Comparing Single-node and Multi-node Cluster in Cassandra using YCSB Benchmark

Michael - 13514108
Computer Science/Informatics
School of Electrical Engineering and Informatics
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13514108@std.stei.itb.ac.id

***Abstract*—Cassandra is a distributed storage system that is used on top of a system that is composed of a lot of nodes. On some system, controlling many nodes means more works and affects the performance. Ensuring performance even if the system has a lot of nodes is important for a distributed system. To check the performance of a distributed system, YCSB benchmark can be used.**

Keywords—Cassandra; YCSB; benchmark;

I. Introduction

A large system such as Facebook is used by a lot of people around the world, which means the user of the system can be very far away from the company headquarter. If the user must always access the server which is located at the company headquarter, the latency will be very high for user who is very far away. Other than that, if only one server serving a lot of clients, once the server is down, clients cannot access the data, resulting in a single point of failure. The solution of these problems is using a distributed system, where the data of the users can be spread out to other servers, and the client can access the data from the nearest server.

In a distributed file system, the most important aspect is availability, which means the data should be available every time a client needs it. Most distributed system ensure the availability through replication, which is having the same data stored in more than one server, but most of the time not all the servers, because that would need a lot of storage.

Having multiple copies of data replicated in several servers would ensure the availability of the data. However, if the user requests a data from a server which does not have the data, the server would need to contact the other server to get the data before giving it to the user, therefore needing more time than accessing the server which hold the data.

A distributed system which can hold the availability aspect but needs a lot of time to serve the client is only a bit better than the single server system. Performance is also an important aspect for a distributed system. Client must always be able to access their data, and must not wait for a long time to access that data.

One way to test a distributed system performance is using a benchmarking framework. YCSB benchmark is a framework to test important aspects of a distributed system.

II. Theory

A. Apache Cassandra

Apache Cassandra is a free and open source distributed NoSQL database management system. Cassandra is designed to handle very large amount of data across many servers and providing high availability with no single point of failure.

Cassandra offers very good performance. According to researchers of University of Toronto who studied NoSQL system, Cassandra achieves the highest throughput for the maximum number of nodes in all experiments. But, such high throughput comes with the price of high write and read latencies.

Cassandra was developed by Avinash Lakshman and Prashant Malik. It was initially developed to power the Facebook inbox search. Later, Facebook released Cassandra as an open source project, then it became Apache Incubator Project, and graduated to a top-level project. After that, many new features were released, such as integrated caching, Apache Hadoop MapReduce, Cassandra Query Language (CQL), triggers, and lightweight transactions based on Paxos consensus protocol.

Main features of Cassandra consist of:

1. Decentralized: every node in the cluster has the same role, data is distributed across the cluster, so there is no single point of failure.
2. Replication: replication strategies are configurable. Cassandra is designed for multiple-data center deployment.
3. Scalability: read and write throughput increase linearly as new machines are added.
4. Fault-tolerant: data is replicated to multiple nodes for fault-tolerance. Failed nodes can be replaced without downtime.

5. Tunable consistency: writes and reads offer tunable level of consistency.

6. MapReduce support: Cassandra has Hadoop integration with MapReduce support.

7. Query language: Cassandra has CQL (Cassandra Query Language) as an interface for accessing Cassandra.

B. YCSB

YCSB stands for Yahoo! Cloud Serving Benchmark, a framework for evaluating the performance of computer programs. YCSB is often used to compare the relative performance of NoSQL database systems.

YCSB was developed by the research division of Yahoo! and was released in 2010 with the goal of “facilitating performance comparisons of the new generation of cloud data serving systems”.

YCSB proposed two benchmark tiers for evaluating systems, performance and scalability, two aspects that are very important on a distributed database system.

1. Performance tier of the benchmark focuses on the latency of requests when the database is under load. It aims to characterize the tradeoff between throughput and latency for each database system. It is done by measuring latency as the throughput is increased until the point at which the database system is saturated and throughput stops increasing.

2. Scaling tier of the benchmark examines the impact of adding new machines to the system. Two metrics that are measured in this tier are:

    1. Scaleup: tested by loading a number of servers with data and run the workload, then delete the data, add more servers, load larger amount of data, and run the workload again. A system is said to have good scaleup properties if the performance remains constant as the number of servers, amount of data, and offered throughput scale proportionally.

    2. Elastic speedup: tested by loading a number of servers with data and run the workload, then adding more servers as the workload is running. A system is said to have good elasticity if the performance is improved when new servers are added.

YCSB offers a variety of workload. Some of the core workloads are heavy on read operation, some are heavy on update operation, and some are heavy on insert operation. The workloads are designed to test the two tiers on every aspect that are important for a distributed database system.

III. Case Studies

In order to know better if Cassandra would perform better in single-node cluster or multi-node cluster, YCSB benchmark will be performed on single-node cluster of Cassandra and three-node cluster of Cassandra.

Every analysis will be using nodes with the same specification and the same YCSB workload. The workload configuration is as described below.

```
workload=com.yahoo.ycsb.workloads.CoreWorkload

recordcount=1000000

operationcount=1000

insertstart=0

fieldcount=1

fieldlength=1000

readallfields=true

writeallfields=false

fieldlengthdistribution=constant

readproportion=1.0

updateproportion=0

insertproportion=0

readmodifywriteproportion=0

scanproportion=0

maxscanlength=1000

scanlengthdistribution=uniform

insertorder=hashed

requestdistribution=uniform

hotspotdatafraction=1.0

hotspotopnfraction=1.0

table=usertable_1000000

measurementtype=raw

histogram.buckets=1000

timeseries.granularity=1000
```

The workload is heavy on read operation, and the aspect which is compared is read latencies. Analysis on single-node cluster and three-node cluster will be presented below.

A. Single-node cluster Cassandra

The first step in testing a single-node cluster Cassandra is installing Cassandra on the node which is going to be tested. Then, install YCSB, and run YCSB with the configuration given above.

The result of running YCSB on single-node cluster Cassandra is written below.

```
...
   [READ], Total Operations, 1000

   [READ], Average, 2728.043

   [READ], Min, 968

   [READ], Max, 27615


...
```

B. *Three-node cluster Cassandra*

The first step in testing a three-node cluster Cassandra is installing Cassandra on every node that are going to be made as a cluster, and then change the Cassandra configuration to have 2 replicas of the data. Then, install YCSB, and run YCSB on one node with the configuration given above. Any node is fine for running YCSB because Cassandra treats all the node as the same.

The result of running YCSB on three-node cluster Cassandra is written below.

```
...
   [READ], Total Operations, 1000

   [READ], Average, 4716.426

   [READ], Min, 1593

   [READ], Max, 96262

...
```

IV. Conclusion

Single-node cluster Cassandra has better performance on average, minimum, and maximum read latencies. However, on single-node cluster, once the node is down, the data cannot be accessed and client cannot be served. On three-node cluster with 2 replications, the data can always be accessed unless the whole system is down. Therefore, the difference of the read latencies between single-node cluster and three-node cluster which is not very big can be considered a good tradeoff for more availability.

Acknowledgment

Firstly, I would like to praise and thank God for His blessings throughout writing this paper. I would like to thank my parents for guiding me up until this point. I also would like to express my sincere gratitude to Dr. Ir. Rinaldi Munir, MT, Dr. Eng. Ayu Purwarianti, ST.,MT, and Dessi Puji Lestari ST,M.Eng.,Ph.D. for the guidance throughout the semester in IF3280 course. Last but not least, I would like to thanks everyone that has supported me until I finished writing this paper.

References

[1] A. Lakshman and P. Malik, “Cassandra - A Decentralized Structured Storage System,” in LADIS, October 2009.

[2] B. F. Cooper, A. Silberstein, E. Tam, R. Ramakrishnan, R. Sears, “Benchmarking Cloud Serving Systems with YCSB,” in Proceedings of the 1st ACM symposium on Cloud computing, 2010.

Statement

I hereby declare that this paper is my own work and not a copy, translation, nor plagiarism of somebody else’s work.

Bandung, May 5th, 2017

Michael