Étude comparative entre les performances du volume LVM en allocation dynamique et du fichier image RAW de la plateforme de virtualisation KVM/QEMU

Par

Adil Ratli

Essai présenté au CeFTI

en vue de l'obtention du grade de maître en technologies de l'information

(maîtrise en génie logiciel incluant un cheminement de type cours en technologies de l'information)

FACULTÉ DES SCIENCES

UNIVERSITÉ DE SHERBROOKE

Longueuil, Québec, Canada, Juin 2020

Sommaire

La virtualisation consiste à exécuter plusieurs systèmes d'exploitation (machines virtuelles) sur la même machine physique. Les ressources de cette dernière sont partagées par toutes les machines virtuelles. Par conséquent, une optimisation de l'utilisation de ces ressources s'avère nécessaire pour tirer le maximum de gains et de performances. Une bonne gestion de l'espace de stockage virtuel a un impact majeur sur l'amélioration des performances de la machine virtuelle. L'objectif principal du présent essai consiste à évaluer l'impact du format du stockage des disques virtuels sur les performances d'entrée-sortie. Il se limite à deux formats particuliers : le fichier image RAW et le volume LVM en mode allocation dynamique.

La mise en place d'une infrastructure de virtualisation nécessite de nombreuses décisions quant au choix des techniques les plus appropriées pour obtenir les meilleures performances. Le choix du format de stockage utilisé par les VM ne fait pas exception et un banc d'essai permettant de mesurer les performances de chaque format s'avère utile.

Cette étude propose une solution à cette problématique en suivant une méthodologie qui repose sur une recherche quantitative. Des tests de lecture et écriture sont réalisés pour chaque format de stockage, et la collecte de données est effectuée à l'aide d'un logiciel spécialisé à cet effet. Les résultats sont ensuite comparés pour déduire lequel des deux formats offre les meilleures performances. L'hypothèse stipule que le format LVM en mode allocation dynamique offre des performances meilleures que celles offertes par le format RAW.

Les résultats confirment que le format LVM en mode allocation dynamique performe mieux que le fichier image RAW et ce pour toutes les opérations d'E/S qui ont été testées dans cette étude, avec un gain maximal de 13,28 % pour l'écriture séquentielle et un gain minimal de 1,28 % pour la lecture aléatoire. Néanmoins, les deux formats offrent d'excellentes performances pour les opérations de lectures qui restent très proches des performances obtenues par du matériel physique.

Pour conclure, l'utilisation d'un volume LVM en mode allocation dynamique comme disque virtuel pour une VM, permet à la fois de bénéficier de performances meilleures que celles

obtenues par le format RAW, et de certaines fonctionnalités qui nous permettent d'optimiser l'utilisation de l'espace du stockage et tirer avantage des instantanés.

Remerciements

Je tiens à remercier mon directeur de recherche Pierre-Martin Tardif qui m'a permis de rédiger un essai de qualité grâce à ces conseils et recommandations.

Je remercie également Michel Hébert pour son aide, sa disponibilité et son encouragement.

Mes remerciements s'adressent également à tous les enseignants, les conférenciers et les chargés de cours rencontrés tout au long de mes études au CeFTI.

Finalement, je remercie toutes les personnes, de près ou de loin, de ma famille, mes collègues et mes amis qui m'ont encouragé à mener à bien ce présent travail.

Table des matières

Liste des tableaux

Liste des figures

Glossaire

Mode natif : appelé aussi « mode physique », il fait référence à une utilisation classique du matériel physique sans virtualisation.

Liste des sigles, des symboles et des acronymes

DAS : Direct-Attached Storage

E/S : Entrées/Sorties (*IO, Input/Output*).

HDD : Hard Disk Drive.

IOPS : (*Input/Output Operations per Second*), nombre des opérations d'E/S par seconde.

KVM : Kernel-based Virtual Machine.

LVM : Logical Volume Manager.

MMV : Moniteur de Machine Virtuelle.

NAS : Network-Attached Storage

QEMU : Quick EMUlator.

RAID : Redundant Array of Independent Disks.

SAN : Storage Area Network

SSD : Solid-State Drive

VM : (*Virtual Machine*) Machine virtuelle.

VMM : (*Virtual Machine Monitor*) Moniteur de machine virtuelle (voir MMV).

Introduction

Depuis son apparition dans les années 1960, la virtualisation a connu une grande évolution qui lui a permis de devenir un outil indispensable pour la gestion des grandes infrastructures TI modernes comme l'infonuagique. Parmi les défis que la virtualisation doit surmonter, on trouve la gestion des stockages dans les machines virtuelles qui a un grand impact sur les performances des entrées/sorties. Plusieurs solutions existent pour émuler le fonctionnement d'un disque virtuel. Certaines de ces solutions sont basées sur des fichiers et d'autres sur l'utilisation directe d'une partition physique ou un volume logique. Chaque format de stockage présente des avantages, mais entraîne aussi des inconvénients. Le bon choix est fondé sur les besoins et les spécifications du projet de la mise en place de la plateforme de virtualisation.

Dans le contexte d'une plateforme de virtualisation KVM/QEMU, les deux formats de stockage basés sur un fichier image RAW et un volume LVM constituent deux solutions qui peuvent offrir d'excellentes performances. L'objectif de cet essai et de mener une étude basée sur une recherche quantitative pour comparer les performances du format de stockage RAW et celui de LVM en mode allocation dynamique.

Le premier chapitre présente le contexte de réalisation de l'étude. On y trouve une brève introduction générale à la virtualisation tout en se focalisant sur la plateforme KVM/QEMU, une présentation des notions essentielles à connaître pour comprendre la problématique et le procédé expérimental mise en place pour effectuer les tests.

Le deuxième chapitre présente la revue de la littérature. Les articles scientifiques pertinents à cette étude sont exposés et résumés. On y trouve les résultats des expériences menées à ce jour pour comparer les deux formats de stockages dans d'autres contextes, et les conclusions qu'on peut déduire pour formuler la problématique et élaborer une approche pour effectuer la recherche.

Le troisième chapitre décrit la problématique. Le but est de formuler la question de recherche et émettre une hypothèse. Certains résultats concernant d'autres expériences, qui ont été

présentés dans le deuxième chapitre, sont extrapolés pour quantifier les ordres de grandeur auxquels on devrait s'y attendre pendant l'expérimentation.

Le quatrième chapitre expose les différentes étapes à suivre pour réaliser l'étude. On y trouve la description de l'approche, la mise en place du laboratoire expérimental, le recueillement et traitement des données et l'approche de validation des résultats.

Le cinquième chapitre présente les résultats de l'expérience et leurs analyses. Les quatre opérations mesurées sont l'écriture séquentielle, l'écriture aléatoire, la lecture séquentielle et la lecture aléatoire. On conclut ce chapitre par la validation des résultats et de l'hypothèse de l'étude.

Chapitre 1

Mise en contexte

Le but de ce chapitre est de fournir les concepts fondamentaux de la virtualisation, des formats de stockage et des mesures de performances. La connaissance de ces concepts est essentielle pour comprendre les éléments des chapitres suivants, notamment la problématique. Les principaux mots-clés et notions à retenir sont KVM, QEMU, le format de fichier RAW, le volume LVM en mode allocation dynamique et les mesures de performances E/S.

1.1 Virtualisation

La virtualisation a fait ses débuts pendant les années 1960 [1]. Depuis, plusieurs concepts et techniques sont apparus pour l'améliorer et tirer le maximum de bénéfices de son utilisation.

La virtualisation offre de nombreux avantages comme la réduction de la consommation électrique, la diminution de l'espace utilisé, la haute disponibilité et la tolérance aux pannes [2]. Elle est devenue la solution incontournable pour bâtir les infrastructures TI modernes comme l'infonuagique [3]. L'un des éléments les plus importants de la virtualisation est l'hyperviseur, le logiciel qui permet, entre autres, la création et la gestion des machines virtuelles.

1.1.1 Définition

Bien qu'il n'existe pas de standard qui définit la virtualisation, elle peut être définie comme une technique qui permet un partitionnement logique d'une machine physique en plusieurs machines virtuelles [3]. Elle consiste à utiliser une couche d'abstraction qui permet d'imiter le fonctionnement du matériel d'un ordinateur [4].

1.1.2 Aperçu historique

En 1964, IBM annonce sa nouvelle série d'ordinateurs centraux : les System/360. Au début, ces ordinateurs étaient destinés à ne faire que du traitement par lots [5]. Ils coutaient très cher et, de plus en plus, les compagnies et les organismes cherchaient à mieux les exploiter en utilisant le principe de temps partagé (*Time-sharing*). Pour répondre à ce besoin, IBM sort le CP-40 en janvier 1967, un système d'exploitation à temps partagé qui est complètement basé sur la virtualisation du matériel. Le CP-40 roulait sur le System/360 et pouvait gérer jusqu'à 14 machines virtuelles [1]. IBM continuera le développement de son système jusqu'à l'actuelle version de sa famille de systèmes d'exploitation pour virtualisation : le z/VM [6] [7].

En 1998, VMware introduit une solution de virtualisation pour l'architecture x86. Cette solution était basée sur l'émulation du matériel, ce qui avait un impact négatif sur les performances [8].

En 2005, Intel commercialisait ses nouveaux microprocesseurs incluant des nouveaux jeux d'instructions VT-x [9] permettant de faire de la virtualisation matérielle assistée [4]. Cela permet aux systèmes d'exploitation de virtualisation (hyperviseurs) de mieux gérer les ressources matérielles et d'augmenter les performances [2].

1.1.3 Hyperviseur

L'hyperviseur, aussi appelé « moniteur de machine virtuelle, MMV » (*Virtual Machine Monitor, VMM*), est un logiciel qui sert à créer et gérer les machines virtuelles [4]. La machine sur laquelle on installe ces machines virtuelles est appelée machine « hôte » (*host*), alors qu'une machine virtuelle est appelée « invitée » (*guest*) [10].

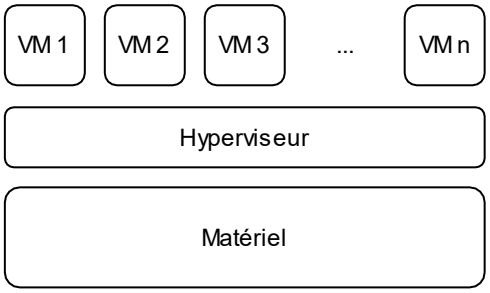


Figure 1.1 : Hyperviseur (*Virtual Machine Monitor*)

On peut distinguer deux types d'hyperviseurs [11] [4] : Type 1 et Type 2.

Type 1 : natif (*bare-metal*)

C'est un logiciel qui s'installe directement sur la machine hôte, et par conséquent, joue aussi le rôle d'un système d'exploitation pour cette machine.

Exemples : VMware ESXi et Microsoft Hyper-V.

Type 2 : hosté (*hosted*)

C'est une application qui a besoin préalablement d'un système d'exploitation pour s'installer, et ne peut en aucun cas jouer le rôle d'un système d'exploitation pour la machine hôte.

Exemples : VMware Workstation et Oracle VirtualBox.

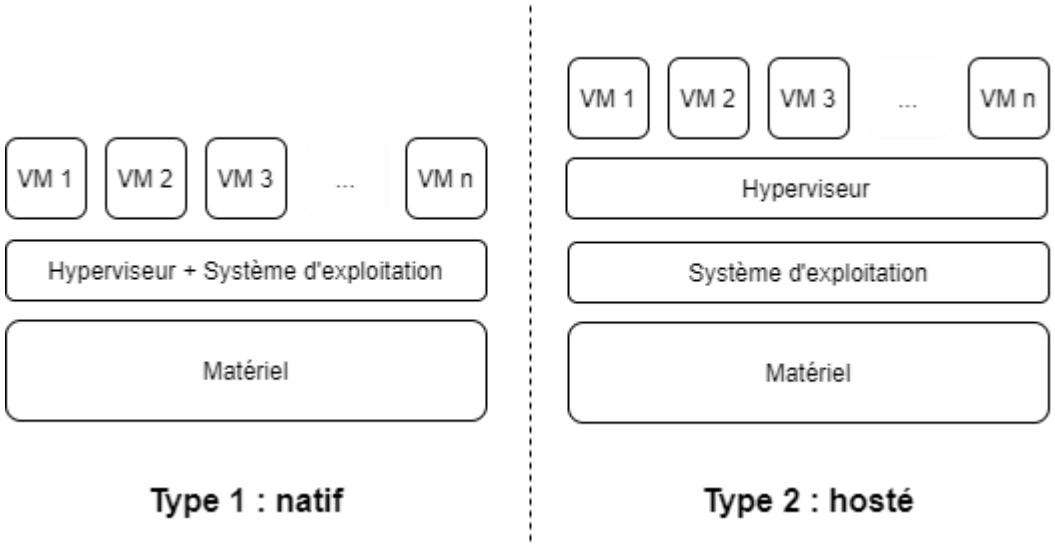


Figure 1.2 : Types d'hyperviseurs

1.2 Techniques de virtualisation

Au fil du temps, plusieurs techniques sont apparues pour améliorer les performances de la virtualisation. Les principales techniques sont : la virtualisation complète, la paravirtualisation et la virtualisation matérielle assistée [4] [10] [12] [13].

1.2.1 Virtualisation complète (*Full Virtualization*)

Pour des raisons de stabilité et de sécurité, la virtualisation complète fait appel à l'émulation pour exécuter certaines instructions du processeur [14]. Cette technique nous permet de faire fonctionner un système d'exploitation sur une machine virtuelle sans aucune modification. Le même système d'exploitation peut aussi fonctionner sur une machine virtuelle que sur une machine physique. Par contre, cette technique limite l'optimisation des performances à cause de l'émulation partielle des instructions matérielles.

1.2.2 Paravirtualisation (*Paravirtualization*)

Le but de la paravirtualisation est d'éviter l'émulation des instructions matérielles et les remplacer par des appels de fonctions (API) implémentées dans l'hyperviseur. Contrairement à la virtualisation complète, le système d'exploitation de la machine virtuelle a besoin d'être modifié pour qu'il puisse utiliser les API de l'hyperviseur. Par contre, on gagne plus de performances, car l'hyperviseur ne fait plus appel à l'émulation pour exécuter le code de l'invitée [2].

1.2.3 Virtualisation matérielle assistée (*Hardware-Assisted Virtualization*)

Cette technique repose sur l'utilisation de nouveau jeu d'instructions du microprocesseur qui permet à l'hyperviseur d'exécuter plusieurs systèmes d'exploitation en même temps sur le même microprocesseur. L'hyperviseur utilise ce jeu d'instructions à la place de l'émulation ou les appels API.

Grâce à cette technique, on obtient des performances meilleures que celles obtenues par la virtualisation complète ou par la paravirtualisation. En outre, aucune modification sur le système d'exploitation invité n'est nécessaire pour le virtualiser avec cette technique. Intel

utilise la technologie VT-x qui représente leur solution de virtualisation matérielle assistée pour l'architecture x86, alors que AMD utilise la technologie AMD-V.

1.2.3.1 KVM

En 2006, la compagnie Qumranet développe KVM (Kernel-based Virtual Machine), un module pour le noyau Linux qui permet à ce dernier d'utiliser les nouveaux jeux d'instructions de virtualisation de l'architecture x86 [15] [16]. Par exemple, le VT-x d'Intel et le AMD-V d'AMD. En 2008, Red Hat annonce l'acquisition de Qumranet [17]. En 2012, elle annonce l'abandon de Xen en faveur de KVM pour ses produits de virtualisation [18]. KVM permet de transformer un système d'exploitation dont le noyau est Linux en un hyperviseur de type 1 lorsqu'on charge son module du noyau (`kvm.ko`) [19].

1.3 Émulation matérielle (*Hardware Emulation*)

Une machine virtuelle a besoin de plusieurs composants : du stockage (disque dur), une carte réseau, des contrôleurs pour les entrées/sorties comme le clavier et la souris, etc. L'émulation matérielle permet de simuler le fonctionnement d'un périphérique matériel, même s'il n'est pas présent physiquement sur la machine hôte [4]. Ainsi, un fichier qui se trouve sur un système de fichier accessible par la machine hôte, peut être vu comme un disque dur (virtuel) par la machine virtuelle. De même, une partition physique ou logique peut être exposée pour une machine virtuelle comme un disque dur. La machine virtuelle peut effectuer toutes les opérations systèmes et d'E/S qu'on peut faire sur un disque physique. Dans le contexte de la virtualisation basée sur KVM, l'hyperviseur utilise le logiciel QEMU qui permet d'émuler les différentes composantes matérielles de la VM, notamment le disque virtuel.

1.3.1 QEMU (Quick EMUlator)

En 2003, Fabrice Bellard développe QEMU (Quick EMUlator), un logiciel qui permet d'émuler plusieurs architectures des processeurs comme x86, ARM, PowerPC, Sparc et MIPS [20]. Il permet aussi d'émuler plusieurs composantes matérielles ou périphériques comme les disques durs, les cartes réseau, les contrôleurs USB, etc.

En combinant KVM et QEMU, on obtient une plateforme de virtualisation complète offrant des performances proches de celles obtenues avec du matériel physique sans virtualisation. Dans cette plateforme, KVM s'occupe de la virtualisation du microprocesseur en exploitant les jeux d'instructions comme VT-x et de la mémoire, tandis que QEMU émule le reste des composantes.

1.3.2 Formats de stockage

QEMU prend en charge de nombreux formats d'image disque pour émuler le périphérique du stockage de la machine virtuelle. Chaque format a ses avantages et ses inconvénients. En voici les plus utilisés [21]:

- RAW : comme son nom l'indique, c'est un fichier qui émule le contenu d'un disque dur.
- QCOW2 : QEMU Copy On Write version 2. Utilise la technique d'optimisation « copie sur écriture » et l'allocation dynamique.
- VMDK : Virtual Machine Disk. Le format utilisé par VMware.
- VDI : Virtual Disk Image. Le format utilisé par Oracle VirtualBox.
- VHD : Virtual Hard Disk. Le format utilisé par Microsoft Hyper-V.
- HDD et HDS : Le format utilisé par la compagnie Parallels.
- FVD : Fast Virtual Disk créé par IBM pour QEMU.

Le format RAW est celui qui offre les meilleures performances [3]. Par contre, il n'offre que peu de fonctionnalités par rapport aux autres. Cet essai se base sur une étude comparative des performances d'E/S, par conséquent, c'est le format RAW qui est choisi pour cette étude.

1.3.2.1 Format RAW

Le format RAW, tel que géré par QEMU, est un fichier qui émule le contenu d'un disque dur secteur par secteur. Ce fichier est généralement hébergé sur un support de stockage physique accessible par le système hôte.

C'est aussi le format le plus simple pour émuler un disque virtuel. Les E/S se font simplement via un mappage entre les blocs du disque virtuel et les blocs du stockage physique [22]. On s'attend donc à obtenir les meilleures performances E/S en utilisant ce format. En revanche,

ce format n'offre que peu de fonctionnalités par rapport aux autres formats du QEMU, tels que l'allocation dynamique ou les instantanés (*snapshot*).

KVM peut aussi utiliser directement une partition physique ou un volume logique (LVM) comme espace de stockage pour les machines virtuelles. Un cas particulier d'un volume LVM est celui en mode allocation dynamique qui est présenté dans la section suivante. C'est le deuxième format de stockage qui sera étudié dans cette étude comparative.

1.3.2.2 LVM

LVM (*Logical Volume Manager*) est une méthode d'allocation d'espace sur les périphériques de stockage basé sur une couche d'abstraction logique qui permet une gestion plus flexible des espaces alloués [23] [24]. On peut, par exemple, étendre un espace, le rétrécir et regrouper des disques ou des partitions dans un seul volume de stockage [25] [26] ; en outre, il prend en charge plusieurs fonctionnalités telles que les instantanés (*snapshot*) et l'allocation dynamique. La gestion des volumes avec LVM n'est qu'une forme des nombreuses formes de virtualisation du stockage [27], et comme toute couche additionnelle au système, on s'attend, en conséquence, à une diminution des performances.
L'espace de stockage d'un volume LVM peut être alloué de deux façons : allocation fixe ou allocation dynamique. Avec l'allocation fixe, la taille totale de la capacité de stockage du volume est allouée sur le stockage physique lors de la création du volume. Dans ce cas, le volume occupe tout l'espace qui lui est alloué dans le stockage physique dès le début de sa création et, par conséquent, l'espace non utilisé de ce volume ne peut être considéré comme espace libre qu'on peut allouer à d'autres volumes. L'inconvénient de cette allocation est que l'utilisation de la capacité totale du stockage n'est pas optimale, car on peut se trouver dans une situation où un espace est alloué sans jamais être utilisé.

Contrairement à l'allocation fixe, l'allocation dynamique, appelée aussi « allocation fine et dynamique », permet une occupation progressive de l'espace sur le stockage physique dont le volume a réellement besoin. Le volume ne consomme que l'espace dont il a besoin initialement et croît avec le temps en fonction du besoin. L'espace alloué, mais non utilisé par le volume, est considéré comme espace libre qu'on peut allouer à d'autres volumes.

Il nous permet même d'allouer une capacité de stockage supérieur à la capacité totale physique disponible. Le danger c'est que si l'espace disque réel est complètement utilisé, toute écriture supplémentaire peut entraîner un dysfonctionnement de toute la plateforme. Par précaution, une surveillance minutieuse de la consommation de l'espace physique disponible est cruciale. L'un des inconvénients de ce mode d'allocation c'est que les performances diminuent encore plus pendant l'allocation d'un nouvel espace pour le volume. Néanmoins, cette diminution ne dure que pendant élargissement du volume. Autre cause de diminution de performances peut apparaitre si la fonctionnalité de la mise à zéro (*zeroing*) est activée [28]. L'allocation dynamique est un concept général qui n'est pas propre à l'utilisation de LVM. On le trouve aussi dans certains formats de stockage tels que VMDK de VMware [29] et QCOW2 de QEMU [30].

1.4 Tests et mesures de performance

Le but d'un test de performance (*Benchmark*) est de mesurer certaines propriétés d'un système particulier. Cette étude utilise particulièrement les tests de performance des entrées/sorties (E/S) des supports de stockage virtuel. Ces E/S se font par des lectures et écritures sur les supports. On distingue plusieurs types de lectures et écritures dont les plus communs sont décrits dans le Tableau 1.1 [31] :

Tableau 1.1 : Types de tests IOPS

Mesure	Description
Lecture aléatoire (*Random Read*)	Nombre moyen d'opérations de lecture en accès direct par seconde
Écriture aléatoire (*Random Write*)	Nombre moyen d'opérations d'écriture en accès direct par seconde
Lecture séquentielle (*Sequential Read*)	Nombre moyen d'opérations de lecture séquentielle par seconde
Écriture séquentielle (*Sequential Write*)	Nombre moyen d'opérations d'écriture séquentielle par seconde

Les unités de mesure les plus utilisées sont : les opérations d'entrée/sortie par seconde (IOPS), le débit/bande passante (*Throuhput/Bandwidth*) et la latence (*Latency*) [3] [32] [22] [33] [17].

1.4.1 Opérations d'entrée-sortie par seconde

Input/Output Operations per Second (IOPS) en anglais. Cette unité de mesure représente le nombre maximum de lectures et d'écritures que le support de stockage peut effectuer par seconde.

1.4.2 Débit (Bande passante)

Cette unité de mesure est liée à la précédente (*IOPS*). Pour comprendre cette liaison, il faut introduire la notion de taille de bloc.

La taille de bloc (*Block Size*) est la plus petite taille que le système de fichier peut allouer. Lorsque le système effectue une lecture ou écriture, l'opération se fait en un multiple entier de

taille de bloc. Les tailles de bloc courantes sont 512 octets, 1 Ko, 2 Ko, 4 Ko, 8 Ko, 16 Ko, 32 Ko et 64 Ko. Ainsi, le débit est défini par la formule suivante :

$$\text{Débit} = \text{IOPS x Taille de bloc}$$

Il se mesure généralement en kilo-octets par seconde (Ko/s) ou méga-octets par seconde (Mo/s).

1.4.3 Latence

En général, la latence mesure le temps entre l'envoi d'une requête et la réception d'une réponse. C'est aussi le temps minimum dont le système a besoin pour effectuer une opération E/S. Elle est souvent mesurée en millisecondes (ms) ou microsecondes (µs).

1.5 Virtualisation et performances E/S

Les machines virtuelles partagent les mêmes ressources matérielles de la machine hôte. Ce partage est géré adéquatement par l'hyperviseur qui assure aussi l'isolement de chaque invité des autres. Comme il fallait s'y attendre, ce partage a un impact sur les performances de toutes les ressources comme le processeur, la mémoire, les E/S, le réseau, etc. Au fur et à mesure que la technologie le permet, plusieurs techniques ont été introduites pour réduire la dégradation des performances.

Le sujet de cette étude se concentre sur les performances E/S des disques virtuels des machines. Parmi les améliorations qu'on peut leur apporter :

1. Choisir le bon format de stockage.
2. La paravirtualisation des interfaces de connexion pour les mémoires de masse comme VirtIO [34].
3. L'utilisation d'une partition physique ou un volume logique.

Le choix de ces améliorations est imposé par le contexte du projet de la mise en place de l'infrastructure de virtualisation. Le choix d'utiliser des partitions physiques ou logiques, comme disques virtuels pour les VM, n'est pas toujours possible. Dans le cas où c'est possible, la 3e amélioration devient envisageable, et le volume logique constitue possiblement une bonne

alternative au format de stockage basé sur des fichiers images enregistrés sur un système de fichier, tout en apportant leur flexibilité et leur performance. Dans cet essai, on présume, bien sûr, que le choix d'utiliser des volumes logiques est possible. Le prochain chapitre présente les travaux effectués à jour pour comparer les performances des différents formats de stockages virtuels en focalisant sur ceux qui traitent principalement le format de fichier image RAW et le volume logique LVM en mode allocation dynamique.

Chapitre 2

Revue de la littérature

Ce chapitre résume l'état des connaissances dans le domaine de la virtualisation surtout celles en rapport avec le sujet d'étude, à savoir, les performances des formats de stockages. Il présente les résultats de quelques articles pertinents à cette étude. Certains d'entre eux sont plus intéressants que d'autres dépendamment de l'objectif de chaque test par rapport au sujet de cette recherche. Néanmoins, les méthodologies et les expériences qu'on trouve dans ces articles sont d'une grande importance pour l'élaboration de la problématique et les approches à préconiser pour la résoudre.

2.1 Méthodologie de recherche

Cette revue s'appuie sur des articles de revues ou de conférences, des livres de référence, des travaux académiques et divers documents électroniques. Les services utilisés pour les trouver sont :

- Institute of Electrical and Electronics Engineers (IEEE Xplore).
- Association for Computing Machinery (ACM Digital Library).
- Service des bibliothèques et archives de l'Université de Sherbrooke.
- Service des bibliothèques de l'Université de Montréal (Atrium).
- arXiv.
- Google Scholar.

Les mots-clés utilisés dans les requêtes de recherches sont composés d'un ou plusieurs mots (combinaisons) parmi les mots suivants :

- KVM
- QEMU
- LVM
- Thin provisioning

- Virtual Storage
- Virtualization IO
- Virtual Hard Disk Performance
- Virtual Storage Benchmarking

L'utilisation des mots-clés « KVM », « QEMU » et « LVM » a permis d'obtenir environ 7600 d'articles et documents. La majorité sort du champ d'intérêt de cette étude. Les résultats sont d'abord filtrés par sujet, langue et année de publication. Ensuite, d'autres mots-clés supplémentaires ont été utilisés, comme « Thin provisioning » et « RAW », pour raffiner les résultats et ne garder que les articles pertinents à cette étude.

2.2 Performances du RAW

Grâce à sa simplicité, qui consiste à présenter le contenu d'un disque virtuel secteur par secteur via un mappage simple et direct entre les blocs du disque virtuel et les blocs physiques [22], le format RAW nous permet d'obtenir les meilleures performances E/S. C'est ce qui a été démontré dans plusieurs études [3] [32] [22].

Joshi et al [3] ont étudié la charge de travail d'E/S sur un système de fichiers local, c'est-à-dire un support de stockage à connexion directe (DAS). Les résultats montrent que le choix du format de stockage peut augmenter considérablement les performances des opérations d'E/S. On trouve, par exemple, que le format RAW excelle généralement dans les opérations de lecture séquentielle et aléatoire indépendamment de la taille du bloc. C'est aussi le format qui détient la plus basse latence. Les formats de stockage testés sont RAW, QCOW2, VHD, VDI, VMDK et HDD. Les unités de mesure utilisées sont IOPS, le débit et la latence.

Malheureusement les graphiques n'incluent pas les mesures des E/S au niveau physique (sans virtualisation), car on aurait pu évaluer aussi les pertes de performances causées par chaque format par rapport aux maximales offertes sans virtualisation du stockage. De même, les volumes logiques (LVM) n'ont pas été inclus dans cette étude, parce que l'étude focalise sur les disques virtuels basés sur des fichiers hébergés sur le système de fichier de l'hôte. Ces résultats restent néanmoins très intéressants parce que les tests ont été effectués sur un DAS, et c'est le cas aussi dans cet essai.

Zhang et al [32] ont fait une évaluation, puis ont proposé une optimisation pour améliorer les E/S sur une plateforme de virtualisation KVM. Les résultats montrent qu'en lecture séquentielle le débit du format RAW est très proche de celui du mode natif, alors qu'en écriture séquentielle le débit du format RAW est environ la moitié de celui du mode natif. Les tests sont plus globaux puisqu'ils incluent aussi d'autres E/S comme celles du réseau et du CPU. RAW est le seul format utilisé dans leurs tests.

Contrairement à l'étude précédente, l'avantage de cette étude c'est qu'elle nous donne une idée sur le comportement du format RAW par rapport au mode natif (sans virtualisation). Ainsi, on peut évaluer le pourcentage de perte de performance due à son utilisation par rapport au maximum disponible, mais aussi, penser à d'éventuelles solutions pour réduire cette perte, c'est ce que les auteurs proposent en fusionnant les instructions d'E/S successives et en supprimant les opérations redondantes dans le système invité.

L'auteur de [22] a étudié les performances du format FVD dans le contexte de l'utilisation dans l'infonuagique. Malgré que son article se penche plus sur le format FVD, en trouve le RAW parmi les formats comparés. Les résultats montrent que ce dernier performe seulement 2,4 % moins que le mode natif pour les E/S aléatoires. Ils montrent aussi que la machine virtuelle performe 63 % mieux lorsque son disque virtuel utilise une partition du disque physique au lieu d'un fichier RAW sur le système de fichier ext3. Le fichier RAW est stocké sur le disque physique local (DAS). On constate aussi que l'utilisation de l'interface VirtIO augmente le débit de 35 % par rapport à l'interface classique IDE.

Toutefois, l'impact de la taille de bloc sur les performances n'a pas été examiné dans cet article, puisqu'aucune indication concernant cette variable n'a été citée, et on présume que ces résultats sont valides pour la taille de bloc par défaut de la plateforme de l'infonuagique d'IBM. Pour résumer, on peut conclure de cette section que le format RAW offre les meilleures performances parmi les autres formats prise en charge par QEMU, et que ces performances sont très proches de celles obtenues sans virtualisation.

2.3 Performances du LVM

LVM utilise à peu près le même principe que le format RAW pour virtualiser le stockage, à savoir, le mappage d'adresses entre le volume logique et le support physique. Ce mappage se fait entre les extensions physiques (PE) et les extensions logiques (LE) [24].

On a vu dans le premier chapitre que le principal avantage de l'utilisation des volumes LVM est leur flexibilité au niveau de la gestion de la capacité de stockage et de l'approvisionnement [23] [25]. Néanmoins, on devrait s'attendre à une diminution au niveau de performances à cause de cette couche de virtualisation additionnelle que les échanges doivent passer à travers pour transmettre de l'information au stockage physique. En effet, une étude réalisée en 2015 a examiné les performances des volumes LVM suivant plusieurs configurations [35]. Les auteurs démontrent par des calculs théoriques et tests expérimentaux qu'un volume LVM performe toujours moins qu'un disque ou une partition physique. En plus, les résultats montrent que plus la configuration de LVM est complexe plus les performances diminuent. Ils ont comparé le débit E/S dans trois cas, le premier sans l'utilisation de LVM, le deuxième avec un LVM configuré sur deux partitions physiques et le dernier avec un LVM configuré sur une seule partition physique.

Ils ont trouvé que le volume logique du deuxième cas a des performances inférieures à celui du troisième cas. En outre, pour les petits fichiers, une partition physique sans LVM a des performances supérieures de 63 % par rapport à un LVM configuré sur une seule partition physique, et de 16 % pour les fichiers plus larges. Les auteurs recommandent clairement d'éviter les LVM configurés sur un grand nombre de disques ou partitions physique, et les remplacer avec des LVM configurés le plus simplement possible pour éviter la dégradation de performances. Malgré le fait que leur étude est limitée à une comparaison du comportement de LVM par rapport au mode natif, l'idée d'utiliser un volume LVM au lieu d'un fichier image pour améliorer davantage les performances d'une VM est envisagée dans autres études [17].

L'objet de cet essai est de comparer deux formats possibles pour la virtualisation des disques des VM dans leurs configurations les plus optimales possible. Ainsi, les résultats de cette étude sont très intéressants parce qu'ils font la lumière sur des situations à éviter dans l'expérimentation et qui sont dues à de simples mauvaises configurations. On conclut de cette

étude que le volume logique performe moins avec les petits fichiers et avec les configurations complexes. Cela a orienté l'étude de l'essai en évitant ces deux cas lors de la conception de l'expérimentation. En effet, le volume a été configuré sur une seule partition physique, et la taille de fichier des tests à 1Go.

2.4 Interfaces de connexion

Dans une plateforme de virtualisation KVM, on peut utiliser plusieurs interfaces de connexion pour les disques virtuels comme IDE et SATA. Pour cela, QEMU procède à de l'émulation pour simuler leur fonctionnement comme dans le cas de n'importe quel matériel physique. Le problème, tel que mentionné dans le premier chapitre, c'est que l'émulation cause des pertes de performances significatives. Une des solutions envisageables est le recours à la paravirtualisation pour éviter ces pertes causées par l'émulation. C'est précisément ce que le pilote VirtIO nous permet de faire [34]. C'est l'interface, parmi les autres, qui nous permet d'avoir les meilleures performances E/S. L'une des études qui le confirment était réalisée en 2013 par Bujor et Dobre [33]. Cette étude est pertinente pour cet essai parce qu'elle inclut aussi une comparaison entre les formats RAW et LVM. L'objectif principal de cette étude n'était pas de faire une comparaison directe entre ces deux formats, mais plutôt de vérifier si un volume LVM peut améliorer les performances lors de l'utilisation de l'interface VirtIO pour les échanges de données entre les invités et l'hôte. Les 3 formats testés sont : RAW, QCOW et LVM.

Ils ont testé les performances pour les 3 interfaces IDE, SCSI et VirtIO. La conclusion est que cette dernière est celle qui donne la plus grande optimisation disponible. Dans ce cas, l'utilisation d'un volume LVM, qui est censé améliorer encore les performances, n'apporte aucun gain substantiel. Quant aux systèmes de fichiers, les résultats montrent que les formats ext4 et reiserfs sont les plus adéquats pour héberger les fichiers images (RAW ou QCOW2).

Pour résumer, cette étude confirme que le VirtIO reste la meilleure option à choisir pour interfacer les disques virtuels et que dans le cas de l'utilisation du format RAW, le système de fichier ext4 est l'un des meilleurs choix possible pour les héberger. Une amélioration était souhaitable si les auteurs avaient inclus des tests plus détaillés concernant les latences et les débits.

Une autre étude réalisée en 2013 [36] qui montre, d'une part, que le fichier RAW est le format qui donne les meilleures performances parmi les formats fichiers testés, et d'autre part, qu'il performe moins qu'un volume LVM en mode allocation fixe. Elle présente des résultats de tests de performance E/S effectués sur une plateforme de virtualisation KVM. Les formats comparés sont RAW, QCOW2, volume logique (LVM) en allocation fixe et le mode natif. Ce dernier sert comme un repère pour les comparaisons et représente une performance de 100%, ce qui est très pratique. Il est égal au maximum théorique qu'on peut estimer avoir. Le volume logique en mode allocation dynamique n'est pas inclus dans cette étude. Les résultats montrent que dans tous les cas, le volume logique est plus performant que le format RAW, mais dans le cas des E/S aléatoires la différence devient minime. L'ajout de mode natif comme une référence dans les tests était une bonne idée, car elle permet aussi de comparer les performances du format RAW et volume LVM individuellement par rapport au maximum offert par les ressources de l'hôte. Ces résultats sont importants pour l'étude de cet essai, car ils nous montrent que dans tous les cas, le volume logique en allocation fixe est plus performant que le fichier image RAW. Mais, le volume logique en mode allocation dynamique performe légèrement moins qu'un volume logique en allocation fixe à cause de la mise à zéro et au principe du fonctionnement de l'allocation dynamique (voir section 2.5). Reste à savoir si ces deux causes vont aller jusqu'à faire dégrader les performances du volume logique en allocation dynamique par rapport à celles du fichier image RAW. Ces résultats ont orienté surtout la question et l'hypothèse de l'essai. La quantification de la question de la recherche a été extrapolée à partir de ces résultats. D'autre part, les performances du volume logique en allocation fixe et le fichier image RAW ont été comparées avec celles du mode physique (natif) avec des rapports en pourcentages. Cela constitue un moyen additionnel pour valider les résultats obtenus de l'expérience de l'essai.

2.5 Allocation dynamique

L'allocation dynamique est un concept général qui n'est pas propre à LVM. Il existe deux causes principales qui peuvent causer une baisse de performance lors de l'utilisation de l'allocation dynamique pour un disque virtuel [29] :

- L'allocation d'un nouvel espace.

- La mise à zéro (*zeroing*) de ce nouvel espace.

La première cause se produit durant l'élargissement de la capacité de l'espace, car l'allocation sur le support physique réel se fait à ce moment-là. Donc, une diminution de performances devrait être observable à cause de l'écriture supplémentaire due à la réservation de l'espace additionnel demandé sur le support physique. Dans le cas où aucun élargissement de la capacité de l'espace virtuel n'est requis, le volume devrait avoir les mêmes performances que celles obtenues dans l'allocation fixe.

La deuxième est liée à la première parce que, pour des raisons de sécurité, lorsqu'un disque virtuel se procure de nouveaux blocs, leurs contenus sont d'abord remplacés par des zéros. Cela veut dire encore une charge de travail supplémentaire qui impacte les performances.

La mise à zéro n'est pas obligatoire et peut être désactivée, mais la perte de performance due à la création de nouveaux blocs est inévitable, parce que c'est une conséquence du principe du fonctionnement de l'allocation dynamique lui-même. Néanmoins, dans un contexte où la capacité du volume virtuel n'est pas censée d'augmenter fréquemment, l'allocation dynamique constitue une solution rentable.

Ce qu'on peut conclure de cette revue c'est qu'avec tous les résultats des différents tests de performance recensés dans ce chapitre, le ficher RAW, en général, reste le meilleur choix pour le disque virtuel d'une VM pour avoir le maximum de performance des E/S. Dans le cas d'utilisation d'une partition physique au lieu d'un fichier, une partition ext3 ou ext4 les performances E/S obtenues sont encore meilleures, alors qu'un volume logique LVM constitue une bonne alternative avec certaines configurations.

Malheureusement, aucune étude n'a été trouvée qui compare les formats RAW et LVM en mode allocation dynamique. L'objet de cette étude est de combler cette lacune, en proposant une méthodologie expérimentale inspirée des diverses études citées dans ce chapitre.

On a vu dans le premier chapitre qu'un volume logique en mode allocation dynamique offre plus de flexibilité concernant la gestion de l'espace total alloué. On peut mener une réflexion de savoir si l'activation de ce mode n'affecte pas considérablement les performances E/S de la VM. Dans ce cas, peut-on garder des performances meilleures que celles d'un fichier RAW? Le prochain chapitre pousse cette réflexion en soulevant cette problématique.

Chapitre 3

Problématique

Ce chapitre expose la problématique concernant le choix du format de stockage utilisé dans une plateforme de virtualisation basée sur KVM. Le choix n'est pas univoque puisque chaque format a des avantages et des inconvénients. Les spécifications du projet de mise en place de l'infrastructure de virtualisation motivent le bon choix.

On a vu, dans le chapitre précédent, que le fichier RAW et le volume LVM en allocation fixe offrent de bonnes performances. En outre, un volume LVM en allocation fixe performe mieux qu'un fichier RAW. L'utilisation du fichier RAW se fait au détriment de la flexibilité de gestion de stockage et de la fonctionnalité de l'allocation dynamique. En contrepartie, il demeure le format de type fichier qui offre les meilleures performances parmi les autres. Contrairement à RAW, LVM offre plus de flexibilité pour la gestion de l'espace ; notamment avec l'allocation dynamique. L'utilisation de ce mode d'allocation a un impact sur les performances. La question qui se pose c'est de savoir si on peut, par l'utilisation d'un LVM en mode allocation dynamique, garder des performances encore meilleures que celles d'un fichier RAW, tout en bénéficiant de la flexibilité en ce qui concerne la gestion de l'espace.

3.1 Description

Lors de la mise en place d'une infrastructure de virtualisation, le stockage représente l'un des défis les plus importants à relever. En effet, le support physique du stockage global disponible est partagé par plusieurs machines virtuelles. Une fois ce support est installé et configuré, une optimisation de cette ressource s'avère très utile pour éviter tout impact négatif sur les machines virtualisées. Ces dernières devraient être capables d'y accéder et de l'utiliser sans aucune contrainte qui peut nuire à leurs performances.

Les premières optimisations sont effectuées sur le matériel physique. D'abord, par le choix de type du support, par exemple, magnétique (HDD) ou mémoire flash (SSD). Ensuite, par le choix de type d'attachement, par exemple, local (DAS) ou réseau (NAS et SAN).

Les optimisations suivantes portent sur le niveau logique par des techniques de virtualisation du stockage comme, par exemple, RAID et LVM. Ou par le type d'approvisionnement comme l'allocation fixe et dynamique qu'on a présenté dans le premier chapitre.

Quant aux disques virtuels des VM, plusieurs formats sont utilisés dépendamment des fonctionnalités dont on a besoin et les objectifs qu'on veut atteindre. Plusieurs d'entre eux, de la famille QEMU, ont été présentés dans le premier chapitre. Parmi eux, le format RAW qui est généralement considéré comme celui qui offre les meilleures performances E/S [3] [33].

L'utilisation d'un format de stockage basé sur un fichier n'est pas obligatoire. On peut directement utiliser une partition physique ou logique comme disque virtuel pour l'invité. Cela permet de réduire la couche de virtualisation responsable du transfert des données entre le contenu du fichier et le système de fichier de l'hôte. Un volume LVM offre plus de flexibilité par rapport à une partition physique. Aussi, ces performances E/S restent proches de celles d'une partition physique [35] [36] lorsque sa configuration n'est pas complexe.

On a vu que le concept de l'allocation dynamique (*Thin Provisioning*) n'est pas propre aux volumes LVM et que d'autres formats de stockage prennent en charge ce mode d'approvisionnement. En général, les performances en mode d'allocation dynamique sont légèrement inférieures à celles en mode d'allocation fixe. Mais en améliorant certaines configurations, comme la désactivation de la propriété de remise à zéro (*zeroing*), on peut avoir presque les mêmes performances pour les deux modes [29] [28]. En conclusion, on a deux moyens de stockages qui offrent de bonnes performances proches de celles obtenues sur du matériel non virtualisé : RAW et LVM en allocation dynamique.

3.1.1 Objectifs et hypothèses

On a, d'une part, le format RAW qui est un simple fichier représentant le contenu du disque virtuel de la VM secteur par secteur qui nous n'offre que peu de fonctionnalités par rapport aux autres formats [21], mais comme mentionné plus haut, offre de bonnes performances. D'autre part, un volume LVM offre à son tour de bonnes performances, mais aussi la fonctionnalité de

l'approvisionnement intelligent grâce à l'allocation dynamique. Mais qu'en est-il des performances du format RAW par rapport au volume LVM en allocation dynamique ?

Les résultats de [36], nous montrent que le volume LVM en allocation fixe performe 16,66 % plus qu'un fichier RAW. Or, les performances d'un volume LVM en allocation dynamique sont inférieures à celles d'un volume LVM en mode allocation fixe. Cette dégradation se manifeste surtout pendant l'élargissement du volume. Donc, on peut estimer que si le volume LVM en allocation dynamique performe mieux qu'un fichier RAW, alors cette amélioration ne peut dépasser approximativement le taux d'amélioration de 16,66 % effectué par un volume LVM en mode fixe.

Ce qui est intéressant c'est de savoir si cette stratégie apporte réellement les améliorations attendues. D'où la question de recherche suivante : **L'utilisation d'un volume LVM en mode allocation dynamique comme moyen de stockage pour une VM permet-elle d'améliorer les performances d'au plus de 16,66 % par rapport à l'utilisation du format RAW** ? L'hypothèse de cette étude stipule que, oui, l'utilisation du LVM en mode allocation dynamique permet d'avoir des performances supérieures à celles obtenues par l'utilisation de mode RAW.

3.1.2 Limites

La recherche se base sur une étude comparative de deux formats de stockage. Elle focalise seulement sur les performances E/S de ces derniers. Les autres mesures telles que la bande passante réseau, l'utilisation de la mémoire, etc. ne seront pas incluses dans cette étude. Les disques virtuels des VM peuvent être hébergés sur différents types de supports :

- Local : SATA, M.2, USB, etc.
- Réseau : NAS, SAN, etc.
- Distribué : Ceph, GlusterFS, etc.
- Autres.

L'expérience de cette étude est inspirée des travaux effectués par Joshi et al [3], Tan [22] et Huynh [36]. Pour ce qui concerne le type de support, l'étude sera limitée au stockage local de type local (DAS). En plus, la configuration sera simple sans aucune couche logique additionnelle, comme le RAID par exemple.

L’impact du choix d’un système de fichier particulier sur les performances E/S sera exclu et le seul système de fichier utilisé est ext4. De même, la variation des performances en fonction de la taille de données (fichiers) ne sera pas incluse; en revanche, la même taille de données sera utilisée dans tous les tests des formats de stockages. Tous les tests concernant le format LVM en mode allocation dynamique se feront sans la mise à zéro (*zeroing*).

3.2 Méthodologie proposée

L’expérience consiste à utiliser un serveur physique muni d’un hyperviseur KVM. Une machine virtuelle est créée dont le disque virtuel représente le format de stockage RAW ou le volume logique LVM en mode allocation dynamique. Des tests multiples de performances sont lancés pour obtenir les performances E/S de chaque format. La moyenne est calculée pour chacun d’eux et les résultats sont ensuite comparés pour confirmer ou infirmer l’hypothèse de l’étude.

3.2.1 Type de recherche

L’expérimentation repose sur une recherche quantitative. Le choix du format de stockage représente ici la variable indépendante. Cette dernière peut avoir deux valeurs : format du stockage RAW et LVM en mode allocation dynamique.

On réalise des tests de performance en lecture et écriture. Une comparaison des résultats déterminera quel format permettra d’avoir les meilleures performances E/S. Les valeurs de ces dernières représentent la variable dépendante.

Pour résumer, l’expérimentation permet de voir l’impact du choix du format de stockage de la VM sur les performances E/S de cette dernière. La Figure 3.1 décrit le cadre conceptuel de la recherche :

Figure 3.1 : Cadre conceptuel de la recherche

3.3 Mise en œuvre

De nombreux outils peuvent être utilisés pour effectuer les tests de performance. On peut citer, à titre d'exemple, Flexible I/O Tester (FIO), Bonnie++, Flexible File System Benchmark (FFSB), Postmark et IOzone. Ce dernier sera utilisé dans cette étude. Il permet de mesurer les performances d'E/S décrites dans le Tableau 1.1.

L'expérimentation a nécessité la création d'un laboratoire informatique spécifique à cet effet. Sa description est décrite en détail dans le chapitre suivant. On y trouve aussi la démarche suivie pour mener les tests et traiter les résultats.

Chapitre 4

Approche proposée

Ce chapitre expose les démarches proposées pour répondre à la question de la recherche telle que soulevée dans le chapitre précédent. On y trouve des descriptions détaillées du laboratoire aménagé pour effectuer les tests, du procédé d'expérimentation, de la collecte des données et de la validation et présentation des résultats.

4.1 Description de l'approche

Cet essai se base sur une étude quantitative corrélationnelle prédictive. Une approche constituée de plusieurs étapes est élaborée afin d'affirmer ou infirmer l'hypothèse de cette recherche. Cette approche est composée des étapes suivantes :

1. Mise en place d'un laboratoire informatique configuré et dédié à l'expérimentation.
2. Lancement des tests et recueillement des données.
3. Traitement des données et validation des résultats.
4. Présentation et analyse des résultats.
5. Validation de l'hypothèse.

Les sous-titres de cette section décrivent chacune de ces étapes. La description des détails techniques se trouve dans la section 4.4.

4.1.1 Mise en place du laboratoire

Un laboratoire informatique est aménagé pour réaliser les expérimentations. Il se compose principalement d'un ordinateur de type serveur, qui représente ici le système hôte. Le support physique de stockage est composé d'une seule mémoire de masse de type locale (DAS). Pour réduire davantage la latence, un disque à base de mémoire flash est utilisé au lieu d'un disque dur traditionnel. Il est découpé en trois parties : la première contient les partitions systèmes qui

serviront à héberger l'hyperviseur, la deuxième sert à héberger les volumes LVM et la troisième est destinée principalement pour héberger les fichiers images RAW.

Une carte réseau, qui va servir au début à mettre à jour le système hôte et installer les packages nécessaires via Internet, est désactivée lorsque l'installation et les configurations seront terminées, pour isoler complètement le système de l'extérieur. Cette mesure nous permet d'éliminer toute influence ou interférence causée par des entrées/sorties indésirables et qui peuvent fausser les résultats.

On crée une machine virtuelle dont le disque virtuel est un fichier RAW ou un volume LVM en mode allocation dynamique. Ces derniers possèdent la même taille d'allocation et sont formatés avec le même système de fichier. L'utilisation d'une seule VM nous permet de garantir la disposition des mêmes ressources pour effectuer les opérations de lectures et écritures sur les deux formats de stockage testés ; et ainsi, nous assurer de la qualité des résultats recueillis.

Toute installation ou configuration inutile ou non indispensable à l'expérience est éliminée. Ainsi, aucun système d'exploitation n'est installé sur les disques virtuels des VM. Un système d'exploitation autonome est chargé dans la mémoire vive de chaque VM via un média optique (Live CD). On y trouve aussi les outils de mesures pour lancer les tests. De cette façon, on s'assure que les seules données qui seront lues et écrites sur les disques virtuels viennent de l'outil de mesure. Quant à l'hyperviseur de l'hôte, son installation est minimale avec les configurations recommandées par Red Hat. Une fois le système est isolé et les différents formats de stockage sont prêts pour les tests, le seul moyen d'y accéder se fera via un terminal (TTY) pour l'hôte et la VM.

4.1.2 Lancement des tests et recueillement des données

Les tests se font en deux modes : physique et virtuel. Le mode physique, tel que défini dans cet essai, fait référence à une utilisation classique de la machine physique sans utilisation de la virtualisation. Ce mode sera testé avant tout pour connaître les performances maximales que le support physique peut offrir. Elles serviront comme une référence principale dans les comparaisons qui suivent. Le mode virtuel fait référence à l'utilisation de l'hyperviseur KVM pour tester les deux formats de stockages RAW et LVM en mode allocation dynamique en tant que disques virtuels de la VM.

Afin de s'assurer que les tests dans les deux modes se font dans les mêmes conditions, le même système d'exploitation et le même outil de mesure seront utilisés dans les deux modes. Ainsi, l'hyperviseur n'est pas utilisé dans le mode physique, et sera remplacé par le même système d'exploitation utilisé pour la VM dans le mode virtuel (un Live CD). Le Tableau 4.1 résume les différents formats de stockages testés dans les deux modes.

Tableau 4.1 : Modes et format de stockage testés

Mode	Format de stockage testé
Physique	Partition physique
Virtuel	Fichier RAW
	Volume LVM en mode allocation dynamique

Pour chaque mode, des actions successives sont exécutées pour compléter les tests et recueillir les données.

4.1.2.1 Mode physique

Les actions suivantes sont exécutées dans cet ordre :

1. Démarrer le serveur (physique) en chargeant le système d'exploitation et l'outil de mesure dans sa mémoire vive (RAM).
2. Formater la partition physique (ext4). Par cette mesure, on veut éviter les influences dues à la fragmentation.
3. Lancer les tests en utilisant l'outil de mesure.
4. Enregistrer et sauvegarder les données.
5. Arrêter le serveur.

Au total, trois tests sont effectués dans ce mode, et dans chaque test ces actions sont exécutées.

4.1.2.2 Mode virtuel

Les actions suivantes sont exécutées dans cet ordre. Un seul format de stockage est testé à la fois.

1. Démarrer la VM en chargeant le système d'exploitation et l'outil de mesure dans sa mémoire vive (RAM).
2. Formater le disque virtuel (ext4). Par cette mesure, on veut éviter les influences dues à la fragmentation. En plus, dans le cas du volume LVM, il faut s'assurer que ce dernier n'a pas servi pour des tests antérieurs pour éviter les influences dues à l'élargissement initial du volume, puisqu'il est censé être en allocation dynamique. Donc, à chaque test un nouveau volume LVM est créé et utilisé.
3. Lancer les tests en utilisant l'outil de mesure.
4. Enregistrer et sauvegarder les données.
5. Arrêter la VM.

Comme pour le mode physique, trois tests sont effectués pour chaque VM, et dans chaque test ces actions sont exécutées.

4.1.2.3 Données recueillies

L'outil de mesure de performances utilisé est *IOzone*. C'est un outil qui permet de générer un grand nombre de modèles d'accès (*Access Pattern*) pour collecter des statistiques sur les performances [37]. Ces modèles d'accès sont configurés suivant plusieurs paramètres. En voici quelques-uns :

- Le mode d'accès (lecture séquentielle, écriture séquentielle, lecture aléatoire, écriture aléatoire, etc.).
- La taille de bloc utilisé dans le test.
- La taille de fichier utilisé dans le test.
- Etc.

Le modèle d'accès automatique produit une sortie qui couvre toutes les opérations pour les tailles de blocs et les tailles de fichiers décrites dans le Tableau 4.2.

Tableau 4.2 : Tailles de blocs et de fichiers utilisées par IOzone en mode automatique

Taille de bloc en Ko	4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384
Taille de fichier en Ko	64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384, 32768, 65536, 131072, 262144, 524288

Les combinaisons de toutes les valeurs de ces paramètres fournissent un nombre de modèles d'accès très grand. Par conséquent, une limitation de certaines valeurs de ces paramètres s'impose. On va s'inspirer des travaux de Xu et al [38], Josh et al [3] et Soriga et Barbulescu [17] pour fixer certaines valeurs et obtenir un modèle d'accès adéquat et suffisant pour cette étude. Les données qui seront recueillies par l'outil de mesure de performances pour cette étude sont présentées dans le Tableau 4.3.

Tableau 4.3 : Données des opérations recueillies par l'outil de mesure

Données d'opération	**Option IOzone**
Le débit, l'IOPS et la latence de l'écriture séquentielle	Write
Le débit, l'IOPS et la latence de la lecture séquentielle	Read
Le débit, l'IOPS et la latence de l'écriture aléatoire	Random Write
Le débit, l'IOPS et la latence de la lecture aléatoire	Random Read

La taille de fichier de test sera fixée à 1024 Mo [3]. La taille de bloc prendra toutes les valeurs du Tableau 4.2. Ces données « brutes » doivent être traitées afin de pouvoir les exploiter pour les futures analyses.

4.1.3 Traitement des données et validation des résultats

Les données résultantes des tests effectués précédemment sont traitées par diverses opérations telles que des sommations, des fusions, des calculs de moyennes, etc. Mais avant

ça, on les valide en vérifiant qu'il n'y a pas de grandes différences entre les différentes données des tests pour un format de stockage donné. On s'attend que les écarts entre ces données ne soient pas grands entre un test et un autre, parce que, à part le format testé, toutes les autres ressources restent inchangeables. Dans le cas où des écarts sont détectés, il faut vérifier qu'il n'y a pas d'erreur d'installation ou de configuration entre un test et un autre, ensuite refaire les tests. Après la validation, on calcule la moyenne des données obtenues pour chaque format.

4.1.4 Présentation et analyse des résultats

Aussitôt que les données sont traitées et validées, les résultats sont regroupés dans des tableaux et présentés dans des graphiques. L'outil principal utilisé est un tableur (Excel). Les graphiques nous permettent d'avoir une vue globale qui facilite la comparaison des résultats et leur analyse.

Pour un format de stockage donné, il y aura un tableau pour chaque opération du Tableau 4.3, c'est-à-dire, quatre tableaux par format de stockage. Le Tableau 4.4 représente un exemple pour l'opération d'écriture.

Tableau 4.4 : Exemple résultats obtenus pour l'opération d'écriture

Taille de bloc (Ko)	Débit écriture (Ko/s)	IOPS (Opérations/s)	Latence (µs)
4	39438	23137	0.04
8	72374	20398	0.05
16	124787	17236	0.06
…	…	…	…
4096	386014	189	5.27
8192	409332	98	10.20
16384	414938	49	20.18

Voici la description des colonnes :

- Taille de bloc : représente la taille de bloc utilisée par l'outil de mesure pour effectuer une opération. Elle se mesure en kilo-octet.
- Débit écriture : représente le débit de l'opération en question. Il se mesure en kilo-octet par seconde.
- IOPS : nombre d'opérations par seconde.
- Latence : le temps moyen pour effectuer une opération. Elle se mesure en microseconde.

On peut générer trois graphiques à partir de ce tableau. Ils représentent la variation du débit, de l'IOPS et de la latence en fonction de la taille de bloc. Les Figure 4.1, Figure 4.2 et Figure 4.3 montrent des exemples pour l'opération d'écriture.

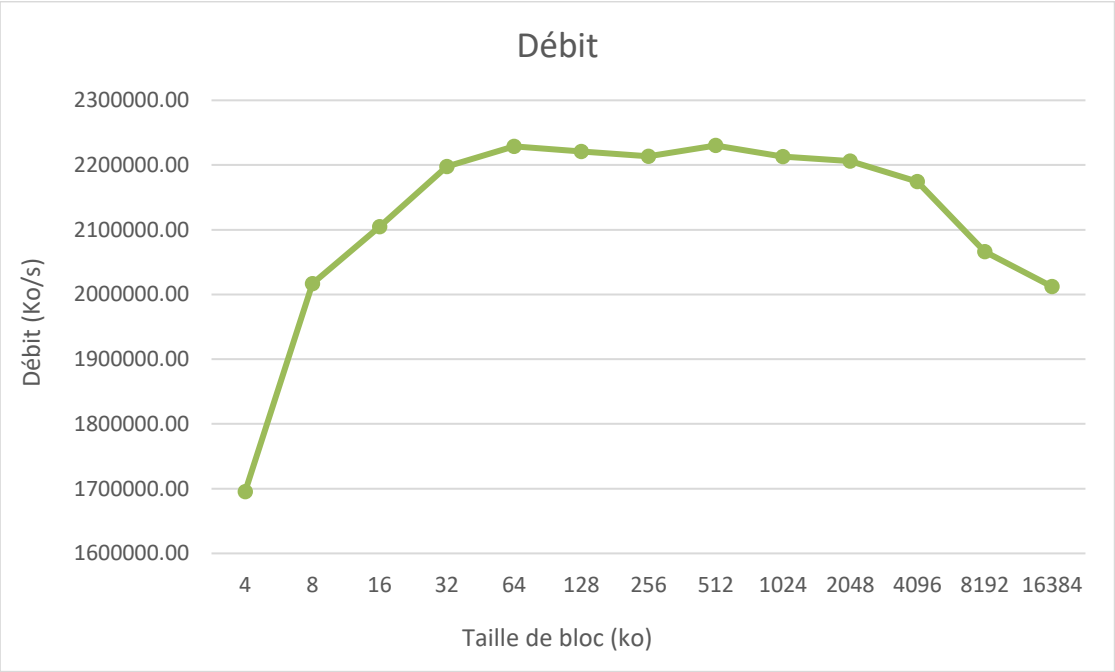


Figure 4.1 : Exemple de présentation du débit

Figure 4.2 : Exemple de présentation de l'IOPS

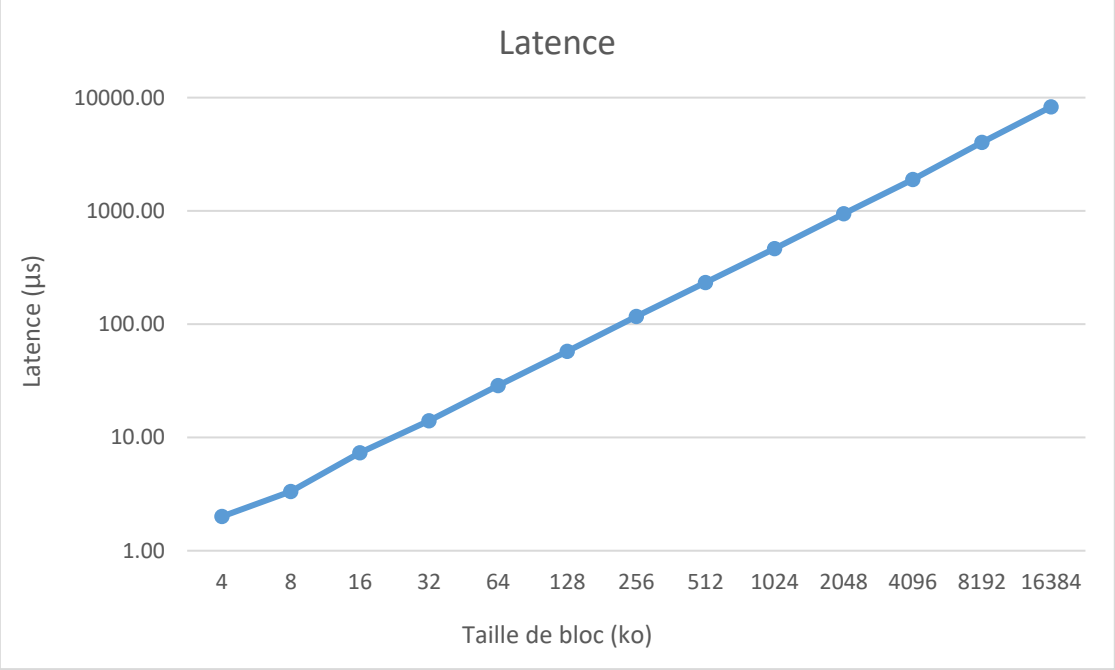


Figure 4.3 : Exemple de présentation de la latence

Au total, il aura 12 tableaux et 36 graphiques pour présenter tous les résultats des tests des différents formats de stockage.

4.1.5 Validation de l'hypothèse

L'hypothèse de cette recherche stipule que l'utilisation d'un volume LVM en allocation dynamique permet d'obtenir des performances meilleures d'au plus de 16,66% que celles obtenues en utilisant un fichier RAW. En formulant mathématiquement cette hypothèse, il faut avoir :

$$1 < \frac{\text{Débit (LVM Thin)}}{\text{Débit (RAW)}} \leq 1{,}1666$$

Dès lors, un tableau représentant les différentes valeurs du rapport (pourcentage) des deux débits en fonction de la taille de bloc s'avère fort utile. Le Tableau 4.5 est un exemple du cas où l'hypothèse est confirmée.

Tableau 4.5 : Rapport des deux débits si l'hypothèse est confirmée

Taille de bloc (Ko)	Débit (LVM Thin) / Débit (RAW) (%)
4	109,02
8	118,05
16	95,52
…	…
4096	102,63
8192	111,69
16384	88,71
Moyenne	**104,27**

Un graphique synoptique, qui représente les débits des trois formats testés, rend facile la comparaison entre les performances de chacun. La Figure 4.4 représente un exemple pour une hypothèse vraie.

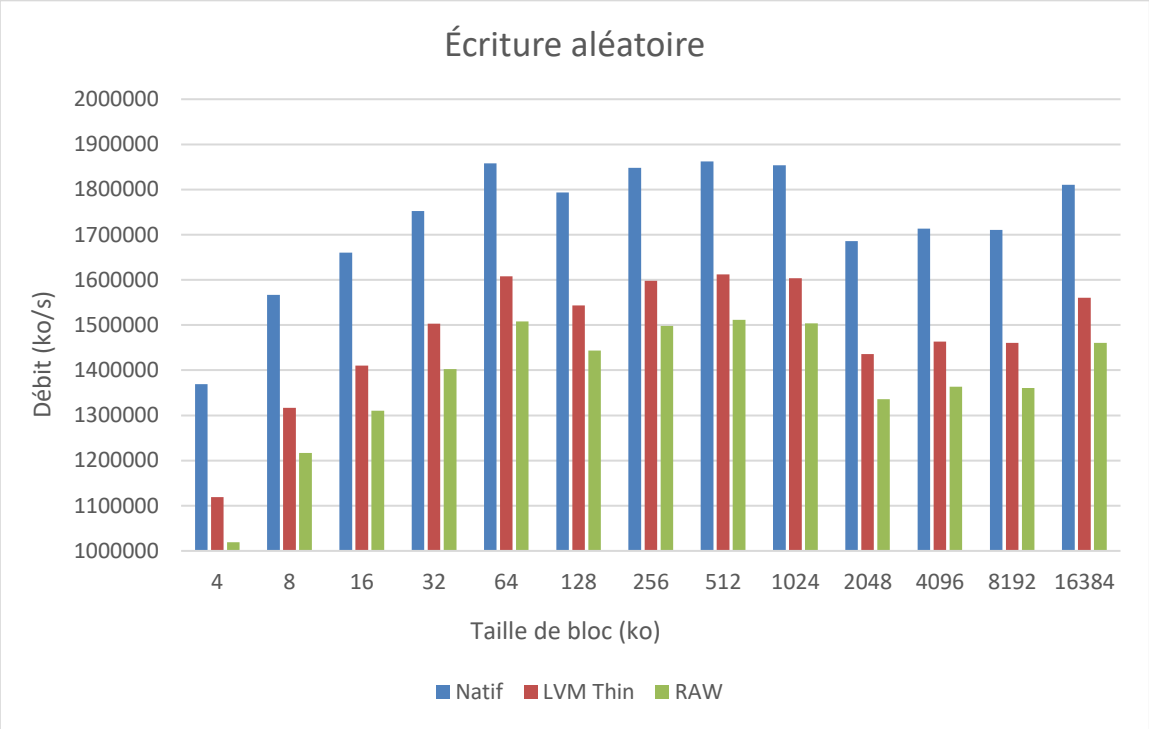


Figure 4.4 : Exemple comparaison des trois débits pour l'opération d'écriture aléatoire

4.2 Validité de l'essai

Plusieurs mesures ont été prises pour assurer la réussite des tests :

- L'isolation du système pendant les tests. Cela va permettre d'éliminer toute activité de l'extérieur, par exemple des mises à jour automatisées, et qui peuvent influencer les tests.
- Exclusion de certains cas particuliers. Par exemple, éviter d'utiliser le même volume LVM pour les trois tests. En effet, durant le premier test, le volume LVM subit un élargissement puisqu'il est censé être en mode d'allocation dynamique. Si ce volume est utilisé pour des tests ultérieurs, alors les conditions initiales ne sont plus les mêmes

puisque le volume avait déjà acquis une allocation réelle sur le support de stockage physique.

- Reformatage du système de fichier au début de chaque test pour éliminer au maximum les effets dus à la fragmentation.
- Exclusion des tests concurrentiels dont lesquels les deux formats de stockages sont testés simultanément. L'objectif est de restreindre l'utilisation des ressources au test d'un format à la fois, d'où l'utilisation d'une seule VM pour les tests des deux formats de stockage.
- Utilisation d'un système d'exploitation autonome (Live CD) pour éviter d'éventuelles activités d'écritures sur le format à tester par un système d'exploitation installé (fichiers temporaires, swap, etc.).

4.3 Approche de validation des résultats

La validation des données se fait principalement par deux modalités :

- Les conclusions et les résultats des expériences précédentes de la revue de littérature.
- Les cohérences des résultats expérimentaux.

Un exemple de la première modalité c'est qu'on doit s'attendre à ce que les performances du format RAW soient inférieures à celles du format physique en mode physique ; car tous les résultats des expériences, qu'on a vus dans le chapitre 2, vont dans ce sens. De même, les performances des deux formats RAW et LVM en allocation dynamique devraient être proches de celles du mode physique.

Un exemple de la seconde modalité, on s'attend à des écarts faibles entre les trois tests effectués pour un format donné. Dans le cas contraire, une vérification des différentes composantes du laboratoire et des configurations s'impose avant d'accepter les résultats des tests.

4.4 Mise en œuvre

L'installation du laboratoire informatique nécessite, premièrement le choix du matériel et logiciel à utiliser, et deuxièmement la configuration de ces différentes composantes.

4.4.1 Installation du laboratoire

La Figure 4.5 illustre le schéma du laboratoire utilisé.

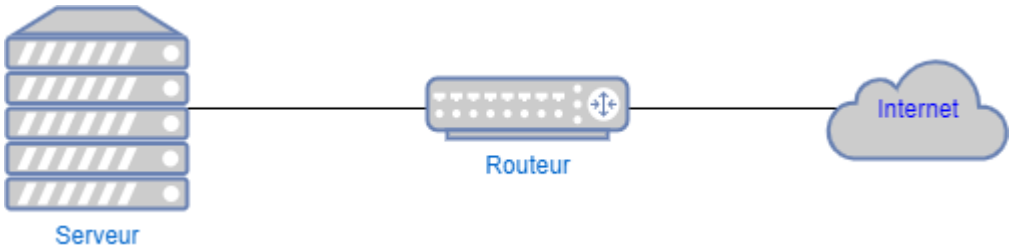


Figure 4.5 : Schéma du laboratoire

Au début, le routeur va servir comme un point pour accéder à Internet pour l'installation de l'hyperviseur et les différents packages dont on a besoin pour son fonctionnement. Ensuite, le serveur sera débranché du routeur pour isoler le système de l'extérieur. Les principales composantes du serveur sont décrites dans le Tableau 4.6.

Tableau 4.6 : Les principales composantes du serveur

Composante	Description
CPU	Intel Xeon E5-2690 v3 12 Cœurs 2.60GHz
RAM	128 Go
Stockage	Mémoire flash 1To (NVMe / M.2)

L'hyperviseur utilisé est basé sur la distribution *CentOS 8* avec une installation minimale. Seuls les packages nécessaires au fonctionnement de la plateforme de virtualisation KVM/QEMU sont installés en se conformant aux recommandations de Red Hat. Le système d'exploitation autonome (Live CD) utilisé est *SystemRescueCD 5.3.2*. Il est dérivé de la distribution *Gentoo* et contient l'outil principal de mesure *IOzone* dans sa version 3.471.

4.4.2 Configuration du laboratoire

Le support physique du serveur est divisé en trois parties (Hyperviseur, LVM et Physique/RAW). La Figure 4.6 illustre chacune des parties et à quoi elles vont servir. Les éléments en couleur verte représentent les formats de stockage qui seront testés. Le seul système de fichier utilisé est ext4.

Au début, la partition physique `/dev/sda5` va servir à tester les performances en mode physique. Ensuite, elle sera utilisée pour héberger le fichier RAW dans le mode virtuel.

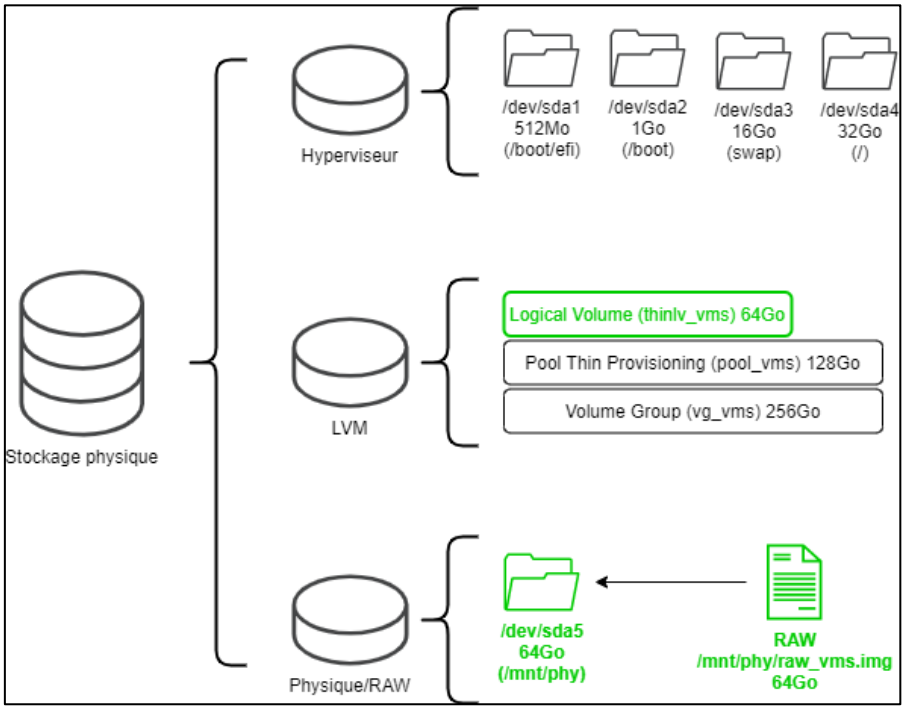


Figure 4.6 : Configuration du stockage physique

La machine virtuelle est créée à l'aide de la commande `virt-install`. On lui alloue les ressources suivantes (Tableau 4.7) :

Tableau 4.7 : Les ressources attribuées à la machine virtuelle

Composante	Description
vCPU	1 (4 cœurs)
vRAM	2 Go
Disque virtuel	Volume LVM Thin 64 Go (VirtIO)
	Fichier RAW 64 Go (VirtIO)

VirtIO est la seule interface de connexion utilisée pour les deux formats de stockages. Microsoft Excel est le tableur utilisé pour le traitement des données et la génération des graphiques. Les résultats de l'expérimentation sont présentés et analysés dans le chapitre suivant.

Chapitre 5

Analyse des résultats

Ce chapitre présente les résultats obtenus des quatre opérations d'E/S pour les différents formats de stockage testés. Ces opérations sont l'écriture séquentielle, l'écriture aléatoire, la lecture séquentielle et la lecture aléatoire. Pour chacune d'elles, les unités qui sont mesurées sont le débit, l'IOPS et la latence.

L'analyse des résultats, tels que présentés dans ce chapitre, a principalement pour but de comparer les différentes mesures obtenues pour chaque format de stockage. Ces comparaisons vont nous servir par la suite de valider l'hypothèse de cette étude. Les annexes renferment les résultats détaillés des mesures pour chaque format de stockage. Dans tout ce chapitre, les abréviations PHY, RAW et LVM font référence à la partition physique, au fichier image RAW et au volume LVM en mode allocation dynamique respectivement.

5.1 Écriture séquentielle

Les Figure 5.1, Figure 5.2, Figure 5.3 présentent les débits, les IOPS et les latences des trois formats de stockage. On constate que la différence entre les débits n'est pas grande. En effet, le débit de LVM présente, en moyenne, 86,14 % du celui de PHY. Alors que celui de RAW en présente 79,86 % environ. On remarque aussi que le maximum des trois débits est atteint pour les tailles de bloc entre 32 Ko et 4 Mo. Alors que pour la taille de bloc 4 Ko, les trois formats performent le moins avec des débits minimaux inférieurs à ceux obtenus pour le reste des tailles de bloc.

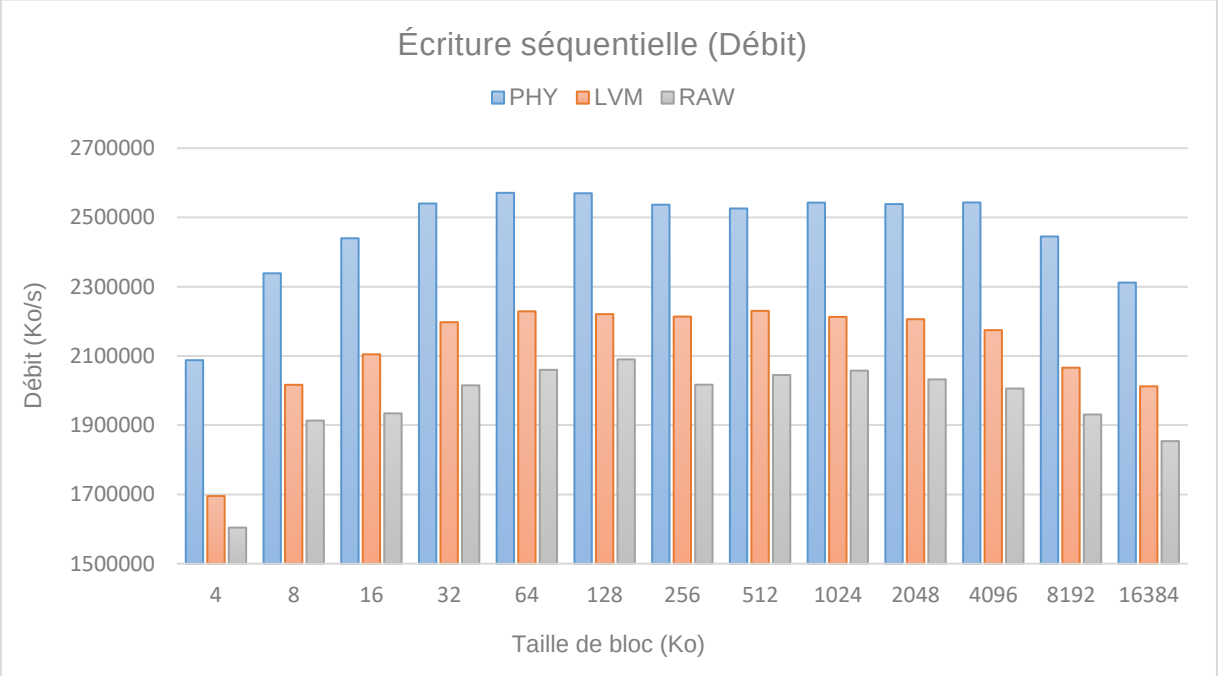


Figure 5.1 : Débit de l'écriture séquentielle

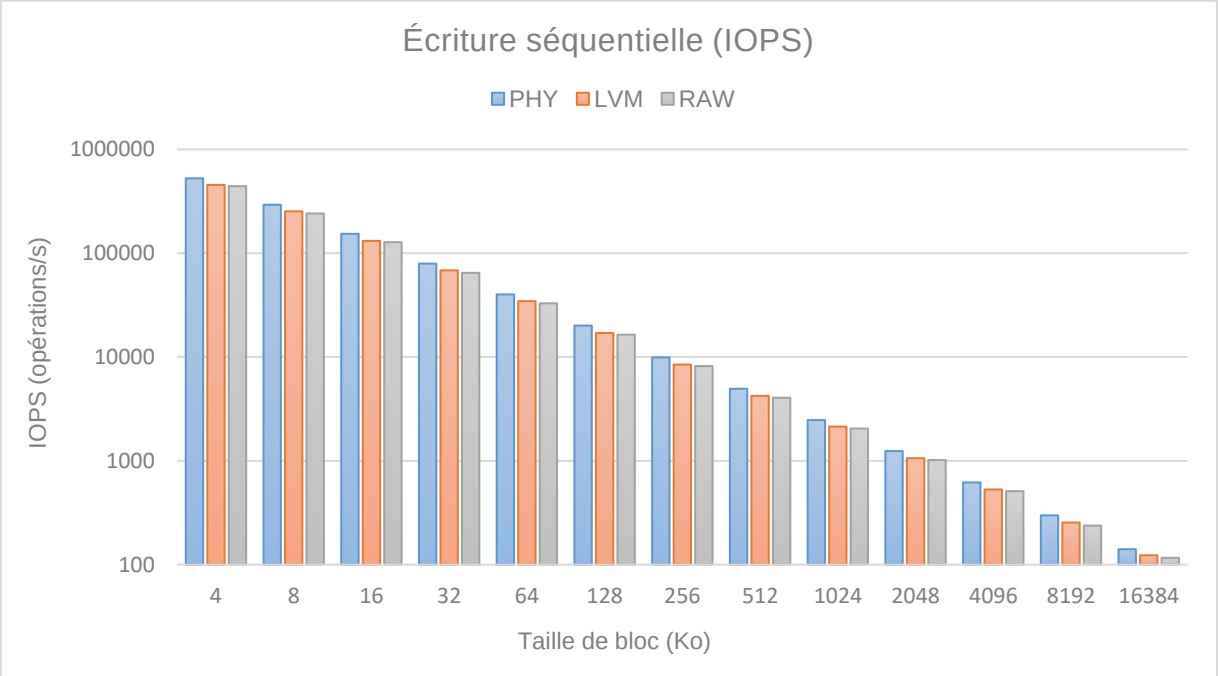


Figure 5.2 : IOPS de l'écriture séquentielle

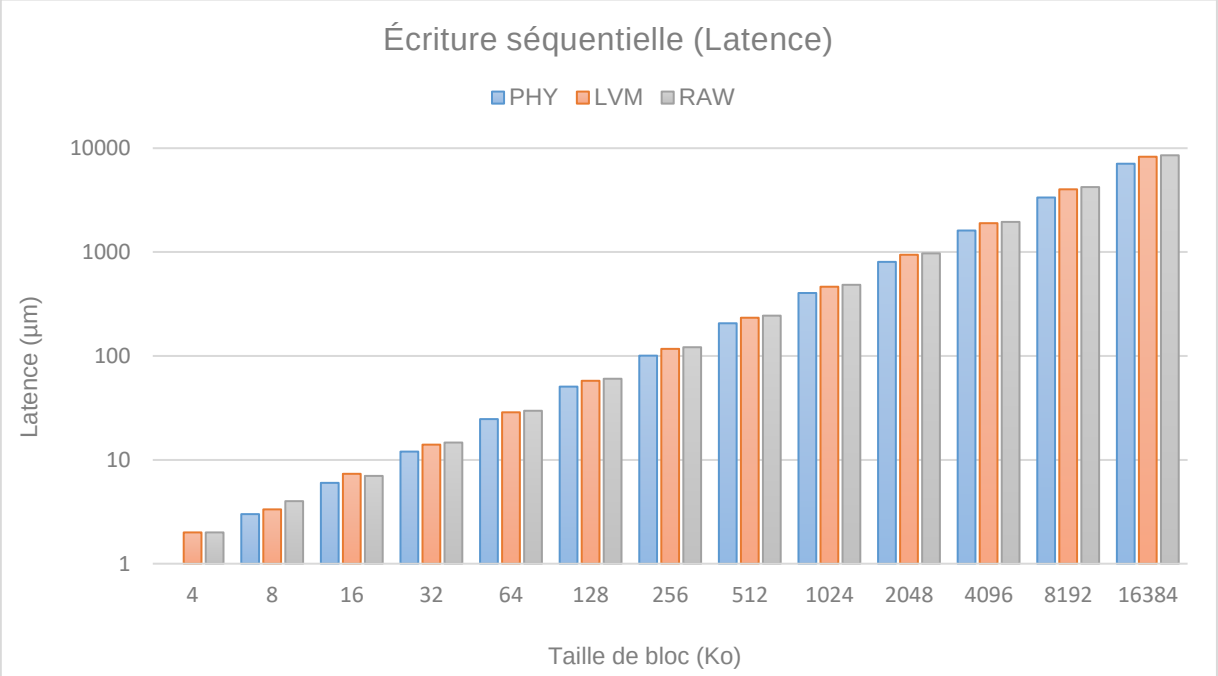


Figure 5.3 : Latence de l'écriture séquentielle

Par contre, la taille de bloc 4 Ko est celle qui possède le plus grand nombre d'IOPS et la plus petite latence (temps de réponse). Le nombre d'IOPS décroit rapidement pour les autres tailles de bloc jusque ce qu'il atteint quelque centaine d'opérations par seconde, alors qu'en même temps, la latence augmente réciproquement. Cela s'explique, évidemment, par le fait que les petites tailles de bloc sont plus rapides à écrire que les plus grandes, donc un temps de réponse plus court pour l'opération.

La Figure 5.4 présente le rapport entre le débit du LVM par rapport à celui du RAW en fonction de la taille de bloc. On découvre que les valeurs de ce rapport sont toujours encadrées par les deux valeurs seuils 100 % et 116,66 % de la formule de l'hypothèse de la section 4.1.5 et ce pour toutes les tailles de bloc. Ce qu'on peut conclure de cette section, c'est que, pour l'écriture séquentielle, le volume LVM en mode allocation dynamique performe mieux qu'un fichier RAW peu importe la taille de bloc utilisée.

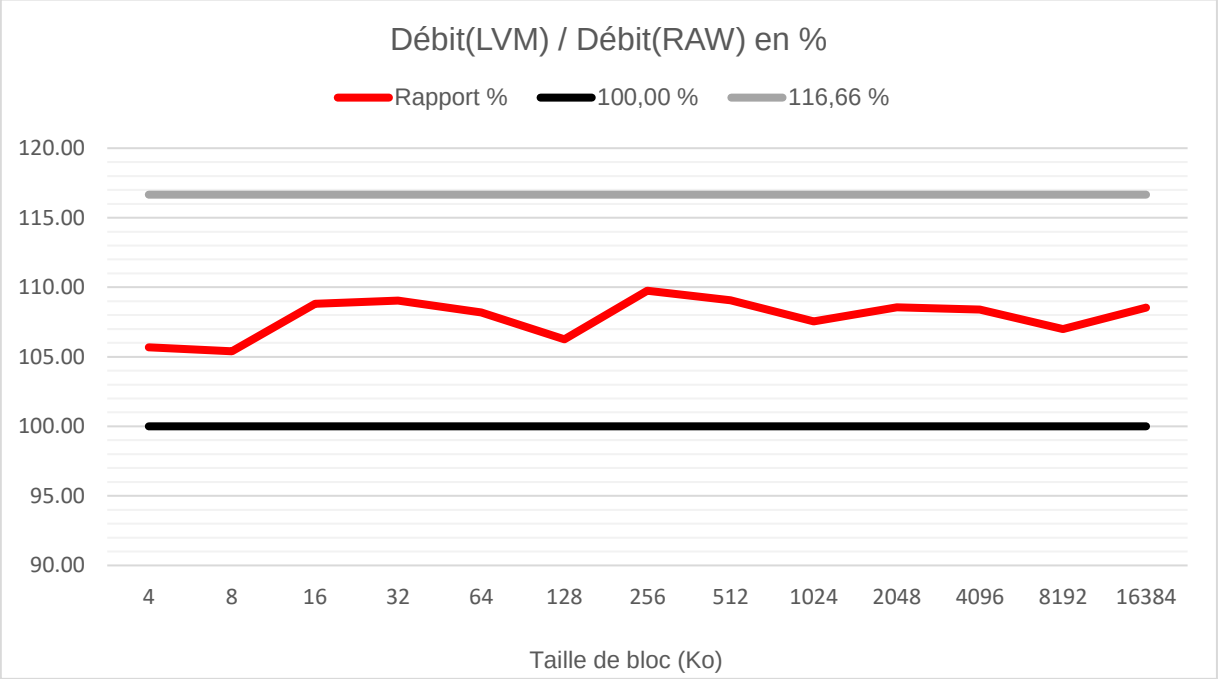


Figure 5.4 : Rapport débit(LVM) et débit(RAW)

5.2 Écriture aléatoire

Les Figure 5.5, Figure 5.6, Figure 5.7 présentent les débits, les IOPS et les latences des trois formats de stockage. On constate que le débit de LVM présente, en moyenne, 78,89 % du celui PHY. Alors que celui de RAW en présente 69,60 % environ. On remarque aussi que le maximum des trois débits est atteint pour les tailles de bloc entre 128 Ko et 4 Mo. Alors que pour la taille de bloc 4 Ko, les trois formats performent le moins avec des débits minimaux inférieurs à ceux obtenus pour le reste des tailles de bloc. Mais la chose la plus remarquable c'est les résultats obtenus pour les deux tailles de bloc 4 Ko et 8 Ko. En effet, ces deux tailles de bloc font exception pour les raisons suivantes :

- Les débits atteignent leurs valeurs les plus faibles.
- Les débits de LVM et RAW sont médiocres et ne font que 44,81 % et 43,74 % de celui de PHY respectivement pour la taille de bloc 4 Ko, et 69,03 % et 57,79 % pour la taille de bloc 8 Ko.

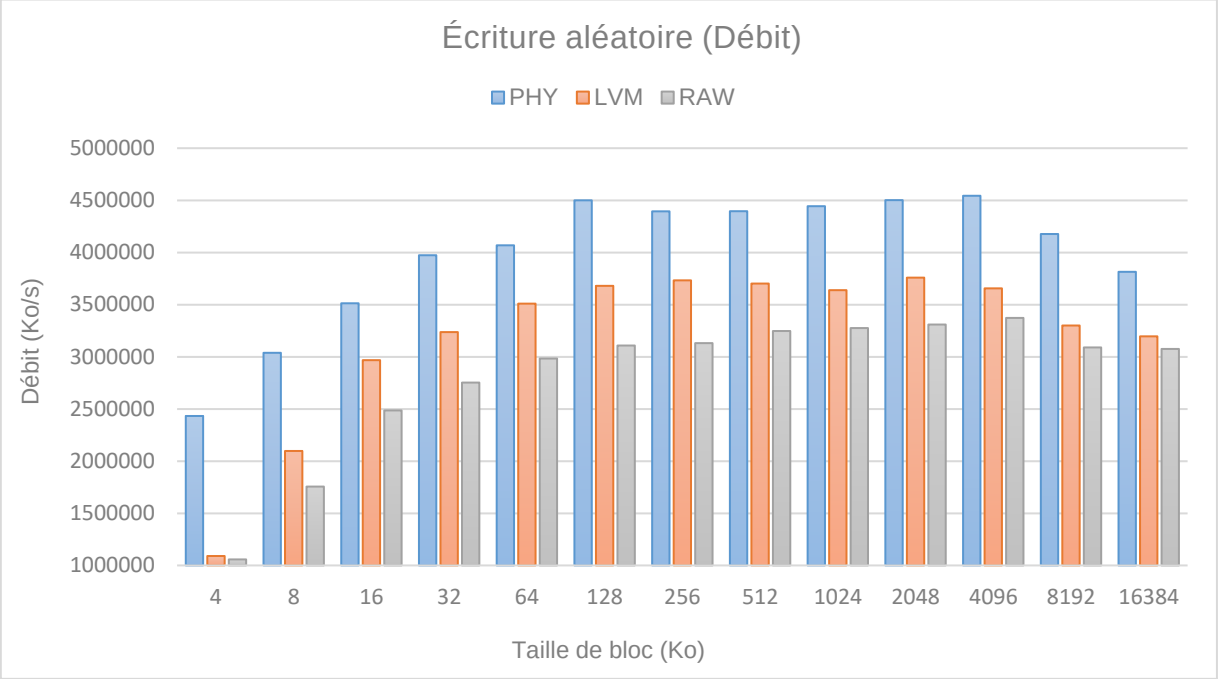


Figure 5.5 : Débit de l'écriture aléatoire

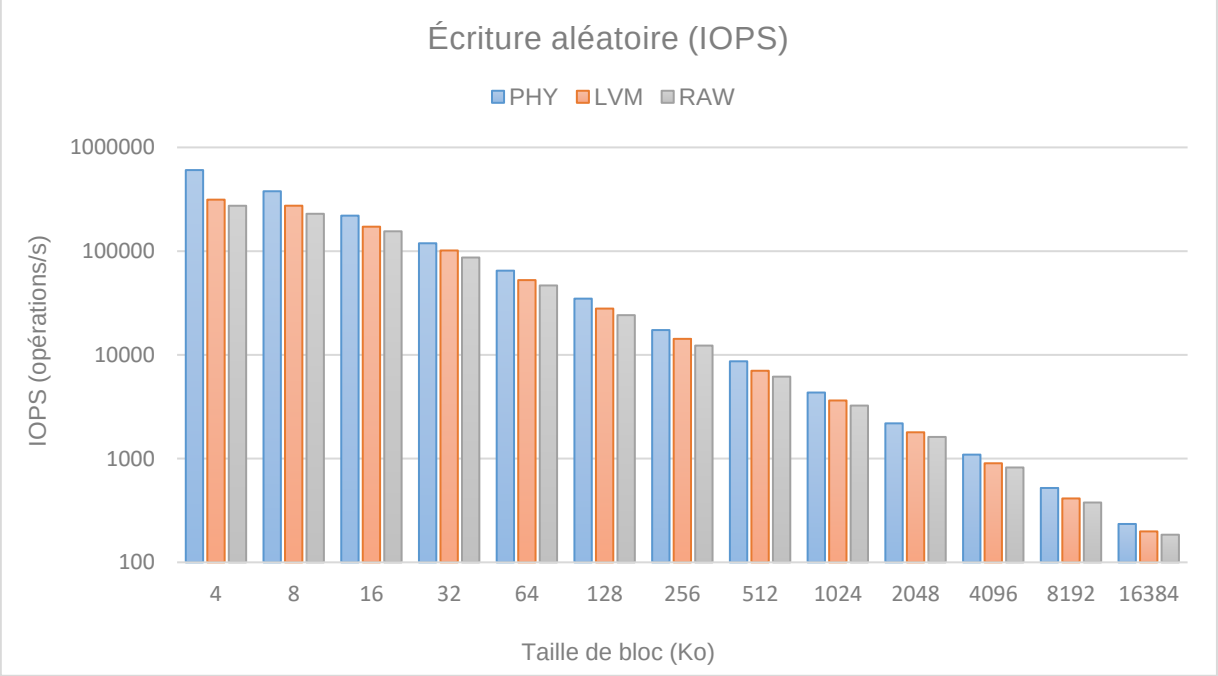


Figure 5.6 : IOPS de l'écriture aléatoire

Comme pour l'opération d'écriture séquentielle, la taille de bloc 4 Ko est celle qui possède le plus grand nombre d'IOPS et la plus petite latence (temps de réponse). Aussi, le nombre d'IOPS décroit rapidement pour les autres tailles de bloc jusqu'à ce qu'il atteigne quelque centaine d'opérations par seconde ; alors qu'en même temps, la latence augmente réciproquement.

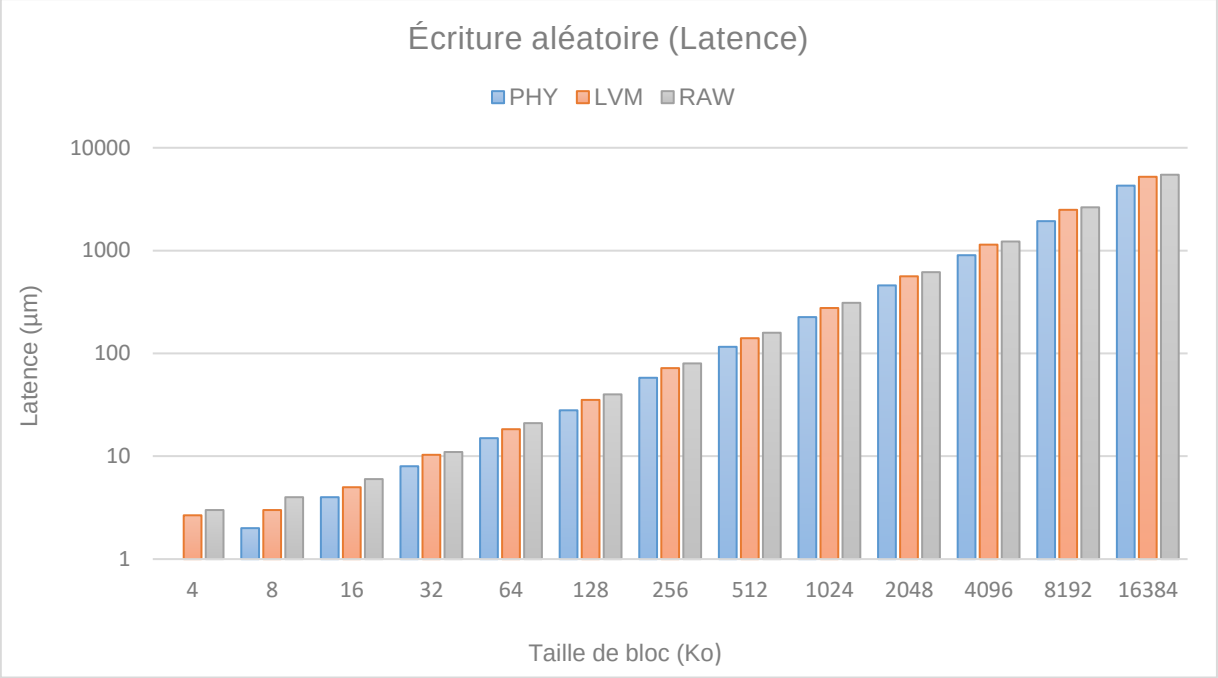


Figure 5.7 : Latence de l'écriture aléatoire

La Figure 5.8 présente le rapport entre le débit du LVM par rapport à celui du RAW en fonction de la taille de bloc. On découvre que, pour les tailles de bloc de 8 Ko jusqu'au 256 Ko inclusivement, les valeurs de ce rapport sont supérieures à 116,66 % ; alors que pour les autres tailles de bloc les valeurs sont encadrées par les deux valeurs 100 % et 116,66 %. Pour conclure cette section, on peut dire que LVM performe mieux que RAW quelle que soit la taille de bloc.

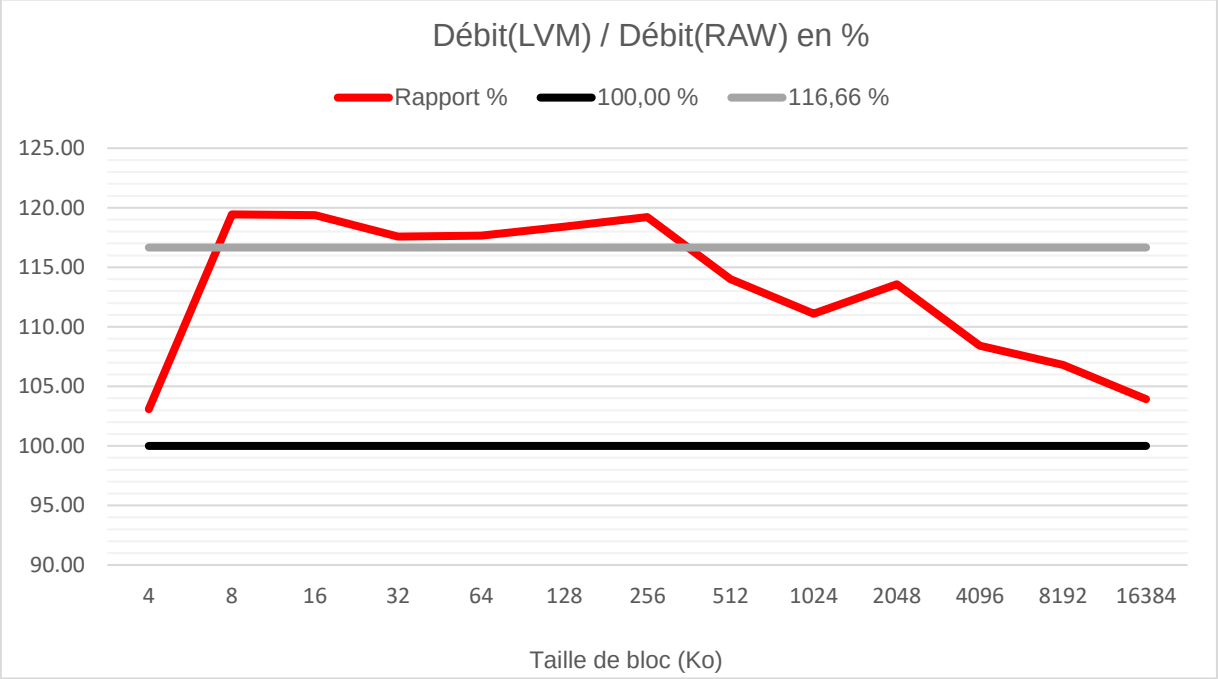


Figure 5.8 : Rapport débit(LVM) et débit(RAW)

5.3 Lecture séquentielle

Les Figure 5.9, Figure 5.10 et Figure 5.11 présentent les débits, les IOPS et les latences des trois formats de stockage. Le fait le plus marquant pour l'opération de lecture séquentielle reste l'excellence des débits effectués par LVM et RAW. En effet, le débit de LVM atteint 99,90 % du débit de PHY alors que celui de RAW en fait 98,42 %. On remarque aussi que le maximum des trois débits est atteint pour les tailles de bloc entre 32 Ko et 128 Ko. Alors que pour la taille de bloc 4 Ko, les trois formats performent le moins avec des débits minimaux inférieurs à ceux obtenus pour le reste des tailles de bloc, mais tout en restant très proches l'un des autres.

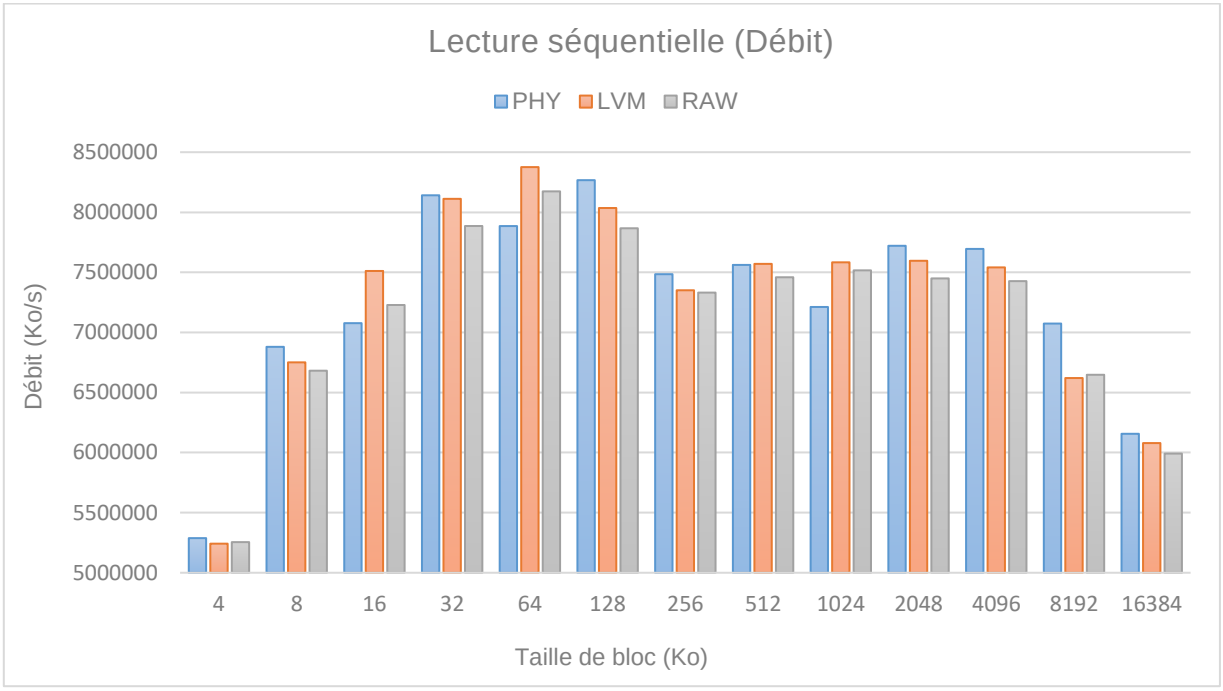


Figure 5.9 : Débit de la lecture séquentielle

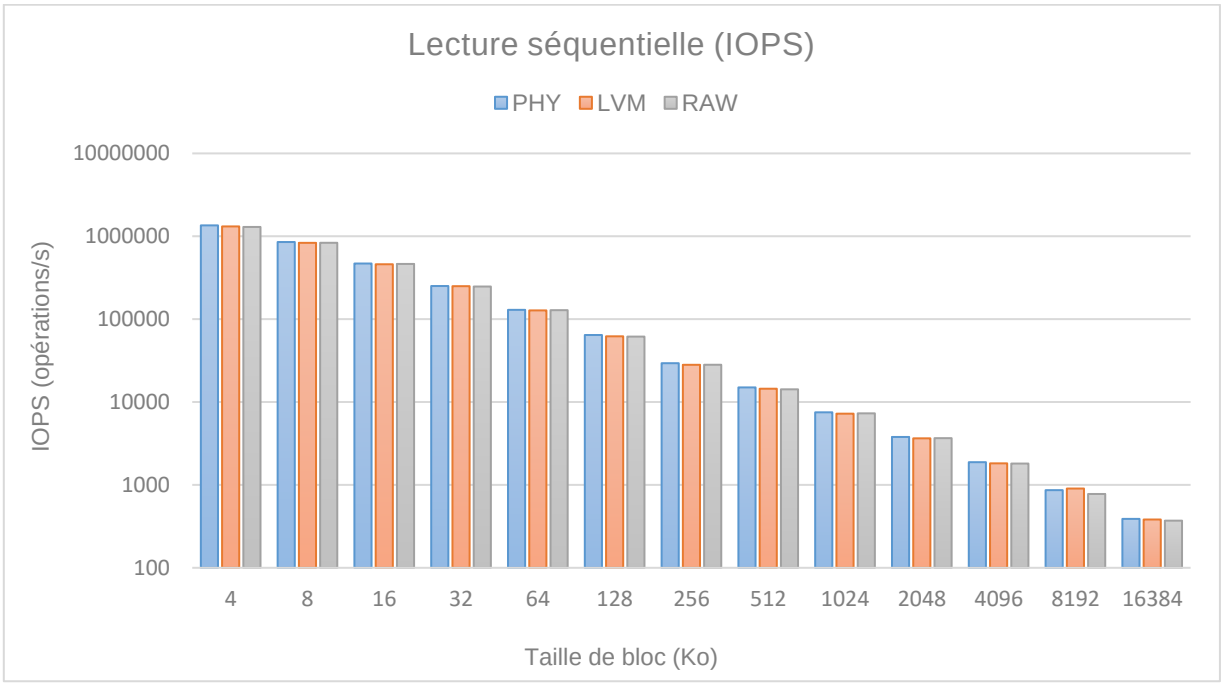


Figure 5.10 : IOPS de la lecture séquentielle

Comme pour les opérations précédentes, la taille de bloc 4 Ko est celle qui possède le plus grand nombre d'IOPS et la plus petite latence. Aussi, le nombre d'IOPS décroit rapidement pour les autres tailles de bloc jusqu'à ce qu'il atteigne quelque centaine d'opérations par seconde ; alors qu'en même temps, la latence augmente réciproquement.

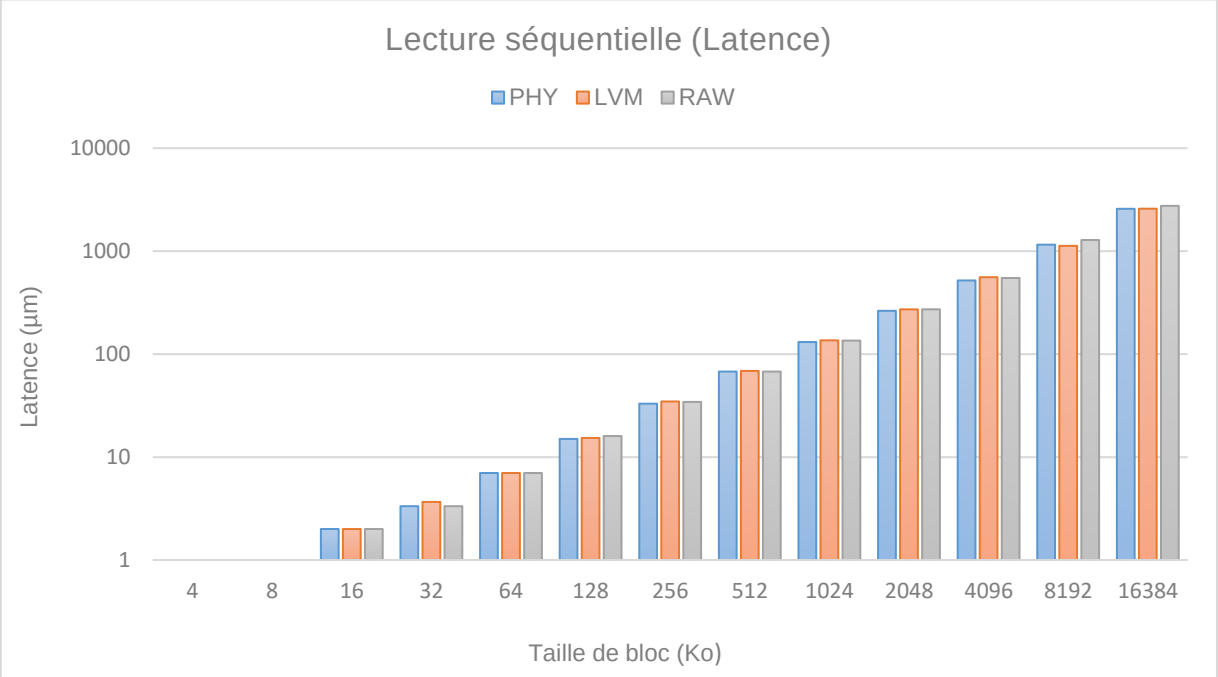


Figure 5.11 : Latence de la lecture séquentielle

La Figure 5.12 présente le rapport entre le débit du LVM par rapport à celui du RAW en fonction de la taille de bloc. Ce rapport se situe légèrement au-dessus du seuil 100 % ; cela est dû aux deux débits qui sont très proches l'un de l'autre comme on l'a vue au début de cette section. Pour conclure cette section, on peut dire que LVM performe légèrement mieux que RAW et que, avec l'opération de lecture séquentielle, on obtient d'excellentes performances qui sont très proches de celui de PHY.

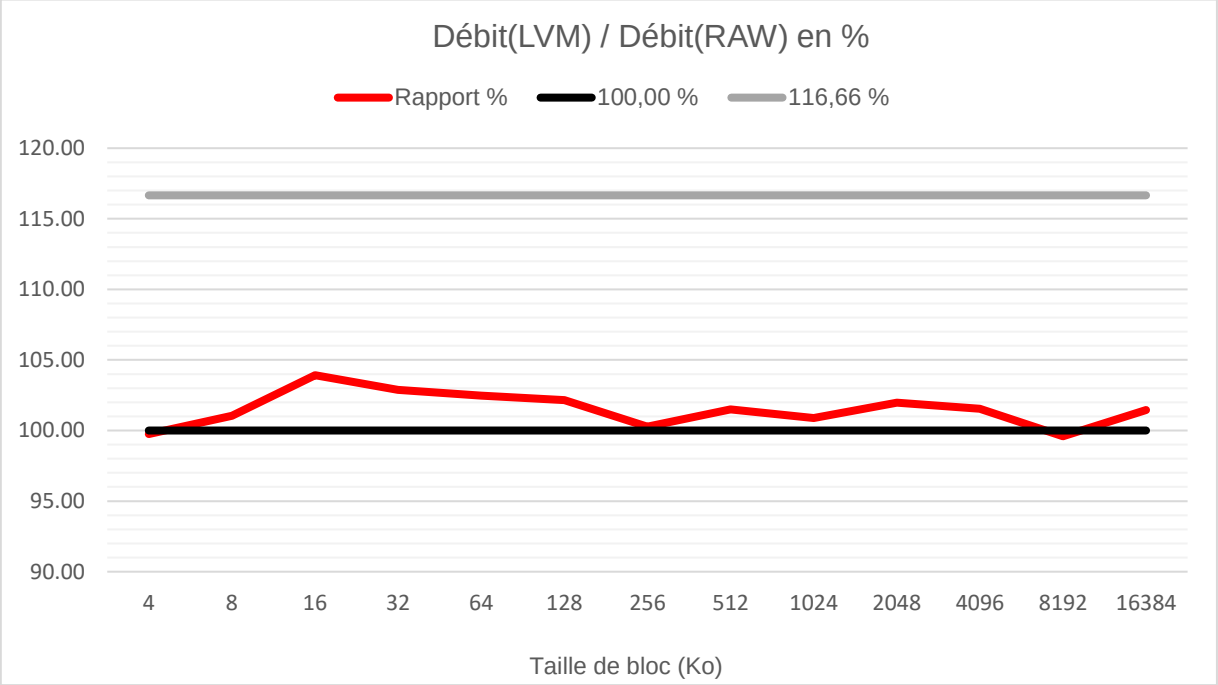


Figure 5.12 : Rapport débit(LVM) et débit(RAW)

5.4 Lecture aléatoire

Les Figure 5.13, Figure 5.14 et Figure 5.15 présentent les débits, les IOPS et les latences des trois formats de stockage. On peut dire que ce qu'on a constaté pour la lecture séquentielle reste à peu près valable pour la lecture aléatoire. C'est vrai que les débits de LVM et RAW sont moins proches de celui du PHY, mais ils dépassent les 90 % de ce dernier. Le débit LVM fait 96,61 % et celui du RAW fait 95,41 %. Les meilleurs débits sont atteints pour les tailles de bloc 64 Ko et 128 Ko et un peu moins pour les tailles de bloc entre 1 Mo et 4 Mo. De la même manière que les opérations précédentes, les débits performent le moins pour la taille de bloc 4 Ko, mais tout en restant très proches l'un des autres.

Tout comme les opérations précédentes, les meilleurs scores du nombre des IOPS et la latence sont ceux obtenus par la taille de bloc 4 Ko. Même remarques pour leurs décroissances et croissance respectivement.

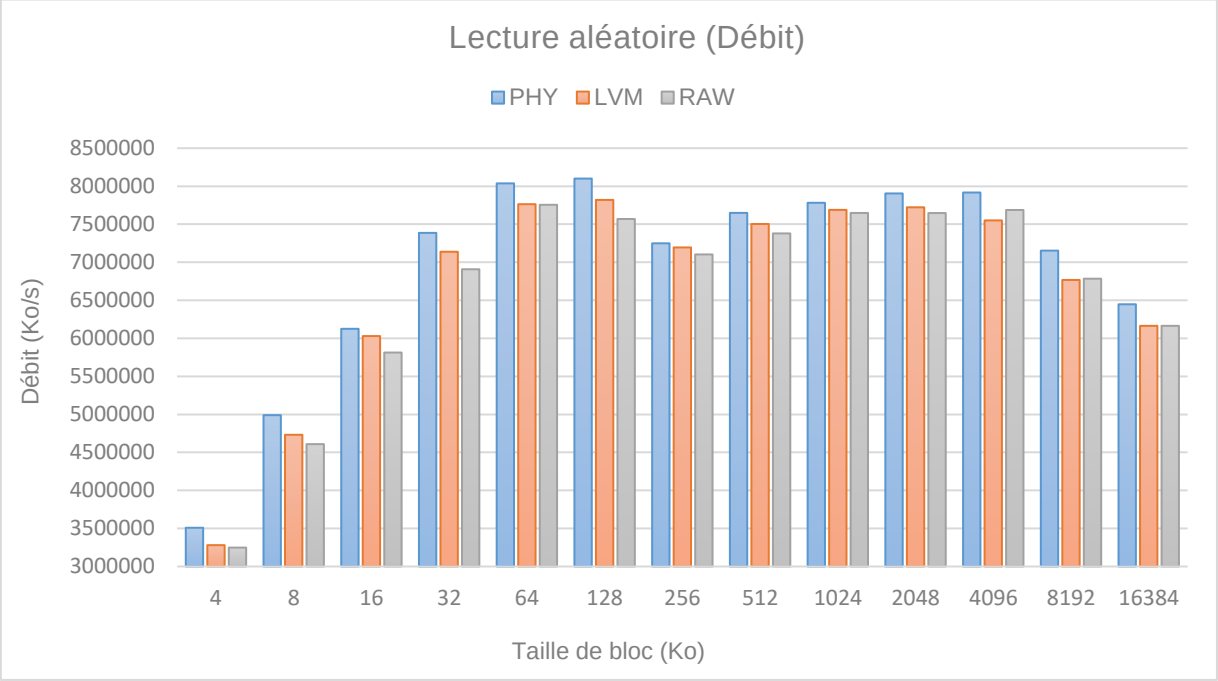


Figure 5.13 : Débit de la lecture aléatoire

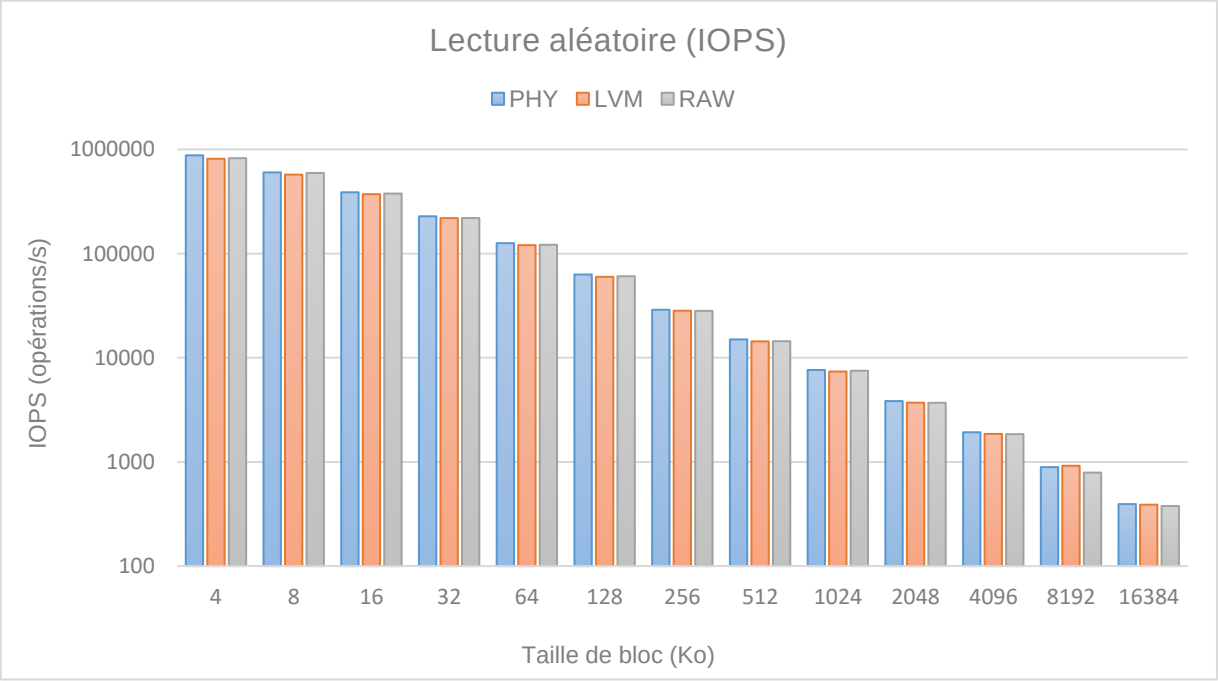


Figure 5.14 : IOPS de la lecture aléatoire

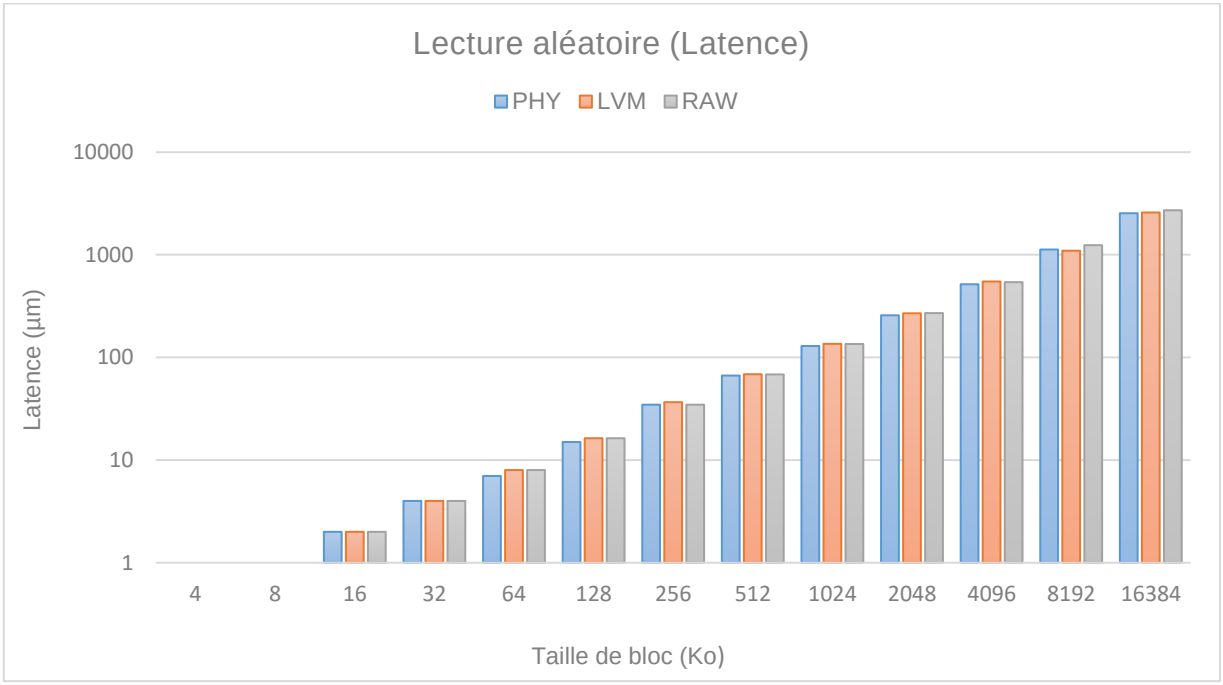


Figure 5.15 : Latence de la lecture aléatoire

La Figure 5.16 présente le rapport entre le débit du LVM par rapport à celui du RAW en fonction de la taille de bloc. Tout comme l'opération précédente, ce rapport se situe légèrement au-dessus du seuil 100 % et pour les mêmes raisons. Pour conclure cette section, on peut dire que LVM performe légèrement mieux que RAW et que, avec l'opération de lecture aléatoire, on obtient d'excellentes performances qui sont un peu moins que celles obtenues pour l'opération de lecture séquentielle.

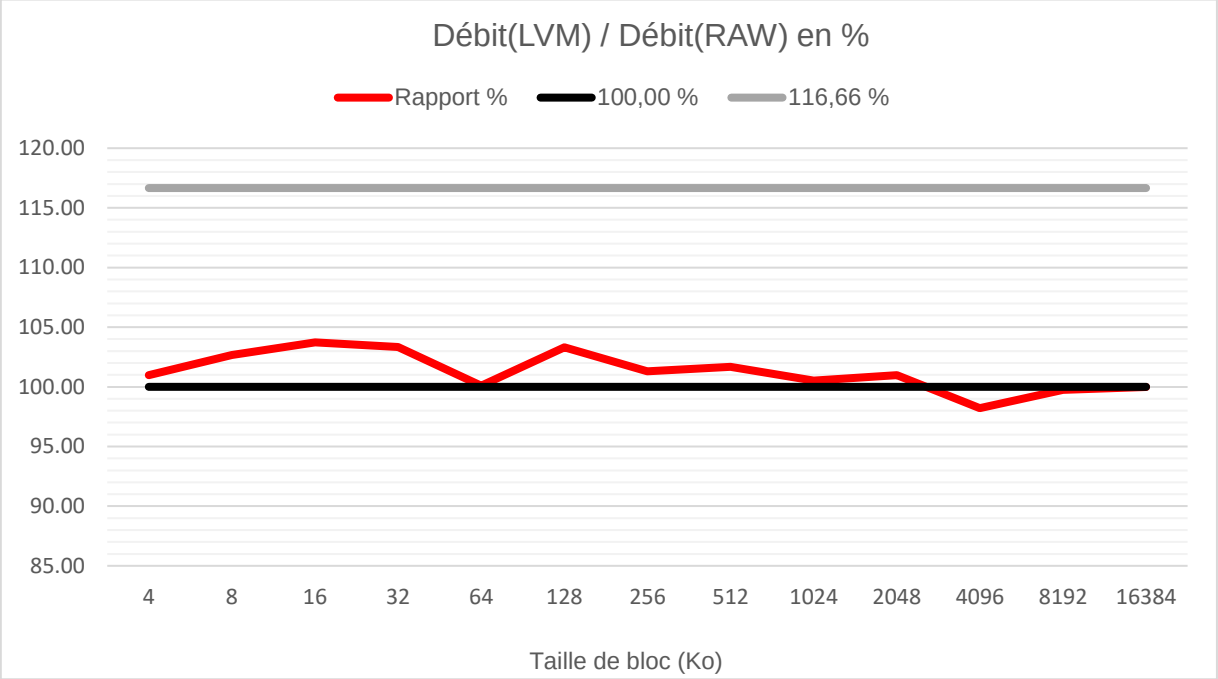


Figure 5.16 : Rapport débit(LVM) et débit(RAW)

5.5 Retour sur les hypothèses

La formule mathématique de l'hypothèse stipule que le rapport entre le débit LVM et celui de RAW est supérieur à 100 % et inférieur ou égal à 116,66 % (section 4.1.5). Le Tableau 5.1 présente la moyenne de ce rapport pour chaque opération d'E/S sur toutes les tailles de bloc :

Tableau 5.1 : Rapport débit(LVM) et débit(RAW) pour toutes les opérations d'E/S

Opération	Débit(LVM) / Débit(RAW) (%)	Hypothèse confirmée ?
Écriture séquentielle	107,87	Oui
Écriture aléatoire	113,28	Oui
Lecture séquentielle	101,49	Oui
Lecture aléatoire	101,28	Oui

On constate que les valeurs de ce rapport vérifient bien la formule de l'hypothèse, et par conséquent, cette dernière est confirmée pour toutes les opérations d'E/S. On remarque aussi que c'est pour les opérations d'écriture qu'on réalise le maximum de gain pour le débit LVM sur celui du RAW. Par contre, pour les opérations de lecture, on peut considérer que ce gain est négligeable et que le bénéfice de l'utilisation de LVM, dans ce cas, réside plus sur ces fonctionnalités que sur son débit.

5.6 Démonstration de la validité des résultats

On a vu dans la section 4.3 que la validation des données va se faire principalement par deux modalités :

- Les conclusions et les résultats des expériences précédentes de la revue de littérature.
- Les cohérences des résultats expérimentaux.

Les résultats de l'expérience nous montrent que les opérations des lectures pour LVM et RAW performent aussi bien que celles de PHY. Or, des résultats similaires ont été obtenus par d'autres auteurs comme Zhang et al [32] et Tang [22]. Même constatation peut être faite pour les autres opérations des écritures qui offrent de bonnes performances, mais moins que celles obtenues pour les lectures.

L'écart entre les résultats des tests pour un format de stockage donné et pour une opération particulière est faible. Cela prouve que le « bruit » dû à des E/S indésirables a été bien contrôlé grâce aux mesures prises comme l'isolation du système, le reformatage des partitions après chaque test, l'utilisation d'un système d'exploitation autonome, etc.

Conclusion

Tout au long de cet essai, plusieurs étapes ont été suivies pour mener une étude de comparaison entre les deux formats de stockage basé sur un fichier image RAW et un volume LVM en mode allocation dynamique. Plusieurs études ont démontré que le fichier image RAW est le format de stockage qui offre les meilleures performances pour une VM parmi tous les autres formats basés sur des fichiers. Malheureusement, ce format ne possède que peu de fonctionnalités. Des fonctionnalités supplémentaires sont parfois nécessaires pour faciliter certaines tâches comme la migration, la duplication, la gestion optimale de l'espace, etc.

Quant aux volumes LVM, ils offrent généralement des performances légèrement supérieures que celles obtenues par le fichier RAW. Par contre, le volume LVM dispose davantage de fonctionnalités intéressantes comme la gestion flexible de l'espace, les instantanés, etc. Le mode d'allocation dynamique permet une gestion de l'espace plus intelligente que celle obtenue par le mode d'allocation fixe. Néanmoins, l'élargissement dynamique du volume et la mise à zéro peuvent avoir un impact négatif sur les performances.

Le but de cette étude est d'évaluer ces impacts négatifs et vérifier si l'utilisation d'un volume LVM en mode allocation dynamique permet de garder un gain en performances par rapport au fichier image RAW. Pour réaliser cette étude, une recherche quantitative a été menée en s'inspirant d'autres expériences précédentes [3] [22] [36] afin de comparer les performances de ces deux formats de stockage dans un contexte bien défini. L'hypothèse de l'étude postule que le volume LVM en mode allocation dynamique performe mieux qu'un fichier RAW, et que le pourcentage moyen de cette amélioration ne dépassera pas les 16,66 %. L'hypothèse a été confirmée pour toutes les opérations d'E/S testées dans l'expérimentation avec un gain maximal de 13,28 % pour l'écriture séquentielle et un gain minimal de 1,28 % pour la lecture aléatoire. Un laboratoire informatique a été aménagé pour réaliser les expérimentations. Seul le support physique de type local (DAS) a été utilisé. Le système de fichiers utilisé dans tous les tests est ext4. Les opérations d'E/S ont permis de calculer le débit, l'IOPS et la latence de chaque format de stockage en fonction de la taille de bloc dont les valeurs varient entre 4 Ko

et 16 Mo. L'expérience s'est limitée aux fichiers larges et au volume logique configuré sur une seule partition physique. Plusieurs mesures ont été prises pour assurer la réussite des tests telles que l'isolation du système et l'exclusion des tests concurrentiels par l'utilisation d'une seule VM pour les deux formats de stockage. La cohérence des résultats expérimentaux et les résultats des expériences précédentes de la revue de littérature ont permis de valider les données recueillies.

Ces résultats ne sont valides que dans le contexte de cette étude et les limites imposées par l'expérimentation. Ainsi, il serait pertinent d'examiner les effets d'autres variables sur les résultats. Par exemple :

- Explorer les performances d'un volume LVM en mode allocation dynamique qui est configuré sur plusieurs disques ou partitions physiques. Malgré le fait que les auteurs de [35] déconseille une configuration complexe d'un LVM sur plusieurs disques, une éventuelle recherche sur ce cas peut s'avérer importante pour nous montrer les limites à partir desquelles des dégradations de performances commencent à apparaitre.
- Dans un milieu professionnel, les entreprises font souvent recours à d'autres techniques de virtualisation de stockage comme le RAID, principalement pour des raisons de sécurité liées à la tolérance aux pannes. Ainsi, connaitre l'impact de ce dernier sur les performances d'un volume LVM serait intéressant.
- Refaire la même étude pour d'autre système de fichiers comme le xfs, zfs, reiserfs, etc.
- Utiliser d'autre architecture de stockage comme NFS pour le réseau ou Ceph pour les systèmes distribués.
- Examiner l'effet de la taille des fichiers sur les performances au lieu de la taille de bloc.

Le rôle de la virtualisation dans le développement des infrastructures TI modernes est incontestable. En effet, des technologies telles que l'infonuagique (*Cloud*) n'auraient pu voir le jour sans les nombreux avantages que la virtualisation apporte. L'optimisation des ressources, la réduction de la consommation énergétique, la gestion simplifiée, l'abstraction du matériel, etc. sont des bénéfices qu'on peut tirer de cette technique. D'autres techniques ont été développées et qui poussent la virtualisation encore loin dans ces limites. On peut citer la conteneurisation (*containerization*) et la virtualisation imbriquée (*nested virtualization*) qui jouent un rôle important dans les infrastructures en tant que service (*IaaS*) et autres. Le but

d’utiliser la virtualisation est de tirer le maximum de bénéfices de ses avantages tout en réduisant au minimum la perte des performances. On a vu dans cet essai plusieurs techniques qui se sont succédé pour améliorer les performances comme l’émulation partielle, la paravirtualisation et la virtualisation matérielle assistée. L’utilisation de la paravirtualisation au niveau des E/S a permis d’atteindre des résultats proches de celles obtenues par du matériel physique. L’utilisation de VirtIO, comme interface de connexion pour les disques virtuels dans l’expérience de cet essai, a permis d’atteindre environ 98% des performances d’une interface physique pour les opérations de lectures. La paravirtualisation est aussi utilisée pour les interfaces réseau virtuelles, et elle permet d’atteindre des performances si proches de celle obtenue par des interfaces réseau physiques. L’objectif de cet essai consiste à apporter une contribution pour encore améliorer cet aspect ; à savoir, les performances au niveau du stockage en menant une étude comparative entre le format fichier RAW et le volume logique LVM en mode allocation dynamique.

Liste des références

[1] M. Varian, «VM and the VM Community: Past, Present, and Future,» Princeton University, 1997.

[2] S. Bose, P. Mishra, P. Sethuraman et R. Taheri, «Benchmarking Database Performance in a Virtual Environment,» *Springer TPCTC,* vol. 5895, pp. 167-182, 2009.

[3] G. Joshi, S. Shingade et M. Shirole, «Empirical Study of Virtual Disks Performance with KVM on DAS,» *IEEE International Conference on Advances in Engineering & Technology Research,* pp. 1-8, 2014.

[4] E. Bugnion, J. Nieh et D. Tsafrir, Hardware and Software Support for Virtualization, Margaret Martonosi, Princeton University, 2017.

[5] D. Marinescu et R. Kröger, State of the art in autonomic computing and virtualization, Wiesbaden, Germany: Distributed Systems Lab, 2007.

[6] R. White, «45 Years of Mainframe Virtualization: CP-67/CMS and VM/370 to z/VM,» GSE/IBM Technical University, Mainz, 2012.

[7] B. Bitner et S. Greenlee, «z/VM - A Brief Review of Its 40 Year History,» IBM Corporation, 2012.

[8] E. Bugnion, S. Devine , M. Rosenblum, J. Sugerman et E. Y. Wang, «Bringing Virtualization to the x86 Architecture with the Original VMware Workstation,» *ACM Transactions on Computer Systems (TOCS),* vol. 30, n° %14, p. Article 12, 2012.

[9] G. Neiger, A. Santoni, F. Leung, D. Rodgers et R. Uhlig, «Intel Virtualization Technology: Hardware Support for Efficient Processor Virtualization,» *Intel Technology Journal,* vol. 10, n° %13, pp. 167-177, 2006.

[10] H. D. Chirammal, P. Mukhedkar et A. Vettathu, Mastering KVM Virtualization, Birmingham, UK: Packt Publishing, 2016.

[11] R. P. Goldberg, Architectural Principles for Virtual Computer Systems, Harvard University, 1972.

[12] W. Paper, Understanding Full Virtualization, Paravirtualization, and Hardware Assist, VMware, Inc., 2007.

[13] H. Lee, «Virtualization Basics : Understanding Techniques and Fundamentals,» Indiana University, Indiana, 2014.

[14] O. Agesen, A. Garthwaite, J. Sheldon et P. Subrahmanyam, «The Evolution of an x86 Virtual Machine Monitor,» *ACM SIGOPS Operating Systems Review,* vol. 44, n° %14, pp. 3-18, 2010.

[15] A. Kivity, Y. Kamay, D. Laor, U. Lublin et A. Liguori, «kvm: the Linux Virtual Machine Monitor,» chez *In Proceedings of the 2007 Ottawa Linux Symposium*, Ottawa, Ontario, Canada, 2007.

[16] W. Paper, KVM: Kernel-based Virtualization Driver, Qumranet Inc., 2006.

[17] S. G. Soriga et M. Barbulescu, «A comparison of the performance and scalability of Xen and KVM hypervisors,» chez *2013 RoEduNet International Conference 12th Edition: Networking in Education and Research*, Iasi, Romania, 2013.

[18] M. Portnoy, Virtualization Essentials, USA: John Wiley & Sons, Inc., 2016.

[19] Red Hat, «What is KVM?,» Red Hat, [En ligne]. Available: https://www.redhat.com/en/topics/virtualization/what-is-KVM.

[20] F. Bellard, «QEMU, a Fast and Portable Dynamic Translator,» USENIX Association, 2005.

[21] «QEMU the FAST! processor emulator,» [En ligne]. Available: https://www.qemu.org/index.html.

[22] C. Tang, «FVD: a High-Performance Virtual Machine Image Format for Cloud,» chez *Proceedings of the 2011 USENIX conference on USENIX annual technical conference*, Berkeley, CA, USA, 2011.

[23] T. R. Weiss, «Logical volume manager,» *Computerworld,* vol. 37, n° %111, p. 42, 2003.

[24] D. Teigland et H. Mauelshagen, «Volume Managers in Linux,» chez *2001 USENIX Annual*, Boston, Massachusetts, USA, 2001.

[25] . S. Powers, «LVM Demystified,» *Linux Journal,* vol. 2013, n° %1236, p. 7, 2013.

[26] H. Mauelshagen, «LVM2 Resource Page,» [En ligne]. Available: https://www.sourceware.org/lvm2/.

[27] G. Zhang, J. Shu, W. Xue et W. Zheng, «Design and Implementation of an Out-of-Band Virtualization System for Large SANs,» *IEEE Transactions on Computers,* vol. 56, n° %112, pp. 1654 - 1665, 2007.

[28] «LVMTHIN,» Red Hat, Inc, [En ligne]. Available: http://man7.org/linux/man-pages/man7/lvmthin.7.html.

[29] M. Kulkarni et S. Satnur, «Performance Study of VMware vStorage Thin Provisioning,» VMware, Inc., Palo Alto, CA, USA, 2009.

[30] P. Bonzini, «Thin-provisioned disks with QEMU and KVM,» chez *Red Hat, Inc.*, 2014.

[31] H. Simitci, Storage Network Performance Analysis, Indianapolis, IN, USA: Wiley Publishing, Inc., 2003.

[32] B. Zhang, X. Wang, R. Lai, L. Yang, Z. Wang, Y. Luo et X. Li, «Evaluating and Optimizing I/O Virtualization in Kernel-based Virtual Machine (KVM),» *IFIP International Federation for Information Processing,* vol. 6289, pp. 220-231, 2010.

[33] A.-C. Bujor et R. Dobre, «KVM IO Profiling,» chez *2013 RoEduNet International Conference 12th Edition*, Iasi, Romania, 2013.

[34] R. Russell, «virtio: towards a de-facto standard for virtual I/O devices,» *ACM SIGOPS Operating Systems Review,* vol. 42, n° %15, p. 95–103, 2008.

[35] B. Djordjevic et V. Timcenko, «LVM in the Linux environment: performance examination,» *Technical Gazette,* vol. 22, n° %15, p. 1157+, 2015.

[36] K. Huynh, «Exploiting The Latest KVM Features For Optimized Virtualized Enterprise Storage Performance,» chez *IBM Linux Technology Center*, 2013.

[37] W. D. Norcott, «IOzone Filesystem Benchmark,» [En ligne]. Available: http://www.iozone.org/.

[38] X. Xu, F. Zhou, J. Wan et Y. Jiang , «Quantifying Performance Properties of Virtual,» *2008 International Symposium on Information Science and Engieering,* vol. 1, pp. 24-28, 2008.

Bibliographie

W. Graniszewski et A. Arciszewski, «Performance analysis of selected hypervisors (Virtual Machine Monitors - VMMs),» International Journal of Electronics and Telecommunications, vol. 62, n° %13, p. 231–236, 2016.

D.-Y. Hong, J.-J. Wu, P.-C. Yew, W.-C. Hsu, C.-C. Hsu, P. Liu, C.-M. Wang et Y.-C. Chung, «Efficient and Retargetable Dynamic Binary,» IEEE Transactions on Parallel and Distributed Systems, vol. 25, n° %13, pp. 622 - 632, 2014.

I. Habib, «Virtualization with KVM,» Linux Journal, vol. 2008, n° %1166, p. 1, 2008.

T. Clark, Storage Virtualization: Technologies for Simplifying Data Storage and Management, Boston: Addison-Wesley Professional, 2005.

J. E. Smith et R. Nair, Virtual Machines : Versatile Platforms for Systems and Processes, USA: Elsevier Inc., 2005.

S. Hajnoczi, «An Updated Overview of the QEMU Storage Stack,» chez IBM Linux Technology Center, 2011.

A. Silberschatz, P. B. Galvin et G. Gagne, Operating System Concepts, Hoboken, NJ, USA: John Wiley & Sons, Inc., 2018.

D. Kusnetzky, Virtualization: A Manager's Guide, Sebastopol: O'Reilly, 2011.

P. S. Subramanian, Getting Started with Red Hat Enterprise Virtualization, Birmingham, UK: Packt Publishing, 2014.

T. Cerling, J. Buller, C. Enstall et R. Ruiz, Mastering Microsoft Virtualization, Indianapolis, Indiana: Wiley Publishing, Inc., 2010.

A. Finn, P. Lownds, M. Luescher et D. Flynn, Windows Server 2012 Hyper-V, Indianapolis, Indiana: Sybex, 2013.

M. Reitz et K. Wolf, «qcow2 - Why (not)?,» RedHat, 2015.

D. Thain et M. Livny, «The Case for Sparse Files,» chez 2001 USENIX Annual Technical Conference, Madison, Wisconsin, USA, 2001.

Annexe I

Résultats détaillés de la partition physique

On trouve dans cette annexe les résultats complets des quatre opérations d'E/S pour les tests effectués sur la partition physique.

Données du débit en Ko par seconde

Test 1	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	2063318	2345125	2402522	2556404	2568892	2556791	2535510	2528102	2552303	2542070	2557358	2468922	2312455
Lecture séquentielle	5311991	6862452	6291102	8111650	6765529	8277812	7481919	7435340	6243639	7635277	7827815	7129240	5756064
Lecture aléatoire	3500399	4898698	6113722	7413265	7948155	8104438	7236345	7599316	7724205	7968572	7948026	7191134	6480049
Écriture aléatoire	2416719	3077981	3454307	4077191	4003572	4560176	4400716	4403485	4433989	4562456	4594181	4200690	3867271

Test 2	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	2097479	2338759	2462705	2538187	2569799	2585756	2538494	2527560	2547357	2536431	2545401	2433206	2314180
Lecture séquentielle	5285480	6880597	7435439	8182417	8500067	8411629	7827007	7785570	7747608	7764588	7606200	7040163	6293257
Lecture aléatoire	3495277	5063772	6264231	7352804	8134301	8110640	7286791	7638613	7766884	7883020	7924723	7126434	6384447
Écriture aléatoire	2430646	3025555	3656149	3770487	4048649	4473505	4405986	4398907	4449868	4478107	4576378	4164404	3780681

Test 3	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	2102433	2332216	2454462	2526031	2574633	2567232	2536443	2522058	2528864	2537621	2526938	2433076	2308931
Lecture séquentielle	5267053	6897661	7505376	8130392	8390427	8112703	7145611	7465193	7644859	7763722	7651493	7052475	6420337
Lecture aléatoire	3532022	5007479	5998308	7394080	8030944	8087114	7225232	7710796	7856273	7864049	7880243	7143125	6479215
Écriture aléatoire	2450830	3011721	3429318	4074275	4154617	4467517	4377987	4385402	4448189	4466206	4461987	4167530	3796892

Moyenne	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	2087743.333	2338700	2439896.333	2540207.333	2571108	2569926.333	2536815.667	2525906.667	2542841.333	2538707.333	2543232.333	2445068	2311855.333
Lecture séquentielle	5288174.667	6880236.667	7077305.667	8141486.333	7885341	8267381.333	7484845.667	7562034.333	7212035.333	7721195.667	7695169.333	7073959.333	6156552.667
Lecture aléatoire	3509232.667	4989983	6125420.333	7386716.333	8037800	8100730.667	7249456	7649575	7782454	7905213.667	7917664	7153564.333	6447903.667
Écriture aléatoire	2432731.667	3038419	3513258	3973984.333	4068946	4500399.333	4394896.333	4395931.333	4444015.333	4502256.333	4544182	4177541.333	3814948

Données des IOPS en opération par seconde

Test 1	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	525473	293110	153370	79277	40159	20112	9898	4925	2464	1241	617	297	141
Lecture séquentielle	1352889	849357	465894	252241	128817	64594	30037	15280	7638	3832	1888	854	386
Lecture aléatoire	870996	601592	379533	225240	126517	63120	29776	14806	7716	3840	1926	872	392
Écriture aléatoire	600095	377377	215485	122544	63408	34834	17725	8567	4340	2194	1091	507	233

Test 2	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	527329	292740	154252	79610	39960	20092	9907	4948	2468	1233	621	299	141
Lecture séquentielle	1358511	855789	475398	246266	127515	64982	29502	14538	7344	3804	1863	884	393
Lecture aléatoire	874433	605273	397021	230018	125143	62864	28443	14891	7632	3833	1929	903	398
Écriture aléatoire	602574	379677	228826	117568	63457	34908	17100	8596	4343	2187	1094	529	237

Test 3	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	526660	292529	153416	79327	40206	20028	9946	4949	2474	1250	618	299	140
Lecture séquentielle	1349352	853611	466087	254942	132330	63640	28734	15230	7568	3737	1892	857	389
Lecture aléatoire	885574	597931	385821	228731	126042	62856	28389	15247	7534	3837	1922	898	391
Écriture aléatoire	611112	376569	215047	117065	67659	34797	17212	8849	4323	2180	1092	527	233

Moyenne	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	526487.3333	292793	153679.3333	79404.66667	40108.33333	20077.33333	9917	4940.666667	2468.666667	1241.333333	618.6666667	298.3333333	140.6666667
Lecture séquentielle	1353584	852919	469126.3333	251149.6667	129554	64405.33333	29424.33333	15016	7516.666667	3791	1881	865	389.3333333
Lecture aléatoire	877001	601598.6667	387458.3333	227996.3333	125900.6667	62946.66667	28869.33333	14981.33333	7627.333333	3836.666667	1925.666667	891	393.6666667
Écriture aléatoire	604593.6667	377874.3333	219786	119059	64841.33333	34846.33333	17345.66667	8670.666667	4335.333333	2187	1092.333333	521	234.3333333

Données de la latence en microseconde

Test 1	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	1	3	6	12	25	53	101	213	403	801	1605	3356	7060
Lecture séquentielle	0	1	2	3	7	15	32	69	130	262	513	1173	2565
Lecture aléatoire	1	1	2	4	7	15	34	67	131	254	518	1131	2537
Écriture aléatoire	1	2	4	8	15	28	58	116	229	461	900	1947	4304

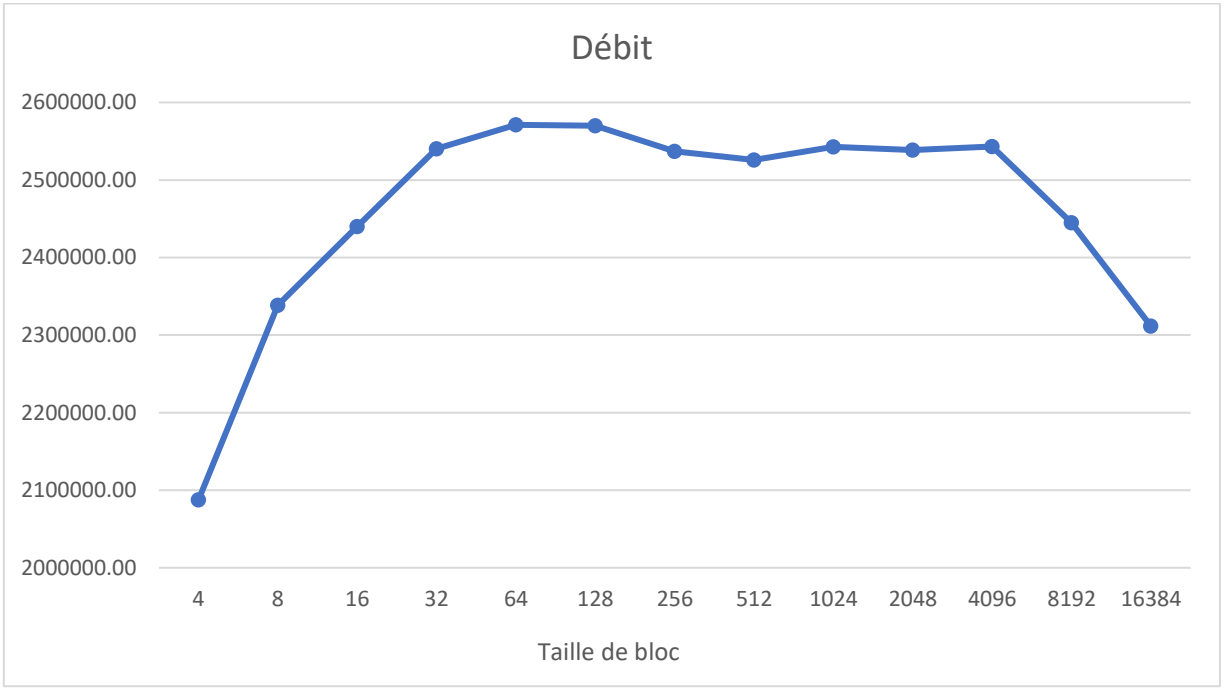
Test 2	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	1	3	6	12	25	49	101	203	404	809	1622	3366	7138
Lecture séquentielle	0	1	2	3	7	15	34	67	130	264	521	1163	2599
Lecture aléatoire	1	1	2	4	7	15	35	67	129	260	519	1126	2560
Écriture aléatoire	1	2	4	8	15	28	58	116	224	458	915	1907	4238

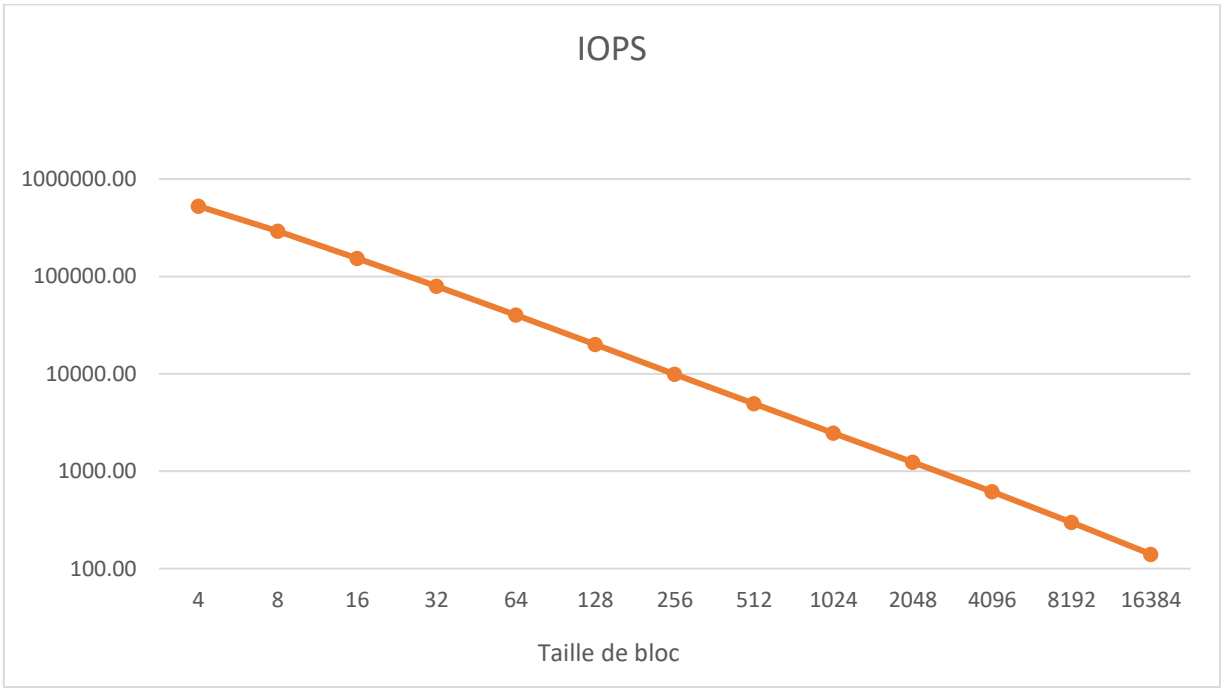
Test 3	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	1	3	6	12	24	50	100	203	404	805	1612	3341	7076
Lecture séquentielle	0	1	2	4	7	15	33	67	133	264	524	1131	2568
Lecture aléatoire	1	1	2	4	7	15	35	66	128	259	514	1124	2533
Écriture aléatoire	1	2	4	8	15	28	58	116	224	458	892	1949	4313

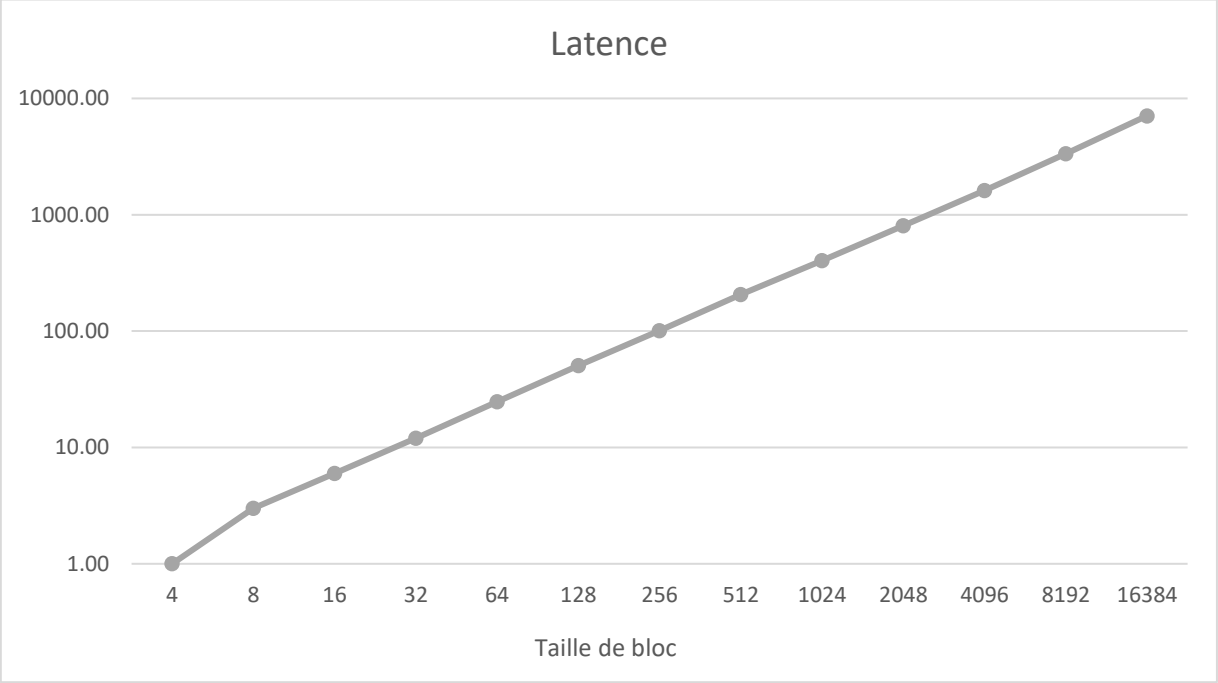
Moyenne	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	1	3	6	12	24.66667	50.66667	100.6667	206.3333	403.6667	805	1613	3354.333	7091.333
Lecture séquentielle	0	1	2	3.333333	7	15	33	67.66667	131	263.3333	519.3333	1155.667	2577.333
Lecture aléatoire	1	1	2	4	7	15	34.66667	66.66667	129.3333	257.6667	517	1127	2543.333
Écriture aléatoire	1	2	4	8	15	28	58	116	225.6667	459	902.3333	1934.333	4285

Écriture séquentielle

Taille de bloc (Ko)	Débit écriture (Ko/s)	IOPS (Opérations/s)	Latence (µs)
4	2087743.33	526487.33	1.00
8	2338700.00	292793.00	3.00
16	2439896.33	153679.33	6.00
32	2540207.33	79404.67	12.00
64	2571108.00	40108.33	24.67
128	2569926.33	20077.33	50.67
256	2536815.67	9917.00	100.67
512	2525906.67	4940.67	206.33
1024	2542841.33	2468.67	403.67
2048	2538707.33	1241.33	805.00
4096	2543232.33	618.67	1613.00
8192	2445068.00	298.33	3354.33
16384	2311855.33	140.67	7091.33

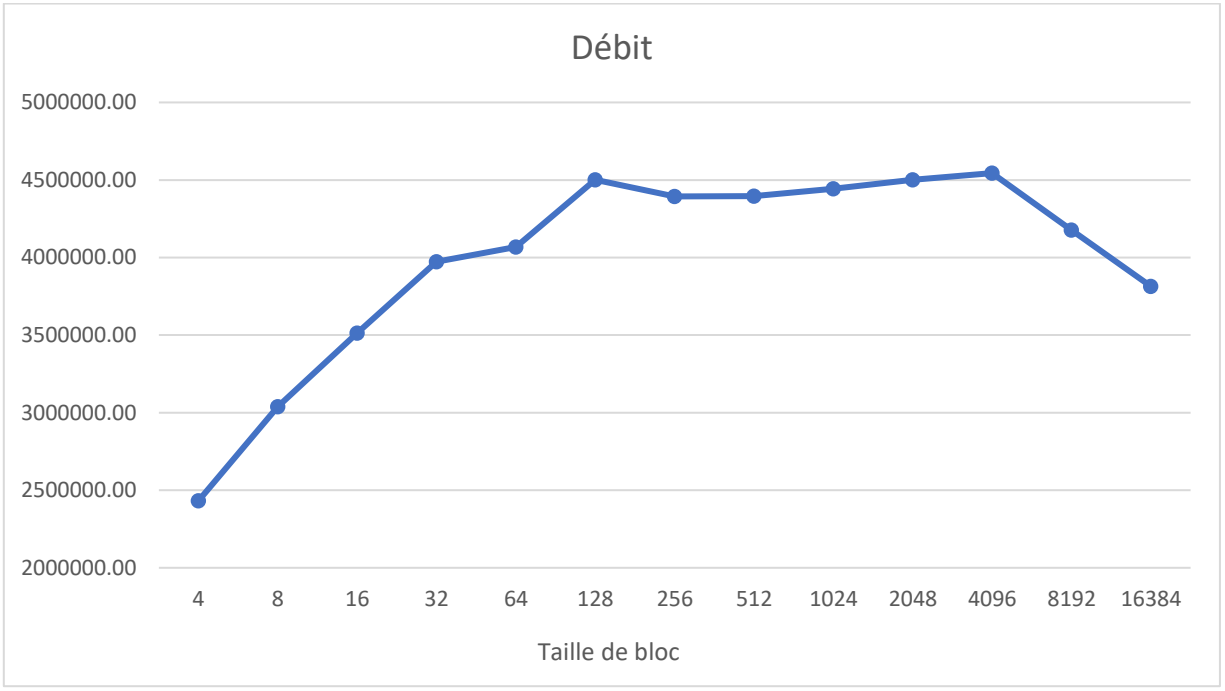

Débit
2600000.00
2500000.00
2400000.00
2300000.00
2200000.00
2100000.00
2000000.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc

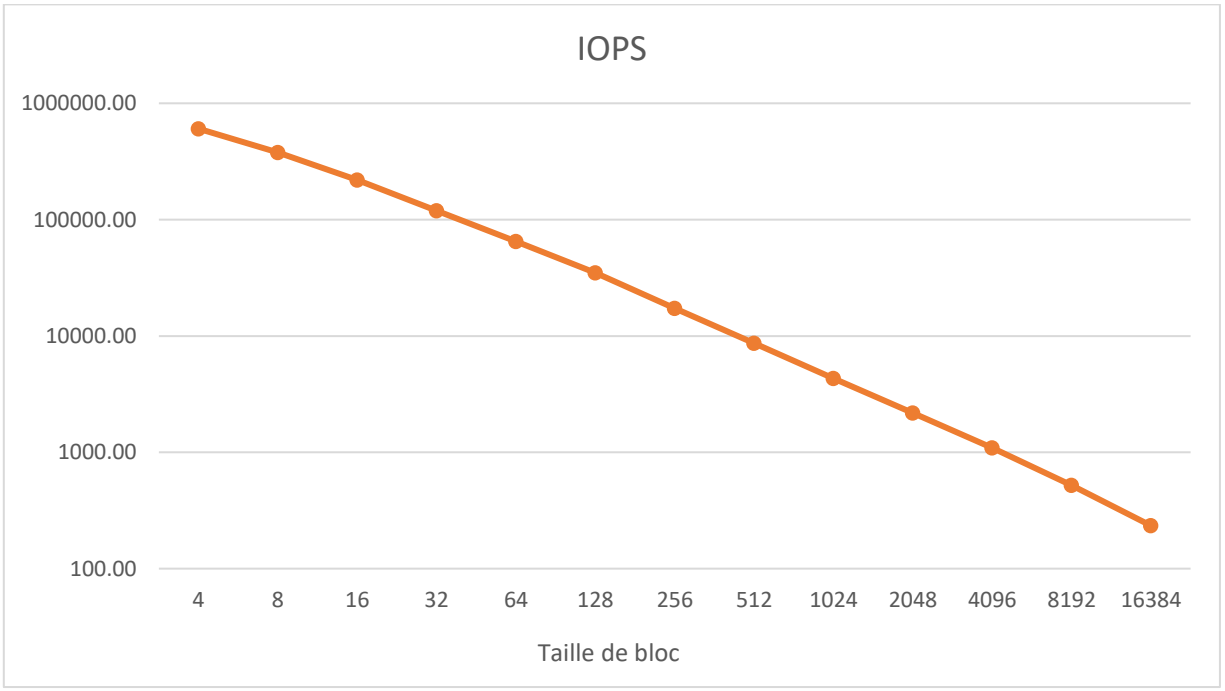

IOPS
1000000.00
100000.00
10000.00
1000.00
100.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc

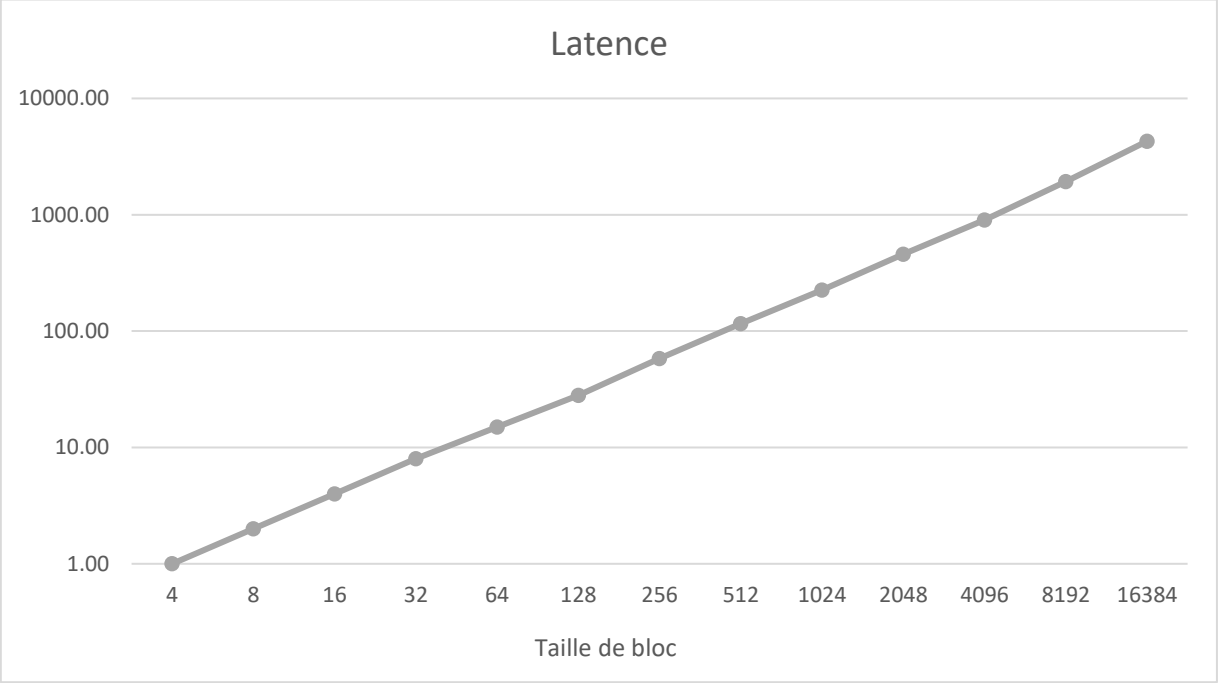


Écriture aléatoire

Taille de bloc (Ko)	Débit écriture (Ko/s)	IOPS (Opérations/s)	Latence (µs)
4	2432731.67	604593.67	1.00
8	3038419.00	377874.33	2.00
16	3513258.00	219786.00	4.00
32	3973984.33	119059.00	8.00
64	4068946.00	64841.33	15.00
128	4500399.33	34846.33	28.00
256	4394896.33	17345.67	58.00
512	4395931.33	8670.67	116.00
1024	4444015.33	4335.33	225.67
2048	4502256.33	2187.00	459.00
4096	4544182.00	1092.33	902.33
8192	4177541.33	521.00	1934.33
16384	3814948.00	234.33	4285.00

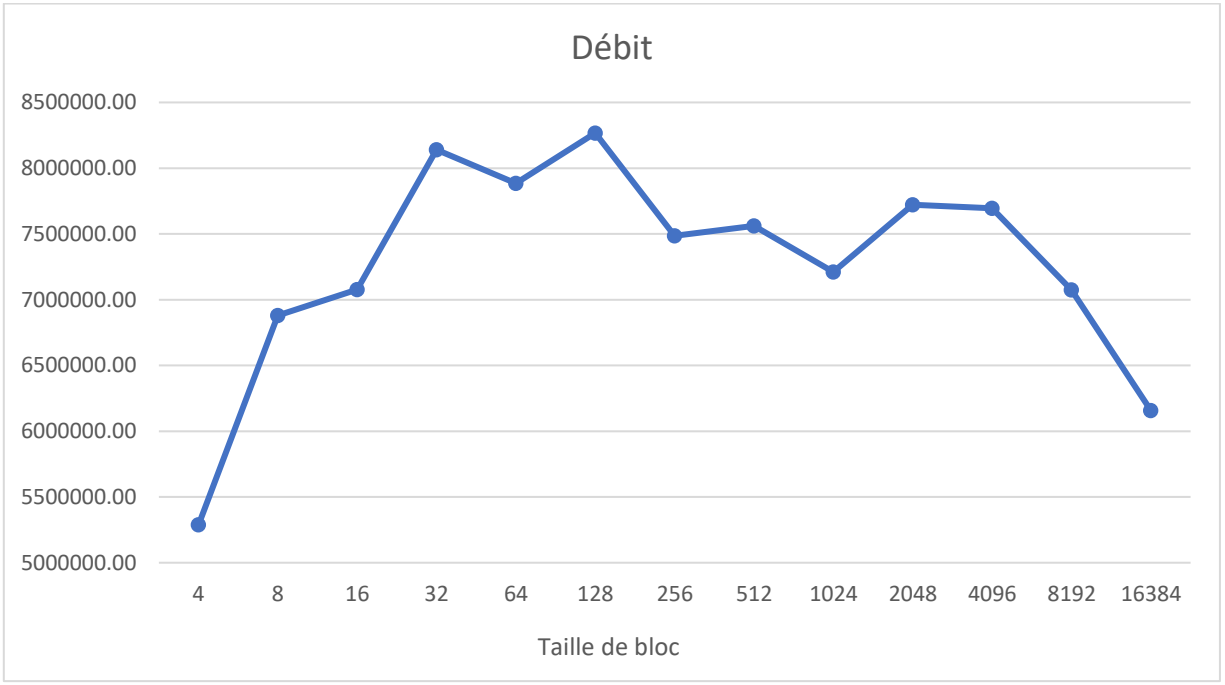

Débit
5000000.00
4500000.00
4000000.00
3500000.00
3000000.00
2500000.00
2000000.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc

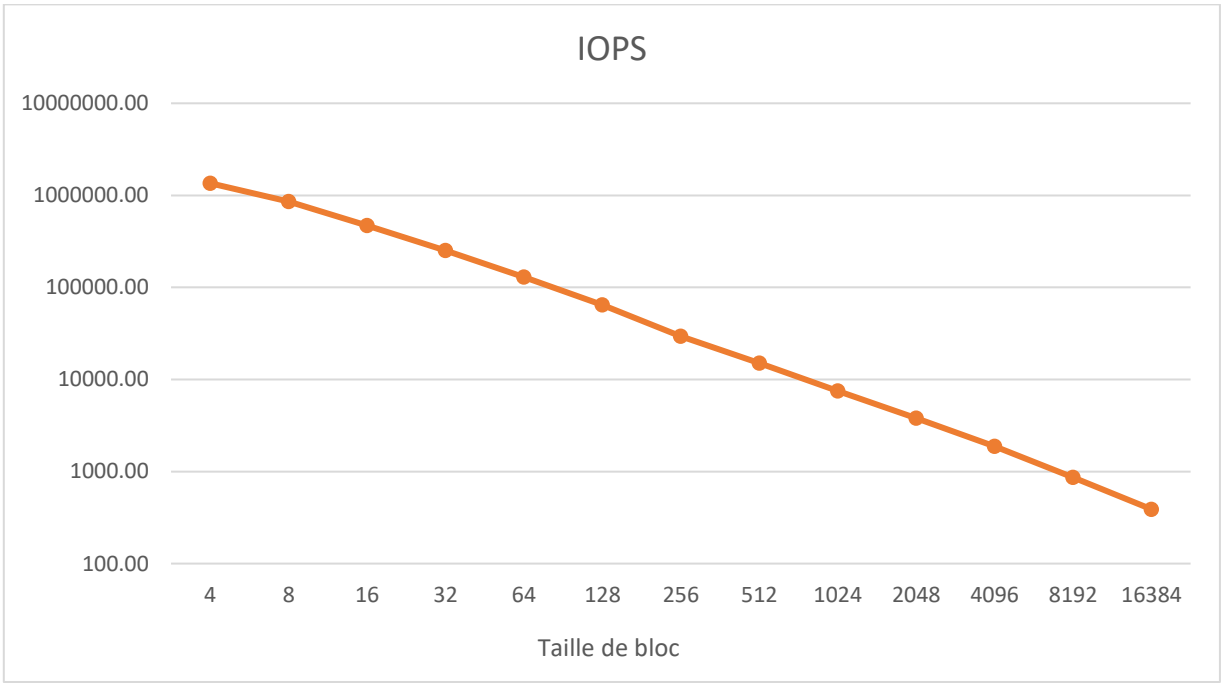

IOPS
1000000.00
100000.00
10000.00
1000.00
100.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc

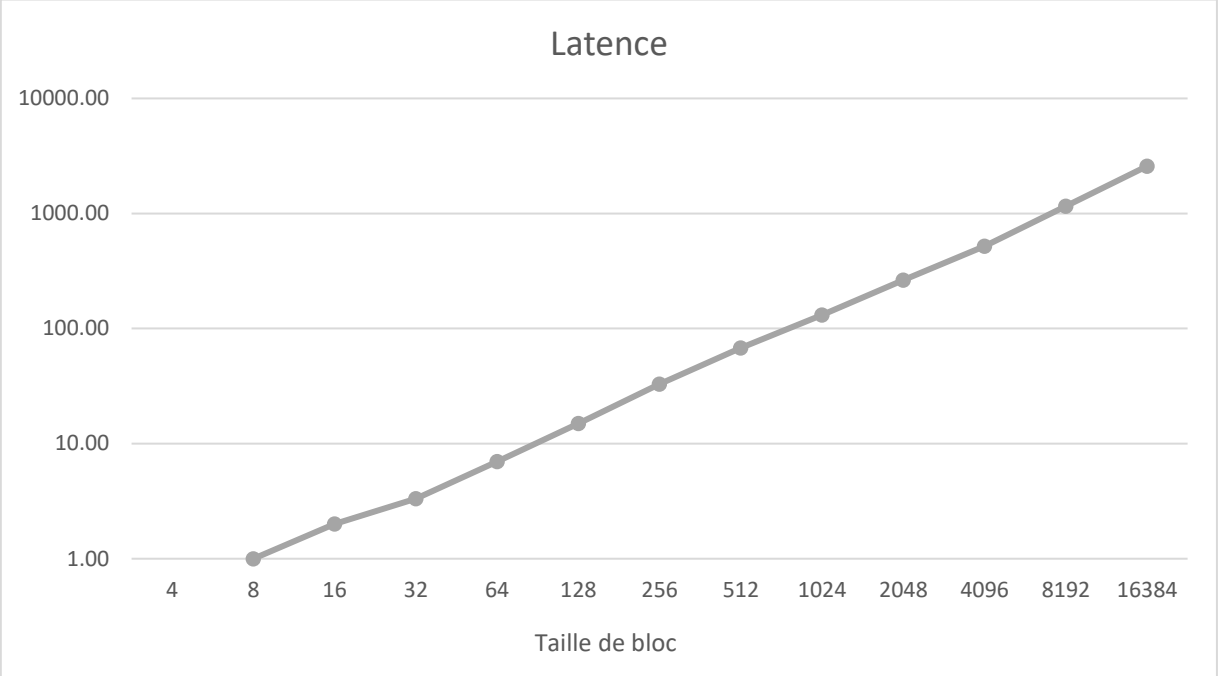


Lecture séquentielle

Taille de bloc (Ko)	Débit écriture (Ko/s)	IOPS (Opérations/s)	Latence (µs)
4	5288174.67	1353584.00	0.00
8	6880236.67	852919.00	1.00
16	7077305.67	469126.33	2.00
32	8141486.33	251149.67	3.33
64	7885341.00	129554.00	7.00
128	8267381.33	64405.33	15.00
256	7484845.67	29424.33	33.00
512	7562034.33	15016.00	67.67
1024	7212035.33	7516.67	131.00
2048	7721195.67	3791.00	263.33
4096	7695169.33	1881.00	519.33
8192	7073959.33	865.00	1155.67
16384	6156552.67	389.33	2577.33

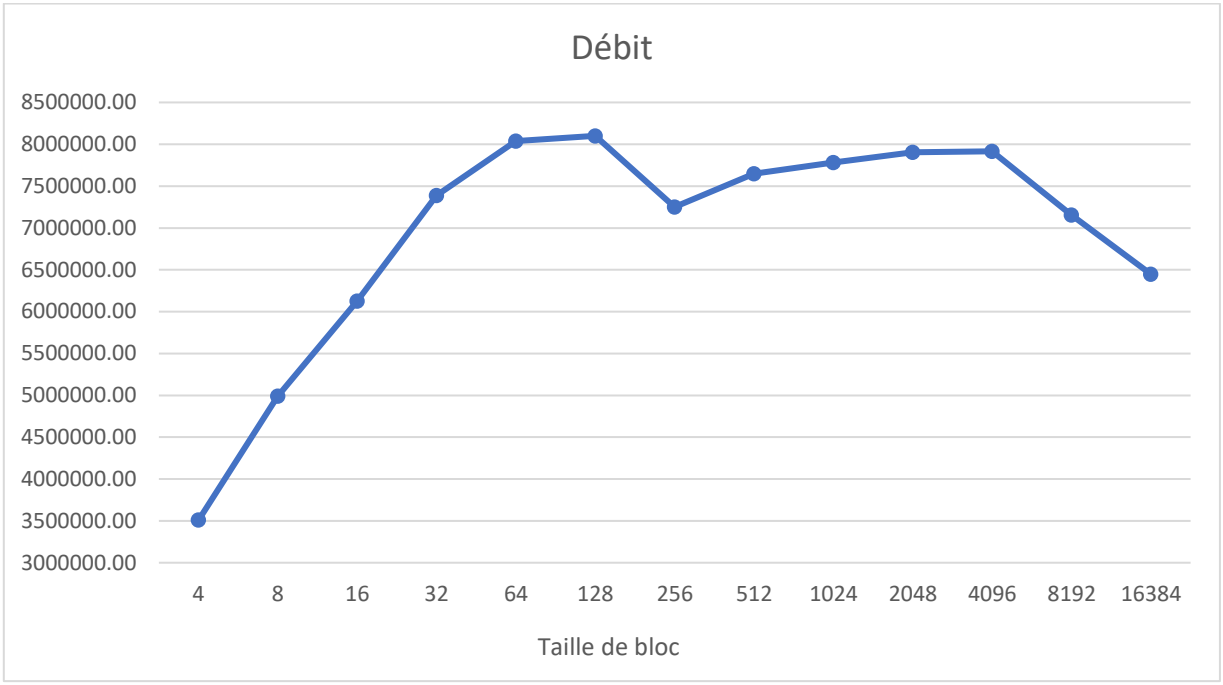

Débit
8500000.00
8000000.00
7500000.00
7000000.00
6500000.00
6000000.00
5500000.00
5000000.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc

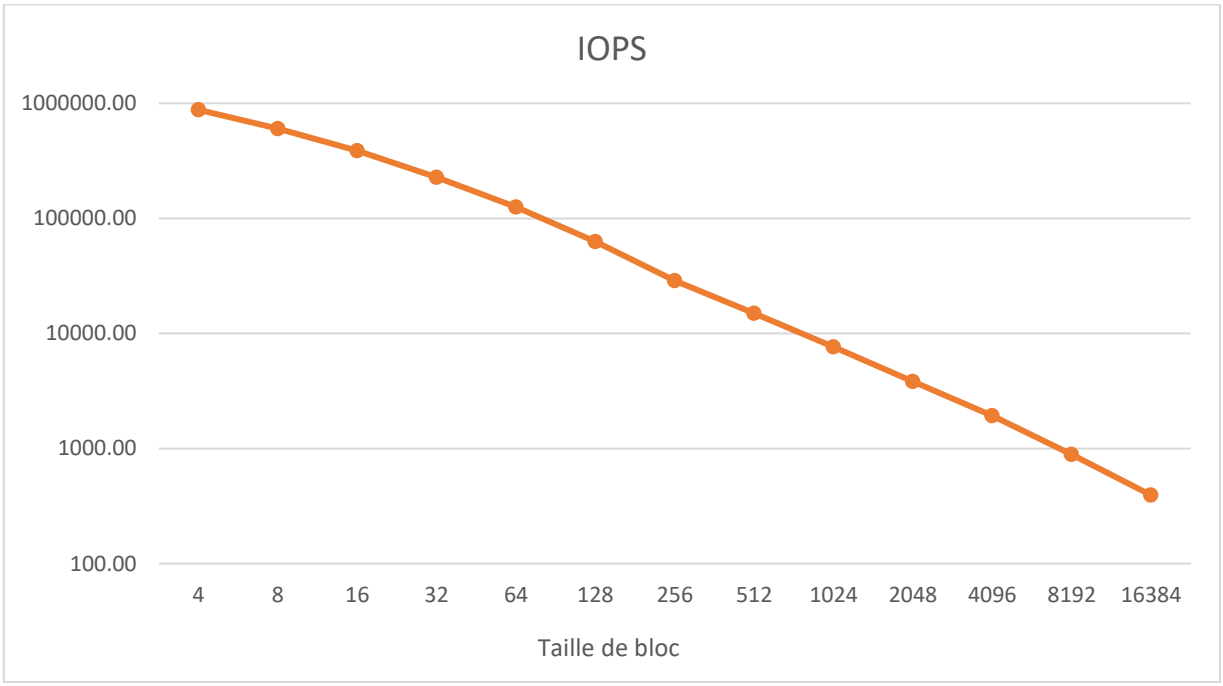

IOPS
10000000.00
1000000.00
100000.00
10000.00
1000.00
100.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc

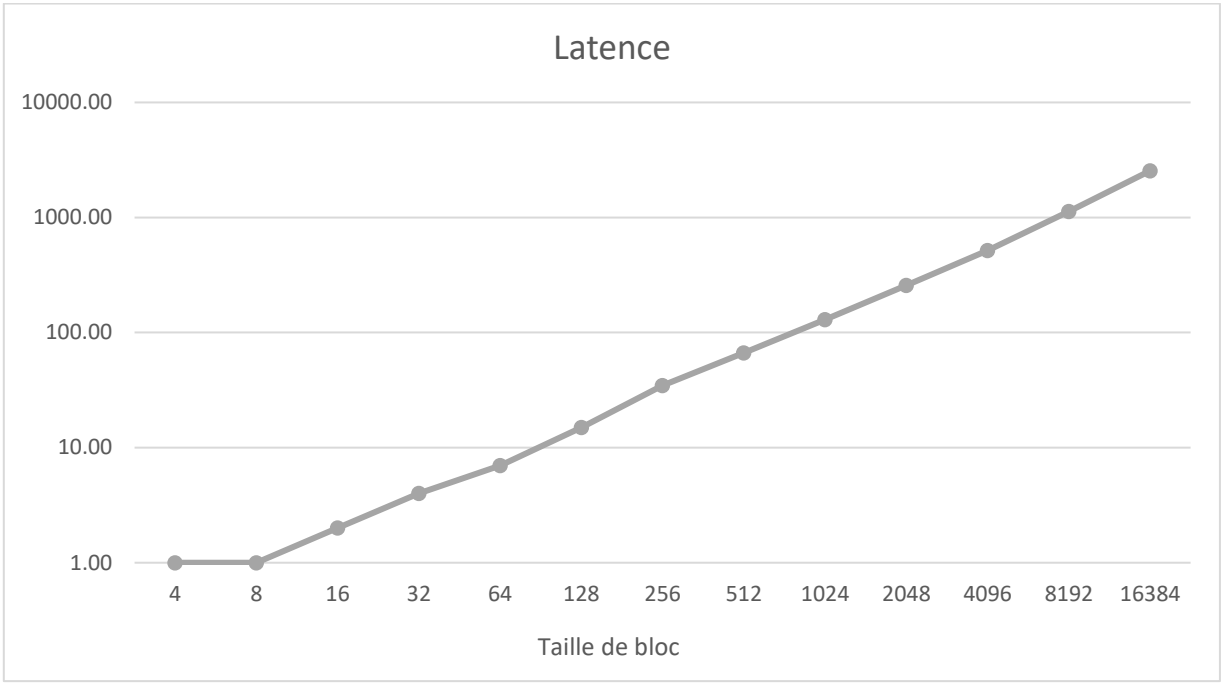


Lecture aléatoire

Taille de bloc (Ko)	Débit écriture (Ko/s)	IOPS (Opérations/s)	Latence (µs)
4	3509232.67	877001.00	1.00
8	4989983.00	601598.67	1.00
16	6125420.33	387458.33	2.00
32	7386716.33	227996.33	4.00
64	8037800.00	125900.67	7.00
128	8100730.67	62946.67	15.00
256	7249456.00	28869.33	34.67
512	7649575.00	14981.33	66.67
1024	7782454.00	7627.33	129.33
2048	7905213.67	3836.67	257.67
4096	7917664.00	1925.67	517.00
8192	7153564.33	891.00	1127.00
16384	6447903.67	393.67	2543.33


Débit
8500000.00
8000000.00
7500000.00
7000000.00
6500000.00
6000000.00
5500000.00
5000000.00
4500000.00
4000000.00
3500000.00
3000000.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc


IOPS
1000000.00
100000.00
10000.00
1000.00
100.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc


Latence
10000.00
1000.00
100.00
10.00
1.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc

Annexe II

Résultats détaillés du volume LVM en allocation dynamique

On trouve dans cette annexe les résultats complets des quatre opérations d'E/S pour les tests effectués sur le volume LVM en mode allocation dynamique.

Données du débit en Ko par seconde

Test 1	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	1392651	1993354	2069605	2182942	2177438	2232794	2200758	2220759	2187988	2192568	2165168	2065305	2003539
Lecture séquentielle	5274180	6818587	7315255	7924303	8383035	8046785	7418032	7466631	7598816	7487317	7416405	6424978	6136222
Lecture aléatoire	3286279	4726573	5983483	7022577	7729501	7877343	7128810	7522122	7619413	7654683	7421231	6477451	6212495
Écriture aléatoire	1031618	2049721	2947490	3130947	3534973	3635914	3750700	3603737	3500281	3744299	3637730	3266427	3242841

Test 2	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	1835539	2006786	2063439	2209724	2243159	2212475	2182396	2240609	2198715	2197277	2150440	2072501	1980425
Lecture séquentielle	5217810	6711831	7563960	8148588	8312008	7953698	7104464	7559597	7446376	7607641	7568154	6893537	6070325
Lecture aléatoire	3254082	4699688	6000467	7053992	7637953	7692467	6961311	7550120	7653620	7714032	7466303	7040491	6172455
Écriture aléatoire	1116730	2110022	2980739	3282442	3481073	3673941	3698173	3730407	3613039	3706930	3645066	3312408	3160722

Test 3	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	1857813	2049829	2181048	2199849	2265962	2217312	2257963	2229111	2251907	2228594	2208020	2060419	2053004
Lecture séquentielle	5234346	6722715	7655902	8263328	8433196	8106629	7530389	7686101	7705245	7694442	7639452	6543129	6031328
Lecture aléatoire	3301104	4769005	6105562	7338442	7924431	7890203	7496901	7437975	7795007	7800514	7764462	6783649	6107310
Écriture aléatoire	1122220	2132188	2975687	3297792	3513748	3731123	3750500	3772981	3804082	3826290	3686562	3321632	3185379

Moyenne	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	1695334.333	2016656.333	2104697.333	2197505	2228853	2220860.333	2213705.667	2230159.667	2212870	2206146.333	2174542.667	2066075	2012322.667
Lecture séquentielle	5242112	6751044.333	7511705.667	8112073	8376079.667	8035704	7350961.667	7570776.333	7583479	7596466.667	7541337	6620548	6079291.667
Lecture aléatoire	3280488.333	4731755.333	6029837.333	7138337	7763961.667	7820004.333	7195674	7503405.667	7689346.667	7723076.333	7550665.333	6767197	6164086.667
Écriture aléatoire	1090189.333	2097310.333	2967972	3237060.333	3509931.333	3680326	3733124.333	3702375	3639134	3759173	3656452.667	3300155.667	3196314

Données des IOPS en opération par seconde

Test 1	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	455951	249708	134806	67038	34524	17062	8384	4213	2152	1056	529	250	121
Lecture séquentielle	1324743	831295	465732	250631	125945	61213	28978	14478	7281	3647	1833	902	376
Lecture aléatoire	825704	594395	374701	213622	120276	59367	28249	14436	7451	3736	1890	910	384
Écriture aléatoire	299737	272960	186197	95786	54912	27956	14343	7016	3686	1792	892	405	197

Test 2	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	459465	254733	134648	69074	34206	17339	8571	4214	2143	1060	538	260	126
Lecture séquentielle	1304563	830058	457691	247758	129435	62803	28107	14374	7222	3686	1829	912	391
Lecture aléatoire	811310	561540	370575	222501	122253	60478	28604	14182	7275	3732	1839	926	398
Écriture aléatoire	307717	272270	186369	104921	49047	28758	14385	6919	3646	1786	932	426	202

Test 2	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	451133	255087	124794	68982	34796	16680	8426	4246	2116	1064	522	253	122
Lecture séquentielle	1317710	837890	455994	251859	127252	62261	27335	14589	7218	3613	1800	900	381
Lecture aléatoire	802610	562410	369543	220970	118686	59520	27986	14460	7388	3641	1839	918	383
Écriture aléatoire	333122	277096	143710	103822	53798	27119	14080	7149	3569	1802	881	408	197

Moyenne	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	455516.3333	253176	131416	68364.66667	34508.66667	17027	8460.333333	4224.333333	2137	1060	529.6666667	254.3333333	123
Lecture séquentielle	1315672	833081	459805.6667	250082.6667	127544	62092.33333	28140	14480.33333	7240.333333	3648.666667	1820.666667	904.6666667	382.6666667
Lecture aléatoire	813208	572781.6667	371606.3333	219031	120405	59788.33333	28279.66667	14359.33333	7371.333333	3703	1856	918	388.3333333
Écriture aléatoire	313525.3333	274108.6667	172092	101509.6667	52585.66667	27944.33333	14269.33333	7028	3633.666667	1793.333333	901.6666667	413	198.6666667

Données de la latence en microseconde

Test 1	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	2	3	7	14	28	57	114	227	457	924	1913	4034	8462
Lecture séquentielle	0	1	2	3	7	15	34	66	135	272	562	1142	2659
Lecture aléatoire	1	1	2	4	8	16	36	68	134	267	552	1105	2681
Écriture aléatoire	3	3	5	11	19	35	72	139	280	558	1193	2515	5417

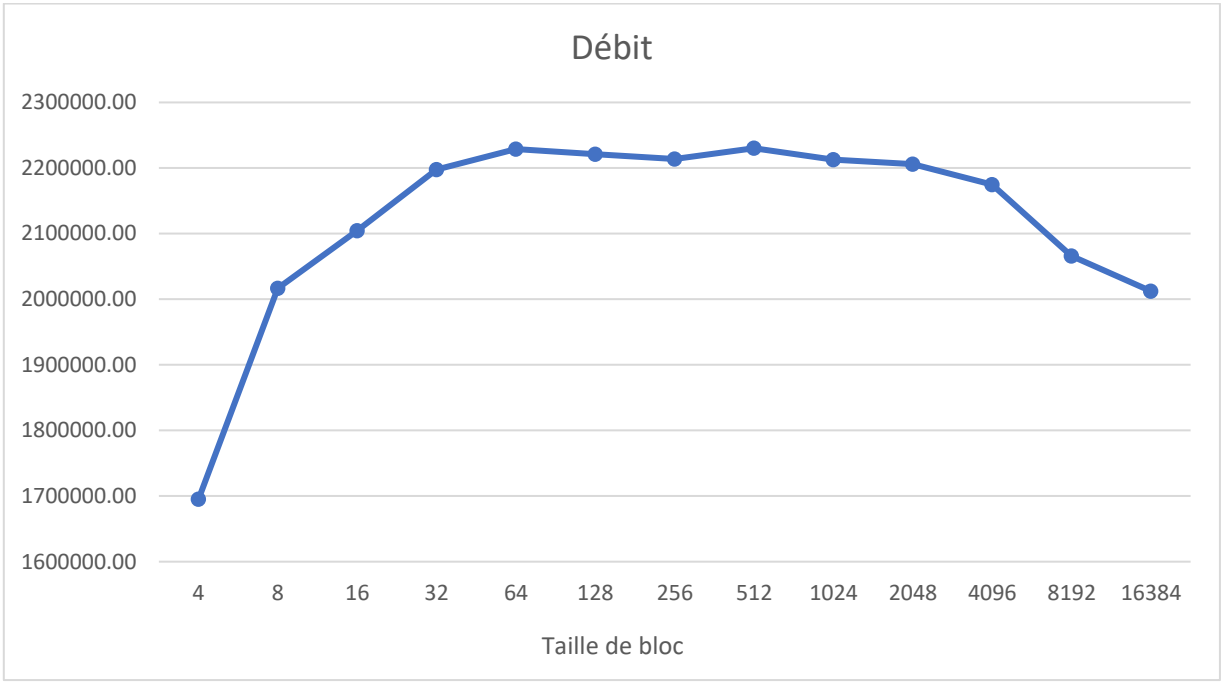
Test 2	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	2	3	8	14	29	59	117	241	466	963	1891	4021	8109
Lecture séquentielle	0	1	2	4	7	15	36	70	138	273	556	1124	2501
Lecture aléatoire	1	1	2	4	8	16	37	70	138	273	549	1093	2522
Écriture aléatoire	3	3	5	10	18	37	70	146	274	575	1114	2490	5009

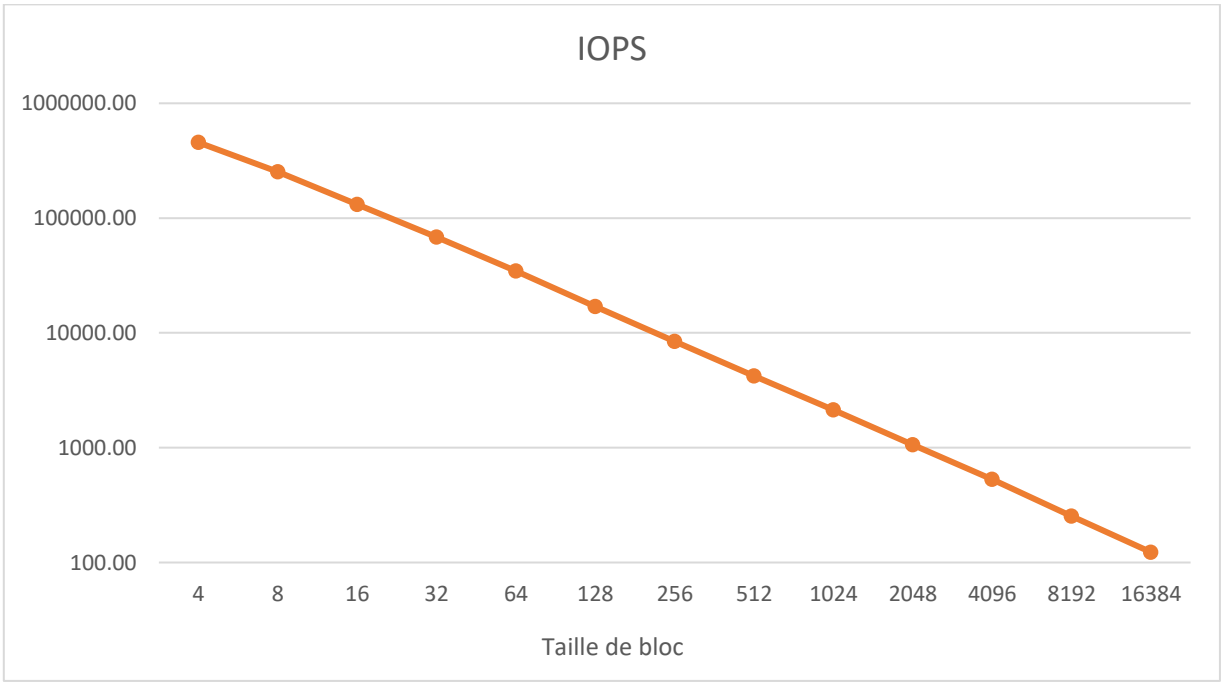
Test 3	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	2	4	7	14	29	57	120	232	469	941	1891	4005	8275
Lecture séquentielle	0	1	2	4	7	16	34	70	135	271	557	1110	2593
Lecture aléatoire	1	1	2	4	8	17	37	68	135	268	549	1095	2557
Écriture aléatoire	2	3	5	10	18	34	74	137	276	555	1121	2474	5246

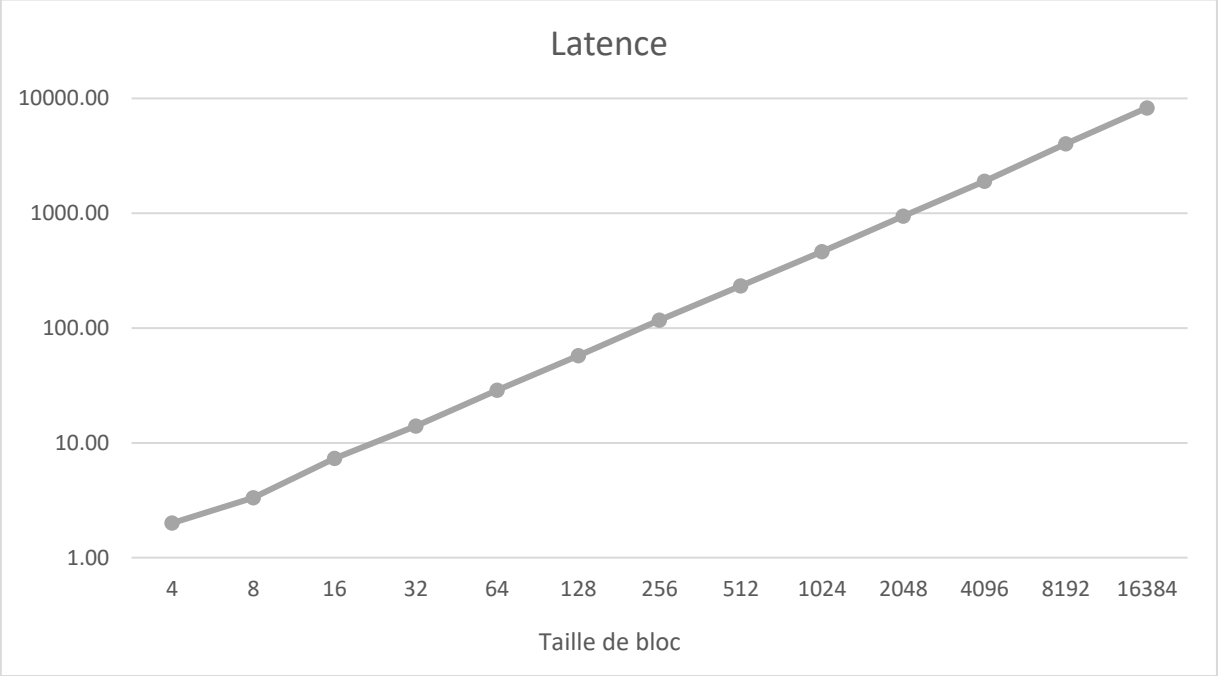
Moyenne	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	2	3.333333333	7.333333333	14	28.66666667	57.66666667	117	233.3333333	464	942.6666667	1898.333333	4020	8282
Lecture séquentielle	0	1	2	3.666666667	7	15.33333333	34.66666667	68.66666667	136	272	558.3333333	1125.333333	2584.333333
Lecture aléatoire	1	1	2	4	8	16.33333333	36.66666667	68.66666667	135.6666667	269.3333333	550	1097.666667	2586.666667
Écriture aléatoire	2.666666667	3	5	10.33333333	18.33333333	35.33333333	72	140.6666667	276.6666667	562.6666667	1142.666667	2493	5224

Écriture séquentielle

Taille de bloc (Ko)	Débit écriture (Ko/s)	IOPS (Opérations/s)	Latence (µs)
4	1695334.33	455516.33	2.00
8	2016656.33	253176.00	3.33
16	2104697.33	131416.00	7.33
32	2197505.00	68364.67	14.00
64	2228853.00	34508.67	28.67
128	2220860.33	17027.00	57.67
256	2213705.67	8460.33	117.00
512	2230159.67	4224.33	233.33
1024	2212870.00	2137.00	464.00
2048	2206146.33	1060.00	942.67
4096	2174542.67	529.67	1898.33
8192	2066075.00	254.33	4020.00
16384	2012322.67	123.00	8282.00

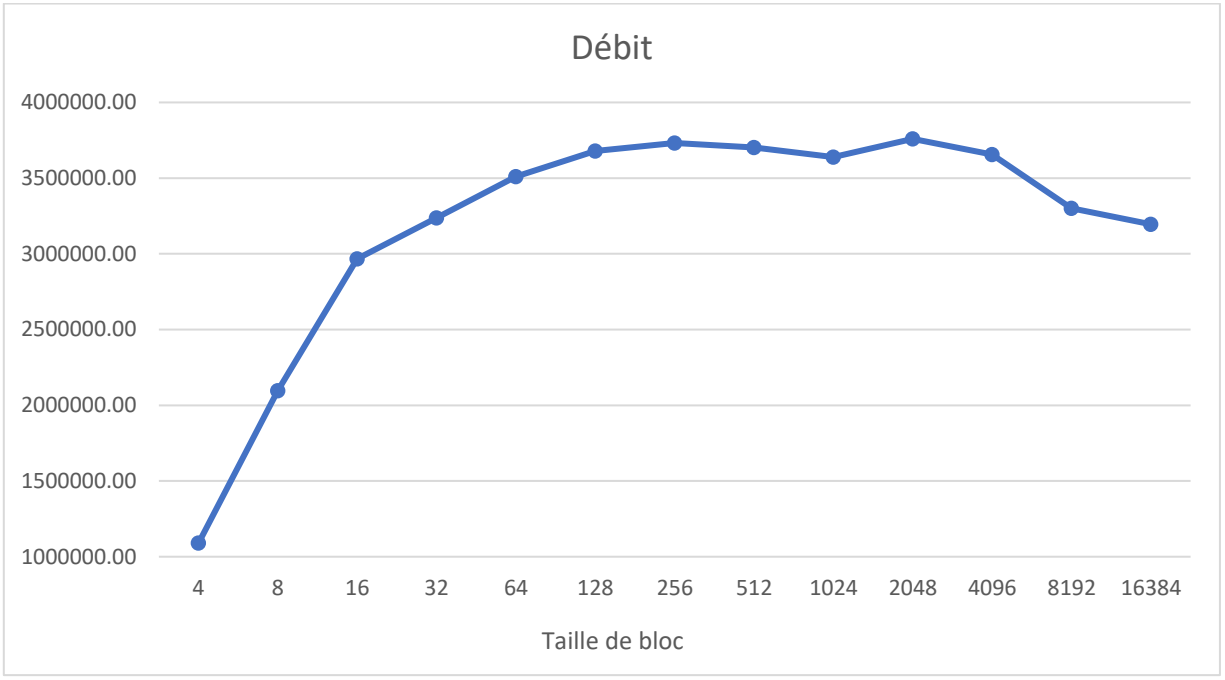

Débit
2300000.00
2200000.00
2100000.00
2000000.00
1900000.00
1800000.00
1700000.00
1600000.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc

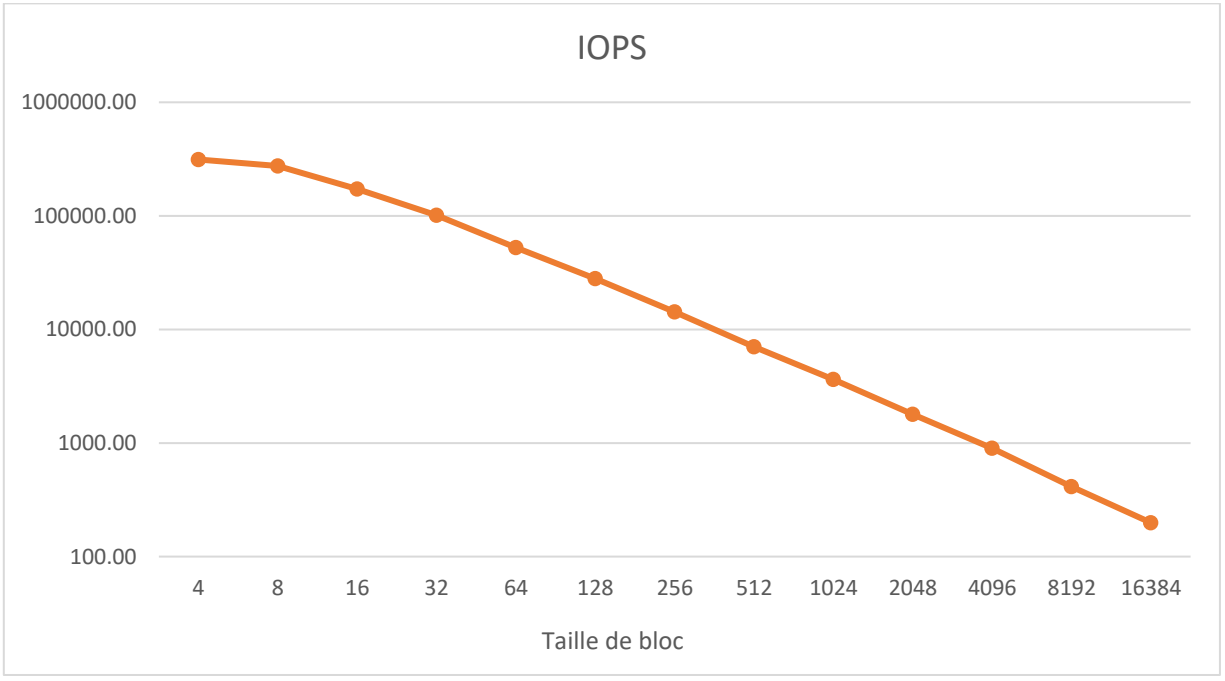

IOPS
1000000.00
100000.00
10000.00
1000.00
100.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc

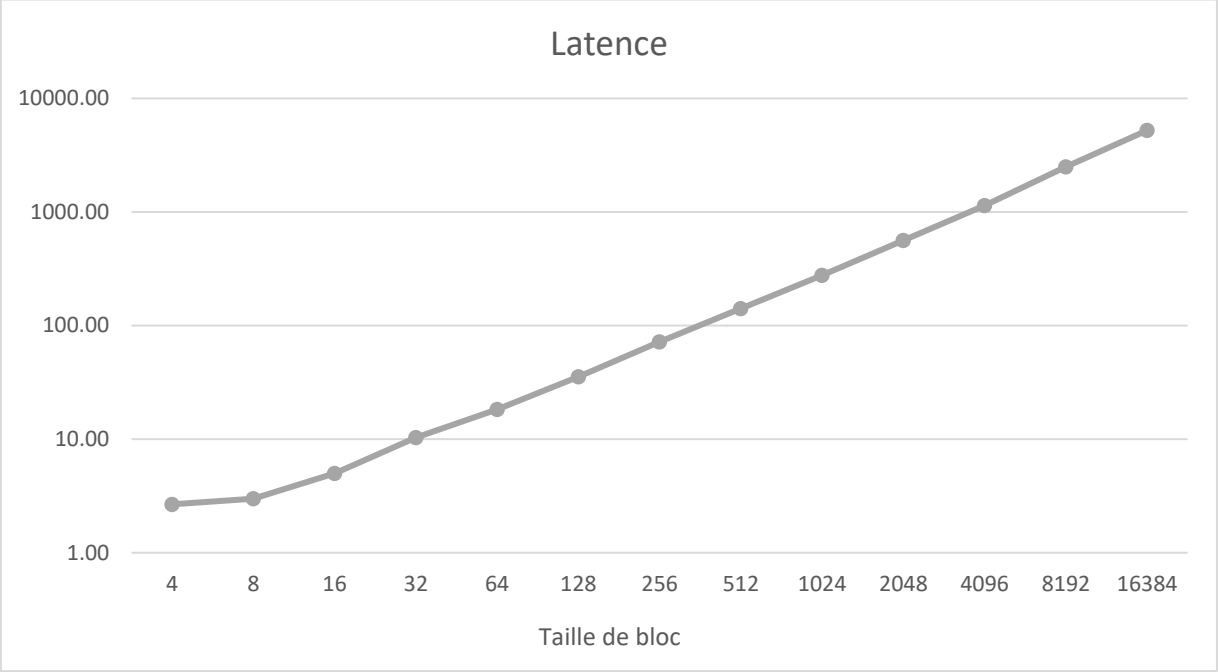


Écriture aléatoire

Taille de bloc (Ko)	Débit écriture (Ko/s)	IOPS (Opérations/s)	Latence (µs)
4	1090189.33	313525.33	2.67
8	2097310.33	274108.67	3.00
16	2967972.00	172092.00	5.00
32	3237060.33	101509.67	10.33
64	3509931.33	52585.67	18.33
128	3680326.00	27944.33	35.33
256	3733124.33	14269.33	72.00
512	3702375.00	7028.00	140.67
1024	3639134.00	3633.67	276.67
2048	3759173.00	1793.33	562.67
4096	3656452.67	901.67	1142.67
8192	3300155.67	413.00	2493.00
16384	3196314.00	198.67	5224.00

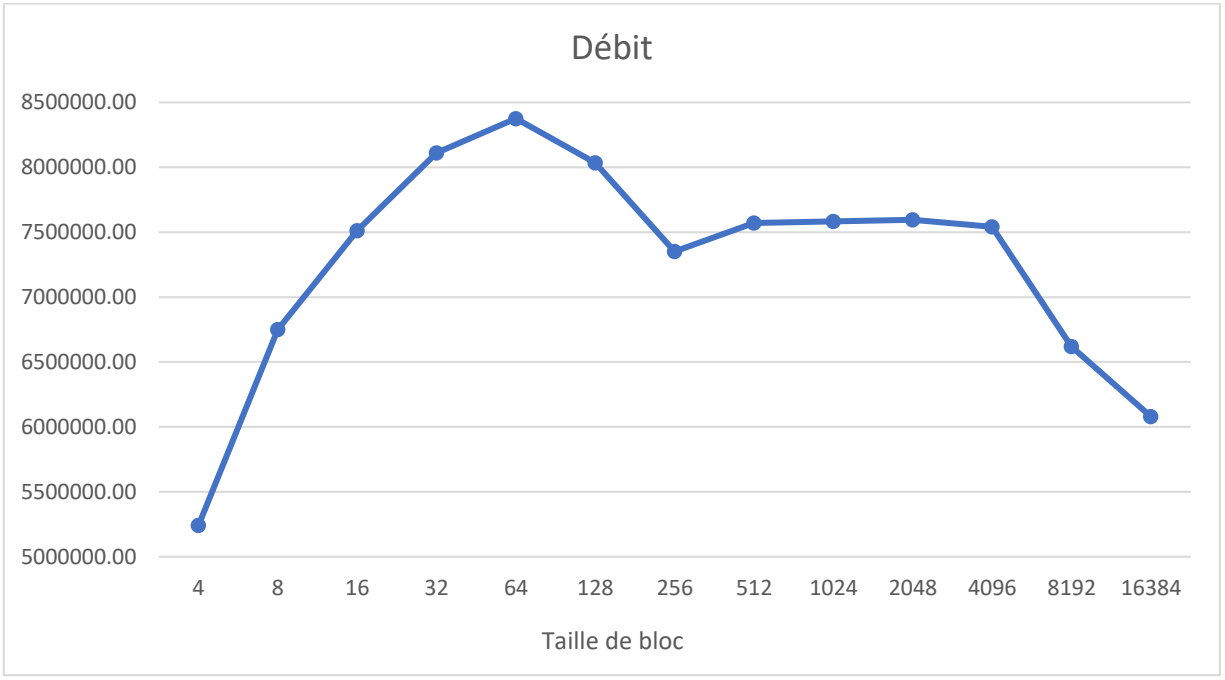

Débit
4000000.00
3500000.00
3000000.00
2500000.00
2000000.00
1500000.00
1000000.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc

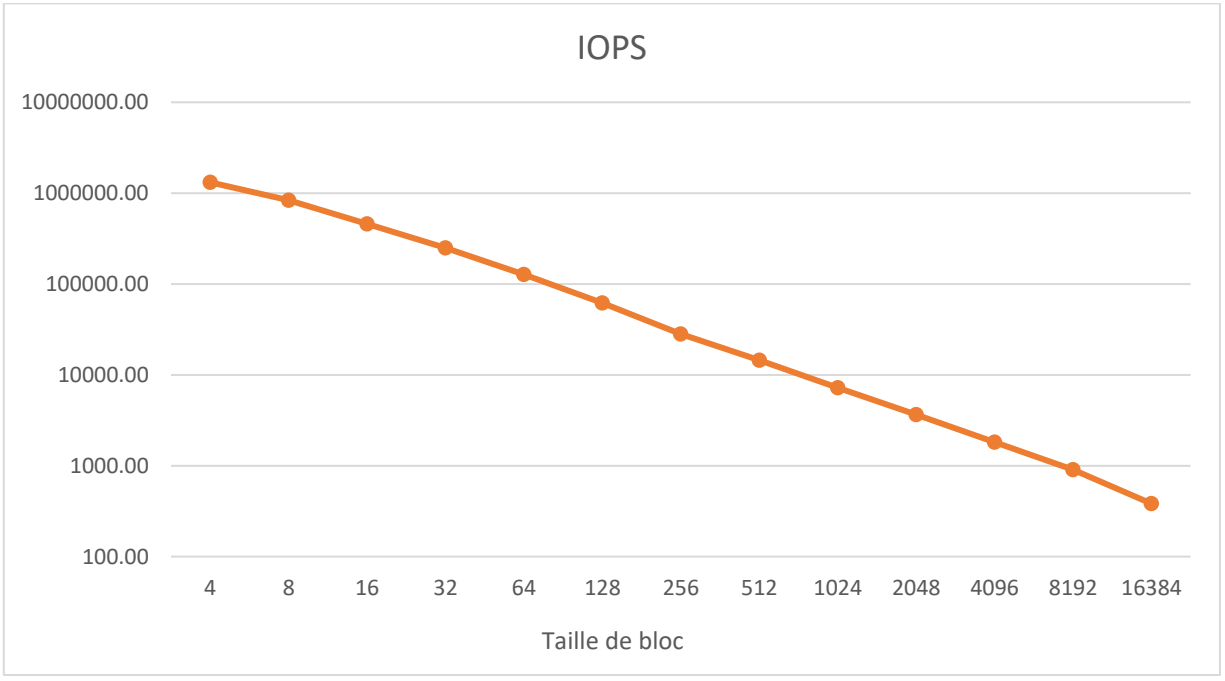

IOPS
1000000.00
100000.00
10000.00
1000.00
100.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc

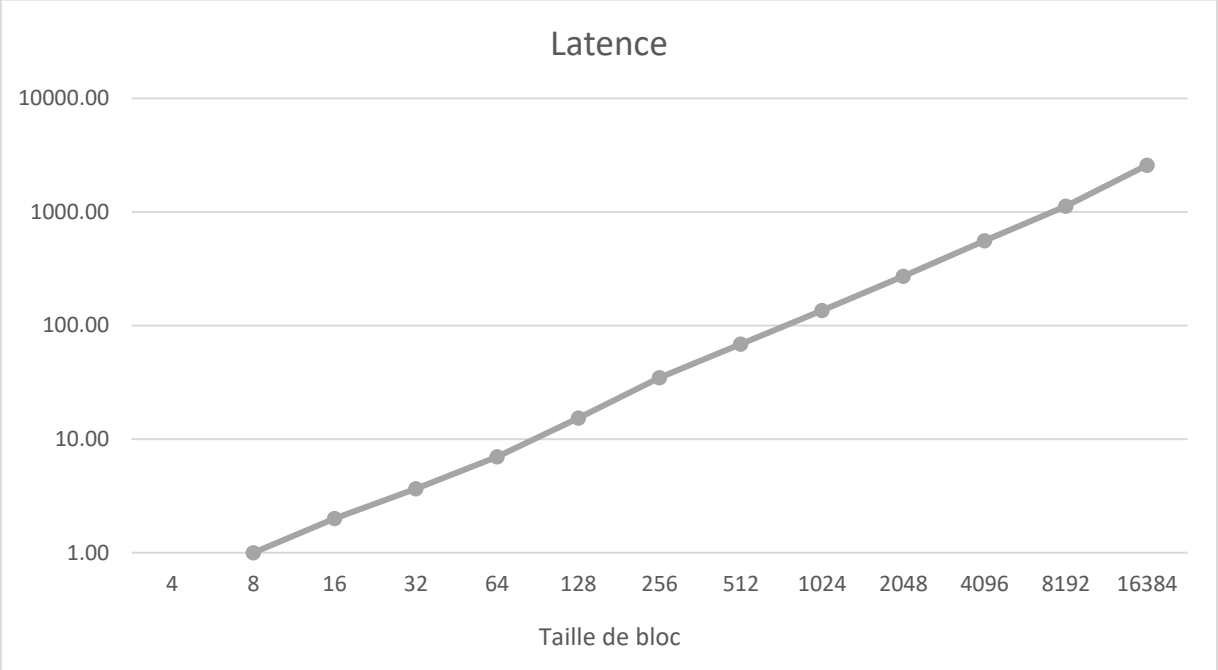


Lecture séquentielle

Taille de bloc (Ko)	Débit écriture (Ko/s)	IOPS (Opérations/s)	Latence (µs)
4	5242112.00	1315672.00	0.00
8	6751044.33	833081.00	1.00
16	7511705.67	459805.67	2.00
32	8112073.00	250082.67	3.67
64	8376079.67	127544.00	7.00
128	8035704.00	62092.33	15.33
256	7350961.67	28140.00	34.67
512	7570776.33	14480.33	68.67
1024	7583479.00	7240.33	136.00
2048	7596466.67	3648.67	272.00
4096	7541337.00	1820.67	558.33
8192	6620548.00	904.67	1125.33
16384	6079291.67	382.67	2584.33

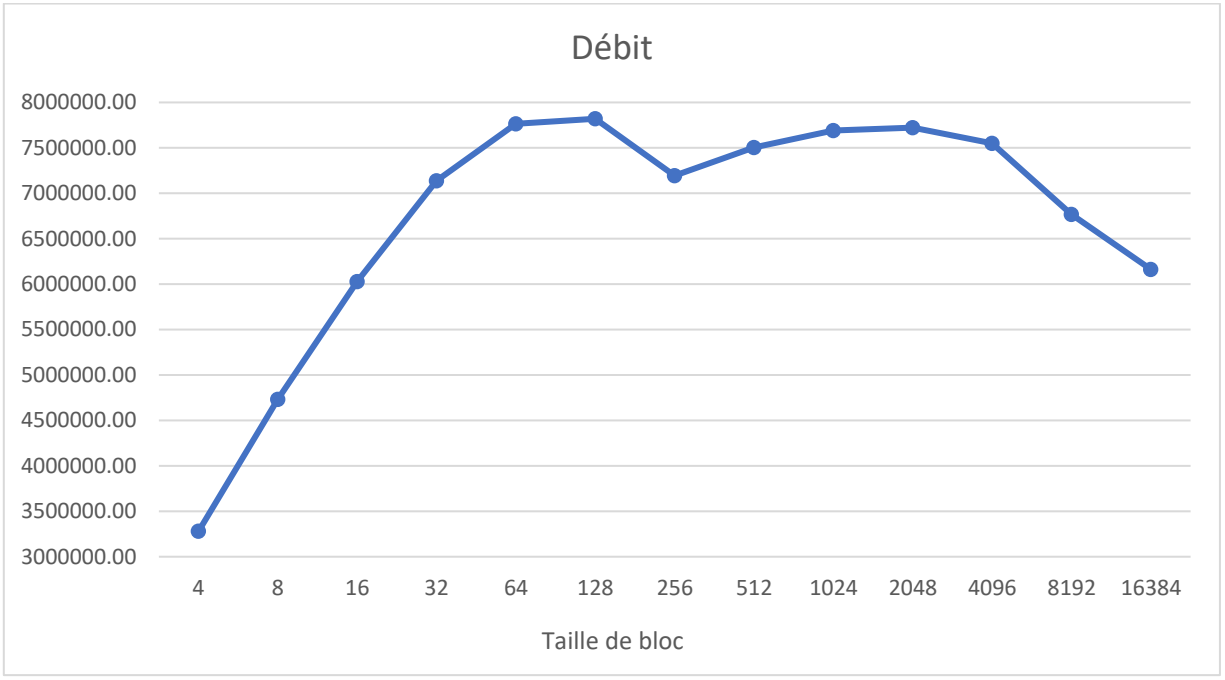

Débit
8500000.00
8000000.00
7500000.00
7000000.00
6500000.00
6000000.00
5500000.00
5000000.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc

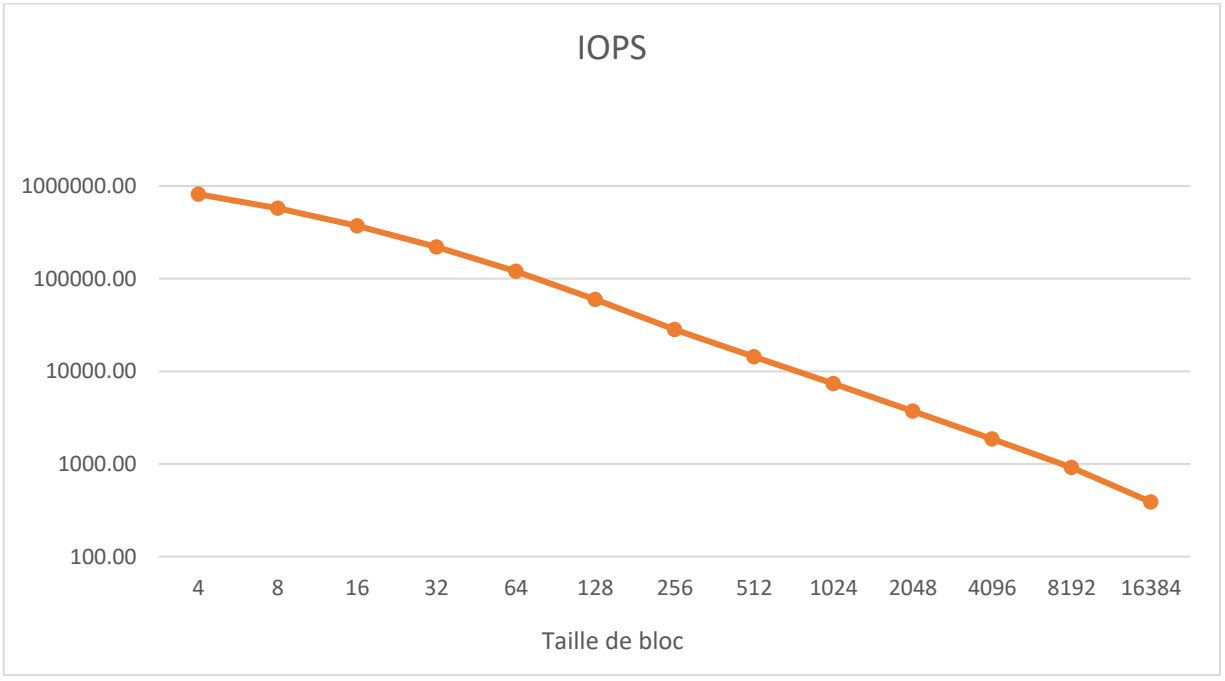

IOPS
10000000.00
1000000.00
100000.00
10000.00
1000.00
100.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc

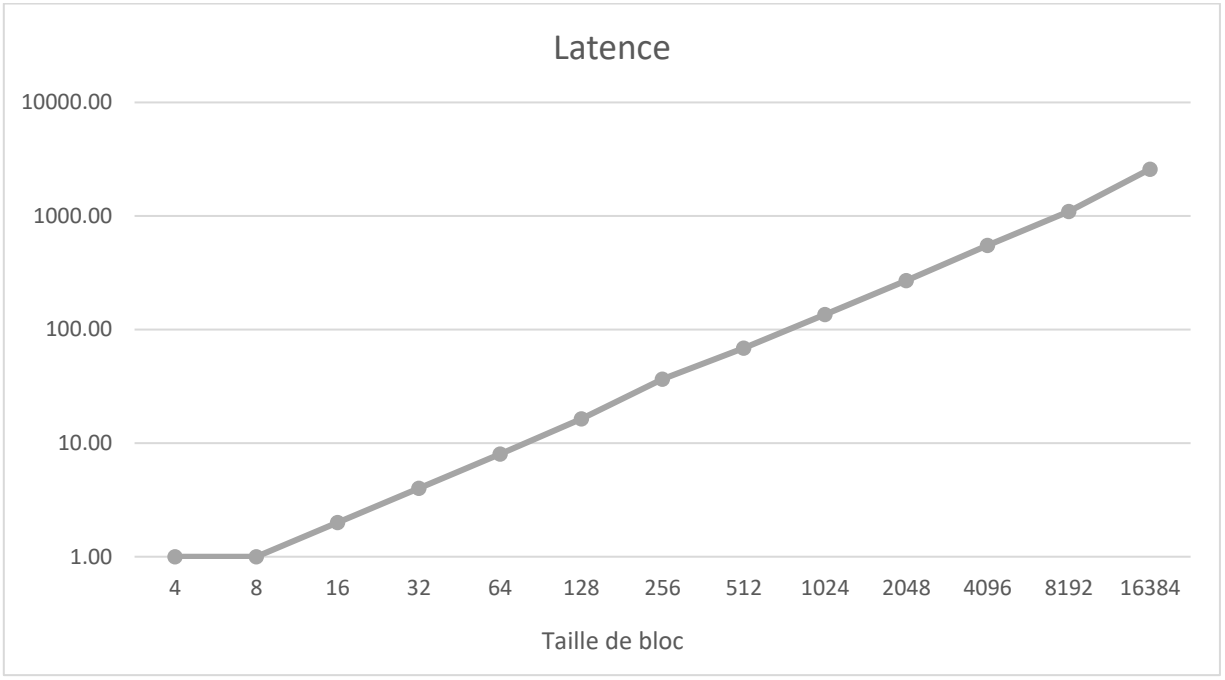


Lecture aléatoire

Taille de bloc (Ko)	Débit écriture (Ko/s)	IOPS (Opérations/s)	Latence (µs)
4	3280488.33	813208.00	1.00
8	4731755.33	572781.67	1.00
16	6029837.33	371606.33	2.00
32	7138337.00	219031.00	4.00
64	7763961.67	120405.00	8.00
128	7820004.33	59788.33	16.33
256	7195674.00	28279.67	36.67
512	7503405.67	14359.33	68.67
1024	7689346.67	7371.33	135.67
2048	7723076.33	3703.00	269.33
4096	7550665.33	1856.00	550.00
8192	6767197.00	918.00	1097.67
16384	6164086.67	388.33	2586.67


Débit
8000000.00
7500000.00
7000000.00
6500000.00
6000000.00
5500000.00
5000000.00
4500000.00
4000000.00
3500000.00
3000000.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc


IOPS
1000000.00
100000.00
10000.00
1000.00
100.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc


Latence
10000.00
1000.00
100.00
10.00
1.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc

Annexe III

Résultats détaillés du fichier image RAW

On trouve dans cette annexe les résultats complets des quatre opérations d'E/S pour les tests effectués sur le fichier image RAW.

Données du débit en Ko par seconde

Test 1	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	1274134	1808812	1840323	1843569	1988806	1944933	1909382	1920050	1954719	1897576	1898947	1825032	1799320
Lecture séquentielle	5190636	6601669	7256078	7799290	8131969	7998052	7465454	7225825	7218614	7484316	7472218	6549625	6269102
Lecture aléatoire	3246074	4474496	5692936	6816815	7584851	7471418	6897572	7281622	7756478	7587541	7765325	6507072	6225482
Écriture aléatoire	1032920	1728500	2397665	2743469	2994819	3077484	3136171	3250995	3244385	3281742	3349260	3014769	3135280

Test 2	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	1761811	1958787	1982518	2087619	2119829	2134806	2059099	2107880	2159025	2100360	2090295	2000873	1916548
Lecture séquentielle	5212229	6727114	7267198	7848033	8232783	8005864	7597821	7718625	7650662	7395012	7262312	6750508	5952545
Lecture aléatoire	3267955	4643621	5921881	6871942	7763316	7554358	7535471	7370318	7638950	7759705	7546856	7137634	6314171
Écriture aléatoire	1053194	1731763	2555751	2746919	2983460	3135487	3205174	3244738	3285804	3280183	3365664	3126485	3082425

Test 3	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	1776765	1972616	1979633	2114141	2071069	2190704	2082581	2106576	2058978	2098692	2028440	1966738	1846139
Lecture séquentielle	5361890	6716044	7162011	8009962	8157965	7596726	6931585	7433336	7681251	7469977	7546867	6642910	5753752
Lecture aléatoire	3231299	4708677	5824904	7036473	7919754	7682606	6876362	7486828	7549855	7595680	7751547	6707666	5953219
Écriture aléatoire	1086468	1807547	2504636	2768686	2972179	3111990	3052649	3246456	3296890	3366734	3403981	3128574	3009024

Moyenne	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	1604236.667	1913405	1934158	2015109.667	2059901.333	2090147.667	2017020.667	2044835.333	2057574	2032209.333	2005894	1930881	1854002.333
Lecture séquentielle	5254918.333	6681609	7228429	7885761.667	8174239	7866880.667	7331620	7459262	7516842.333	7449768.333	7427132.333	6647681	5991799.667
Lecture aléatoire	3248442.667	4608931.333	5813240.333	6908410	7755973.667	7569460.667	7103135	7379589.333	7648427.667	7647642	7687909.333	6784124	6164290.667
Écriture aléatoire	1057527.333	1755936.667	2486017.333	2753024.667	2983486	3108320.333	3131331.333	3247396.333	3275693	3309553	3372968.333	3089942.667	3075576.333

Données des IOPS en opération par seconde

Test 1	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	438641	239391	124908	63542	33184	16246	8044	4065	2036	1015	510	239	117
Lecture séquentielle	1286306	829406	470331	251833	128278	61489	27429	14520	7306	3584	1847	783	375
Lecture aléatoire	820435	590890	375784	223195	125166	60686	27051	14396	7632	3637	1892	807	379
Écriture aléatoire	275492	237839	155575	84730	47115	24233	12425	6175	3256	1614	823	383	186

Test 2	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	437893	242146	127848	65369	32212	16709	8072	4010	2061	1014	511	236	115
Lecture séquentielle	1305440	838463	452812	248244	129402	61246	28468	13964	7115	3608	1784	779	375
Lecture aléatoire	830279	598721	377391	218951	121786	59089	28573	14733	7267	3722	1790	773	382
Écriture aléatoire	267593	219340	151500	90167	47170	23960	12130	6058	3213	1629	819	376	184

Test 3	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	451746	243008	130331	65074	33284	16290	8414	4081	2042	1023	511	238	117
Lecture séquentielle	1296427	834927	467980	242876	127534	62077	28631	14233	7497	3809	1801	771	362
Lecture aléatoire	820443	592121	377185	216130	117378	61968	28830	14082	7610	3724	1860	789	370
Écriture aléatoire	278176	230253	158957	85475	45782	24245	12321	6267	3265	1610	822	373	184

Moyenne	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	442760	241515	127695.6667	64661.66667	32893.33333	16415	8176.666667	4052	2046.333333	1017.333333	510.6666667	237.6666667	116.3333333
Lecture séquentielle	1296057.667	834265.3333	463707.6667	247651	128404.6667	61604	28176	14239	7306	3667	1810.666667	777.6666667	370.6666667
Lecture aléatoire	823719	593910.6667	376786.6667	219425.3333	121443.3333	60581	28151.33333	14403.66667	7503	3694.333333	1847.333333	789.6666667	377
Écriture aléatoire	273753.6667	229144	155344	86790.66667	46689	24146	12292	6166.666667	3244.666667	1617.666667	821.3333333	377.3333333	184.6666667

Données de la latence en microseconde

Test 1	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	2	4	7	15	29	60	122	244	484	964	1982	4211	8518
Lecture séquentielle	0	1	2	4	7	16	34	68	139	275	541	1293	2730
Lecture aléatoire	1	1	2	4	8	16	36	71	135	275	528	1247	2686
Écriture aléatoire	3	4	6	11	22	40	81	159	315	623	1228	2663	5585

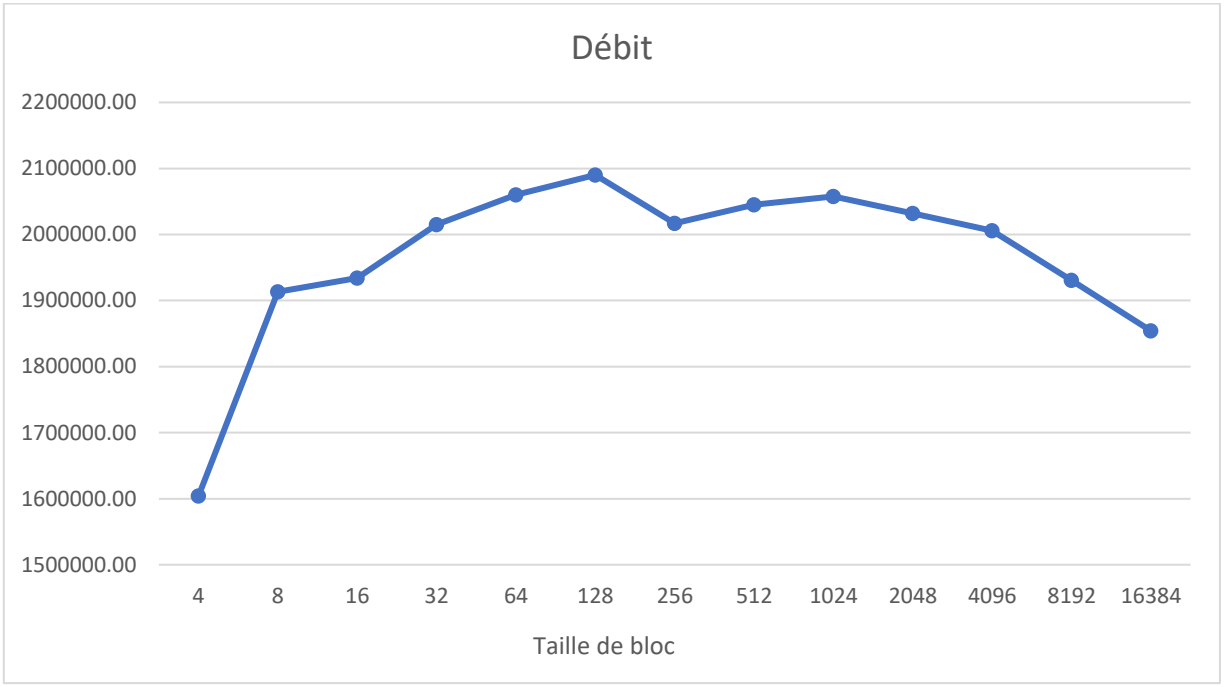
Test 2	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	2	4	7	15	30	61	123	244	489	974	1934	4246	8533
Lecture séquentielle	0	1	2	3	7	16	35	68	134	273	543	1262	2738
Lecture aléatoire	1	1	2	4	8	16	34	68	133	277	545	1232	2701
Écriture aléatoire	3	4	6	11	21	40	79	158	307	623	1225	2635	5371

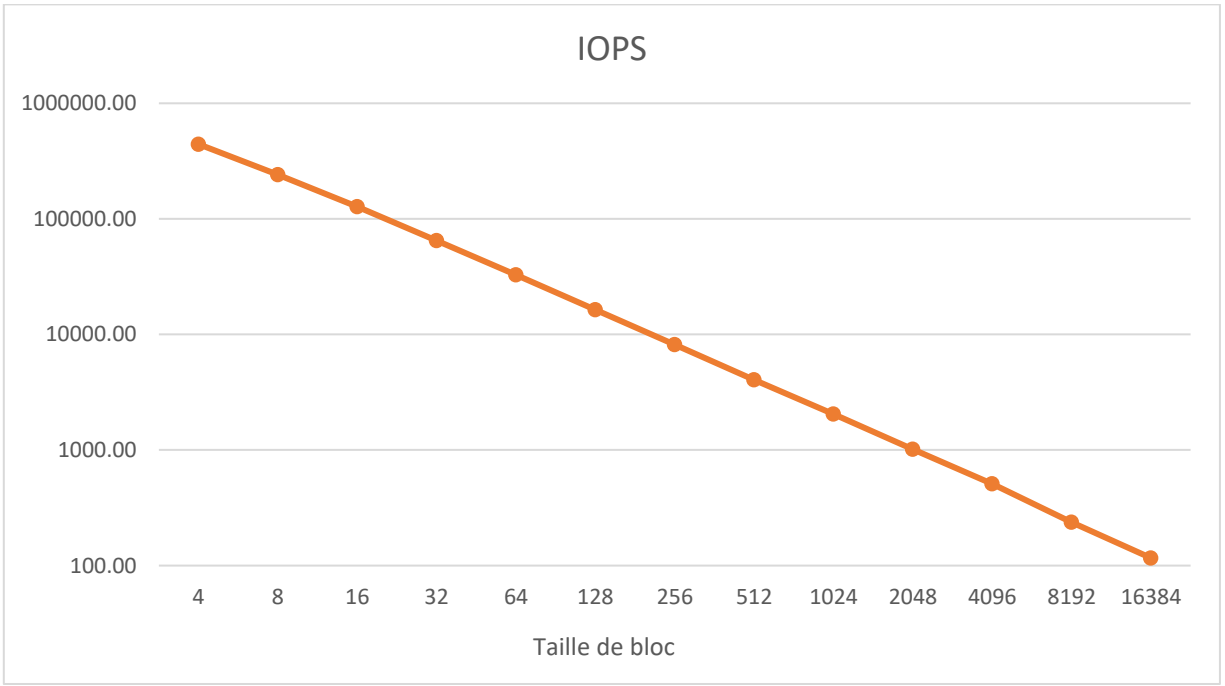
Test 3	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	2	4	7	14	30	60	119	245	479	976	1941	4221	8601
Lecture séquentielle	0	1	2	3	7	16	34	67	133	269	561	1291	2791
Lecture aléatoire	1	1	2	4	8	17	34	66	138	260	553	1247	2769
Écriture aléatoire	3	4	6	11	20	40	80	160	310	606	1227	2612	5452

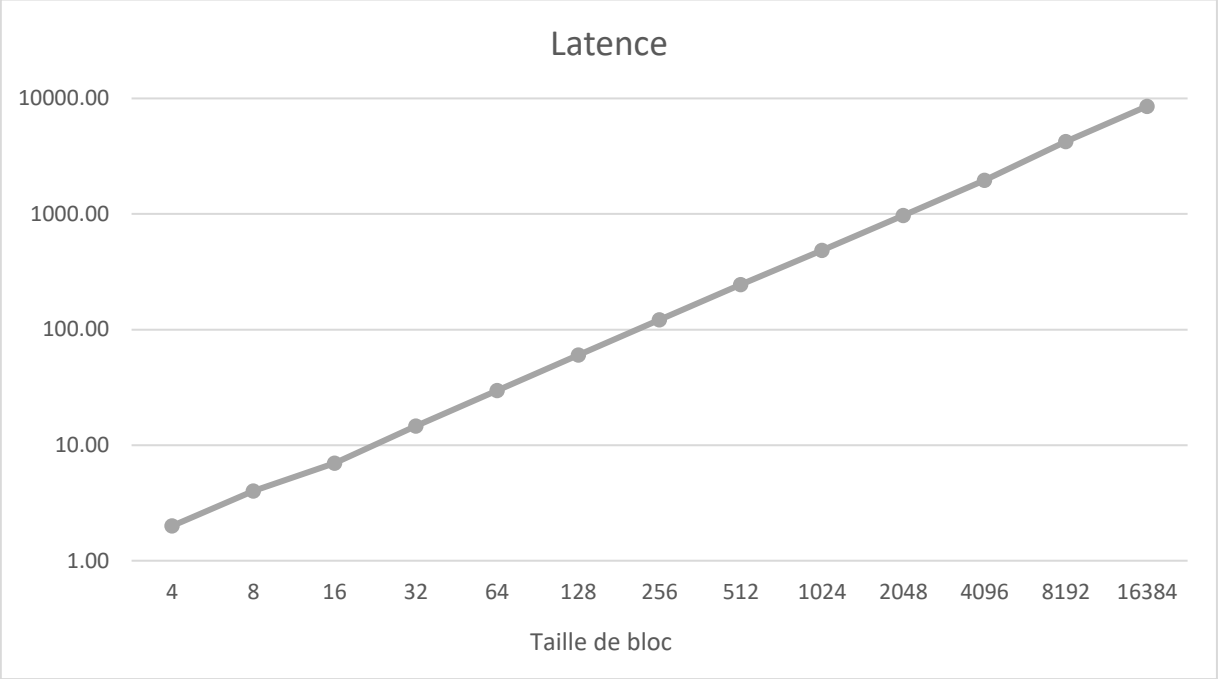
Moyenne	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384
Écriture séquentielle	2	4	7	14.66667	29.66667	60.33333	121.3333	244.3333	484	971.3333	1952.333	4226	8550.667
Lecture séquentielle	0	1	2	3.333333	7	16	34.33333	67.66667	135.3333	272.3333	548.3333	1282	2753
Lecture aléatoire	1	1	2	4	8	16.33333	34.66667	68.33333	135.3333	270.6667	542	1242	2718.667
Écriture aléatoire	3	4	6	11	21	40	80	159	310.6667	617.3333	1226.667	2636.667	5469.333

Écriture séquentielle

Taille de bloc (Ko)	Débit écriture (Ko/s)	IOPS (Opérations/s)	Latence (µs)
4	1604236.67	442760.00	2.00
8	1913405.00	241515.00	4.00
16	1934158.00	127695.67	7.00
32	2015109.67	64661.67	14.67
64	2059901.33	32893.33	29.67
128	2090147.67	16415.00	60.33
256	2017020.67	8176.67	121.33
512	2044835.33	4052.00	244.33
1024	2057574.00	2046.33	484.00
2048	2032209.33	1017.33	971.33
4096	2005894.00	510.67	1952.33
8192	1930881.00	237.67	4226.00
16384	1854002.33	116.33	8550.67

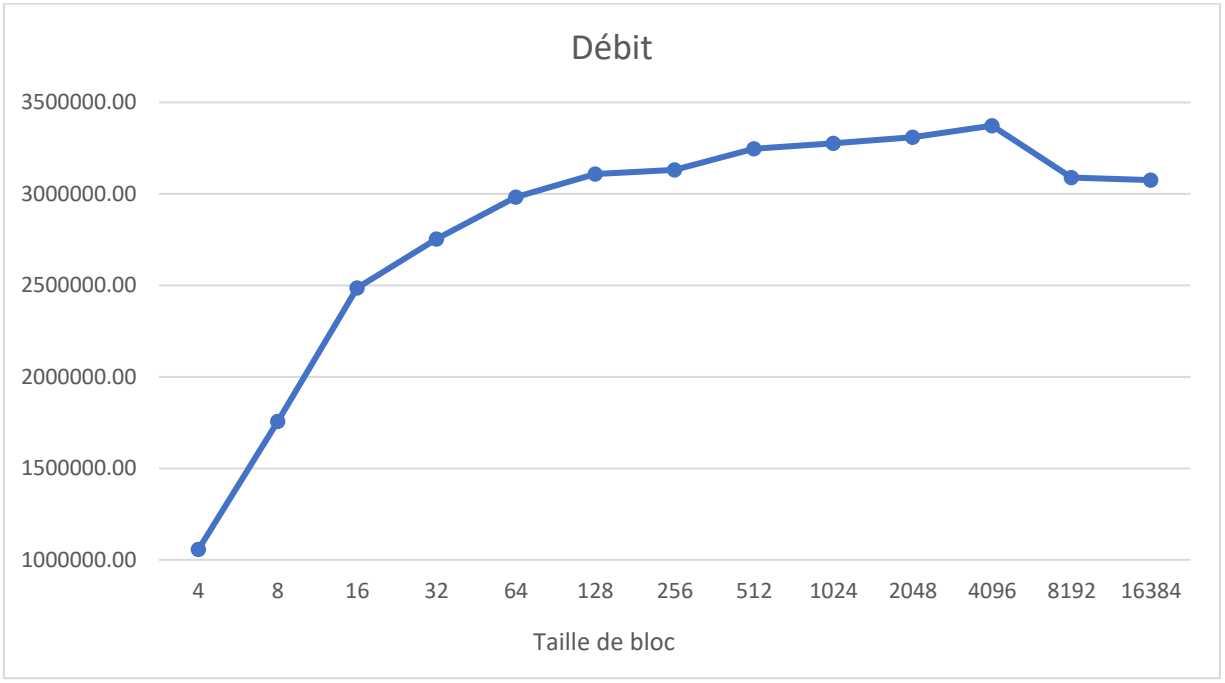

Débit
2200000.00
2100000.00
2000000.00
1900000.00
1800000.00
1700000.00
1600000.00
1500000.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc

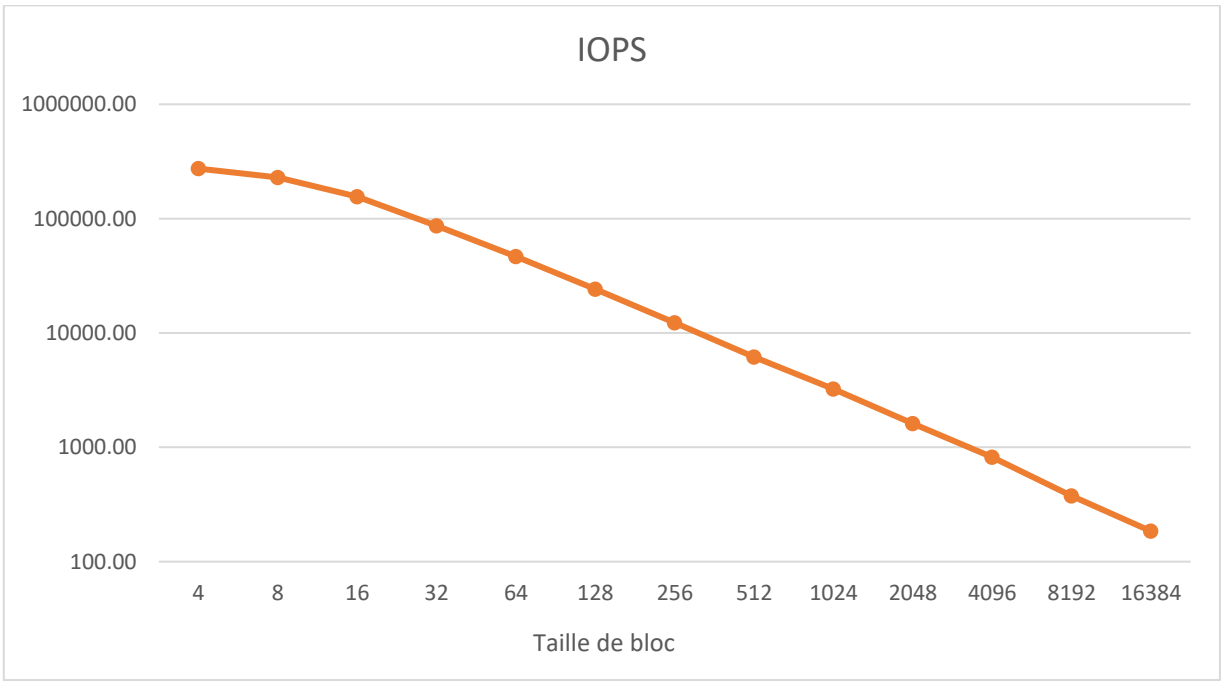

IOPS
1000000.00
100000.00
10000.00
1000.00
100.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc

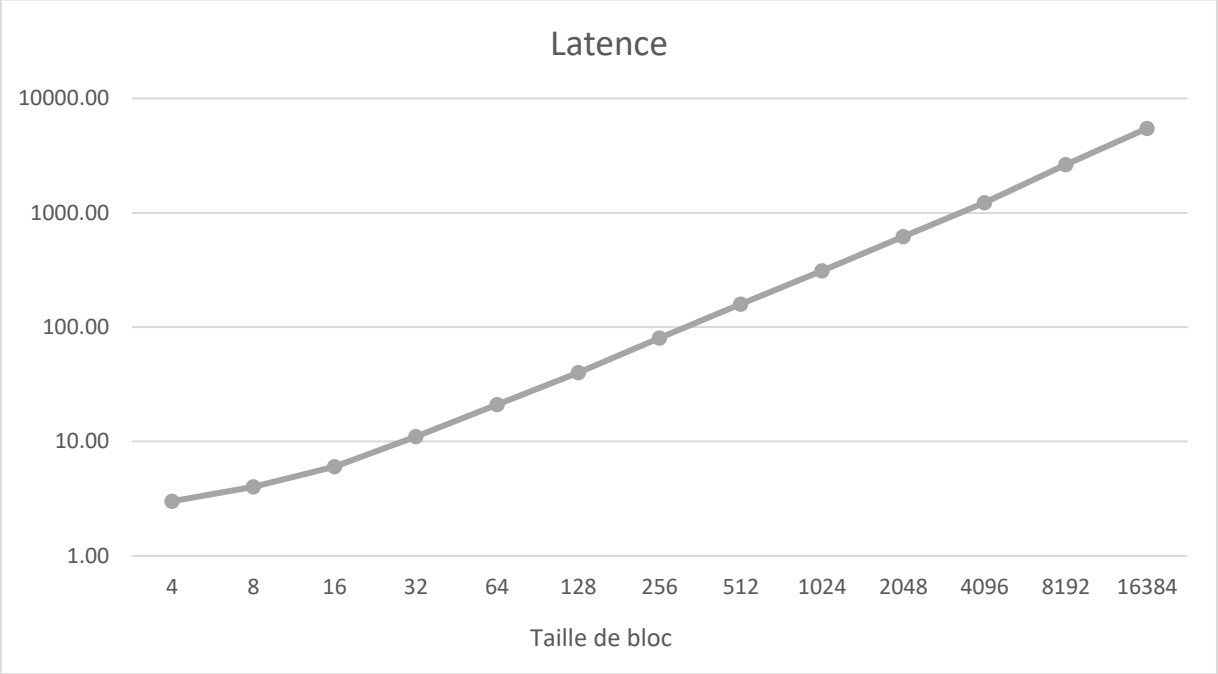


Écriture aléatoire

Taille de bloc (Ko)	Débit écriture (Ko/s)	IOPS (Opérations/s)	Latence (µs)
4	1057527.33	273753.67	3.00
8	1755936.67	229144.00	4.00
16	2486017.33	155344.00	6.00
32	2753024.67	86790.67	11.00
64	2983486.00	46689.00	21.00
128	3108320.33	24146.00	40.00
256	3131331.33	12292.00	80.00
512	3247396.33	6166.67	159.00
1024	3275693.00	3244.67	310.67
2048	3309553.00	1617.67	617.33
4096	3372968.33	821.33	1226.67
8192	3089942.67	377.33	2636.67
16384	3075576.33	184.67	5469.33

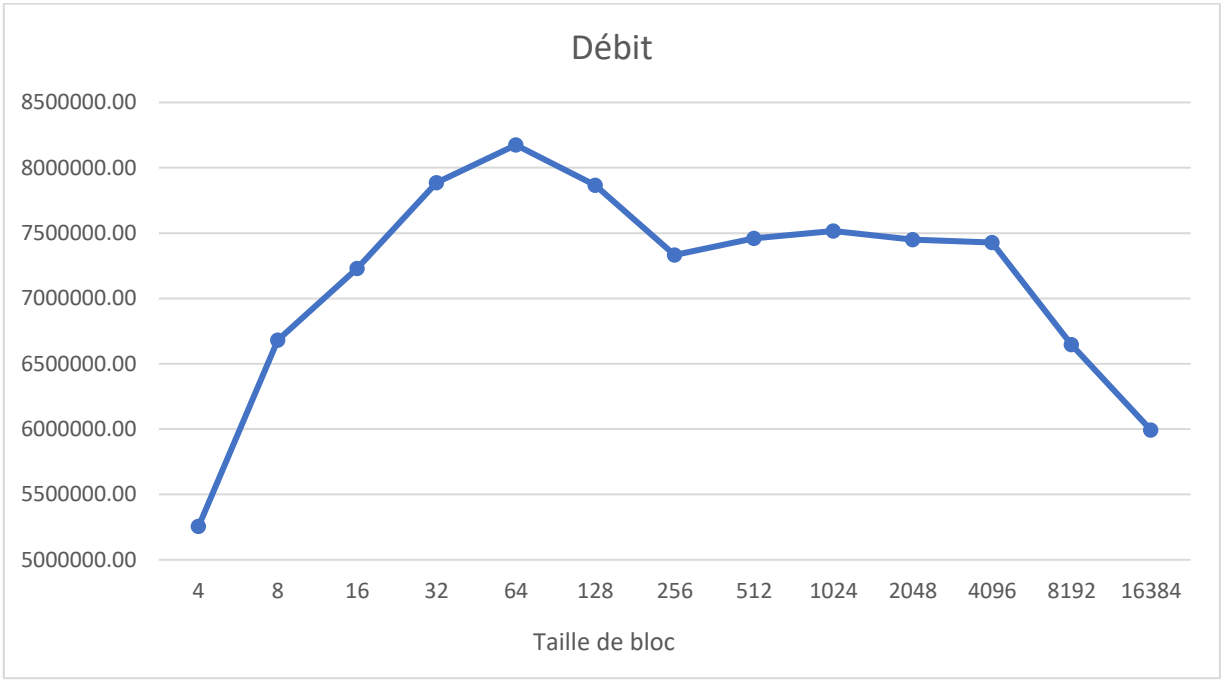

Débit
3500000.00
3000000.00
2500000.00
2000000.00
1500000.00
1000000.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc

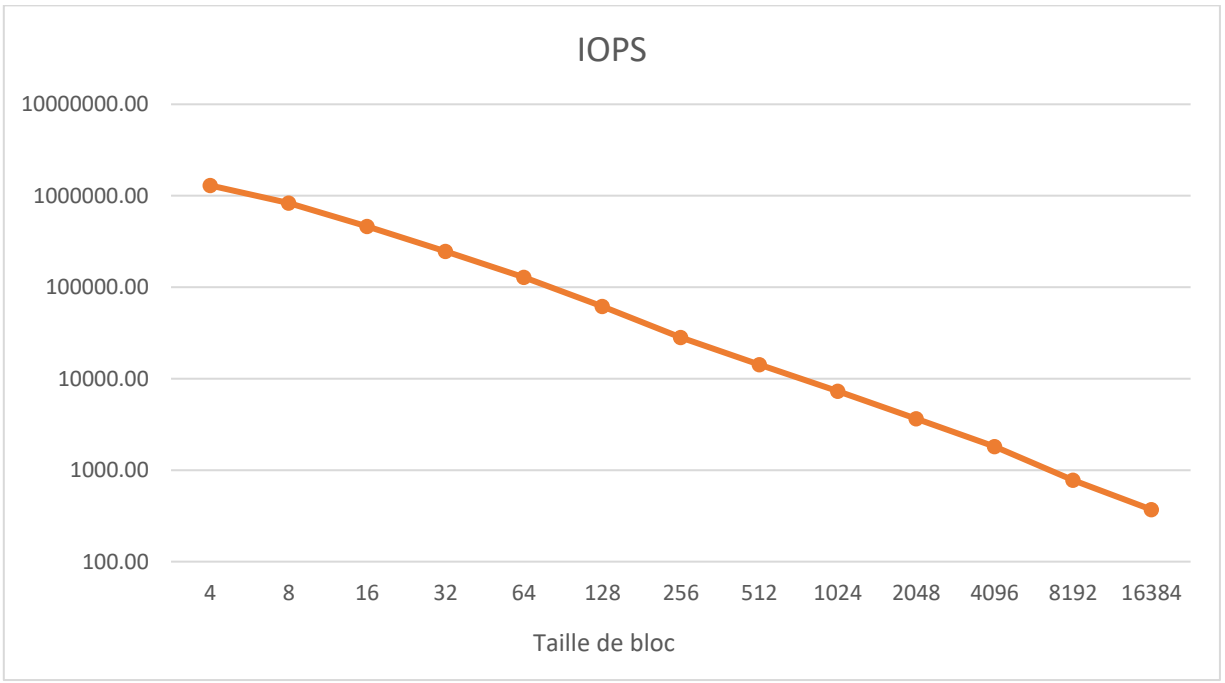

IOPS
1000000.00
100000.00
10000.00
1000.00
100.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc

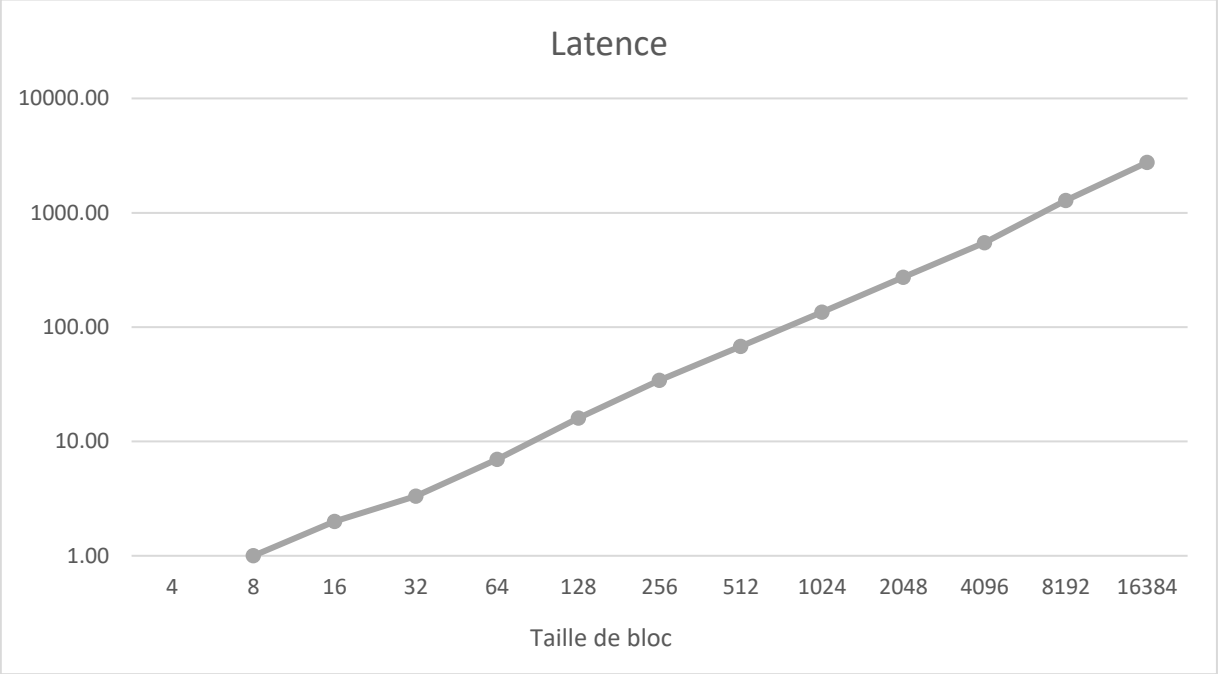


Lecture séquentielle

Taille de bloc (Ko)	Débit écriture (Ko/s)	IOPS (Opérations/s)	Latence (µs)
4	5254918.33	1296057.67	0.00
8	6681609.00	834265.33	1.00
16	7228429.00	463707.67	2.00
32	7885761.67	247651.00	3.33
64	8174239.00	128404.67	7.00
128	7866880.67	61604.00	16.00
256	7331620.00	28176.00	34.33
512	7459262.00	14239.00	67.67
1024	7516842.33	7306.00	135.33
2048	7449768.33	3667.00	272.33
4096	7427132.33	1810.67	548.33
8192	6647681.00	777.67	1282.00
16384	5991799.67	370.67	2753.00

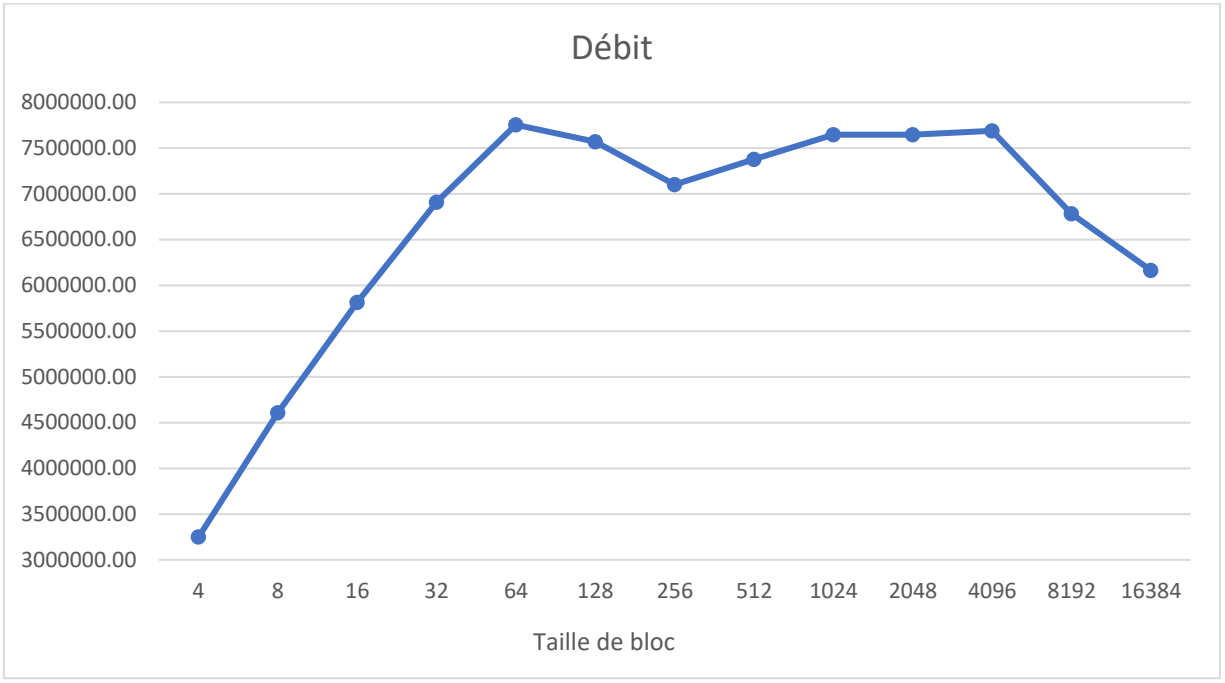

Débit
8500000.00
8000000.00
7500000.00
7000000.00
6500000.00
6000000.00
5500000.00
5000000.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc

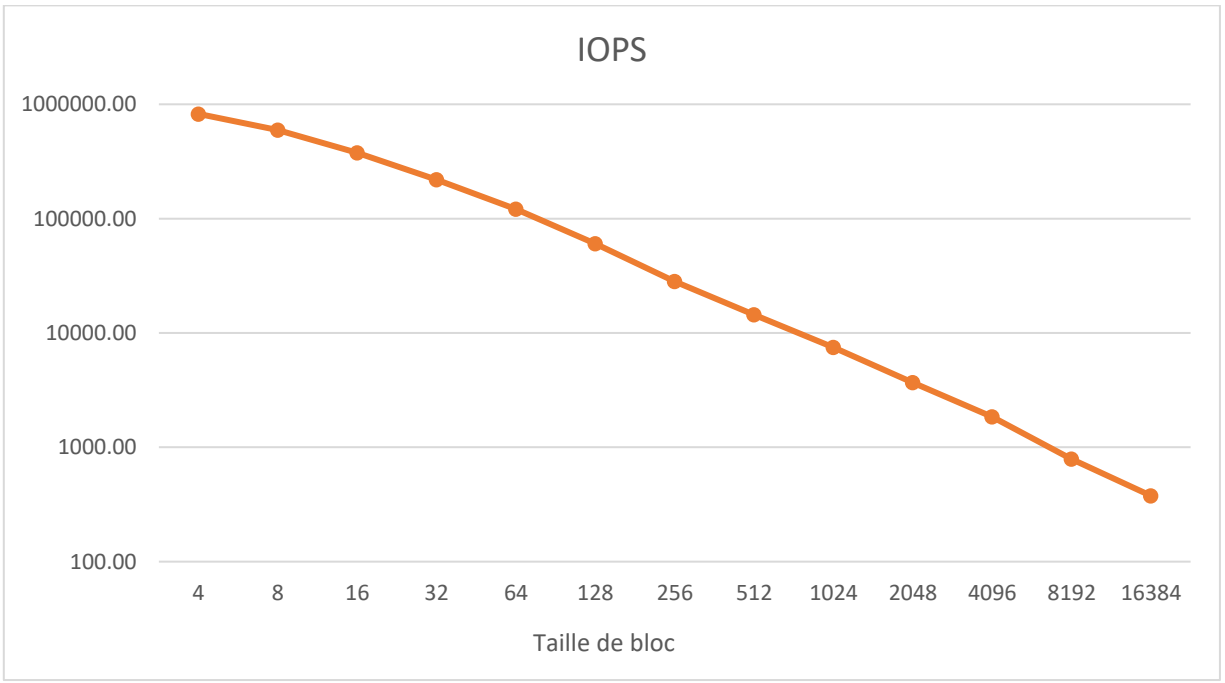

IOPS
10000000.00
1000000.00
100000.00
10000.00
1000.00
100.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc

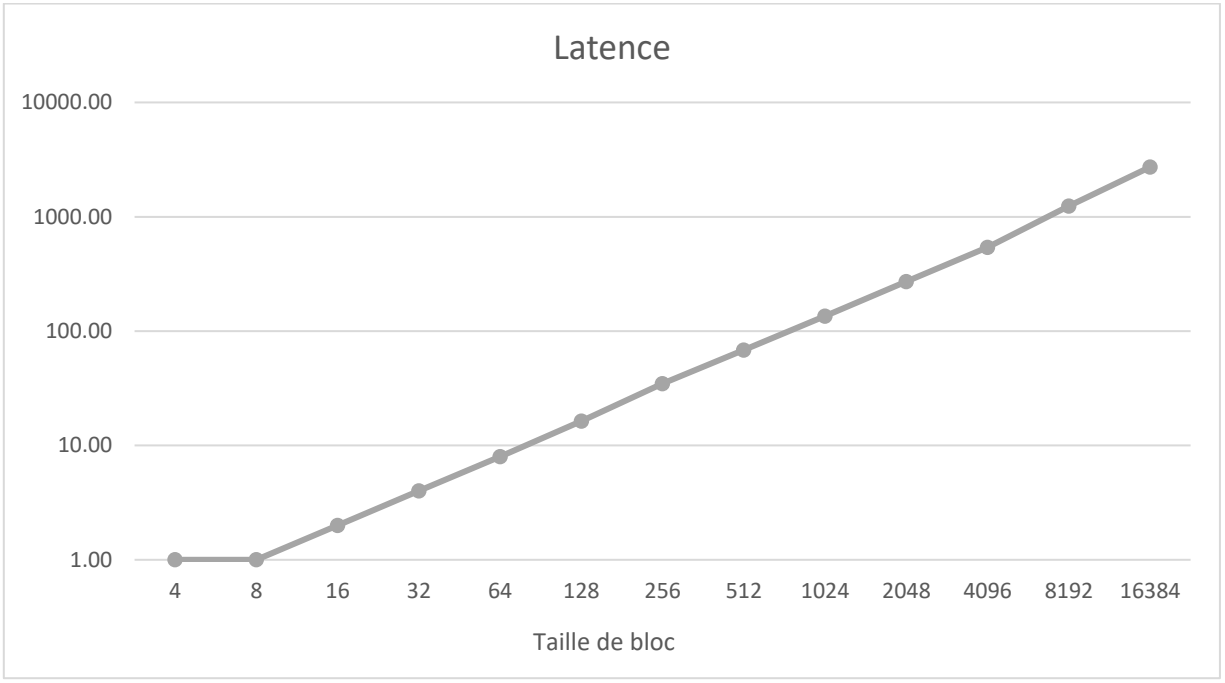


Lecture aléatoire

Taille de bloc (Ko)	Débit écriture (Ko/s)	IOPS (Opérations/s)	Latence (µs)
4	3248442.67	823719.00	1.00
8	4608931.33	593910.67	1.00
16	5813240.33	376786.67	2.00
32	6908410.00	219425.33	4.00
64	7755973.67	121443.33	8.00
128	7569460.67	60581.00	16.33
256	7103135.00	28151.33	34.67
512	7379589.33	14403.67	68.33
1024	7648427.67	7503.00	135.33
2048	7647642.00	3694.33	270.67
4096	7687909.33	1847.33	542.00
8192	6784124.00	789.67	1242.00
16384	6164290.67	377.00	2718.67


Débit
8000000.00
7500000.00
7000000.00
6500000.00
6000000.00
5500000.00
5000000.00
4500000.00
4000000.00
3500000.00
3000000.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc


IOPS
1000000.00
100000.00
10000.00
1000.00
100.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc


Latence
10000.00
1000.00
100.00
10.00
1.00
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
Taille de bloc