BENCHMARK · ARM64 CONTRE X86_64

Imaxe Monitor Solution - Zabbix 7.0 1.2

Comparaison mesurée entre l'image Graviton que nous publions et une compilation x86_64 identique, sur des types d'instance équivalents.

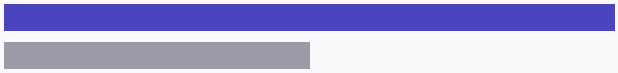
Réf. PERF-zabbix70-1.2-20261107 Généré 2026-11-07 us-east-1

EN UNE PHRASE

Graviton offre +14,1 % de performance médiane sur les tests mesurés.

Convention de couleur dans tout le document : **ARM64 (Graviton)** · **x86_64**. Les deltas portent aussi un symbole (▲ / ▼) pour une lecture en noir et blanc.

1 Résultats

TEST	ARM64 · C7G.LARGE	X86_64 · C6I.LARGE	Δ ARM64	COMPARAISON
sysbench — CPU (1 hilo)	**1113 events/s**	1074 events/s	▲ +3,6 %	
sysbench — CPU (todos los hilos)	**2221 events/s**	1113 events/s	▲ +99,5 %	
sysbench — memoria	**40914 MiB/s**	19077 MiB/s	▲ +114,5 %	
openssl — AES-256-GCM	**4147 MB/s**	8772 MB/s	▼ -52,7 %	
openssl — SHA-256	**1600 MB/s**	1284 MB/s	▲ +24,6 %	
gzip — compresión de un corpus fijo	**129,6 MB/s**	155,6 MB/s	▼ -16,7 %	
fio — lectura aleatoria 4k	**3043 IOPS**	3020 IOPS	▲ +0,8 %	
fio — escritura aleatoria 4k	**3047 IOPS**	3047 IOPS	▼ 0,0 %	
sysbench — CPU, latencia p95	**0,90 ms**	1,82 ms	▲ +102,2 %	
fio — lectura 4k, latencia p99	**5800 µs**	29230 µs	▲ +404,0 %	

Avantage médian de Graviton sur les tests comparables : +14,1 %. Chaque barre est mise à l'échelle du plus grand de sa propre paire, car chaque test a son unité.

2 Montée en charge sur les threads

PARAMÈTRE	ARM64	X86_64
CPU sur 1 thread	**1113 events/s**	1074 events/s
CPU sur tous les threads	**2221 events/s**	1113 events/s
Facteur de montée en charge	**×2,00 (2 vCPU)**	×1,04 (2 vCPU)
Efficacité par rapport à l'idéal	**99,8 %**	51,8 %



Efficacité par rapport à l'idéal

En passant d'un thread à tous les vCPU, Graviton multiplie son travail par 2,00 et l'instance x86_64 par 1,04.

> Un vCPU n'est pas la même chose dans les deux familles : chez Graviton c'est un cœur physique, chez Intel un thread SMT qui partage les unités d'exécution avec son jumeau. C'est pourquoi un seul thread peut donner des chiffres proches et l'écart n'apparaît qu'une fois tous occupés — ce qui est précisément la façon dont le produit travaille : nginx avec ses workers, PHP-FPM avec son pool et la base de données avec ses connexions utilisent tous les vCPU à la fois, pas un seul.

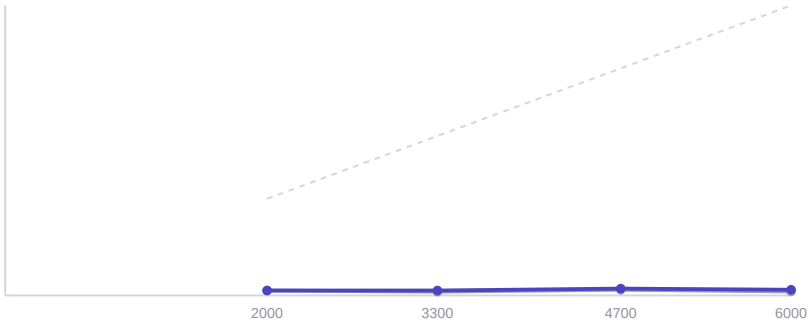
3 Méthodologie

▸ Environnement de test

PARAMÈTRE	ARM64	X86_64
Région	us-east-1	us-east-1
Type d'instance	c7g.large	c6i.large
CPU et mémoire	2 vCPU · 3799 MiB	2 vCPU · 3811 MiB
Processeur	Neoverse-V1	Intel(R) Xeon(R) Platinum 8375C CPU @ 2.90GHz
Noyau	7.0.0-1017-aws	7.0.0-1017-aws

> Chaque architecture est mesurée sur sa propre instance de test, et seulement après que celle-ci a passé les tests du produit. Le produit tourne pendant la mesure, ce qui correspond à la machine réelle d'un acheteur.

4 Courbe de charge



Requêtes servies face aux requêtes demandées. La diagonale en pointillés est le débit demandé : là où une ligne s'en écarte, la machine sature.

DÉBIT DEMANDÉ	ARM64	X86_64	ARM64 P99	X86_64 P99
2000	**103,9** ▲	104,2 ▲	**67057 ms**	56975 ms
3300	**101,7** ▲	78,8 ▲	**49891 ms**	41113 ms
4700	**142,4** ▲	119,5 ▲	**43587 ms**	53015 ms
6000	**119,7** ▲	84,0 ▲	**44825 ms**	62663 ms

▲ marque un palier où la machine ne tient plus le débit demandé. Au-delà, le p99 mesure l'attente en file, pas la latence.

La charge est générée par une machine distincte — même type d'instance pour les deux architectures, dans la même zone de disponibilité — qui sollicite la machine mesurée via son adresse privée à un taux d'arrivée constant. Le client est identique dans les deux passes : seule change la machine qui répond.

▸ Capacité et coût du travail

POSTE	ARM64	X86_64	Δ ARM64
Débit maximal soutenu	**142,4 req/s**	119,5 req/s	**▲ +19,1 %**

La rampe a commencé au-dessus de ce que l'une des deux pouvait encaisser ; pour celle-là il n'y a aucun palier propre et ce qui est montré est le débit maximal observé — un plafond, pas un débit soutenu.

Le débit maximal soutenu est le dernier palier que la machine a servi en entier, pas le sommet du graphique : un palier saturé délivre beaucoup et répond mal, et le compter comme capacité reviendrait à promettre une charge où le produit va déjà mal. Le coût par million de requêtes se calcule à ce débit, et c'est la comparaison qui tient quand les deux machines ne encaissent pas la même charge : le prix horaire dit ce que coûte la machine allumée, pas ce que coûte le travail qu'elle fait.

5 Conclusion

Imaxe Monitor Solution - Zabbix 7.0 est publié sur Graviton (arm64) parce qu'il mesure mieux et coûte moins : +14,1 % d'avantage médian sur l'instance x86_64 équivalente.

> Tous les chiffres proviennent d'une exécution sur l'image publiée. Les tests non mesurables sur les deux architectures sont signalés comme tels et exclus de la médiane.

A Annexe — commandes exécutées

> Exactement ce qui a été exécuté, pour que les chiffres puissent être reproduits ou contestés.

TEST	COMMANDE
sysbench — CPU (1 hilo)	`sysbench cpu --cpu-max-prime=20000 --threads=1 --time=60 run`
sysbench — CPU (todos los hilos)	`sysbench cpu --cpu-max-prime=20000 --threads=$(nproc) --time=60 run`
sysbench — memoria	`sysbench memory --memory-block-size=1M --memory-total-size=100G --threads=2 run`
openssl — AES-256-GCM	`openssl speed -elapsed -evp aes-256-gcm -seconds 3 2>&1`
openssl — SHA-256	`openssl speed -elapsed -evp sha256 -seconds 3 2>&1`
gzip — compresión de un corpus fijo	`f=/mnt/bench/corpus.txt; s=$(stat -c%s $f); t0=$(date +%s%N); i=0; while [ $i -lt 3 ]; do gzip -6 -c $f >/dev/null; i=$((i+1)); done; t1=$(date +%s%N); awk -v s=$s -v a=$t0 -v b=$t1 -v n=3 'BEGIN{printf "throughput: %.2f MB/s\n", n*s/1000000/((b-a)/1000000000)}'`
fio — lectura aleatoria 4k	`fio --name=randread --ioengine=libaio --direct=1 --bs=4k --iodepth=16 --rw=randread --size=4G --runtime=60 --time_based --filename=/mnt/bench/testfile`
fio — escritura aleatoria 4k	`fio --name=randwrite --ioengine=libaio --direct=1 --bs=4k --iodepth=16 --rw=randwrite --size=4G --runtime=60 --time_based --filename=/mnt/bench/testfile`
sysbench — CPU, latencia p95	`sysbench cpu --cpu-max-prime=20000 --threads=$(nproc) --time=60 run`
fio — lectura 4k, latencia p99	`fio --name=randread --ioengine=libaio --direct=1 --bs=4k --iodepth=16 --rw=randread --size=4G --runtime=60 --time_based --filename=/mnt/bench/testfile`