2021 Cloud Report

Authors
Arul Ajmani
John Kendall
Yevgeniy Miretskiy
Jessica Edwards

Cockroach Labs

Table of Contents

Introduction

The 2021 Cloud Report stands on benchmarks.

It is the only cloud performance report to compare the three major cloud providers — Amazon Web Services (AWS), Microsoft Azure (Azure), and Google Cloud Platform (GCP) — on micro and industry benchmarks that reflect critical OLTP applications and workloads.

Written with CockroachDB performance needs in mind, we assess machines from AWS, Azure, and GCP to help our customers understand the performance tradeoffs present within each cloud and its machines.

The 2021 Cloud Report is more precise than ever, capturing our ability to evaluate each of the clouds with data that better tells realistic and universal performance stories.

Working with each of the three major cloud providers, we worked to optimize all microbenchmarks to be more accurate and representative of real-world performance. We evaluated 54 machine configurations and conducted nearly 1,000 benchmark test runs including CPU, Network Throughput, Network Latency, Storage Read Performance, Storage Write Performance, and Cockroach Labs Derivative of TPC-C, the industry-standard OLTP benchmark.

Open Source Testing Methology & Reproduction Steps

In the 2021 Cloud Report, we benchmarked 54 machine configurations across the following broad axes:

- CPU Performance
- Network Performance
- Storage I/O Performance
- OLTP Performance (TPC-C)

A thorough explanation of our testing methodology, the full list of all machines testedare available in Appendix I. Reproduction steps are fully open source available in this repo: https://github.com/cockroachlabs/cloud-report-2021.

All benchmarks and microbenchmarks are focused on machines with network-attached storage. Results for machines with locally-attached storage have been moved to Appendix II.

2021 Cloud Report Results

Benchmark	1st	2nd	3rd
1 core CPU	GCP	Azure	aws
16 core CPU	aws	GCP	Azure
Network Throughput	GCP	aws	Azure
Network Latency	aws	Azure	GCP
Storage I/O Read IOPS	Azure	GCP	aws
Storage I/O Write IOPS	Azure	GCP	aws
Storage I/O Read Throughput	GCP	Azure	aws
Storage I/O Write Throughput	GCP	Azure	aws
Storage I/O Read Latency	aws	GCP	Azure
Storage I/O Write Latency	Azure	GCP	aws
Max. tpm	GCP	Azure	aws
$/tpm	aws	Azure	GCP

2021 CLOUD REPORT TOP-LEVEL EVALUATIONS

AWS Offers Cost Efficiency; Room For Improvement on Storage I/O

AWS offered the most cost-efficient machine on the OLTP benchmark, Cockroach Labs Derivative TPC-C, with the c5a.4xlarge machine coming in 12% lower than GCP and 35% lower than Azure's most cost-efficient machines.

AWS has also provided the best network latencies three years in a row. Finally, AWS machines with the custom-built Graviton2 processor performed best in the multi-core CPU benchmark.

However, AWS demonstrated the lowest performance on seven of the 12 benchmarks, including all storage I/O benchmarks and the single-core CPU benchmark.

2021 CLOUD REPORT TOP-LEVEL EVALUATIONS

Azure's Ultra Disks Deliver on Storage IOPS & Throughput

Azure had comparable raw throughput (tpm) performance with GCP and AWS -- the spread across all three providers' winning machines was razor thin.

Most notably, Azure surpassed AWS in storage I/O performance. Provisioned with ultra disks, their machines led all cloud providers in storage I/O read IOPS, write IOPS, and write latency.

This elite storage performance came at a higher cost. Azure had the highest estimated monthly disk cost and was narrowly the second least cost-efficient cloud provider in terms of $/tpm.

2021 CLOUD REPORT TOP-LEVEL EVALUATIONS

GCP Beats Out AWS & Azure Overall, Shines on Raw Throughput

GCP achieved the best single-core CPU performance and delivered the most throughput at every level, including on the OLTP benchmark, Cockroach Labs Derivative TPC-C.

GCP's machines outperformed Azure and AWS in network and storage I/O (read and write) throughput.

Despite not having an advanced disk option (extreme-pd) available to us, GCP machines with general purpose disks (pd-ssds) attained the highest amount of raw throughput (tpm) but was the least cost efficient provider in terms of $/tpm.

TOP LEVEL FINDINGS

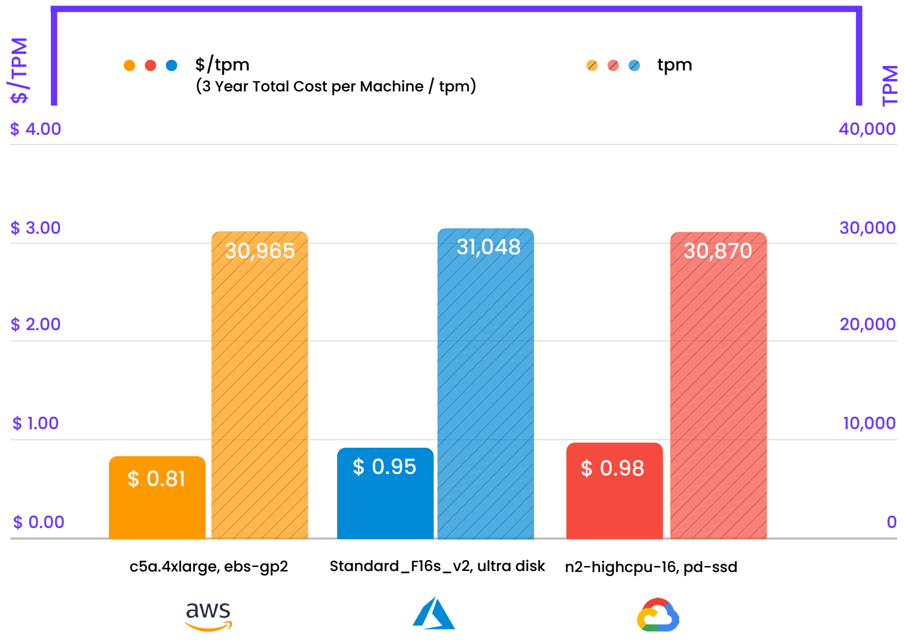
Overall Conclusions on Real World OLTP Applications

The Cockroach Labs Derivative TPC-C Benchmark and its real world OLTP workload is the closest benchmark we have for evaluating the kinds of applications our customers might run.

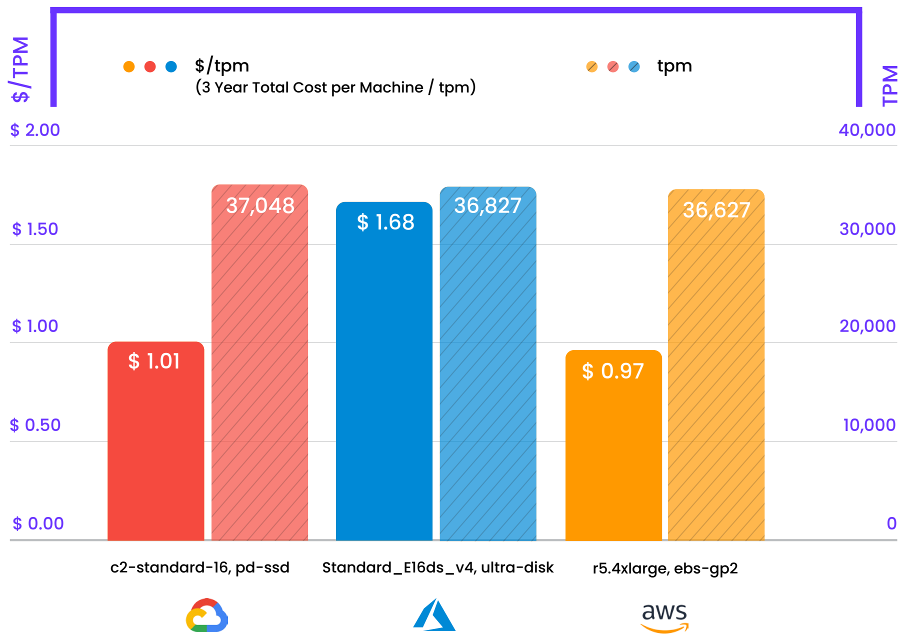
- In the 2021 Cloud Report, AWS provided the most cost-efficient machine at $0.813 in dollars per tpm (3-Year Total Cost Per Machine / tpm).
- On transactional throughput, however, GCP enabled CockroachDB to have the most highest throughput of any of the clouds, running 37,048 transactions per minute (tpm) on a three-node cluster.
- While Azure did not win on either of the Cockroach Labs Derivative TPC-C metrics, Azure's ultra disk -- the more advanced (i.e., expensive) disk -- was a strong competitor, delivering 16% more tpm while being priced only 11% higher than Azure's less expensive "premium" disks.

We discuss these results in greater depth below.
In these charts, the winner in each category is presented first.

MOST COST EFFICIENT ($/TPM) MACHINES FROM EACH CLOUD



TOP-PERFORMING MAXIMUM TPM MACHINES FROM EACH CLOUD



CPU BENCHMARK

Benchmarking AWS, Azure, and GCP with CoreMark

About CoreMark

CoreMark is an open source, cloud-agnostic benchmark that evaluates general CPU performance. CoreMark executes various workloads that are representative of the types of operations often found in real world applications, including list management, list sort and search, matrix manipulation, and checksum computation. The benchmark then reports the number of operations performed, and produces a single-number score (the higher the score, the better the performance).

Measurement

We evaluated each cloud's CPU performance using the CoreMark version 1.0 benchmark[1]. Designed for CPU benchmarking, CoreMark produces a single-number score: the higher the score, the better the performance.

CoreMark replaced stress-ng, which we used to measure CPU performance in the 2020 report[2]. Stress-ng is designed for stress testing, not for measuring general purpose CPU performance, so we collaborated with the cloud providers to find a new framework for this benchmark.

We arrived at CoreMark because it is open-source, cloud-agnostic, and more representative of general CPU performance than stress-ng. For example, CoreMark tests against various real-world workloads like list sort and search.

The CoreMark benchmark can be limited to run on a single vCPU, or it can execute workloads on multiple vCPUs in parallel. We ran CoreMark in both modes ten times and reported the average number of iterations/second for both the single-core results as well as the 16-core results.

A Note on CPU Processors

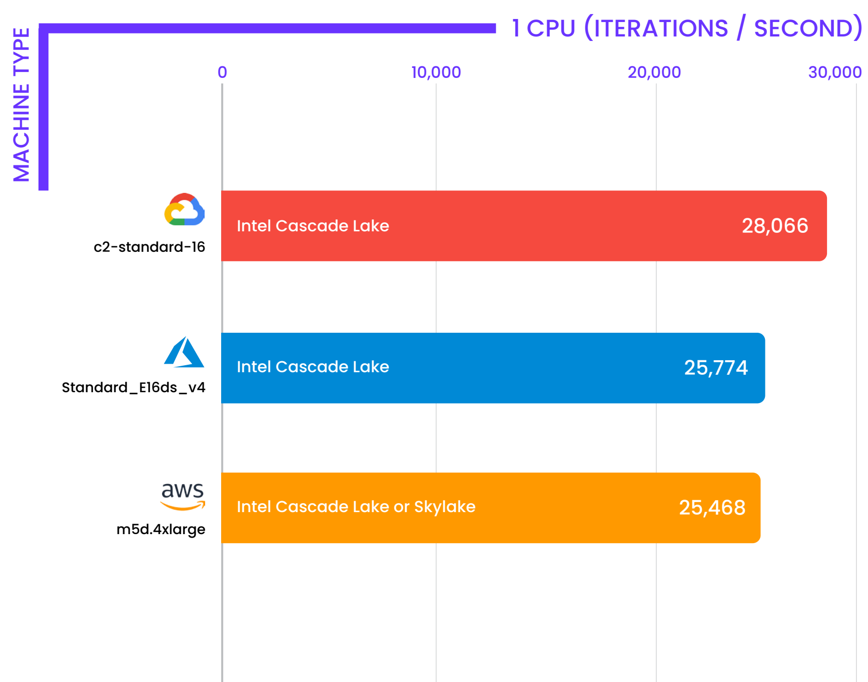
For the single-core runs, all the winning machines ran Intel processors. When we looked at performance on the 16-core CoreMark benchmark, none of the winning machines ran Intel processors. AWS's Graviton2 came in just ahead of GCP and Azure's winning machines, both of which ran AMD processors.

1 https://github.com/eembc/coremark
2 https://kernel.ubuntu.com/~cking/stress-ng/

CPU PERFORMANCE ON SINGLE-CORE

GCP Produces Highest Single-Core Score

We found that CoreMark performance on a Single-Core served as an effective indicator for raw throughput (OLTP) performance. Leading the way, GCP achieved the highest CoreMark score by 10% over best performing compute-optimized machines in AWS and Azure.

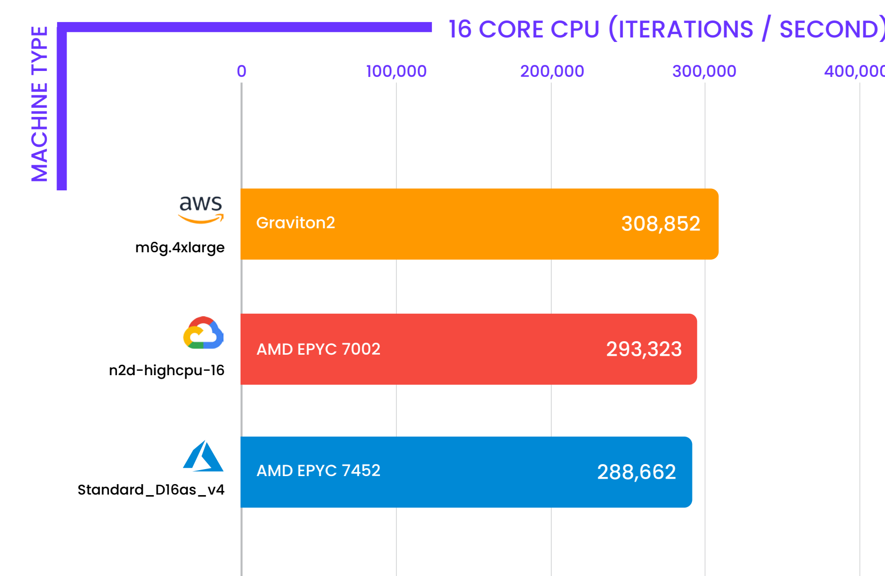


CPU PERFORMANCE ON 16 CORES

AWS Scales with Graviton2 Processor

As a less indicative metric of raw throughput (Cockroach Labs Derivative TPC-C) than CoreMark performance on one core, CoreMark on 16 cores highlights each machine's ability to scale to multiple cores.

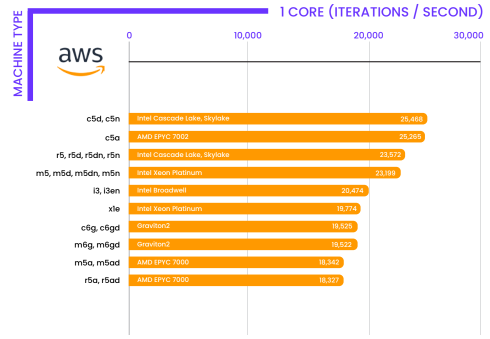
The AWS custom-built Graviton2 Processor, which uses a 64-bit ARM architecture, enabled the top-performing m6g.4xlarge machine to achieve better multi-CPU scaling than the other machines. AWS's m6g.4xlarge hit 308,852 iterations/second, and GCP and Azure's machines produced scores that were 5% and 7% lower, respectively.

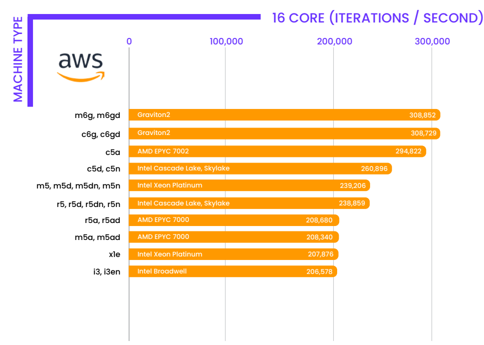


Cockroach Labs
© 2021 Cockroach Labs, Inc

AWS CPU PERFORMANCE

Machines with Intel processors outperformed the others on single-core CPU performance, with the notable exception of the c5a series running AMD EPYC 7002 processors. For CoreMark performance on 16 cores, AWS machines with Graviton2 processors consistently outperformed all other AWS machines (and, indeed, all other machines in the field, period). These machines are also AWS's cheapest.



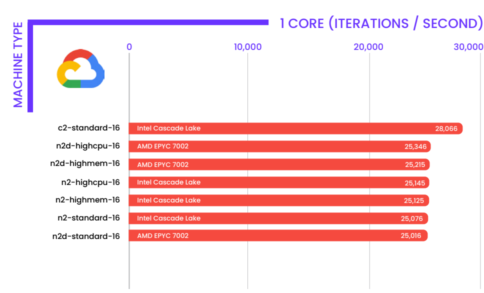


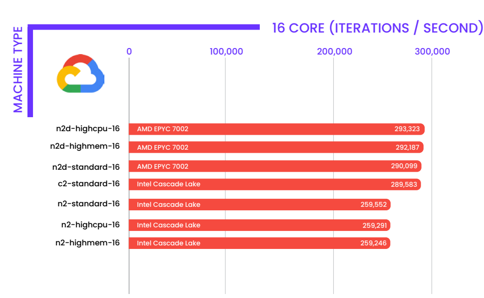
Note: Where AWS machines demonstrated minimal variability, results have been grouped by type and processor to improve readability.

GCP CPU PERFORMANCE

GCP's compute-optimized machine (c2-standard) demonstrated the best raw CPU performance among GCP machines. Beyond the c2-standard machine, there was minimal variation in single-core benchmark among machines tested because the N2 machines have the same processors and only differ in their vCPU to memory ratios.

The same applies to the N2D machines. On the 16-core benchmark, the N2D machines with AMD EPYC 7002 Processors and the c2-standard-16 machines led all other GCP machines.

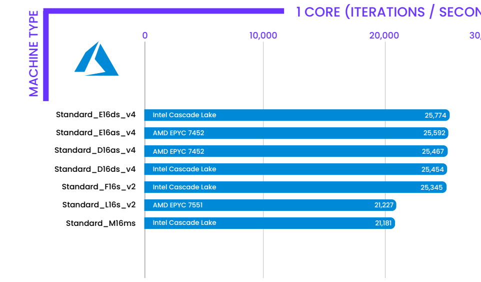


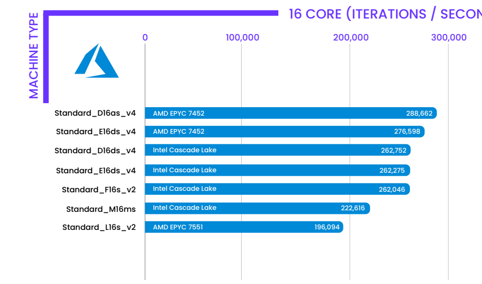


Azure CPU PERFORMANCE

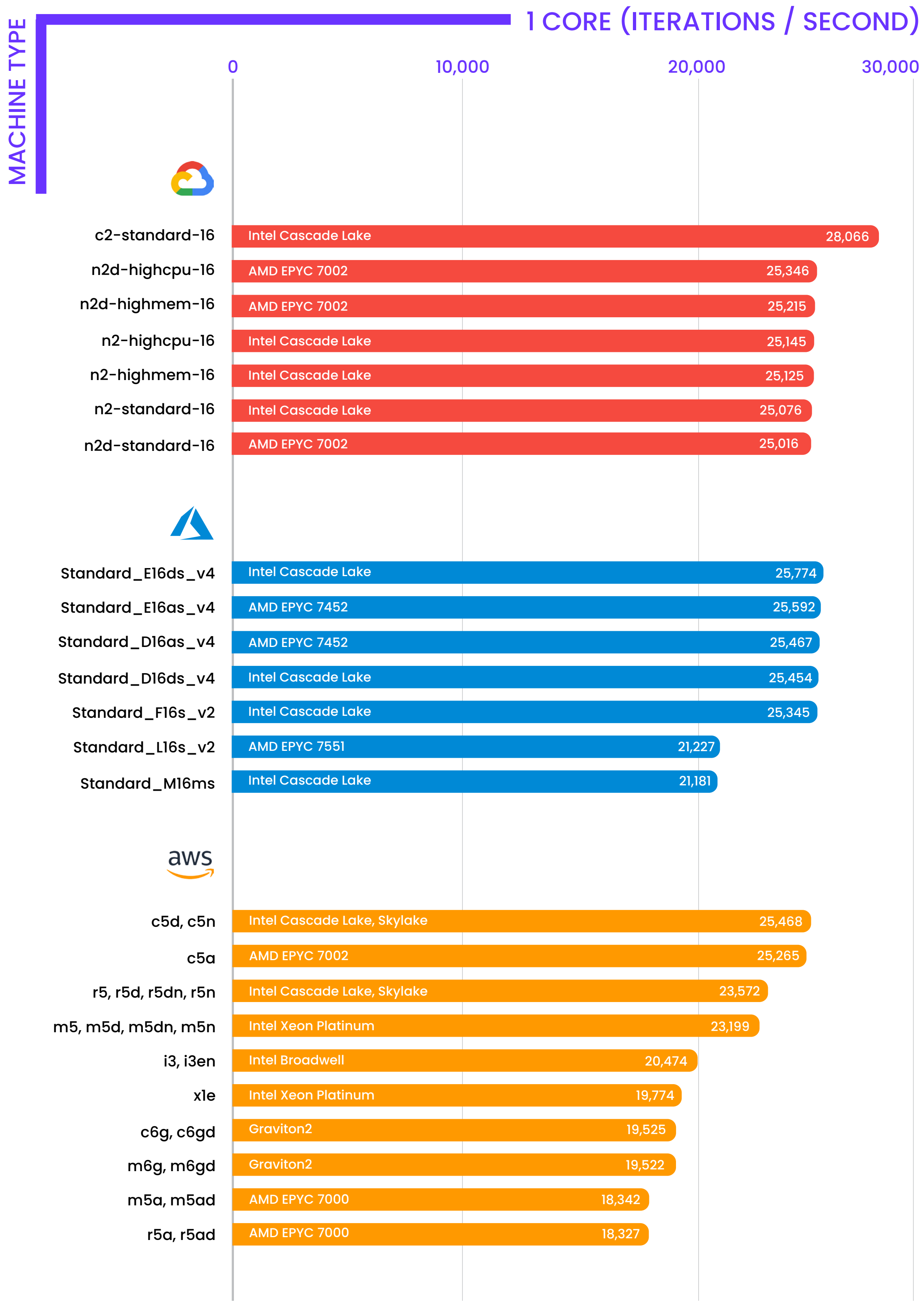
Similar to the single-core benchmark on AWS, Azure machines with AMD processors performed on par with Azure machines with Intel processors. Consistent with both AWS and GCP, Azure machines with AMD processors outperformed Azure machines with Intel processors on the 16-core benchmark.

Notably, the memory-optimized Standard_M16ms machine and the storage-optimized Standard_L16s_v2 machine were the bottom performers among Azure's machines on both metrics.

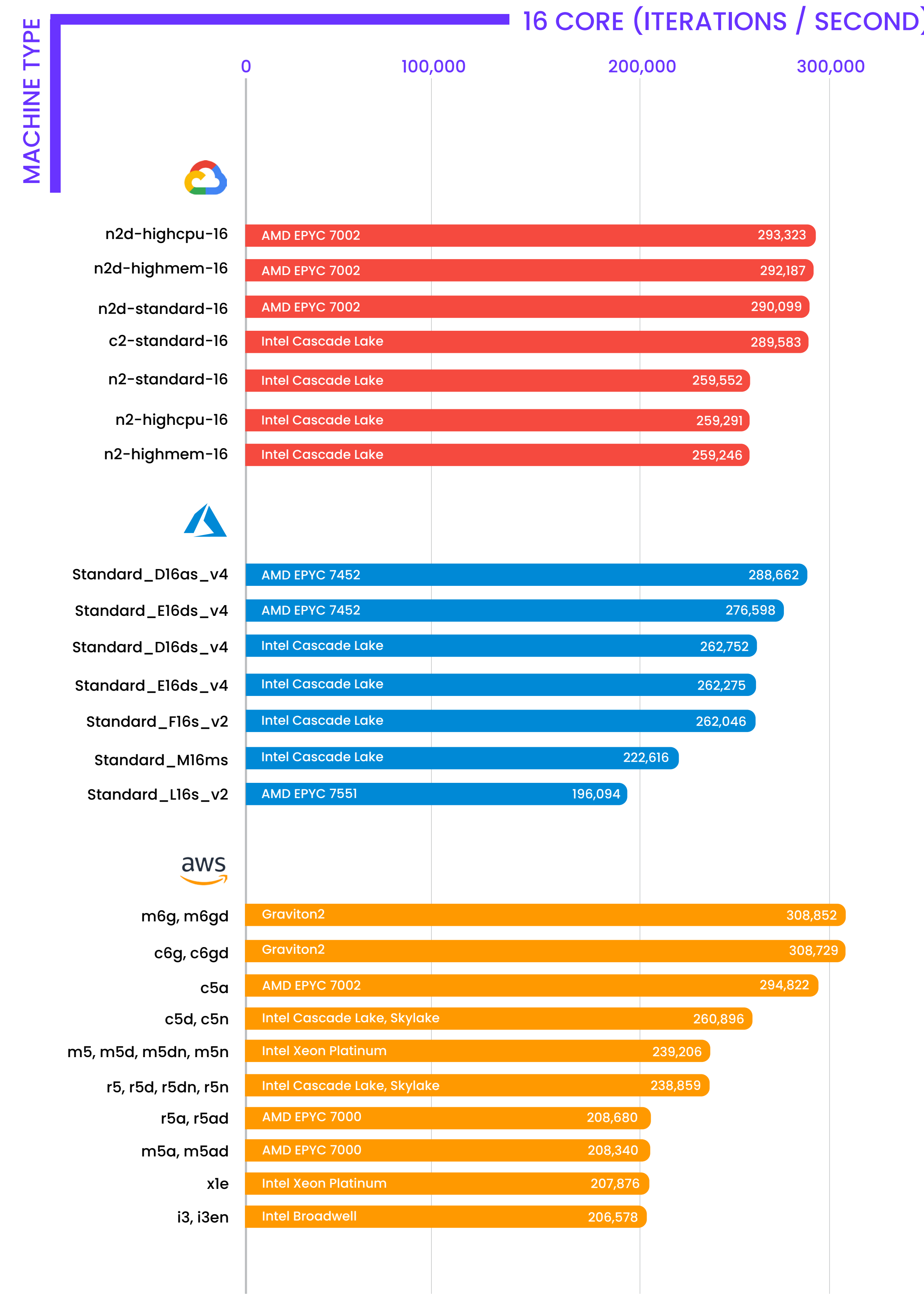




SINGLE CORE COMPARISON



16 CORE COMPARISON



NETWORK BENCHMARK

Measuring Network Throughput and Latency with Netperf

About Netperf

Netperf is a popular benchmark designed for measuring network throughput and latency. It replaces last year's usage of iPerf and ping for measuring throughput and latency, respectively. Netperf measures the latency using TCP requests, which is more representative of the real-world applications. In contrast, ping uses ICMP (Internet Control Message Protocol, an older protocol that serves a different purpose than TCP).

Measurement

To evaluate network throughput and latency, we used the Netperf version 2.6.0 benchmark. We ran this benchmark using a single Netperf process on both a designated client and a designated server VM. Throughput (also known as bandwidth) is the quantity of data being sent and received over time. Latency is the total "round-trip" response for a request.

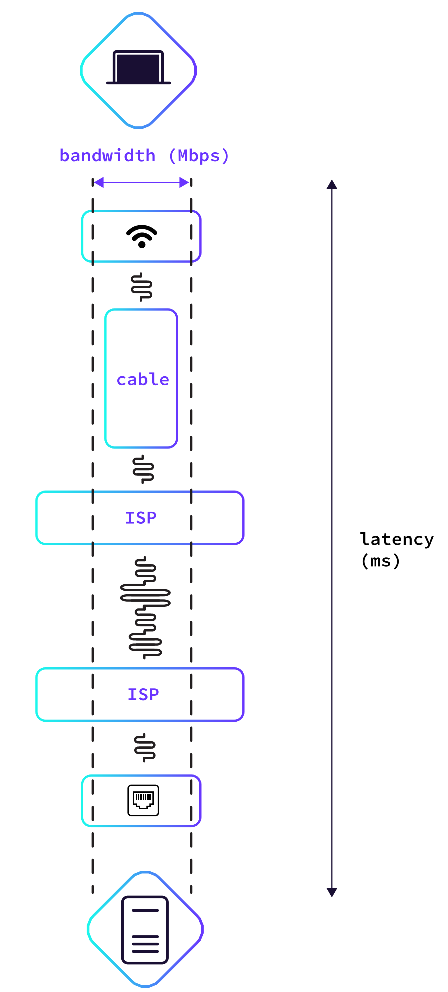
Of the two benchmarks, throughput results were more reproducible. A critical determining factor for latency is where machines are physically placed [see Latency Concepts below]. We provisioned machines in the same availability zone for all our network tests, meaning the latency numbers presented are from within the same availability zone for each of the clouds.

It is important to call out that there may be variance in the latency numbers depending on machines' positions within an availability zone. We did not try to control this using more granular placement policies.

Maximizing availability while minimizing network latency can be a delicate balance. They can seem, at times, to be at odds with one another. To maximize availability, you may want to distribute machines across availability zones, or maybe even regions, but the cost is latency.
The inverse would be to minimize network latency by clustering machines inside a single zone, elevating the risk of a zone failure.

A core principle of CockroachDB is to provide resilience and maximum availability to applications. In practice, CockroachCloud spreads machines across zones to mitigate against zone failure by default.

Additionally, CockroachDB also offers operators the ability to accomplish stronger survivability goals, such as a region failure, by setting data location constraints and replication preferences. In this report, however, we provisioned all instances in the same zone for all our experiments. This was something we could control across the different clouds, and helped level the playing field.



Cloud-Specific Names of Network- Attached Storage Disks:

	General Purpose Network Attached Disks	Advanced Network Attached Disks
AWS	Elastic Block Store general purpose (EBS gp2)	Elastic Block Store Provisioned IOPS Volume (EBS io2)
Azure	Premium Disk	Ultra Disk
GCP	Persistent Disk SSD	Extreme Persistent Disk (was not available for us)

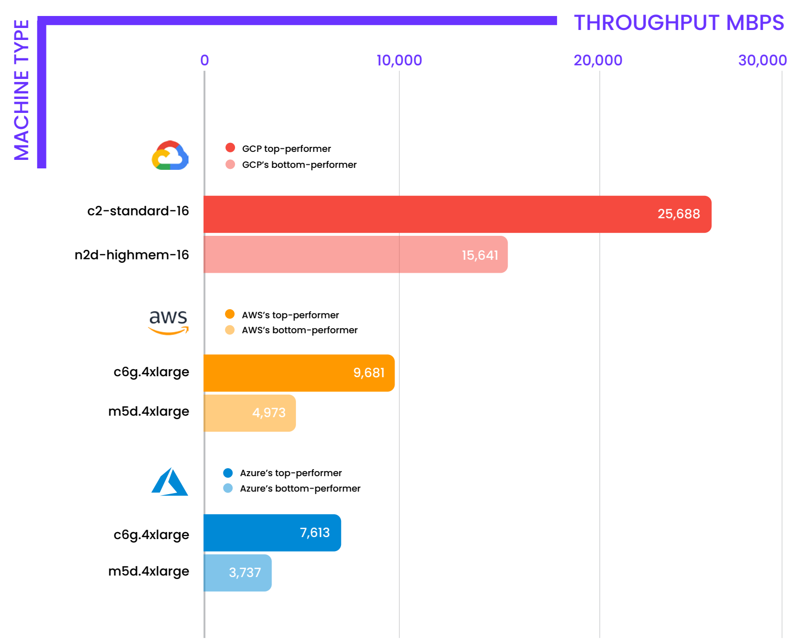
NETWORK THROUGHPUT OVERALL

GCP has nearly 3x the throughput of AWS & Azure

Consistent with last year's report, GCP had more network throughput than AWS and Azure, and it wasn't close. Looking at the cloud providers' top performing network throughput machines, GCP's top-performing machine had 165% and 237% more throughput than AWS and Azure respectively.

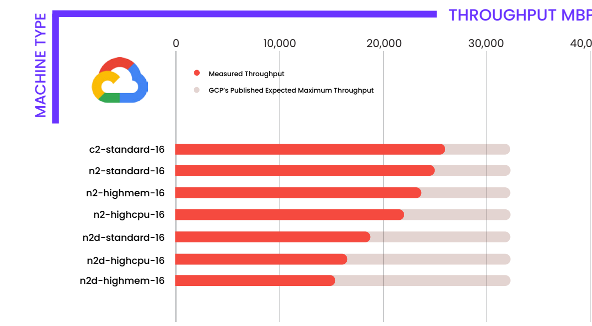
To add, GCP's bottom-performing network throughput machine (n2d-highmem-16) performed 62% and 105% better than AWS' and Azure's top performing network throughput machines.

The throughput results were consistent with how each of the cloud provisioned 16-core VMs: GCP simply made more bandwidth available (up to 32 Gbps) than AWS (up to 10 Gbps, or in some cases up to 25 Gbps) or Azure (up to 8 Gbps). When choosing, we recommend looking at the application and workload that you plan to run, and in particular how much data your application needs to send or receive, before determining whether having less available bandwidth is a disqualifying factor.



GCP NETWORK THROUGHPUT

First let us reiterate that all of GCP's machines outperformed all of AWS and Azure for network throughput. Interestingly, the three machines with AMD processors (n2d-standard-16, n2d-highcpu-16, and n2d-highmem-16) all had the lowest throughput among GCP machines.



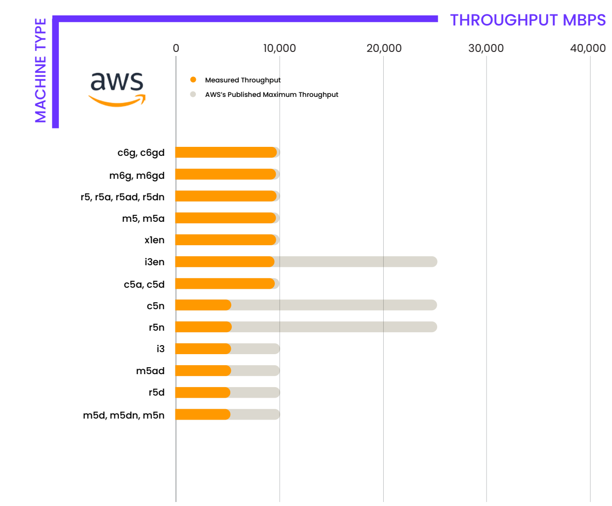
AWS NETWORK THROUGHPUT

For some of the cases, AWS performed within their published maximum bandwidth of up to 10 Gbps. However, some AWS network-optimized machines (c5n.4xlarge, m5dn.4xlarge, and m5n.4xlarge) demonstrated a significant dip in throughput resul ts when compared to their non-optimized counterparts. These machines were expected to have up to 25 Gbps in network throughput but averaged 5 Gbps in our tests.

These results for the network-optimized machines were admittedly surprising. We ran 10 independent experiments for each of the network optimized machines to confirm our findings. This included provisioning new machines for every run. In around half our runs, we saw throughput numbers of around 5 Gbps, while in the other half we saw throughput numbers around 10 Gbps.

More interestingly, we saw a bimodal distribution for p90 latencies along with the bimodal distribution for throughput. This can be seen in the latency graph below as well -- all machines that measured in around 5 Gbps in throughput had p90 latencies above 115 microseconds. Most other machines are clustered around the 60 microseconds p90 latency mark.

Lastly, it is worth noting we did not see the network-optimized machines come close to the maximum 25 Gbps claimed in any of our validation runs. AWS states that the expected network performance is "up to" for instances for 16 vCPUs or less. As we limited our testing to 16 vCPUs, we didn't test larger machines.

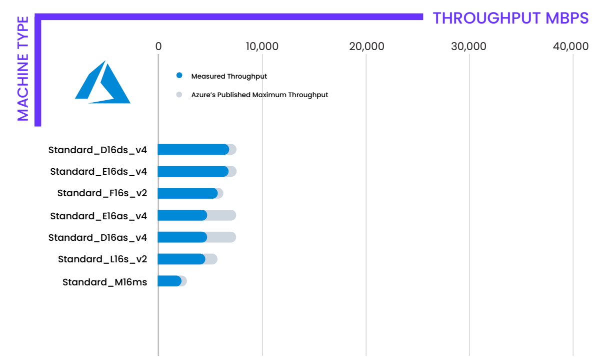


Note: For readability, the AWS machines have been grouped by their machine type as we did not see variations in network throughput. Any outliers have been separated out.

AZURE NETWORK THROUGHPUT

Azure performed close to maximum published expectations.[3] While using their free Accelerated Networking feature to maximize network throughput, Azure's throughput performance on average achieved 90-95% of their published maximum throughput amounts. Notable exceptions to that were the Standard_E16as_v4 and Standard_D16as_v4 machines, which achieved 76% of expected throughput.[4]

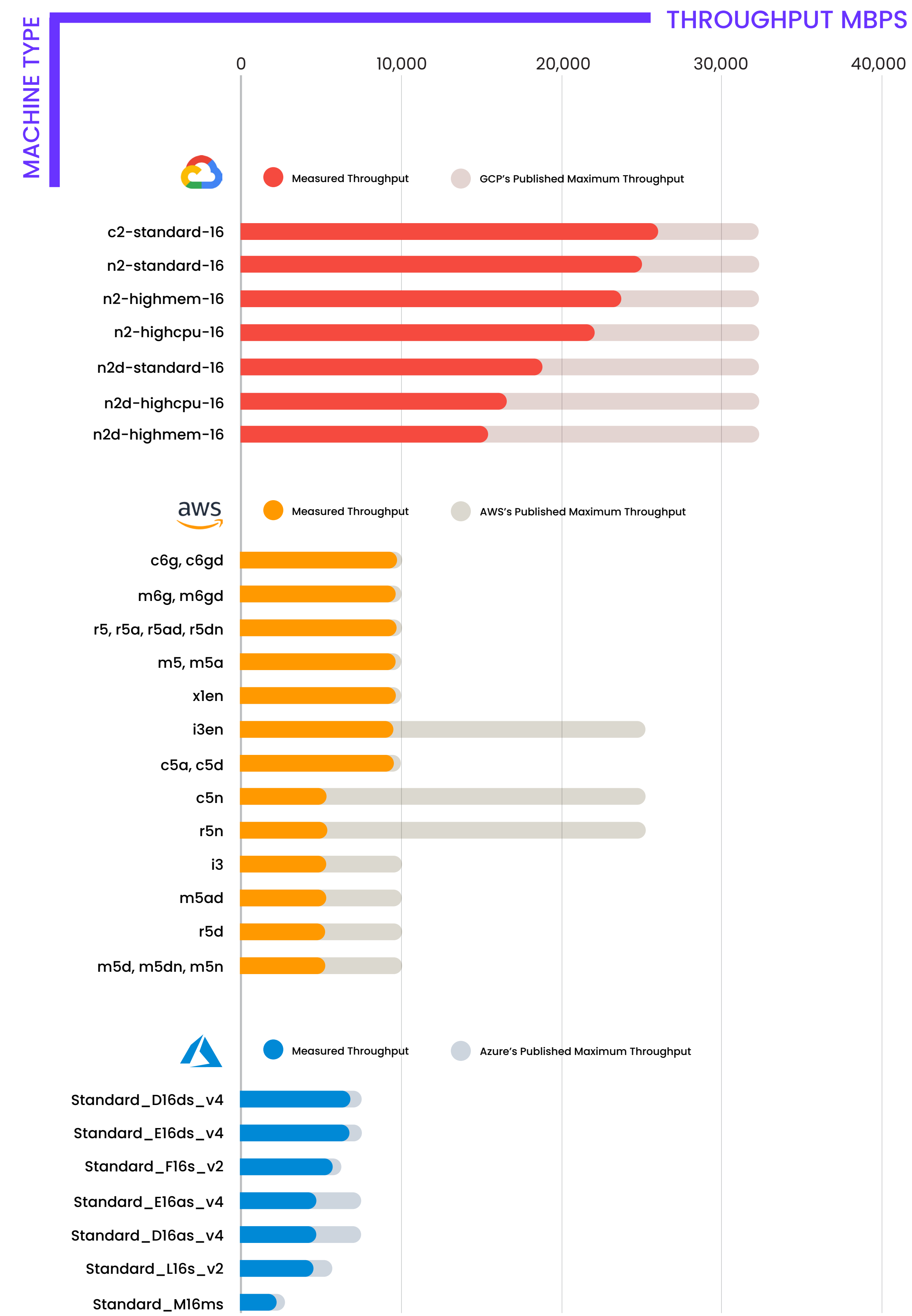
Interestingly, Azure throughput measurements did not exhibit a correlation with their network latency values. In particular, two of the best performing machines on the network throughput test (Standard_D16ds_v4 and Standard_E16ds_v4) performed the worst on the network latency test below. This is in stark contrast to AWS.



[3] According to Microsoft documentation, "Expected network bandwidth is the maximum aggregated bandwidth allocated per VM type across all NICs, for all destinations" (https://docs.microsoft.com/en-us/azure/virtual-machines/ddv4-ddsv4-series?toc=/azure/virtual-machines/linux/toc.json&bc=/azure/virtual-machines/linux/breadcrumb/toc.json#size-table-definitions).

[4] Standard_D16ds_v4: https://docs.microsoft.com/en-us/azure/virtual-machines/dav4-dasv4-series

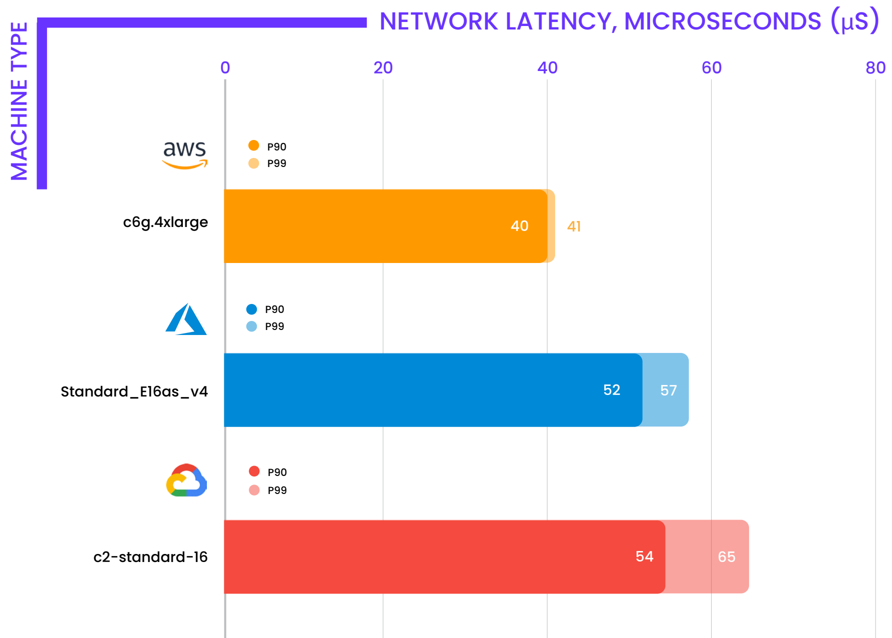
NETWORK THROUGHPUT COMPARISON



OVERALL NETWORK LATENCY

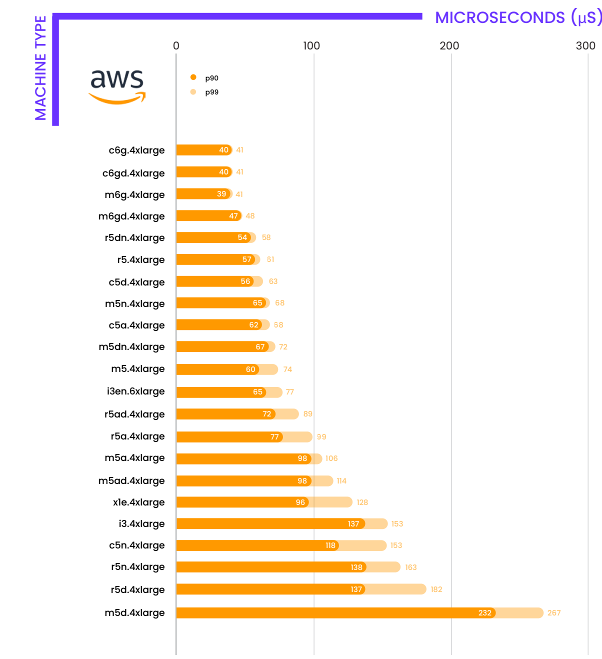
Azure and GCP Can't Match AWS Network Latency

AWS has performed best in network latency for three years running. Their top-performing machine's 99th percentile network latency was 28% and 37% lower than Azure and GCP, respectively. As mentioned in our discussion of placement policies above, it is important to keep in mind possible randomness in the physical distance between instances.



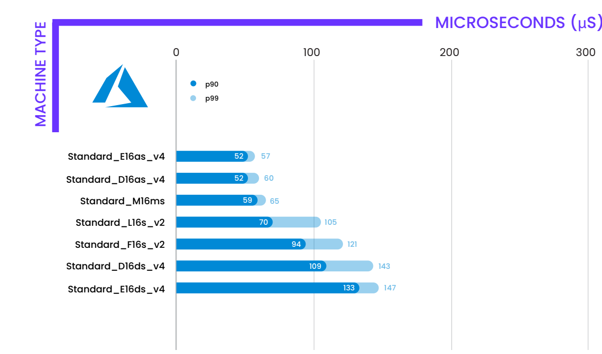
AWS NETWORK LATENCY

Overall, AWS performed well on this benchmark, with some outliers. This can be attributed to the physical placement of the instances. Even though we controlled the experiment by running VMs in the same Availability Zone, the degree of variance we see can be attributed to randomness in VM placement.



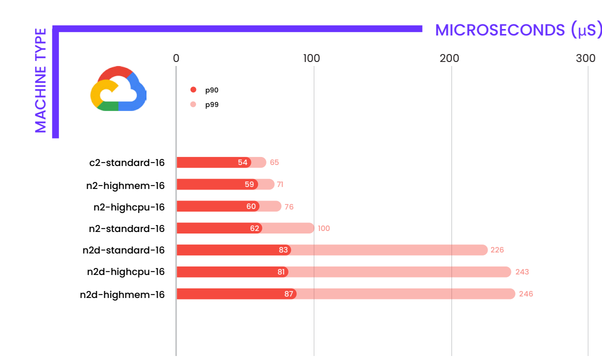
AZURE NETWORK LATENCY

Similar to AWS, Azure's network latency varied. To minimize network latency, we used Azure Accelerated Networking, learned while creating the 2020 Cloud Report. In addition to improving throughput, Accelerated Networking promises to lower latency and is available at no extra cost.



GCP NETWORK LATENCY

GCP's network latency also varied. Demonstrating a pattern driven by the processor, the three GCP AMD machines (prefixed with "n2d") all had high 99th percentile latencies relative to the c2 and n2 machines, which run on Intel processors.



Cockroach Labs

STORAGE I/O BENCHMARK

Measuring Storage IOPS, Throughput, and Latency with FIO

About FIO

FIO (Flexible I/O tester) is an open-source benchmark suite designed for measuring storage I/O performance. FIO allows us to isolate disk performance characteristics by running workloads against the disk itself, as opposed to a file system.

Measurement

We measured storage I/O performance with FIO (Flexible I/O tester) version 3.1. In previous years, we benchmarked storage I/O performance with Sysbench, but switched to FIO because we found its flexibility as a benchmarking tool quite appealing. Among other things, FIO allowed us to isolate disk performance characteristics by running workloads against the disk itself, as opposed to a file system[5].

Overall, we assessed each of the cloud's storage I/O performance by analyzing the following metrics:

- Input/Output Operations Per Second (IOPS): number of disk operations per second.
- Throughput: (also known as bandwidth) is the quantity of data being sent and received over a time period.
- Disk Latency: Similar to network latency, this represents the round-trip time to fulfill a request.

We assessed those benchmarks on both locally-attached solid-state drives (local SSDs) and durable network-attached storage (e.g. AWS Elastic Block Store (EBS)). Unlike in past years, we made network-attached disks the focus of the report. Choosing the correct disk type is a complex task which can vary based on use case. In general, locally-attached storage is cheaper and provides great performance, but it makes certain tradeoffs in availability and durability[6]. CockroachDB allows operators to change the replication factor and pin data to specific localities. These are some of the ways we give operators the flexibility to make informed decisions about tradeoffs when running their deployments, regardless of underlying disk types.

Our usage of network-attached disks in CockroachCloud served as our primary motivation for making them the focus of this year's report. That being said, we did still test locally-attached storage this year, and those results are in the appendix for brevity.

Among the network-attached disks, we looked at general purpose disks, such as GCP's pd-ssds. In addition, we looked at more advanced network-attached disks, such as AWS' io2 disks.

As a more expensive set of disks than general purpose disks, the advanced network-attached disks support reserving a specific number of IOPS and are designed for better performance over general purpose disks.

We have configured each machine to have a 2.5TB disk. In addition, advanced disks (ultra disk on Azure and io2 on AWS) were provisioned with sufficient IOPS allocation to determine the maximum achievable IO performance.

[5] File system performance depends on many factors: including file system type, configuration, memory caching, etc.
[6] https://cloud.google.com/compute/docs/disks/local-ssd#data_persistence

Monthly Cost of Each Disk Type Across Clouds

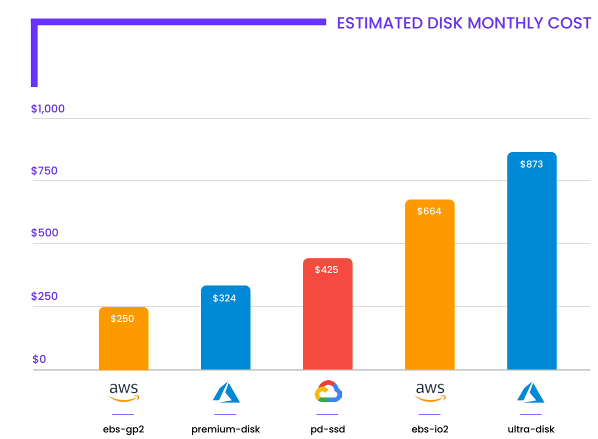
An important aspect when evaluating storage options across Clouds can be cost[7 8 9 10]. This is also a challenging calculation, as it varies greatly depending on the use case. This is made harder still because even as all network disks can be sized, certain options can be tuned on different axis' while others cannot.

However, the io2 disks and ultra disks allow users to configure the exact limit on the number of IOPS as well. Azure takes ultra disks a step further, allowing users to also tune throughput to their needs. This flexibility is great, as you pay for exactly what you need. But it also requires intimate familiarity with the use case (or some trial and error).

We used the following disk configurations for all machines:

- 2500 GB sized disks[11]
- 5400 IOPs for io2 (AWS) and ultra disk (Azure)
- 300 Mbps throughput for ultra disk(Azure)

With so many available variables, we decided to graph an example pricing at a snapshot of values. We chose 300 Mbps as the allocated bandwidth. This roughly represents 80% of the maximum throughput we observed on our Azure ultra disk benchmarks, excluding the Standard_L16s_v2 instance, which was considerably higher. We then picked 5400 as the IOPS limit by distributing the bandwidth across various sized requests[12 13] and applied the same limit to the io2 disk as well.



Based on our assumptions, AWS general purpose (gp2) disks were the most cost-effective option. Azure's premium disk and GCP's pd-ssd were 30% and 70% more expensive respectively. Among the advanced disk options, Azure's ultra disk was 31% more expensive than AWS' io2 disk.

[7] AWS EBS pricing: https://aws.amazon.com/ebs/pricing/
[8] Azure Premium Disk Pricing: https://azure.microsoft.com/en-us/pricing/calculator/?service=storage
[9] Azure ultra disk Pricing: https://azure.microsoft.com/en-us/pricing/details/managed-disks/
[10] GCP Persistent Disk Pricing: https://cloud.google.com/persistent-disk#section-7
[11] Azure does not provide a 2500 GB option; we used the average cost of a 2TB option and scaled it to 2.5TB.
[12] 25% at 4KB, 50% at 8KB, and 25% at 16KB.
[13] Each cloud counts requests up to 16KB as a single request. However, your application may issue requests of varying sizes. It is important to understand your application characteristics in order to properly size your disks.

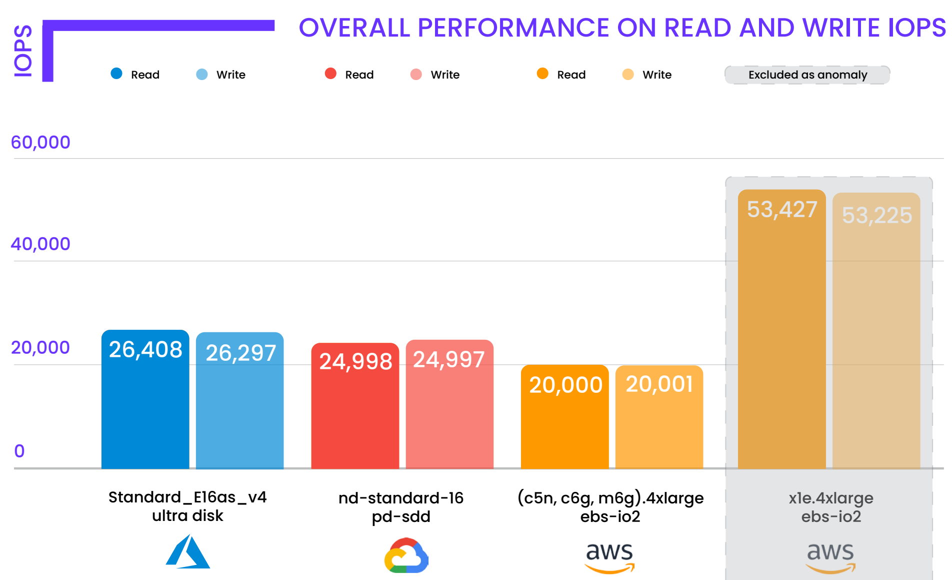
GCP's General Purpose Disk Performs on Par with More Expensive Offerings from AWS and Azure

With GCP's advanced disk type (Extreme PD) being unavailable to us at the time of testing, it was notable to see how well GCP's general purpose (pd-ssd) disk performed against Azure and AWS' advanced disks. In fact, GCP's top-performing machine (n2-standard-16 / pd-ssd) achieved only 5% fewer read IOPS than the top-performing Azure machine (Standard_E16as_v4 / ultra-disk) with the more expensive ultra disk.

What's going on with the AWS x1e.4xlarge?

According to AWS documentation, the x1e.4xlarge should be restricted to a maximum of 10,000 IOPS. We attempted to benchmark x1e.4xlarge with various block sizes, and in each instance, we were bound by bandwidth (1750 Mib/s), but not by IOPS. In our testing, we were able to drive 53,427 IOPS to the io2 disk. This was more than double the performance of both GCP and Azure, and more than five times the machine's expected performance. We believe this result was anomalous and excluded it from the ranking as it did not perform in line with published expectations.

AWS is transparent when it comes to the expected IO performance of each EBS optimized machine.[14] By and large, each AWS machine we benchmarked met its published expectations. Three AWS machines (c5n.4xlarge, c6g.4xlarge, m6g.4xlarge) met their target 20,000 IOPS and bandwidth (over 4Gib/s to EBS). AWS restricted those machines to maximum amounts of 20,000 IOPS, even though the io2 drives themselves were capable of delivering much more. In order to deliver more than 20,000 IOPS, users must upgrade to a larger VM.



[14] https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-optimized.html

AZURE STORAGE (READ AND WRITE) IOPS PERFORMANCE

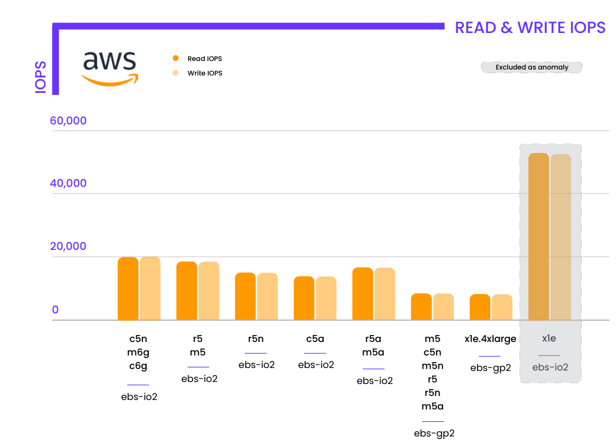
Similar to AWS, Azure machines' IOPS performance correlated with their pricing. The machines with the ultra disks achieved 3.4 times more IOPS than the same machines with premium disks.

GCP STORAGE (READ AND WRITE) IOPS PERFORMANCE

While most of GCP's machines performed identically, GCP's c2-standard-16 machine had by far the worst storage IOPS performance relative to other GCP machines. Specifically, it had 68% and 84% fewer read and write IOPS than the averages of the other GCP machines. These results were in line with GCP's published performance targets for c2 machines: these machines were made for compute/memory dense applications, and they have stricter restrictions on the amount of I/O.

AWS STORAGE (READ AND WRITE) IOPS PERFORMANCE

AWS Machines consistently performed in line with how they were priced. The machines with io2 disks outperformed the machines with gp2 disks. Specifically, the c5, m6g, and c6g machines performed 2.67 times more IOPS than the same machines with gp2 disks (20,000 IOPS vs 7600). Regarding their pricing, the c5.4xlarge machine running on an io2 disk with 20,000 IOPS was substantially more expensive than its counterpart with a gp2 disk ($2,015 vs $739).



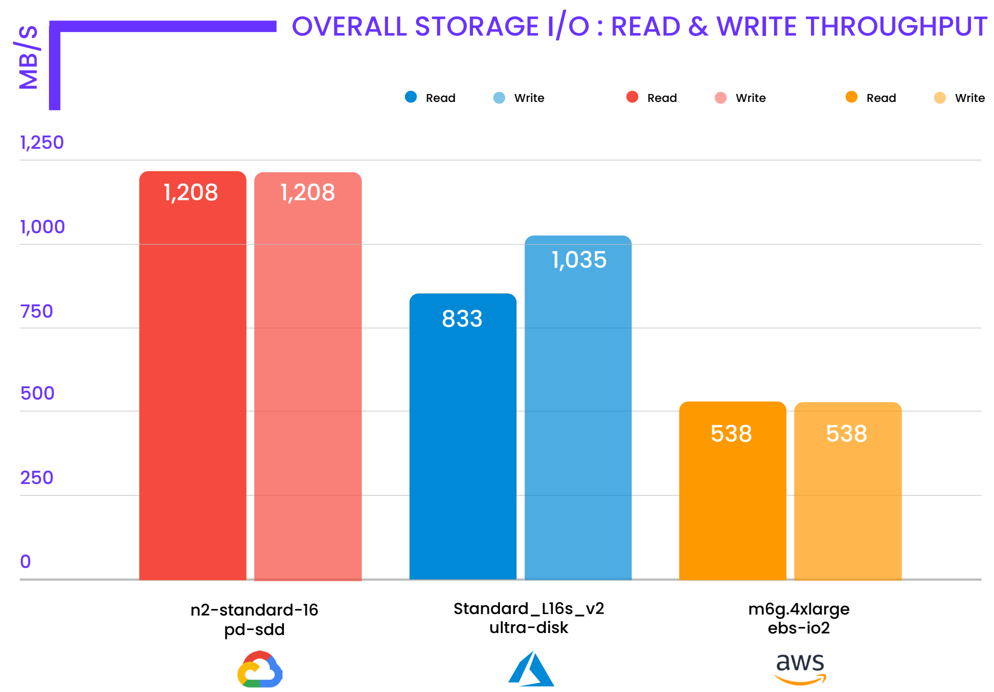
Note: Where AWS machines demonstrated minimal variability, results have been grouped for improved readability.

STORAGE I/O THROUGHPUT OVERALL

With Cheaper Storage, GCP Has More Read and Write Storage I/O Throughput than Azure and AWS

GCP achieved the most storage I/O throughput and proved to be more cost efficient in this experiment. GCP's top-performer (n2-standard-16) achieved 45% and 125% more read throughput than Azure and AWS, respectively. In addition, GCP had 17% and 125% more write throughput than Azure and AWS, respectively. Regarding cost efficiency, GCP's top-performing monthly machine cost was 80% and 12% less than Azure and AWS' monthly machine costs, respectively.

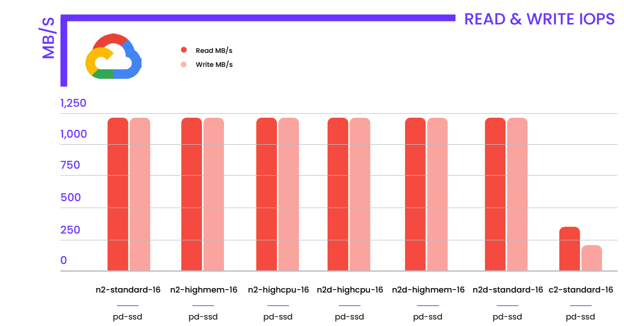
It should be noted that both AWS and Azure restricted the amount of bandwidth to disks based on the machine size (16 cores) and disk type, while GCP restricted only based on the disk size. As a result, lower throughput numbers for Azure and AWS were expected. The only way to increase throughput for Azure and AWS was to use larger machines, which would increase costs. [15][16]



[15] https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-optimized.html
[16] https://docs.microsoft.com/en-us/azure/virtual-machines/lsv2-series

GCP STORAGE THROUGHPUT PERFORMANCE

Consistent with GCP's storage IOPS performance, most of the GCP machines performed identically. As an exception, GCP's c2-standard-16 significantly underperformed relative to the other GCP machines. Specifically, c2-standard-16 had 73% less read and 82% less write throughput than other GCP machine averages.[17]



AZURE STORAGE THROUGHPUT PERFORMANCE

As the top-performer among Azure machines, the Standard_L16s_v2 machine with ultra disk delivered. Consistent with its promise as a storage-optimized machine, Standard_L16s_v2 with ultra disk achieved 163% more read and 261% more write throughput than the averages of the other Azure machines.

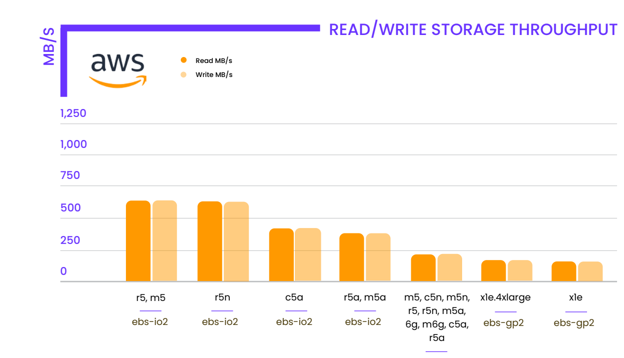
Among all machines tested across cloud providers, machines with ultra disks uniquely allow users to configure bandwidth. However, to maintain consistency across all disks, we only configured IOPS and disk size, not bandwidth.

Considering cost across Azure machines, there was a correlation between throughput performance and pricing. Ultra disks were 169% more expensive than premium disks. At a higher price point, machines with ultra disks had 57% more read throughput and 107% more write throughput than machines with premium disks.

AWS STORAGE THROUGHPUT PERFORMANCE

Similar to the price correlations we observed with AWS storage IOPS performance, machines with the more expensive io2 disks outperformed machines with gp2 disks. Specifically, machines with io2 disks averaged 74% more read throughput than machines with gp2 disks. Regarding pricing, the monthly cost of io2 disks were 166% more expensive than gp2 disks.

It should be noted that the AMD based machines underperformed other machines in this throughput benchmark. This result was consistent with how AWS provisioned those instances: AMD-based instances had ~30% less bandwidth available to them than the Intel-based machines.[18]



Note: Where AWS machines demonstrated minimal variability, results have been grouped for improved readability.

[17] https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-optimized.html
[18] This result is consistent with published performance expectations for c5 machines.

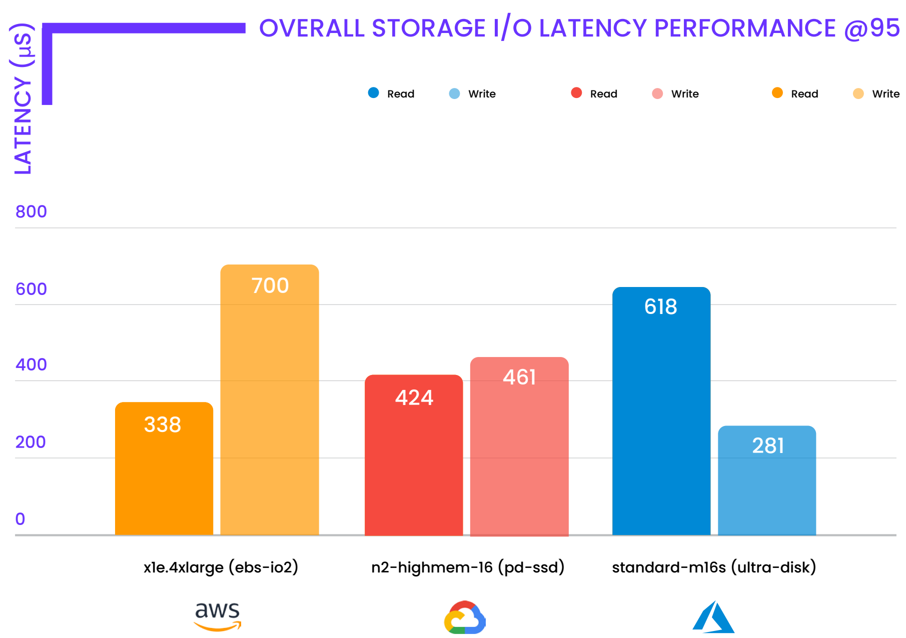
STORAGE I/O LATENCY OVERALL:

Storage I/O Read and Write Latency Performance Varied Among the Clouds

While AWS had the best storage I/O read latency, it had the worst storage I/O write latency. AWS' best read latency was 20% and 45% lower than GCP's and Azure's, respectively. In addition, AWS' best write latency was higher than GCP's and Azure's by 52% and 150%, respectively.

Inverse to AWS, Azure had the lowest write and the highest read latency. Azure's machine had 39% and 60% lower write latency than GCP and AWS, respectively. Azure's machine had 46% and 83% higher read latency than GCP's and AWS'.

Lastly, GCP held its own, coming in second place for both read and write latency while also being the most cost efficient. GCP's top performing machine monthly cost was 77% and 84% lower than AWS and Azure, respectively.



AWS STORAGE I/O LATENCY PERFORMANCE

Consistent with storage IOPS and throughput performance, AWS machine's latency is correlated with its pricing. Overall, the machines with the advanced io2 disks averaged 51% lower latency than the machines with general purpose disks.

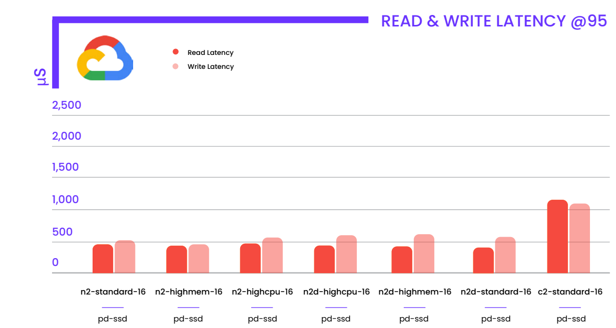
However, there was one anomaly. The memory-optimized x1e.4xlarge machine with a gp2 disk achieved the second lowest read and write latencies among all AWS machines. Specifically, x1e.4xlarge with ebs-gp2 was 83% lower than the average read latency for machines with io2 disks and 92% lower than other machines with gp2 disks.

Note: Where AWS machines demonstrated minimal variability, results have been grouped for improved readability.

GCP STORAGE I/O LATENCY PERFORMANCE

GCP's c2-standard-16 machines continued to perform poorly relative to other GCP machines. It delivered 187% higher read latency and 110% higher write latency than the averages of the other GCP machines. Outside of the c2-standard-16 machine, GCP read and write latency had minimal variance.

The c2 machine performance is consistent with the fact that it has lower bandwidth to storage disk, compared to other machines.



AZURE STORAGE I/O LATENCY PERFORMANCE

In the same way as Azure's storage IOPS and throughput results, Azure's storage I/O latency performance aligned with how its machines are priced. The machines with the more expensive ultra disks had an average read latency 72% lower than that of machines with premium disks. Also, the average monthly cost of machines with ultra disks were 40% more expensive than the average monthly cost of machines with premium disks.

We have observed that ultra disks had higher read latency than write latency. Ultra disks had substantially higher cached read performance, and since our random reads benchmark bypassed the cache, the higher latencies for random reads were expected.

OLTP BENCHMARK

Benchmarking Clouds for OLTP Performance

About Cockroach Labs Derivative TPC-C Benchmark

The Cockroach Labs Derivative TPC-C Benchmark is based on TPC-C, the industry standard benchmark for On-line Transaction Processing (OLTP) performance. It simulates an industry-agnostic business with an OLTP database that manages, sells, or distributes a product.

Cockroach Labs Derivative TPC-C measures databases across two different metrics:

- Throughput: Measured as throughput-per-minute (tpm), which in practical terms measures the number of orders processed per minute.
- Scale: Measured as the total number of warehouses supported. Each warehouse is a fixed data size and has a max amount of tpm that it is allowed to support, so the total data size of the benchmark is scaled proportionally to throughput.

Note: The Cockroach Labs Derivative TPC-C Benchmark is derived from the TPC-C Benchmark. As such, it is not comparable to published TPC-C Benchmark results, as these results do not comply with the complete TPC-C Benchmark Specification. Our implementation can be found here, and the schema can be found here.

Measurement

We measured overall performance of each of the clouds by running the Cockroach Labs Derivative TPC-C Benchmark, a derivative of TPC-C, the industry standard benchmark for On-line Transaction Processing (OLTP) performance. As an OLTP database, CockroachDB performance is ultimately judged by how well it serves real-world workloads such as financial ledgers and e-commerce shopping carts. This benchmark simulates an industry-agnostic business with an OLTP database (see schema below) that manages, sells, or distributes a product.

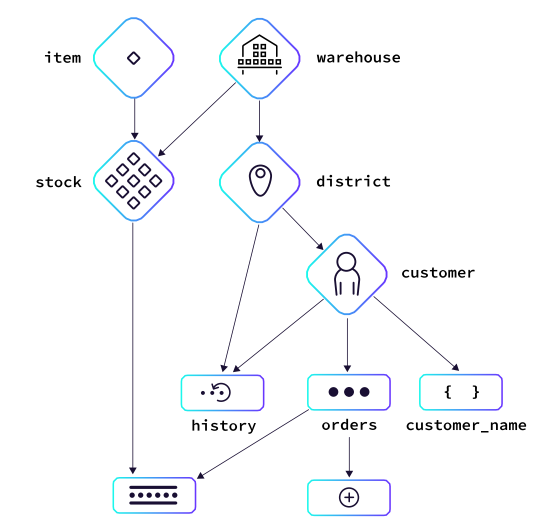
Cockroach Labs Derivative TPC-C measures databases across two different metrics. One is a throughput metric: throughput-per-minute(tpm), which in practical terms measures the number of orders processed per minute.

The other is linear scale, measuring the total number of warehouses supported. Each warehouse is a fixed data size and has a max amount of tpm that it can support, so the total data size of the benchmark is scaled proportionally to throughput.

Building on the tpm metric, we benchmarked the industry-standard measure of price per tpm ($/tpm) to measure cost efficiency. For simplicity in this experiment, we calculated the cost component of $/tpm with the default on-demand pricing published by each cloud.

Machine cost efficiency affects profitability. If you can achieve top performance but pay twice the amount of another machine with comparable performance, it suddenly makes less sense..

Measuring cost efficiency with the standardized $/tpm benchmark allowed for the more fair and accurate comparisons of machines across clouds as well as within clouds.



Cockroach Labs Derivative TPC-C Schema

For this experiment, we used the following criteria, environment, and steps for each Cockroach Labs Derivative TPC-C run:

Benchmark "Passing" Criteria[19]:

- 90th Percentile Latency < 5 Seconds

Environment:

- 3- node CockroachDB cluster
- 4th node for simulating the benchmark load

Steps:

1. Start cluster with 2,500 warehouses
2. Run load for 30 minutes
3. Check if cluster was still in a "passing" state
4. If not "passing," end experiment
5. If "passing," add 250 warehouses
6. Repeat Step 2

Note on Results by Machine Type:
The charts below include only those machines that passed at 2,500 warehouses. All other machines have been excluded from the results.

[19] These Cockroach Labs Derivative TPC-C results are unofficial and have not been audited.

TPC-C FINALE

Cost Efficient Compute- and Memory-Optimized Machines Thrive in OLTP Benchmark Finale

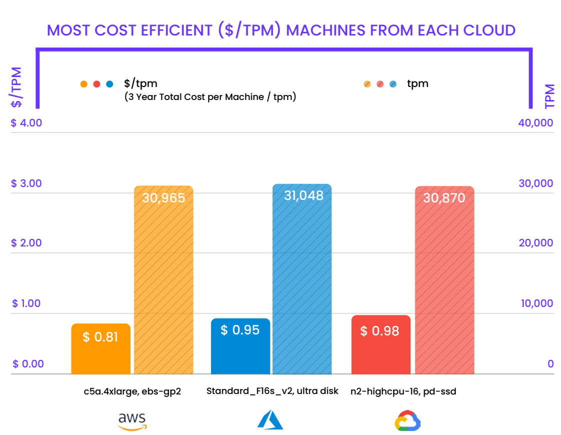
In our final benchmarks, running a simulation of real-world OLTP workloads, cheaper machines with general purpose disks won for both AWS and GCP.

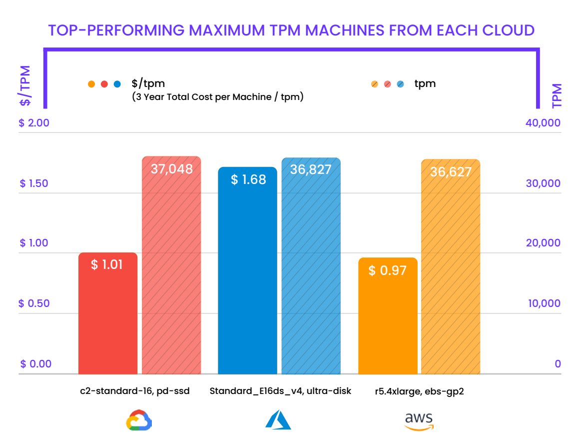
AWS provided the most cost-efficient machine (c5a.4xlarge / ebs-gp2) with a general purpose disk, and GCP's machine (c2-standard-16 / pd-ssd) narrowly achieved the highest maximum tpm amount with a general purpose disk. Although it had the worst storage I/O performance among GCP machines, c2-standard-16 won the maximum tpm benchmark among all machines for all clouds.

Why the discrepancy?
Cockroach Labs Derivative TPC-C is a compute and memory intensive workload. While it also values storage I/O performance, we found that the benchmark did not drive sufficient load at the storage I/O level to prove the value of io2 and ultra disks for this benchmark. As a result, memory- and compute-optimized machines thrived, and storage-optimized machines with advanced disks were underutilized.

In action, our benchmark runs found c2-standard-16 to deliver on its promise as an industry-leading compute-optimized machine. As discovered in the network and storage I/O experiments, both Azure's machines with ultra disks and AWS' machines with io2 disks had additional restrictions to the network and storage capacity that may have affected their tpm results.

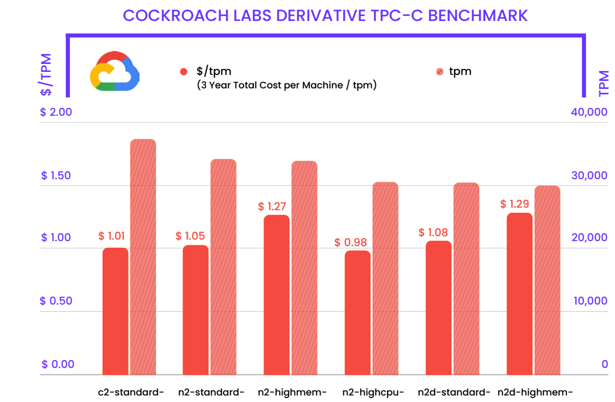
To improve OLTP performance, we found it is better to spend on more nodes, memory, and better CPUs. In use cases outside of Cockroach Labs Derivative TPC-C, machines with advanced disks could be more appropriate for heavier read or write tasks, and critical workloads with demanding latency requirements.





GCP TPM AND $/TPM

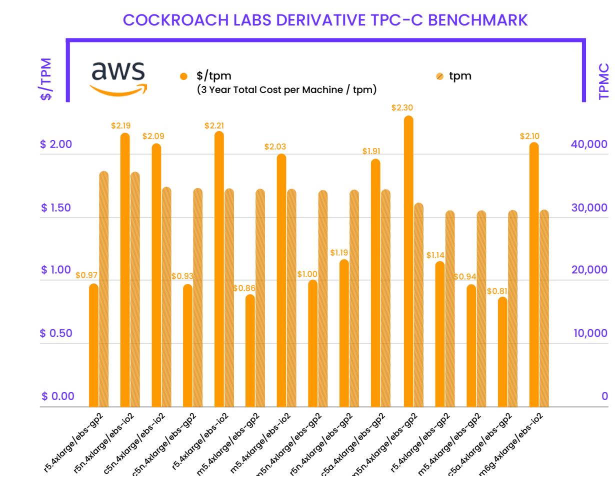
For maximum tpm among GCP machines, the lone C2 machine outperformed N2 and N2D (AMD) machines. As the least cost efficient GCP machine, n2d-highmem-16 had a 20% higher $/tpm amount than the average $/tpm of the other GCP machines.



AWS TPM AND $/TPM

AWS machines with general purpose (gp2) disks thrived in this experiment. Their average $/tpm was 47% less than the average $/tpm of machines with io2[20]. At the same time, machines with gp2 disks demonstrated comparable tpm performance with machines with io2 disks.

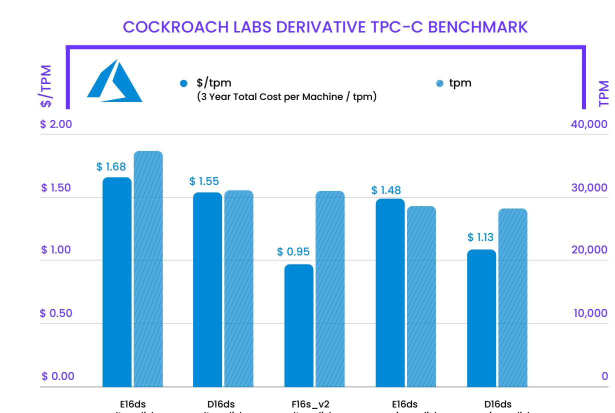
In fact, the top-performing AWS machine for maximum tpm was a machine with a general purpose disk (r5.4xlarge with ebs-gp2). This result makes sense since a large gp2 disk (2,500 GB) delivers 7,500 IOPS -- more than enough for this particular benchmark. In addition, the most cost-efficient machine's maximum tpm was only 15% lower than the top-performing maximum tpm among AWS machines.



AZURE TPM AND $/TPM

Azure's machines with ultra disks were competitive for maximum tpm and $/tpm workloads. They delivered at a minimum 11% more tpm while being priced on par and even less than machines with general purpose disks (and premium disks). This was in stark contrast to AWS, where the difference between their general purpose disk (gp2) and more expensive io2 disk was marginal.

This higher tpm meant the ultra disk machines were more competitive in the $/tpm benchmark as well, so much so that Azure was the only cloud where a machine with a more expensive disk (Standard_F16s_v2 / ultra disk) came out on top in the $/tpm metric. This is partly because of the cheaper base price of the machine, and partly because its premium disk variant did not pass a 2500 warehouse run.



Note: This chart includes only those machines that passed at 2,500 warehouses. All other machines have been excluded from the results. For the purpose of $/tpm calculation, we used disk pricing described in the IO section of the report.

[20] 2500 GB io2 drive with 5400 IOPS

Cockroach Labs

Conclusion: A Winner, But...

For the first time in our three years of putting together the Cloud Report, GCP came out on top. But declaring a winner was much harder to declare than in years past.

Each of the cloud providers stood out in different ways:

- GCP had the best overall performance across the set of microbenchmarks and achieved the best raw throughput on the OLTP benchmark, Cockroach Labs Derivative TPC-C.
- Arguably surpassing AWS for the first time, Azure's ultra disks stood out in the storage I/O experiment.
- Finally, AWS offered the most cost efficient machine on the OLTP benchmark, Cockroach Labs Derivative TPC-C.

It is important to point out that each of the cloud providers showcased overall growth in machine performance since the 2020 report. And since we know that every CockroachDB customer has specific cost and performance needs, we believe each provider has a compelling case for how their machines are ready to support today's applications.

About Cockroach Labs

Cockroach Labs is the creator of CockroachDB, the most highly evolved cloud-native, distributed SQL database on the planet. Helping companies of all sizes—and the apps they develop— to scale fast, survive anything, and thrive everywhere. CockroachDB is in use at some of the world's largest enterprises across all industries, including Equifax, Bose, Comcast and some of the largest companies in banking, retail, and media. Headquartered in New York City, Cockroach Labs is backed by Altimeter, Benchmark, GV, Firstmark, Index Ventures, Redpoint Ventures, Sequoia Capital, Tiger Capital, and Workbench.

For more information, please visit cockroachlabs.com

Appendix I

Machines tested

	Cloud	Machine Type	CPU Platform	Frequency (GHz)	Memory (GiB)	Disk Type
General Purpose	AWS	m5.4xlarge	Intel Cascade Lake or Skylake-SP	3.1 GHz	64	io2, gp2
		m5d.4xlarge	Intel Cascade Lake or Skylake-SP	3.1 GHz	64	2 x 300 NVMe SSD
		m5n.4xlarge	Intel Cascade Lake	3.1 GHz	64	io2, gp2
		m5dn.4xlarge	Intel Cascade Lake	3.1 GHz	64	2 x 300 NVMe SSD
		m5a.4xlarge	AMD EPYC 7000	2.5 GHz	64	io2, gp2
		m5ad.4xlarge	AMD EPYC 7000	2.5 GHz	64	2 x 300 NVMe SSD
		m6g.4xlarge	AWS Graviton2	2.3 GHz	64	io2, gp2
		m6gd.4xlarge	AWS Graviton2	2.3 GHz	64	1 x 950 NVMe SSD
	Azure	Standard_D16ds_v4	Intel Cascade Lake	3.4 GHz	64	Premium Disk, ultra disk
		Standard_D16as_v4	AMD EPYC 7452	3.3 GHz	64	Premium Disk, ultra disk
	GCP	n2-standard-16	Intel Cascade Lake	3.4 GHz	64	PD-SSD
		n2d-standard-16	AMD EPYC 7002	3.3 GHz	64	PD-SSD
Compute Optimized	AWS	c6g.4xlarge	AWS Graviton2	2.3 GHz	32	io2, gp2
		c6gd.4xlarge	AWS Graviton2	2.3 GHz	32	1 x 950 NVMe SSD
		c5n.4xlarge	Intel Skylake-SP	3.4 GHz	32	io2, gp2
		c5.4xlarge	Intel Cascade Lake or Skylake-SP	3.5 GHz	32	io2, gp2
		c5d.4xlarge	Intel Cascade Lake or Skylake-SP	3.5 GHz	32	1 x 400 NVMe SSD
		c5a.4xlarge	AMD EPYC 7002	3.3 GHz	32	io2, gp2
	Azure	Standard_F16s_v2	Intel Cascade Lake	3.4 GHz	32	Premium Disk, ultra disk
	GCP	n2-highcpu-16	Intel Cascade Lake	3.4 GHz	16	PD-SSD
		n2d-highcpu-16	AMD EPYC 7002	3.3 GHz	16	PD-SSD
		c2-standard-16	Intel Cascade Lake	3.8 Ghz	64	PD-SSD
Memory Optimized	AWS	x1e.4xlarge	Intel Haswell	2.3 GHz	448	1 x 480 SSD
		r5.4xlarge	Intel Cascade Lake or Skylake-SP	3.1 GHz	128	io2, gp2
		r5d.4xlarge	Intel Cascade Lake or Skylake-SP	3.1 GHz	128	2 x 300 NVMe SSD
		r5n.4xlarge	Intel Cascade Lake	3.5 GHz	128	io2, gp2
		r5dn.4xlarge	Intel Cascade Lake	3.5 GHz	128	2 x 300 NVMe SSD
		r5a.4xlarge	AMD EPYC 7000	2.5 GHz	128	io2, gp2
		r5ad.4xlarge	AMD EPYC 7000	2.5 GHz	128	2 x 300 NVMe SSD
	Azure	Standard_E16ds_v4	Intel Cascade Lake	3.4 GHz	128	Premium Disk, ultra disk
		Standard_E16as_v4	AMD EPYC 7452	3.4 GHz	128	Premium Disk, ultra disk
		Standard_M16ms	Intel Cascade Lake	2.5 GHz	437.5	Premium Disk, ultra disk
	GCP	n2-highmem-16	Intel Cascade Lake	3.4 GHz	128	PD-SSD
		n2d-highmem-16	AMD EPYC 7002	3.3 Ghz	128	PD-SSD
Storage Optimized	AWS	i3en.6xlarge	Intel Cascade Lake or Skylake-SP	3.1 GHz	192	2 x 7500 NVMe SSD
		i3.4xlarge	Intel Broadwell	2.3 GHz	122	2 x 1.9 NVMe SSD
	Azure	Standard_L16s_v2	AMD EPYC 7551	3.0 GHz	256	Premium Disk, ultra disk
	GCP*	-	-	-	-	-

* GCP does not market a "storage-optimized machine", but they do allow users to configure the number of local SSDs (up to 8 for the n2/c2 machines we tested).

Reproduction Steps

We are making everything required for reproduction—steps and resources, scripts, configurations, and instructions—available through a public repository. Our goal is to ensure these resources will always be free and easy to access. We encourage you to review the specific steps used to generate the data in this report.

Note: if you wish to provision nodes exactly the same as we do, you can use this link to access the source code for Roachprod, our open-source provisioning system.

Testing Methodology

Our testing focused on the following broad axes:

- CPU Performance
- Network Performance
- Storage I/O Performance
- OLTP Performance (Cockroach Labs Derivative TPC-C)

For the machines tested across each cloud provider, we held the operating system constant by using Ubuntu 18.04. Each benchmark ran on a 16 vCPU machine to be consistent with last year's report.

While testing for this year's report, each benchmark was run in the us-east region and with machines provisioned in the same availability zone. We did 10 runs of the CoreMark benchmark and reported the mean for our results.

All other benchmarks were time-based tests. It is worth noting that we expect performance to vary across the time of day, but we did not study the impact of time of day on these benchmarks.

Updated Methology in the 2021 Cloud Report

All microbenchmarks have been revised to be more representative of real-world performance:

- Network Performance Test: We replaced Ping and iPerf with Netperf version 2.6.0.
- CPU Performance Test: We replaced stress-ng with CoreMark version 1.0.
- IO Performance Test: We replaced sysbench with FIO (Flexible I/O tester) version 3.1.

All benchmarks and microbenchmarks are focused on machines with network-attached storage. Results for machines with locally-attached storage have been moved to the appendix.

Machine Types Tested on AWS, Azure and GCP

We tested the newest machines available to us at the time of testing. These machines are bucketed based on use cases by cloud types -- general purpose, memory-, storage-, and compute- optimized.

We also configured each machine using different storage options available to us including both locally attached storage and different tiers of network-attached storage options.

AWS

Of all the three clouds, AWS has the most out of the box machine configurations. Instead of specifying a flag during provisioning to get better network performance or locally-attached disks, AWS users must choose a different machine type. Thankfully, there is a uniform pattern of naming across all instances.

- `a` variants: denotes an AMD processor
- `d` variants: denotes instances with locally-attached storage
- `n` variants: suggests better network performance

These designations can be combined in many cases. Concretely, the m5dn.4xlarge machine is identical to the m5.4xlarge machine, except it claims better network performance and has a locally-attached disk. Finally, AWS is the only cloud that offers ARM-based processors, which they call Graviton2 processors. These are the c6g and m6g series.

Azure

Similar to the other clouds, Azure has its own unique naming scheme. The VM series is referenced by the capital letter preceding the vCPU count.

- `a` suffix: denotes an AMD processor
- `d` suffix: denotes VMs which have locally-attached storage
- `s` suffix: designates the ability to attach a premium managed disk, such as premium disk or ultra disk

There are a few exceptions that deviate from this general naming scheme though — for example, the Standard_D16as_v4 includes locally-attached disks but lacks the `d` suffix and the Standard_L16s_v2 machine uses an AMD processor without a referencing it in its name.

GCP

In contrast to AWS, GCP has very few named machines but they can be configured appropriately to serve user needs. For example, GCP is the only cloud that does not market a "storage-optimized machine," but it is also the only cloud that allows users to configure the number of local SSDs (up to 8 for the n2/c2 machines we tested).

- `d` suffix: denotes an AMD processor (for example, the `n2d` machine is the AMD counterpart to the `n2` machine)
- highcpu/standard/highmem nomenclature: dictates the vCPU to memory (RAM) ratio — 1:1, 1:4, or 1:8 respectively.

A Note on Locally-Attached Storage (Local SSDs):

Machines with locally-attached storage outperformed network-attached storage on storage benchmarks. This was to be expected, as a machine with a disk embedded in, rather than outside the VM, should do better.

CockroachDB and Performance Tuning

However, they are designed to be ephemeral and make certain trade-offs in availability and durability[21]. CockroachDB allows you to make informed choices about these trade-offs by allowing control over replication factor and data locality, among other things.

This year we decided to focus our benchmarking on managed disks, as we leverage them internally in CockroachDB's database-as-a-service offering CockroachCloud. We did evaluate machine types with locally-attached storage as well, however, those results have been moved to the appendix.

[21] https://cloud.google.com/compute/docs/disks/local-ssd#data_persistence

Cloud Terminology

Availability Zones or Zones:

An Availability Zone (AZ) or Zone is an isolated location— often a data center — within a region. A region is made up of multiple zones.

Extreme PD:

A GCP-specific disk type introduced in July 2020. These are suited for workloads requiring the highest performance. None were tested, as they were not available to us at the time of benchmarking.

Gp2 volumes:

Available on AWS, these “General Purpose SSD” disks are intended for most workloads.

Io2 volumes:

Available on AWS, these “Provisioned IOPS SSD” disks are intended for high performance, high throughput, mission-critical workloads.

PD-SSD (SSD persistent disks):

GCP’s pd-ssd disks are suited for applications and high-performance database workloads that require lower latency and more IOPS than standard persistent disks. SSD persistent disks are designed for single-digit millisecond latencies.

Placement Policy / Placement Group:

Rules designating where to place machines within an availability zone.

Premium Disk:

Azure’s “Premium SSD Managed Disks” promise high-performance and low-latencies for input/output (IO)-intensive workloads. Premium SSDs are suited for mission-critical production applications and are priced lower than Azure’s “ultra disks.”

Regions:

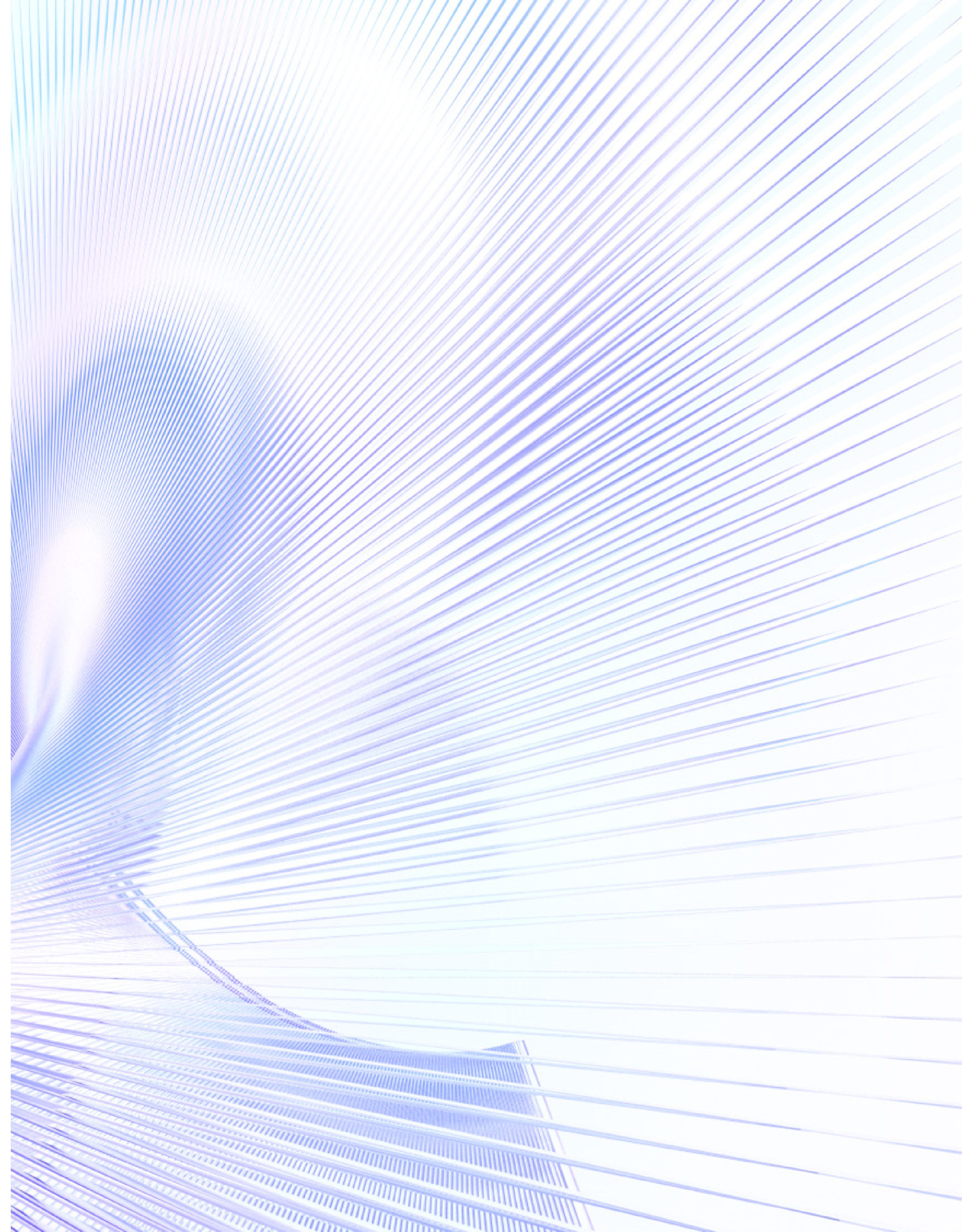
Regions are separate geographic areas that can be distributed around the world. The closer a region is to the end user or application, the better the network latency.

Ultra Disk:

Azure’s “Ultra Disks” promise high throughput, high IOPS, and low latency disk storage. Ultra disks are suited for data-intensive and transaction-heavy workloads. They are priced higher than Azure’s “premium disks.”

Volume:

Term that AWS uses for “disks.”

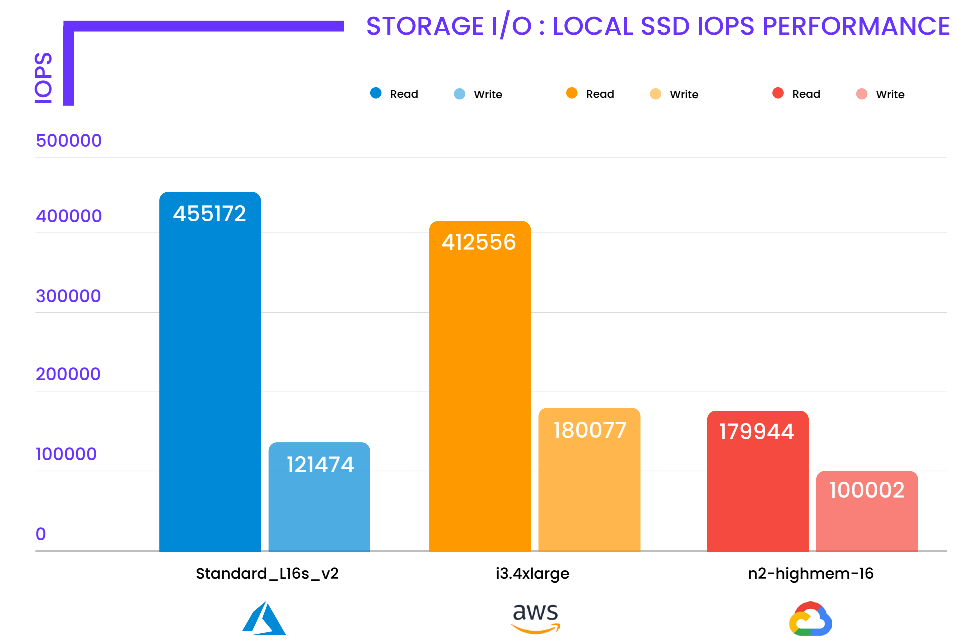


Appendix II

Storage I/O Results for Machines with Locally Attached Storage: Azure Outperforms GCP and AWS

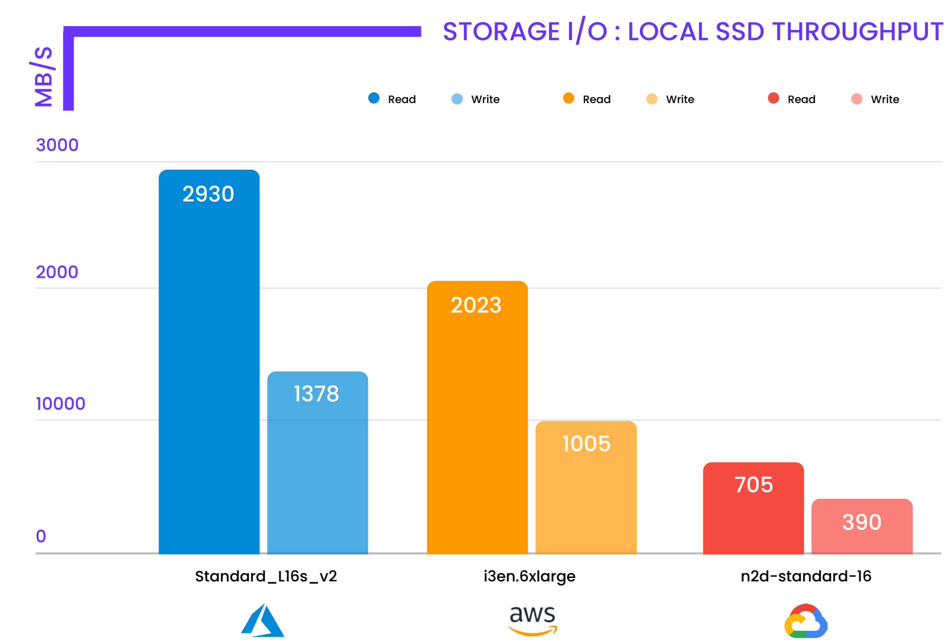
Storage I/O (Read and Write) IOPS Performance Overall (Local SSDs):

Azure's Storage Optimized Machine Outperforms AWS and GCP's Machines in Read IOPS Performance



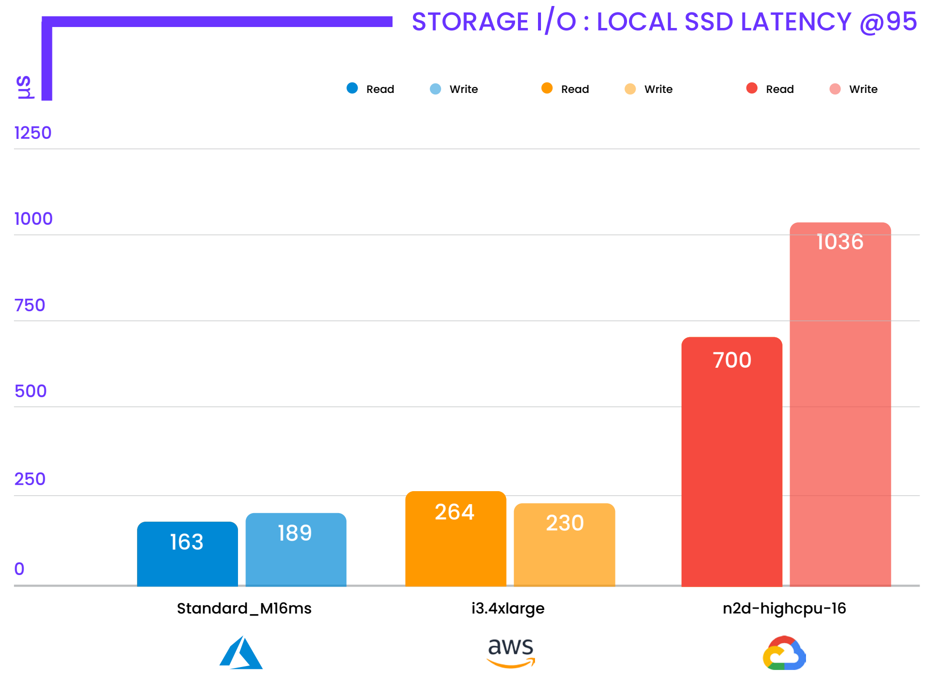
Storage I/O (Read and Write) Throughput Performance (Local SSDs)

Azure's Storage-Optimized Machine Outperforms AWS and GCP's machines in Read and Write Throughput Performance



Overall Storage (Read and Write) Latency Performance (Local SSDs):

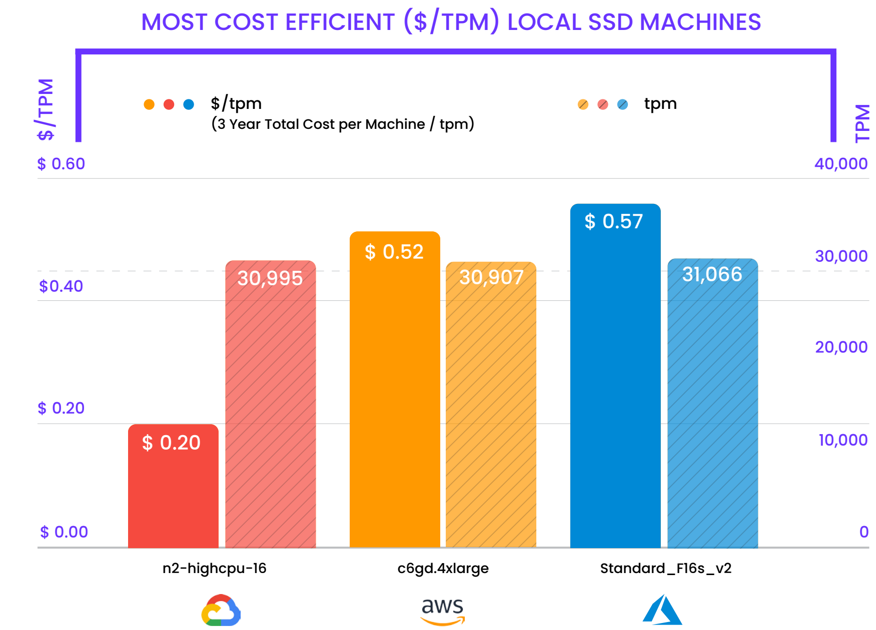
While GCP Performs Worst With Notably Highest Read and Write Latencies, Azure Achieves Lowest Read and Write Latencies



Cockroach Labs Derivative TPC-C: GCP Outperforms Azure and AWS

Lowest $/tpm from Each Cloud (Local-SSDs):

GCP Has Most Cost Efficient Machine Among Machines with Locally Attached Storage



Top-Performing Maximum tpm (Local-SSDs):

GCP Narrowly Surpasses AWS and Azure