnement virtualisé.

Limitations : Ce type de virtualisation ne permet de virtualiser que des systèmes d'exploitation prévus pour la même architecture matérielle que le processeur physique de l'ordinateur hôte. Par exemple, un ordinateur équipé d'un processeur Intel x86 sera incapable de virtualiser un système d'exploi- tation prévu pour fonctionner dans une architecture PowerPC.

Quelques hyperviseurs de virtualisation complète :

- VirtualBox
- VMWare Player, VMWare Workstation

- Parallels Desktop for Windows et Linux

- KVM

**Para-Virtualisation**

La para-virtualisation fait intervenir un hyperviseur. Il s'agit d'un noyau allégé au-dessus duquel viendront se greffer les systèmes invités. Contrairement à un système traditionnel de machines

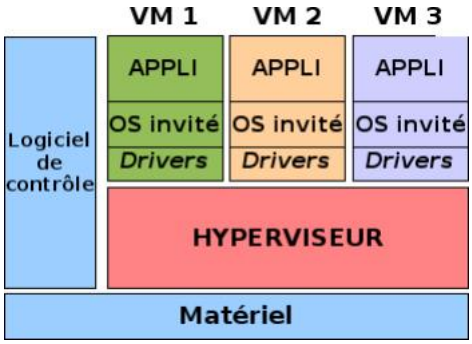


Figure 1.4 – Para-virtualisation

virtuelles où la virtualisation est transparente, avec la para-virtualisation, le système invité doit avoir conscience qu'il tourne dans un environnement virtuel ce qui implique d'employer un noyau modifié. Ce type de virtualisation permet des performances bien plus importantes que la virtualisation totale (assistée par matériel, que nous avons vu plus haut) .

Quelques hyperviseurs de virtualisation assisté :

- XEN

- VMWare ESX/ESXi

- Hyper-V (Microsoft)

- xVM

**Les Isolateurs**

Un isolateur est un logiciel permettant d'isoler l'exécution des applications dans ce qui sont appelés des contextes, ou bien zones d'exécution. L'isolateur permet ainsi de faire tourner plusieurs fois la même application dans un mode multi-instance (plusieurs instances d'exécution) même si elle n'était pas conçue pour ça. Cette solution est très performante, du fait du peu d'overhead (temps passé par un système à ne rien faire d'autre que se gérer), mais les environnements virtualisés ne sont pas complètement isolés. La performance est donc au rendez-vous, cependant on ne peut pas vraiment parler de virtualisation de systèmes d'exploitation.

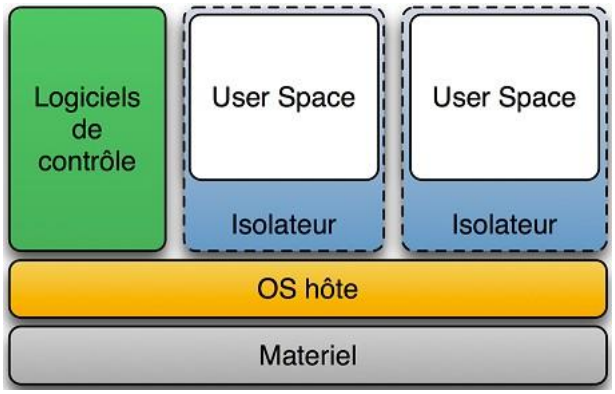


Figure 1.5 – Isolateur

Uniquement liés aux systèmes Linux, les isolateurs sont en fait composés de plusieurs éléments et peuvent prendre plusieurs formes, Par exemple : Linux-VServer (isolation des processus en espace utilisateur) ; chroot (isolation changement de racine) ; BSD Jail (isolation en espace utilisateur) ; OpenVZ : libre, (partitionnement au niveau noyau sous Linux) ; LXC : libre, (usage des C-groups du noyau Linux).

**Les Solutions Principals**

- **XEN**

Xen est une solution de virtualisation open source développée initialement par le département informatique de l'Université de Cambridge. Son développement est aujourd'hui activement sponsorisé par Citrix, qui a racheté l'éditeur initial XenSource. Citrix distribue une version commerciale de Xen, nommée Citrix XenServer, particulièrement adaptée à la virtualisation des OS Microsoft Windows et Linux RHEL et SLES. Elle est dotée d'une interface d'administration avancée, et d'un accès au support technique. Quant aux fonctionnalités, elles sont les mêmes que dans la version distribuée librement. De grandes sociétés comme IBM ont contribué au développement de Xen, et de gros efforts ont été faits par Citrix pour assurer une compatibilité parfaite avec Windows, compatibilité aujourd'hui reconnue par Microsoft.

**KVM**

KVM, Kernel Virtual Machine, est intégré depuis le noyau 2.6.20 et per- met une virtualisation matérielle et donc une accélération de la virtualisation de système d'exploitation.

C'est un système optimisé pour la virtualisation de serveur. Pour virtualiser des systèmes de type desktop, on peut lui préférer virtualbox. KVM semble en effet plus performant en consommation de processeur mais plus lent pour l'émulation du périphérique graphique. L'utilisation d'un bureau virtualisé dans VirtualBox pourra donc laisser une meilleure impression à l'utilisateur. Vous pouvez tout de même préférer KVM pour sa meilleure compatibilité avec des système d'exploitations anciens ou peu populaires.

Néanmoins, KVM est complètement libre, performant et très facile à installer et à utiliser. L'interface graphique virt-manager pourra aider à paramétrer KVM et pourra rendre la vie plus simple aux administrateurs réseaux. Mais Vous ne pouvez pas utiliser KVM en même temps que VirtualBox. Il faudra en effet fermer KVM pour utiliser VirtualBox et vice versa. Ou désactiver le support de la virtualisation processeur dans VirtualBox.

- **VMware ESX**

VMware vSphere est un logiciel d'infrastructure de Cloud computing de l'éditeur VMware, c'est un hyperviseur de type 1 (Bare Metal), basé sur l'architecture VMware ESXi.

VMware vSphere nécessite une configuration matérielle restreinte précisée dans le guide de comptabilité VMware.

La gestion de ce serveur hôte peut se faire via plusieurs possibilités : par le navigateur Web avec une connexion directe, par une console cliente avec une connexion directe ou par un outil de gestion centralisée nommé VMware vCenter Server qui permet d'administrer l'ensemble des machines virtuelles, des hôtes physiques, de leurs ressources et des options de l'environnement (High Availability, vMotion, Storage vMotion, Distributed Resource Schedu- ler, Fault Tolerance) depuis une seule console.

- **Hyper-V**

Hyper-V, également connu sous le nom de Windows Server Virtualisation, est un système de virtualisation basé sur un hyperviseur 64 bits de la version de Windows Server 2008.

Il est possible d'utiliser la console Hyper-V sur Windows 7. Dans le sens in- verse, de nombreux systèmes d'exploitation peuvent tourner à l'intérieur de Hyper-V :

Bien évidemment pour les systèmes d'exploitation Microsoft Windows 8.1, Windows 8, Windows 7 (sauf édition familiale), Windows Vista SP1/SP2 (sauf édition familiale), Windows Server 2008 x64 SP1/SP2 et R2, Windows Server 2003 x64 SP2 et R2 SP2, Windows 2000 SP4, Windows XP Professionnel SP2/SP3 et x64 SP2

Pour les systèmes d'exploitation linux :

SUSE Linux Enterprise Server 10 SP1/SP2 et 11

Red Hat Enterprise Linux 5.2 x64 et versions ultérieures Ubuntu 12.04 LTS et versions ultérieures

- **OpenVZ**

Une des solutions les plus avancées et matures dans le domaine de l'isolation est OpenVZ. Ce produit se présente sous la forme d'un patch pour le noyau Linux, et d'un ensemble d'outils d'administration. Le patch du noyau permet à un système GNU/Linux de gérer des contextes virtualisés. Les outils d'administration permettent de créer, d'instancier, et de contrôler les environnements virtuels. Rappelons que la technologie d'isolation ne permet d'exécuter que des serveurs virtuels Linux sur un hôte OpenVZ, même si ces serveurs peuvent être de distributions différentes. Certaines distributions Linux proposent des versions packagées d'OpenVZ. En particulier, la dis- tribution Debian GNU/Linux, dans les versions Lenny et Squeeze, permet dès l'installation du serveur physique de mettre en place cette solution en quelques secondes via son système de packages. Il faut cependant noter que OpenVZ a été remplacé par LXC dans la version Squeeze, paru en 2013.

Le projet OpenVZ fournit aux systèmes GNU/Linux une méthode de virtua- lisation. Cette virtualisation se situe au niveau du noyau de l'OS. Cela rend possible l'exécution de multiples instances d'OS GNU/Linux sur la même machine. Ces instances fonctionnant de façon complètement sécurisées et partageant intelligemment les ressources du serveur hôte.

- **LXC**

LXC est une solution de virtualisation de type isolateur. Cette solution permet la virtualisation par container au niveau du noyau. LXC est très ré- cent et remplace Linux-VServer et OpenVZ. Aussi, LXC est dès à présent intégré au noyau, ce qui nŠa jamais été le cas des deux solutions citées précédemment.

L'isolateur tire avantage de la possibilité, unique sous UNIX et Linux, de partager le noyau avec dŠautres processus du système. Cette virtualisation à noyau partagé utilise une fonctionnalité nommée chroot. Cette fonctionnalité modifie le système de fichiers racine dŠun processus pour l'isoler de manière à fournir une certaine sécurité. On parle alors de ń prison ż chroot.

**Domaines d'application :**

Voici quelques exemples d'application de ces techniques de virtualisation, dans les domaines où elles sont couramment mises en place.

Les offres d'hébergement étaient traditionnellement distinguées en deux catégories : hébergement dédié et hébergement mutualisé.

Dans un hébergement dédié, le fournisseur met à disposition de son client un ou plusieurs serveurs, configurés selon ses besoins. Selon les cas, le contrat peut prévoir une plus ou moins grande autonomie du client par rapport à la

configuration et l'exploitation de son serveur, mais du moins au plan technique, rien ne s'oppose à ce que le contrôle soit total.

Avec un hébergement mutualisé, le fournisseur utilise un même serveur pour plusieurs de ses clients. Il utilise différentes solutions de cloisonnement 1 pour maintenir une certaine étanchéité entre ces environnements.

Le partage de la ressource serveur permet bien sûr un coût très inférieur, particulièrement attractif pour les sites à faible trafic. Mais l'hébergement mutualisé simple a plusieurs handicaps:

- L'allocation des ressources du serveur n'est pratiquement pas contrôlée, de sorte que la qualité de service de chaque site peut être pénalisée par un pic de trafic, ou par la boucle d'un programme sur un autre site.

- La configuration logicielle est unique, et dictée par l'hébergeur. Elle fait le choix, en général, d'un même serveur Http, mais aussi très souvent d'un même outil de gestion de contenus et de base de données. La simple installation de telle ou telle librairie spécifique nécessaire à l'un des clients n'est en général pas possible.

- En termes d'exploitation, chaque client est extrêmement confiné, de peur qu'il ne perturbe la configuration. Il dispose le plus souvent d'un simple accès en transfert de fichier sur son répertoire privé, et dans tous les cas n'a jamais l'accès root (administrateur) sur le serveur.

Entre ces deux modes d'hébergement, la virtualisation a permis un mode combinant les bénéfices de l'un et de l'autre : le partage de ressources d'une part, l'autonomie et le contrôle d'autre part.

C'est le mode que l'on appelle VDS pour un serveur dédié virtuel (*Virtual Dedicated Server).*

**La Virtualisation de Stockage :**

Dans tout projet de virtualisation se pose, à un moment ou un autre, la question du stockage. Quel que soit la technologie utilisée, une machine virtuelle se compose de deux éléments :

Des ressources : part de CPU alloués, mémoire vive autorisée, nombre de cartes réseau virtuelles...

Des données : comme un serveur normal, on doit disposer d'un système d'exploitation de bibliothèques, d'outils, d'applications et de leurs données.

Le stockage dépend de la technologie de virtualisation utilisée, et surtout de sa profondeur.

Dans les technologies de machine virtuelle, l'hyperviseur ne fournit au système virtualisé qu'un espace de stockage. Il peut s'agir d'un volume, ou simplement d'un fichier, on peut

placer l'intégralité de cet espace sur un disque local, un réseau de stockage ou un autre serveur...

L'utilisation d'un disque local est la solution la plus avantageuse en terme de performances et de facilité d'administration. Cependant, l'utilisation d'un stockage en réseau permet d'ouvrir la voie à de nouvelles fonctionnalités.

**Stockage en Réseau**

Les pleines capacités des hyperviseurs modernes ne peuvent s'exprimer qu'au travers d'un stockage en réseau, en effet les hyperviseurs sont généralement gérés

sous forme de ńpoolsż, formant une ńforce de travailż globale qui se partageront les machines virtuelles à exécuter.

Cette vision n'est possible que si le stockage est lui aussi unifié : sans cela chaque hyperviseur ne peut faire tourner que les serveurs virtuels présents sur son disque local, et n'est donc pas interchangeable.

Disposant d'un réseau de stockage, chaque hyperviseur a accès à toutes les machines virtuelles, et peut donc exécuter n'importe laquelle, et la transférer sans interruption à un autre hyperviseur en fonction de sa charge. Nous allons présenter quelques technologies permettant de mettre en place un réseau de stockage.

- **NAS et NFS**

Un NAS, ou stockage réseau (Network-Attached Storage) est simplement un serveur fournissant leurs fichiers à d'autres serveurs par le réseau.

NFS (Network file storage) est le standard universel pour l'accès aux fichiers sur un réseau, c'est le protocole le plus utilisé dans les NAS.

Dans le cadre d'un isolateur, il permet de stocker l'arborescence du serveur virtuel à distance. Dans le cadre d'une solution de virtualisation complète il permet de stocker à distance les fichiers contenant les disques durs de la machine virtuelle. Ce dernier cas est déconseillé hors des environnements de test : NFS n'est pas adapté à la lecture aléatoire dans un seul fichier. En revanche pour un isolateur, stocker les données en NFS est intéressant, et le deviendra encore plus avec les systèmes de fichier de nouvelle génération tels que ZFS, HAMMER ou btrfs, qui permettent des snapshots instantanés, le visionnement des arborescences et autres fonctionnalités pour l'instant réservées aux baies de stockage haut de gamme.

En plus des matériels dédiés, la plupart des systèmes d'exploitation proposent une implémentation serveur NFS, ce qui permet d'utiliser n'importe quel serveur comme serveur de stockage NFS. Ces derniers utilisent alors soit des disques locaux, soit leur propre réseau de stockage SAN

- **SAN**

Un SAN, ou réseau de stockage (Storage Area Network), est un réseau sur lequel circulent les données entre un système et son stockage. Cette technique permet de déporter tout le stockage interne d'une machine vers un équipe- ment dédié. Les SAN sont des équipements dédiés, qui ne travaillent qu'aux plus basses couches du stockage, la notion de fichier leur est inconnue ; ils travaillent simplement sur des blocs de données et les fournit par le réseau à des serveurs qui eux sauront les utiliser.

Cependant, les SAN les plus hauts de gamme sont dotés de capacités avancées, tel que la prise de cliché, ou encore la copie rapide de volumes. Les deux principaux protocoles d'accès à un SAN sont iSCSI et Fibre Channel.

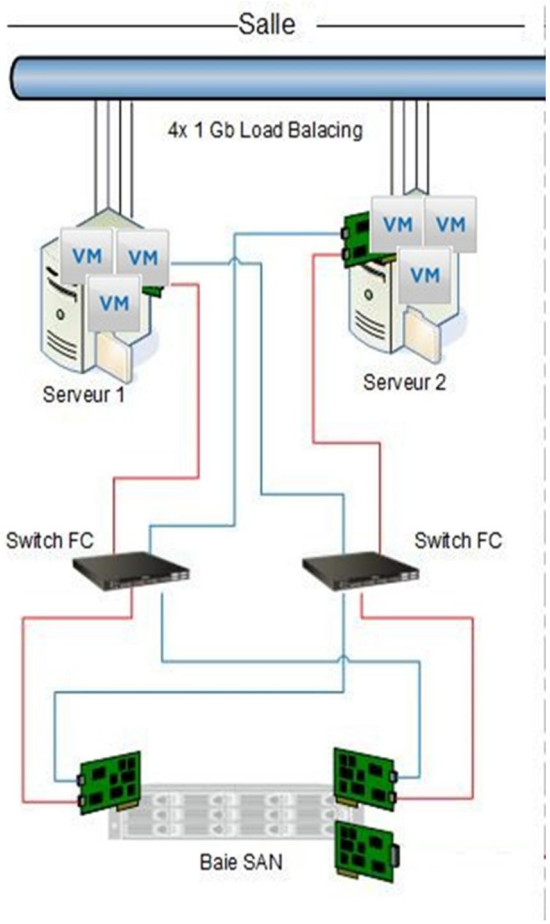


Figure 1.6 – Exemple de stockage SAN

- **iSCSI** est un protocole d'accès disque fonctionnant sur un réseau Ethernet, il permet d'implémenter un réseau de stockage en profitant de la connectique et des équipements de commutation standards. Comme le NFS, il peut être soit implémenté par une baie de stockage dédiée, ce qui assure les meilleures performances, soit par un serveur classique disposant du logiciel adéquat, par exemple IET (iSCSI Entreprise Target) sous Linux. Voici un exemple de SAN : parmi les machines clientes du SAN, on retrouve un NAS : ces deux techniques peuvent être combinées car elles ne travaillent pas au même niveau.

- **Fibre Channel**

La solution la plus haut-de-gamme pour implémenter un réseau de stockage est l'utilisation d'une baie dédiée et du protocole Fibre Channel. Basé sur des fibres optiques il assure une latence et un débit bien meilleurs que iSCSI, à un prix bien sûr plus élevé. Son principe d'utilisation est le même qu'un SAN iSCSI.

1. R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, « Cloud Computing and Emerging it Platforms: Vision, hype, and reality for delivering computing as the 5th utility », *Future Generation computer systems*, vol. 25, no. 6, pp. 599-616, 2009.

2. J. W. Smith and I. Sommerville, « Workload Classification & Software Energy Measurement for Efficient Scheduling on Private Cloud Platforms », *arXiv preprint arXiv:1105.2584*, 2011.

3. Q. Zhang, J. Hellerstein, and R. Boutaba, « Characterizing Task Usage Shapes in Google Cmpute Clusters », 2011.

4. A. K. Mishra, J. L. Hellerstein, W. Cirne, and C. R. Das, « Towards Characterizing Cloud Backend Workloads: Insights From Google Compute Clusters », *ACM SIGMETRICS Performance Evaluation Review*, vol. 37, no. 4, pp. 34- 41, 2010.

5. S. Kavulya, J. Tan, R. Gandhi, and P. Narasimhan, « An Analysis of Traces From a Production MapReduce Cluster », in *Cluster, Cloud and Grid Computing (CCGrid), 2010 10th IEEE/ACM*

*International Conference on*, pp. 94-103, IEEE, 2010.

6. S. Aggarwal, S. Phadke, and M. Bhandarkar, « Characterization of Hadoop Jobs Using Unsupervised Learning », in *Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on*, pp. 748- 753, IEEE, 2010.

7. I. S. Moreno, P. Garraghan, P. Townend, and J. Xu, « Analysis, Modeling and Simulation of Workload Patterns in a Large-Scale Utility Cloud », *IEEE Transactions on Cloud Computing*, vol. 2, pp. 208-221, apr 2014.

8. S. Shen, V. Van Beek, and A. Iosup, « Statistical Characterization of Business-Critical Workloads Hosted in Cloud Datacenters », *Proceedings – 2015 IEEE/ACM 15th International Symposium on Cluster, Cloud, and Grid Computing, CCGrid 2015*,

pp. 465-474, 2015.

9. I. S. Moreno, P. Garraghan, P. Townend, and J. Xu, « An Approach for Characterizing Workloads in Google Cloud to Derive Realistic Resource Utilization Models », in *Service Oriented System Engineering (SOSE), 2013 IEEE 7th International Symposium on*, pp. 49-60, IEEE, 2013.

10. A. Bahga, V. K. Madisetti, *et al.*, « Synthetic Workload Generation for Cloud Computing Applications », *Journal of Software Engineering and Applications*, vol. 4, no. 07, p.396, 2011.

11. A. Beitch, B. Liu, T. Yung, R. Griffith, A. Fox, D. A. Patterson, *et al.*, « Rain: A Workload Generation Toolkit for Cloud Computing Applications », *University of California, Tech. Rep. UCB/EECS-2010-14*, 2010.

12. Y. Chen, A. S. Ganapathi, R. Griffith, and R. H. Katz, « Analysis and Lessons From a Publicly Available Google Cluster Trace », *EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2010-95*, vol. 94, 2010.

13. G. Wang, A. R. Butt, H. Monti, and K. Gupta, « Towards Synthesizing Realistic Workload Traces for Studying the Hadoop Ecosystem », in *2011 IEEE 19th Annual International Symposium on Modelling, Analysis, and Simulation of Computer and Telecommunication Systems*, pp. 400-408, IEEE, 2011.

14. S. Kavalanekar, B. Worthington, Q. Zhang, and V. Sharda, « Characterization of Storage Workload Traces From Production Windows Servers », in *Workload Characterization, 2008. IISWC 2008. IEEE International Symposium on*, pp. 119-128, IEEE, 2008.

15. S. Kavalanekar, D. Narayanan, S. Sankar, E. Thereska, K. Vaid, and B. Worthington, « Measuring Database Performance in Online Services: A Trace-Based Approach », in *Technology Conference on Performance Evaluation and Benchmarking*, pp. 132-145, Springer, 2009.

16. S. Sankar and K. Vaid, « Storage characterization for unstructured

data in online services applications », in *Workload Characterization, 2009. IISWC 2009. IEEE International Symposium on*, pp. 148-157, IEEE, 2009.

17. A. Gulati, C. Kumar, and I. Ahmad, « Modeling workloads and devices for io load balancing in virtualized environments », *ACM SIGMETRICS Performance Evaluation Review*, vol. 37, no. 3, pp. 61-66, 2010.

18. O. Ozmen, K. Salem, M. Uysal, and M. Attar, «Storage workload estimation for database management systems," in *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pp. 377-388, ACM, 2007.

19. J. Wilkes, « Traveling to rome: Qos specifications for automated storage system management », in *International Workshop on Quality of Service*, pp. 75-91, Springer, 2001.

20. B. Abrahao and A. Zhang, «Characterizing application workloads on cpu utilization for utility computing» *Hewlett-Packard Labs*, 2004.

21. J. P. Patwardhan, A. R. Lebeck, and D. J. Sorin, «Communication breakdown: analyzing cpu usage in commercial web workloads» in *Performance Analysis of Systems and Software, 2004 IEEE International Symposium on-ISPASS*, pp. 12-19, IEEE, 2004.

22. S. Huang and W. Feng, « Energy-efficient cluster computing via Accurate Workload Characterization », in *Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*, pp. 68-75, IEEE Computer Society, 2009.

23. D. G. Feitelson, « Workload modeling for performance evaluation », in *IFIP International Symposium on Computer Performance Modeling, Measurement and Evaluation*, pp. 114-141, Springer, 2002.

24. H. Li, «Realistic workload modeling and its performance impacts in largescale escience grids» *IEEE Transactions on Parallel and Distributed Systems*, vol. 21, no.4, pp. 480-493, 2010.

25. Y. Joo, V. Ribeiro, A. Feldmann, A. C. Gilbert, and W. Willinger, « Tcp/ip traffic dynamics and network performance: A lesson in workload modeling, flow control, and trace-driven simulations », *ACM SIGCOMM Computer Communication Review*, vol. 31, no. 2, pp. 25-37, 2001.

26. X. Liu, J. Heo, and L. Sha, « Modeling 3-tiered web applications », in *13th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, pp. 307-310, IEEE, 2005.

27. A. Ganapathi, Y. Chen, A. Fox, R. Katz, and D. Patterson, « Statistics-driven workload modeling for the cloud », in *Data Engineering Workshops (ICDEW), 2010 IEEE 26th International Conference on*, pp. 87-92, IEEE, 2010.