Database management in a 5G network: Apache Geode benchmark and monitoring

Ante Turalija, Ivan Baković, Toni Mastelić
Ericsson Nikola Tesla d.d.
Split, Croatia
ante.turalija@ericsson.com, ivan.bakovic@ericsson.com, toni.mastelic@ericsson.com

Abstract− Apache Geode is a real-time database that provides consistent access to data-intensive applications across widespread cloud architectures which can be used in 5G. This paper brings a short overview for Apache Geode benchmark, which is enabled by the YCSB platform, and database monitoring. YCSB is an open-source platform for benchmarking and it is not configured for database monitoring which can be useful for software developers. Database monitoring is realized using database client. Logs for Geode client, located on the YCSB client side, can be configured in several levels such as error, info, debug and others. Each level provides a limited set of information that helps better monitor background processes.

Keywords: Kubernetes, Apache Geode, microservices, YCSB

I. INTRODUCTION

The core 5G network sets standards that lead to an increase in the stability and scalability of the network itself, and this is primarily achieved by means of a microservice architecture. Microservice architecture is characterized by the fact that it is focused on building single-function modules with well-defined interfaces and operations, which contributes to the agility of the architecture itself. Microservice architecture with other technologies such as Docker and Kubernetes, contributes to the full functionality of the network [1].

By using benchmarks, such as YCSB, it is possible to obtain statistical data of the database when loaded with different types of traffic. YCSB is the most famous NoSQL benchmark package.[2]. Database like Apache Geode is used in 5G. Main features of distributed Apache Geode database are speed, low latency and high scalability. The goal of this paper is to test some aspects of the Geode database. Using YCSB, database latency was tested under different workloads. Another interesting thing are client logs. Client logs from Geode database can be very useful to developers because they may provide insights into background processes. They can be retrieved by configuring YCSB Geode client and that is the focus of this paper. Logs can be configured in several levels (error, info, debug,…) which provides limited set of information and describes how database communicates with client site.

II. TOOLS AND METHOD FOR APACHE GEODE DATABASE MANAGEMENT

The architecture of the 5G network must be highly scalable to enable the implementation and operation of new services that will use the 5G network infrastructure. In order to take advantage of all the possibilities of the 5G network, a "Cloud Native" approach is used, since the network needs to behave agilely for the realization of new services. Cloud Native as a term represents the concept of developing and running scalable applications that are built as a set of microservices that run in Docker containers and are orchestrated using the Kubernetes system. Docker is an open-source containerization platform that allows developers to package applications in containers i.e., standardized executables that combine application source code with operating system (OS) files and files needed to run that code in any environment, while Kubernetes is an open-source platform for management of containerized services and workloads that facilitates declarative configuration and automation [3] [4]. The installation of the Apache Geode database can be realized by downloading the Docker image for Geode, and using Kubernetes, or Docker, to define the locator, server and necessary services that use the necessary tags in order to harmonize Kubernetes pods with other applications.

Access to Kubernetes services is possible through DNS, which is available for all requests from the application, and using ENV variables for each active service of a single node. Locators and servers are defined through the Kubernetes pod, which represents the smallest execution unit in Kubernetes and can contain one or more applications.

By raising the pods for the locator and server, and other pods such as services, PV and PVC, the necessary infrastructure for the Geode database is realized. By accessing the pod from the locator, it is possible to run "gfsh" inside it and thus enable the Apache Geode database.

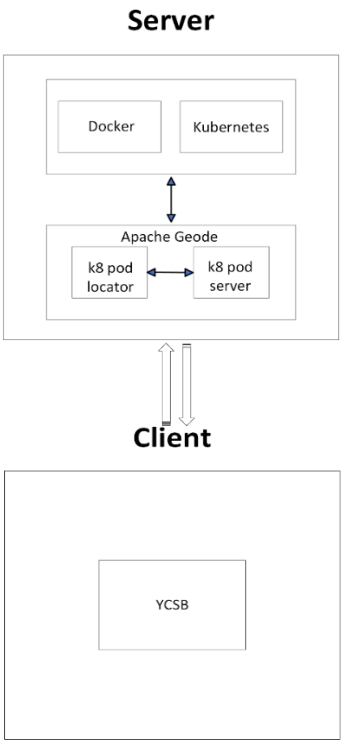

Fig. 1. Component architecture

III. SOLUTION FOR CLIENT LOGS

By running gfsh and entering information into the database, logs are created on the server side that allow us to see the steps that are performed when managing the database. On the client side, we distinguish two types of logs, benchmark logs and Geode client logs. Benchmark logs refer specifically to the logs of the application itself (YCSB) and describe the steps performed by the application in the background, while the Geode client logs describe how the client side communicates with the database located on the server side. YCSB is "open-source", which allows developers to adapt it to the needs of the project. In order to retrieve the logs on the client side, it is necessary to rearrange the code in such a way as to define the log-level and log-file for the cache factory.

```
cf.set("locators", locatorStr);
cf.set("log-level", "DEBUG");
cf.set("log-file", "client2.log");
```

It is also necessary to create a properties file for log4j2 with all the necessary parameters. Some important parameters are shown below.

```
name=PropertiesConfig
appender.file.type = File
appender.file.name = LOGFILE
appender.file.fileName=mylogger.log
logger.file.name=site.ycsb.db
logger.file.level = debug
```

By setting the log-level to DEBUG level, a more detailed insight into the background processes on the client side, that is, on the Geode client side, is enabled.

IV. BENCHMARK STATISTICS

The YCSB benchmark provides us with Geode database statistics. We can obtain statistics based on several predefined workloads (e.g. workload A, which represents a large workload with a 50/50 combination of reading and writing data).

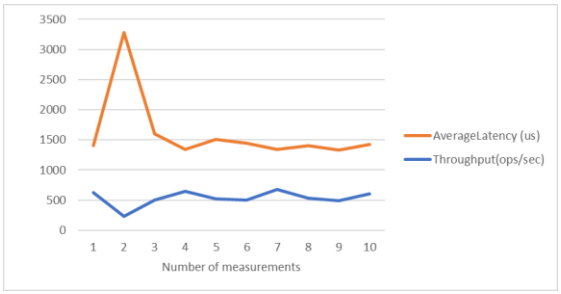

Fig. 2. Average latency and throughput for workload A

The values of latency and throughput for each individual workload is shown in the table below.

TABLE I. BENCHMARK STATISTIC FOR EACH WORKLOAD

Workload	Average throughtput	AverageLatency
A	531.9	1076.2
B	611.3	749.9
C	624.3	732.1
D	615.7	760.5
E	555.7	919.2
F	548.3	887.6

TABLE II. WORKLOAD DESCRIPTION [5]

Workload	Description
A	Mix of 50/50 reads and writes
B	Mix of 95/5 reads and writes
C	Read only
D	Read latest workload
E	Short ranges of records
F	Read-modify-write

REFERENCES

[1] „What are Microservices? Code Examples, Best Practices, tutorials and More“ ,https://stackify.com/what-are-microservices/ ,from Internet 07.05.2022.

[2] „The Ultimate YCSB Benchmark Guide (2021)“, https://benchant.com/blog/ycsb ,from Internet 15.06.2022.

[3] „What is Docker?, “What is Docker? (redhat.com),from Internet 21.05.2022.

[4] „Docker“, https://www.ibm.com/cloud/learn/docker, from Internet 16.05.2022.

[5] „CoreWorkloads“,https://github.com/brianfrankcooper/YCSB/wiki/Core-Workloads, from Internet 15.06.2022.