Performance comparison between MinIO and Amazon S3 for Starburst Presto SQL

JULY 2019

Performance comparison between MinIO and Amazon S3 for Starburst Presto SQL

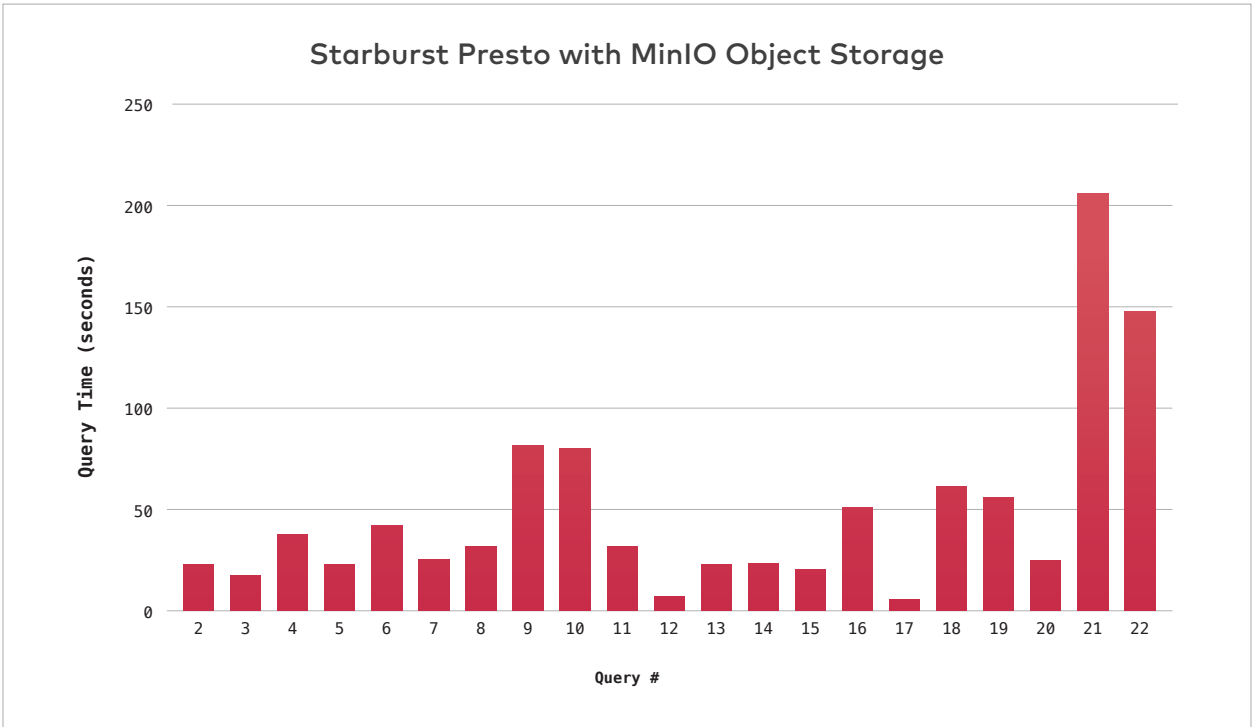
MinIO is a high-performance, object storage server designed for AI and ML workloads. It is fully compatible with the Amazon S3 API.

Machine learning, big-data analytics, and other AI workloads have traditionally utilized the map-reduce model of computing where data is local to the compute jobs. Modern computing environments have adopted a cloud-native architecture where storage and compute are disaggregated. This enables computing to become stateless, elastic, and scalable independent of storage. Object storage has become the de-facto standard for this architecture.

Presto is a distributed query engine that can analyze billions of records at very high speeds by distributing computational tasks across multiple servers. It is used by hyperscalers like Facebook, AirBnB and Dropbox.

The performance of underlying storage directly is critically important at this scale.

This document describes the benchmarking tests and the results of measuring the performance of Starburst Presto distribution with MinIO Object Storage.

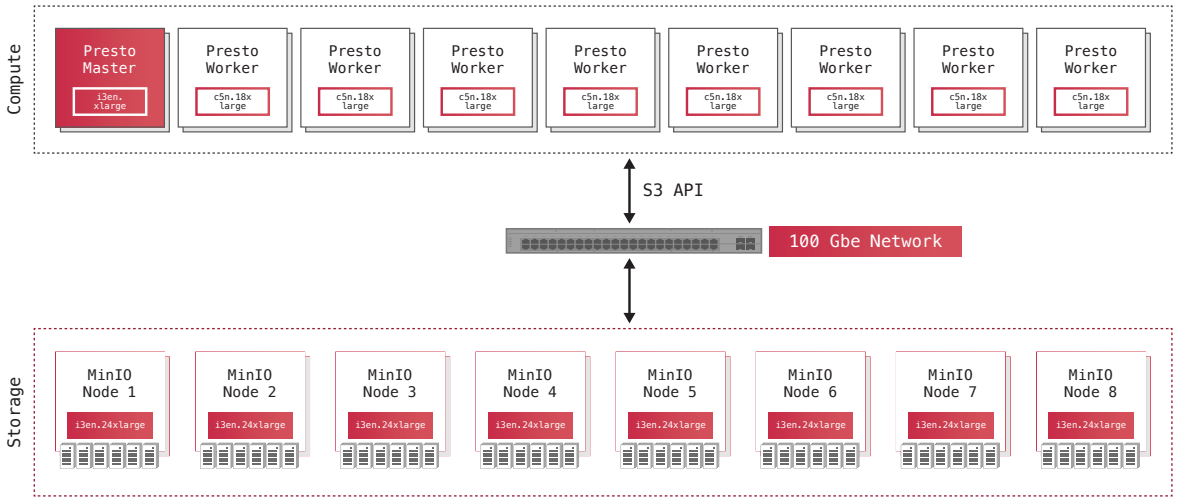


1. Benchmark Environment

1.1 Hardware

For the purpose of this benchmark, MinIO utilized AWS bare-metal, storage-optimized instances with local NVMe drives and 100 GbE networking. The nodes running Presto (c5n.18xlarge) have the highest available bandwidth to the S3 service.

Instance	# Nodes	AWS Instance type	CPU	MEM	Storage	Network
Presto Master	1	i3en.xlarge	4	32 GB	1 x 2500 GB	Upto 25 Gbps
Presto Worker	8	c5n.18xlarge	72	192 GB	EBS	100 Gbps
MinIO Server	8	i3en.24xlarge	96	768 GB	8 x 7500 GB	100 Gbps



1.2 Software

Property	Value
Presto	Starburst Distribution: 302-E.11
MinIO	Minio-RELEASE.2019-06-15T23-07-18Z
Benchmark	TPC-H™ Benchmark Scaling Factor: 1000
Server OS	CentOS Linux 7 (Core)

TPC-H™ benchmark

TPC Benchmark™ H is comprised of a set of business queries designed to exercise system functionalities in a manner representative of complex business analysis applications. These queries portray the activity of a wholesale supplier and add necessary context to the components of the benchmark.

Dataset

TPC Benchmark™ H, provides its own dataset comprising of eight tables representing a complex business environment. The tables are interrelated to each other, facilitating complex queries across multiple tables.

The size of the dataset is variable and chosen based on the underlying storage system. The size of the dataset is determined based on a scaling factor. A scaling factor of one leads to a dataset approximately 1GB in size, scaling factor 100 generates a dataset approximately 100GB in size and so on. The scaling factor is plugged into a dataset generation tool to generate the data.

This benchmark used scaling factor 1000. A summary detailing the dataset is presented below:

Table	# Records	# Records (SF: 1000)
Customer	150,000 * SF	150,000,000
Orders	1,500,000 * SF	1,500,000,000
Lineitem	6,000,000 * SF	6,000,000,000
Supplier	10,000 * SF	10,000,000
Part	200,000 * SF	200,000,000
PartSupp	800,000 * SF	800,000,000
Region	5	5
Nation	25	25
Total		8.66 Billion

The data was formatted in ORC (Optimized Row Columnar) format, and stored in a MinIO bucket. Converting to this format automatically compresses the data, which shrunk the data size to 273 GB.

Starburst Presto Performance Tuning

Starburst Presto was configured to utilize 1TB of aggregate memory across 8 worker nodes using the following settings:

```
$ cat etc/config.properties | grep 'query.max-memory'
query.max-memory=1024GB
query.max-memory-per-node=128GB
```

JVM configuration was updated to complement the above settings

```
$ cat etc/jvm.properties | grep 'Xmx'
-Xmx192G
```

The Hive connector was utilized to connect Starburst Presto to Minio. Hive connector was tuned using the following parameters:

Parameter	Value
hive.s3.max-connections	4000
hive.s3.multipart.min-file-size	128M
hive.s3.multipart.min-part-size	128M
hive.s3select-pushdown.enabled	false

In addition to the above optimizations, Starburst Presto was configured to run queries only on the higher capacity worker nodes and never run on the master node. This was achieved with the following setting:

```
$ cat etc/config.properties | grep 'node-scheduler'
node-scheduler.include-coordinator=false
```

2. Benchmark Results

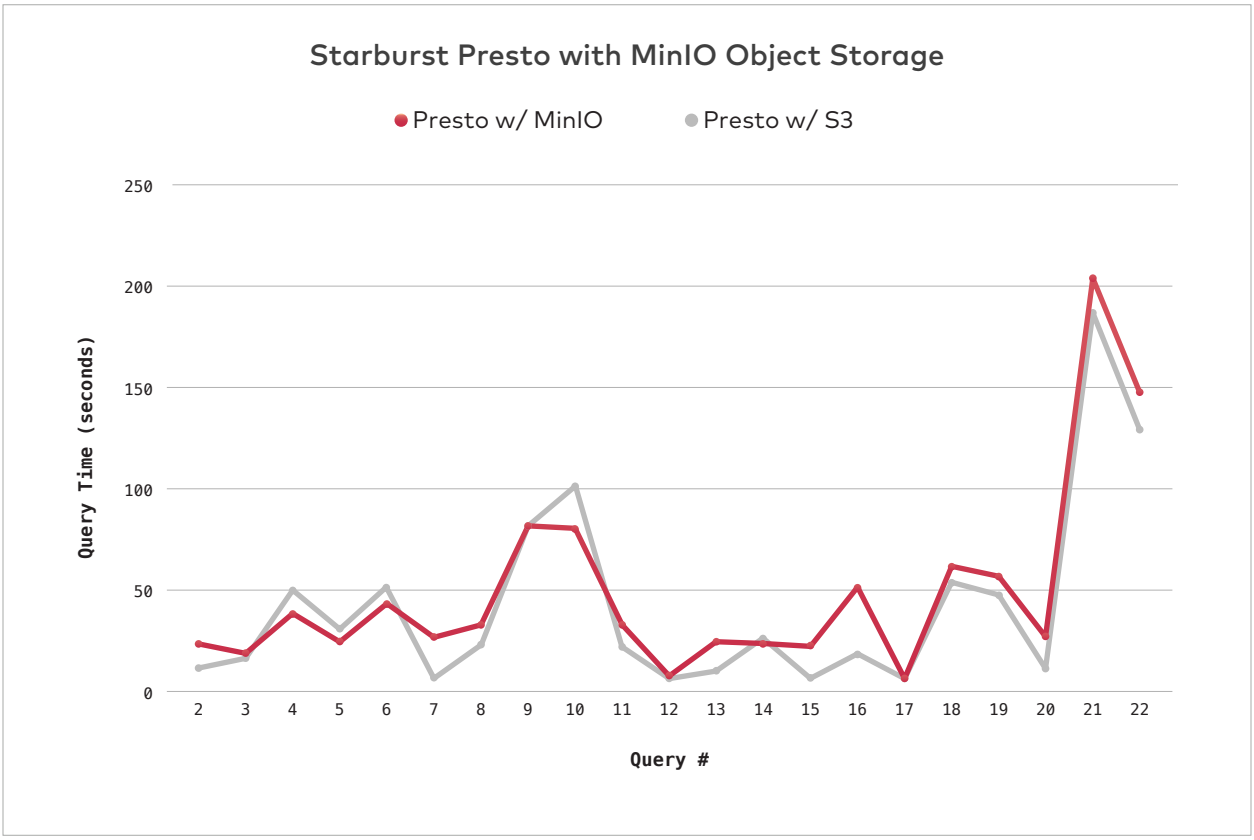
The time taken for each of the 22 TPC-H™ queries is presented below:

Node #	Query Execution Time (seconds)
	Presto w/ MinIO
1	23.1
2	17.9
3	38.4
4	23.8
5	43.2
6	25.9
7	32.7
8	82.2
9	81.0
10	32.8
11	7.48
12	23.6
13	23.9
14	21.5
15	51.6
16	5.91
17	61.8
18	57.3
19	26.1
20	206.4
21	148.2
22	7.7
Average	47.4

The values presented above serve to provide a reference point that will be used to benchmark future versions of MinIO and other applications serving similar use cases.

3. Comparing MinIO to Amazon S3

The same benchmark tests were run against data stored in Amazon S3 using the same hardware for Starburst Presto. It should be noted that MinIO is strictly consistent, whereas Amazon S3 in only eventually consistent. The performance was largely the same with some queries slower than MinIO and others faster. A graph summarizing the query times comparing MinIO and S3 for Starburst Presto workloads is presented below:



4. Results

The results show that the performance difference between running Starburst Presto backed by S3, as compared to Starburst Presto backed by MinIO is negligible - making MinIO an attractive alternative for large scale, high performance, data-intensive workloads in private cloud environments.