대규모 트래픽 처리가 가능한 원격 화상회의 시스템 설계

*김하늘, *박정훈, *이예지, *진소린, *최용석
*한성대학교 컴퓨터공학부
e-mail: jnsorn@hansung.ac.kr

Designing a remote video conferencing system that can handle large traffic

*Ha-Neul Kim, *Jung-Hoon Park, *Ye-Ji Lee, *So-Rin Jin, *Yong-Seok Choi
*Dept of Computer Engineering, Han-Sung University

요 약

대규모 트래픽을 처리할 수 있는 원격 화상회의 시스템 아키텍처 설계로서 다량의 메시지 송수신을 감당 할 수 있으며 사용자에게는 원활한 응답성을 보장 할 수 있는 시스템을 설계한다.

1. 서론

화상회의 시스템이 다양한 분야에서 활발히 사용되고 있는 지금, 대규모의 사용자가 만들어 내는 트래픽을 원활히 처리하면서 사용자에게는 빠른 응답성을 보장할 수 있는 시스템을 설계하려고 한다. 텍스트・음성・화상 채팅 및 파일 공유, 화면 공유 등의 기능을 제공하는 시스템을 설계하고, 설계한 시스템을 개발해 대규모 트래픽을 분석 및 테스트하려고 한다.

2. 본론

2.1. 관련연구

2.1.1 RDBMS와 NoSQL

기존 RDBMS는 대용량의 데이터를 처리하기에 많은 부담과 문제가 있다. 특히 RDBMS는 SQL 조인이 필요하기 때문에 데이터 처리시간이 길어지며 데이터양이 증가할수록 시스템 성능이 저하된다. 참고문헌[1]에 따르면, PostgreSQL과 Redis의 성능을 동일한 조건에서 실험한 결과 실행시간과 메모리사용량 모두 데이터의 양이 증가할수록 Redis의 성능이 향상되는 것을 확인 할 수 있었다.

따라서 사용자 증가에 따른 대규모 트래픽을 처리하기 위해 데이터를 Disk에 저장하는 기존의 전통적인 RDBMS 대신 Redis를 주 데이터베이스로 채택하여 사용자에게 낮은 지연시간을 제공하도록 설계하였다.

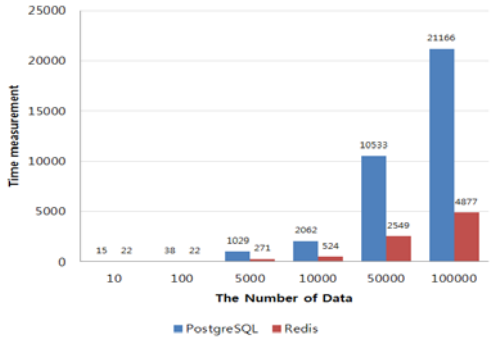

Fig.1 PostgreSQL과 Redis의 Update 실행시간

2.2. 적용기술

2.2.1 Kafka를 이용한 대규모 메시지 처리

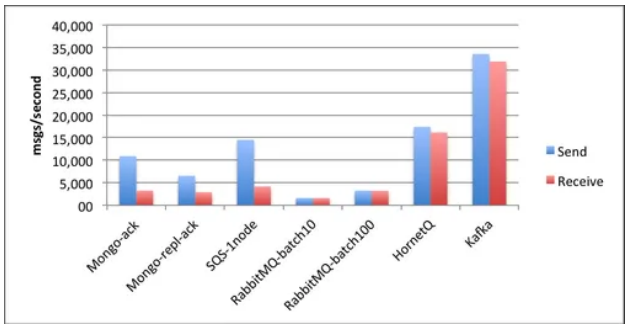

Fig.2 Kafka와 타 메세징 큐의 성능 비교

Apache Kafka는 Apache 소프트웨어 재단에서 개발한 오픈소스 분산 Message 처리 시스템으로서 Apache Kafka는 대용량 Message의 실시간 처리에 특화되어 있으

며, 분산 시스템을 기본으로 설계되었기 때문에 대용량 메시징 에 사용되는 시스템에 적합하며 확장성이 좋다.

[Fig.2]의 그림은 Apache Kafka와 타 메시지 처리 시스템에 초당 송수신 가능 메시지 성능을 측정한 표이다. Apache Kafka가 타 메시지 처리 시스템에 비해 성능이 뛰어난 모습을 보여주고 있다. 본 논문에서는 Apache Kafka 메시지 처리 시스템을 이용하여 대용량 트래픽을 감당하는 한편, 여러 대의 서버로 나눠 메시지를 처리하도록 설계하였다.

2.2.2 Key-Value 모델과 Redis 서버 복제 및 분산

Redis는 기본적으로 Key-Value Store 모델을 사용한다. 이것은 대용량의 데이터를 다루기 위해 고안되었고, 데이터를 스키마 없이 저장할 수 있도록 한다. Key의 값이 유일하기 때문에 Key만 조회하면 해당 객체에서 바로 결과를 얻어낼 수 있다. 즉, 분산 시스템에서 K-V 모델이 유리하다. NoSQL에서 K-V 모델을 사용하는 Redis를 이용하여 데이터를 설계하였다.

또한, 클러스터링을 통해 복제 서버를 구축하였으며, 이는 주 서버 장애 시에도 요청이 여러 서버로 분산되므로 향상된 읽기 성능을 지원하고 신속하게 복구가 가능하다.

2.2.3 MSA구조의 서버 아키텍처

Micro Service Architecture(MSA)를 적용함으로써 각각의 서비스에 대해 독립된 서버로 나누어 부하를 줄일 수 있으며, 독립시킴으로써 타 컴포넌트간 의존성 없이 배포할 수 있다. 또한, 서비스에 대해서 오류 발생도 독립적으로 일어나므로 오류 해결이 용이하다.

2.3. 시스템구성

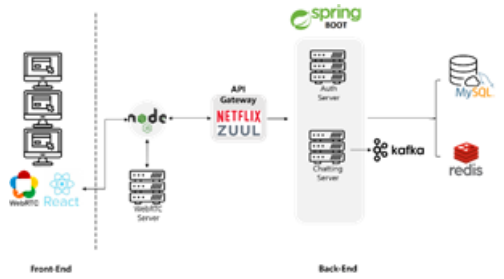

Fig.5 시스템 구성

3. 결론

3.1. 메시지 지연시간 측정 및 부하

위에서 설계한 시스템을 토대로 소프트웨어를 구현하고, 직접 개발하여 테스트를 진행하였다. 'Meetview'라고 이름 지은 Kafka 및 Redis를 이용한 대용량 원격 화상회의 시스템을 이용해서 부하를 걸고 테스트를 진행하였다. 첫 메시지 송신 시간부터 마지막 메시지 수신 시간을 측정하였다. 테스트는 I5-7200U, 8GB RAM 환경에서 진행하였으며 네트워크는 오직 테스트를 위한 통제된 환경에서 진행하였다.

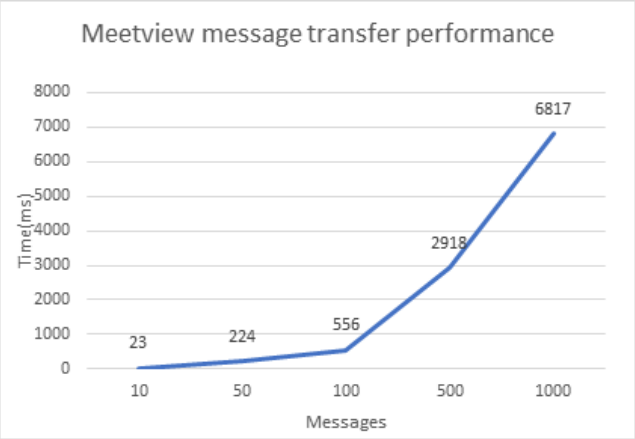

Fig.6 메시지 양에 따른 지연시간 측정

메시지 양에 따라 안정적인 시간을 보여주고 있으며 사용자에게는 높지 않은 지연시간으로 메시지를 표시하고 있다. 게다가 클러스터링을 적용해 높은 서버 안정성으로 적절한 지연시간을 보여줄 수 있다.

3.2 Redis 서버 분산을 통한 데이터 부하 방지와 Clustering을 통한 장애 안정성 확보

Redis 서버에 장애가 발생하더라도 시스템이 중단되지 않도록 Redis Clustering을 통해 분산 서버를 구축하였다. 3개의 Master node와 3개의 Slave node를 생성하여 클러스터로 묶어주었으며, 적절한 키 분산을 위해 Hash slot 방식을 사용하여 데이터를 균등하게 분배해주었다. 이와 같은 작업은 3개의 서버가 Hash slot을 나누어 가지게 되어 데이터가 한쪽에 쏠리는 현상을 방지한다.

또한, 장애 복구 시간을 최소화하기 위해 인프라 복구 후에는 전체 기능이 자동 복구되도록 failover 기능을 활용하였다. 실제로 server1의 master 서버를 down시킬 경우 server1의 slave node가 master로 승격됨을 확인 하였다.

참고문헌

[1] Web Service Performance Improvement with the Redis, Journal of the Korea Institute of Information and Communication Engineering 19(9), 2015.9, 2064-2072)

[2] Comparisons of NoSQL for Big Data Management, 한국정보과학회 학술발표논문집 , 2015.12, 1439-1441

[3] Kafka: a Distributed Messaging System for Log Processing

Acknowledgement

[본 논문은 과학기술정보통신부 정보통신창의인재양성 사업의 지원을 통해 수행한 ICT멘토링 프로젝트 결과물입니다.]