Facebook's Monitoring Systems

조영일

목차

- ODS
  - Introduction
  - Motivation
  - Architecture
  - Performance
- Scribe
  - Introduction
  - Design Goals
  - Architecture
  - Evaluation

ODS - Introduction

- ODS(Operational Data Store)
  - Facebook에서 2012년에 발표
    - 실시간 모니터링과 경향을 위한 시계열 데이터를 처리하는 시스템
  - 25억 건 데이터 / 분, 1.6만 건 읽기 / 분을 감당함

ODS - Motivation

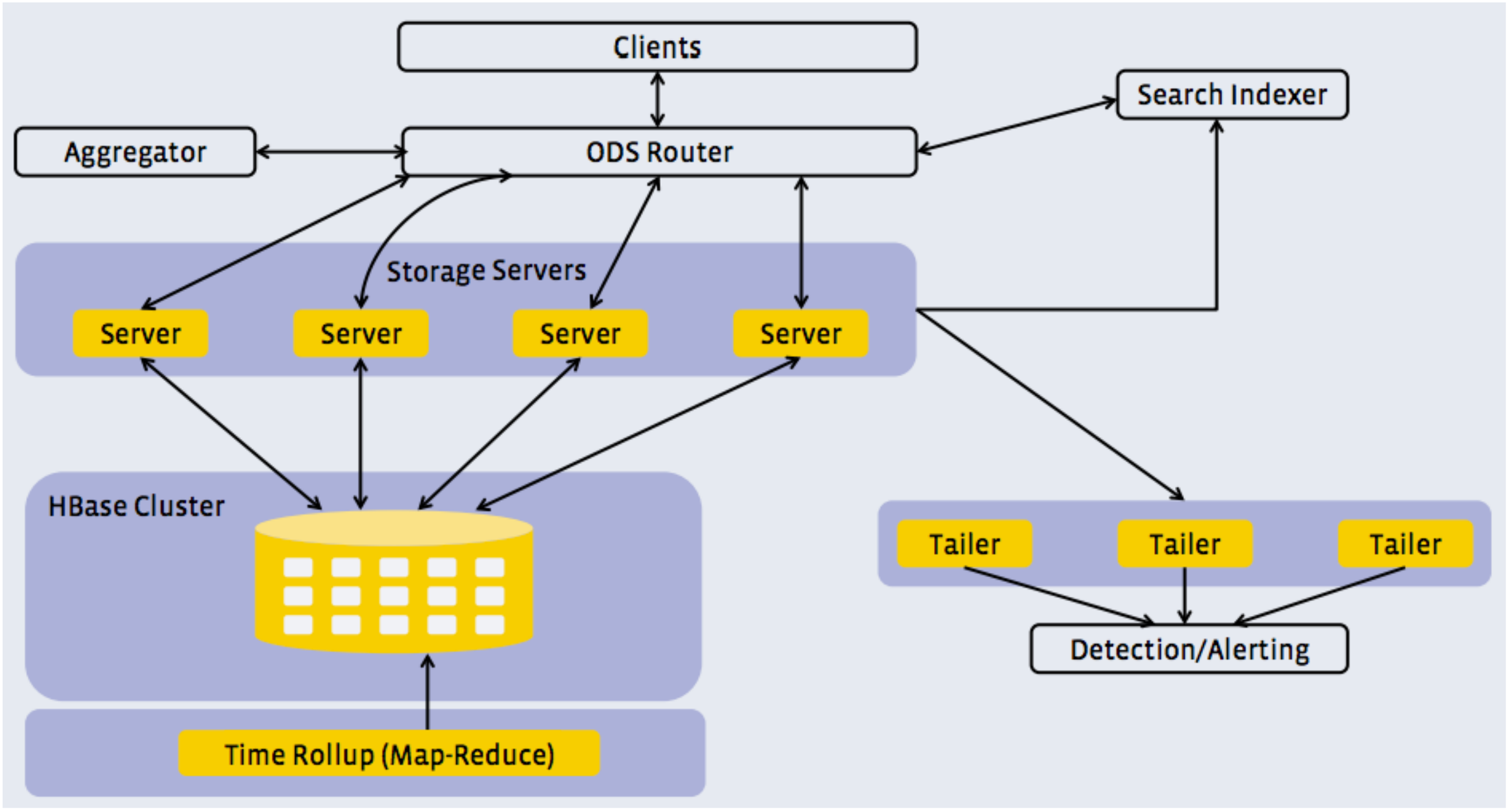
- MySQL 시스템의 문제점
  - 하드웨어 장애 시의 낮은 가용성
  - 제한된 throughput
    - 테이블 크기 제한
    - sharding하면 hotspot 생김
- 대안
  - Memcached x
  - Hadoop x
  - Cassandra, Tokyo Cabinet, Hive, … ?

ODS - Motivation

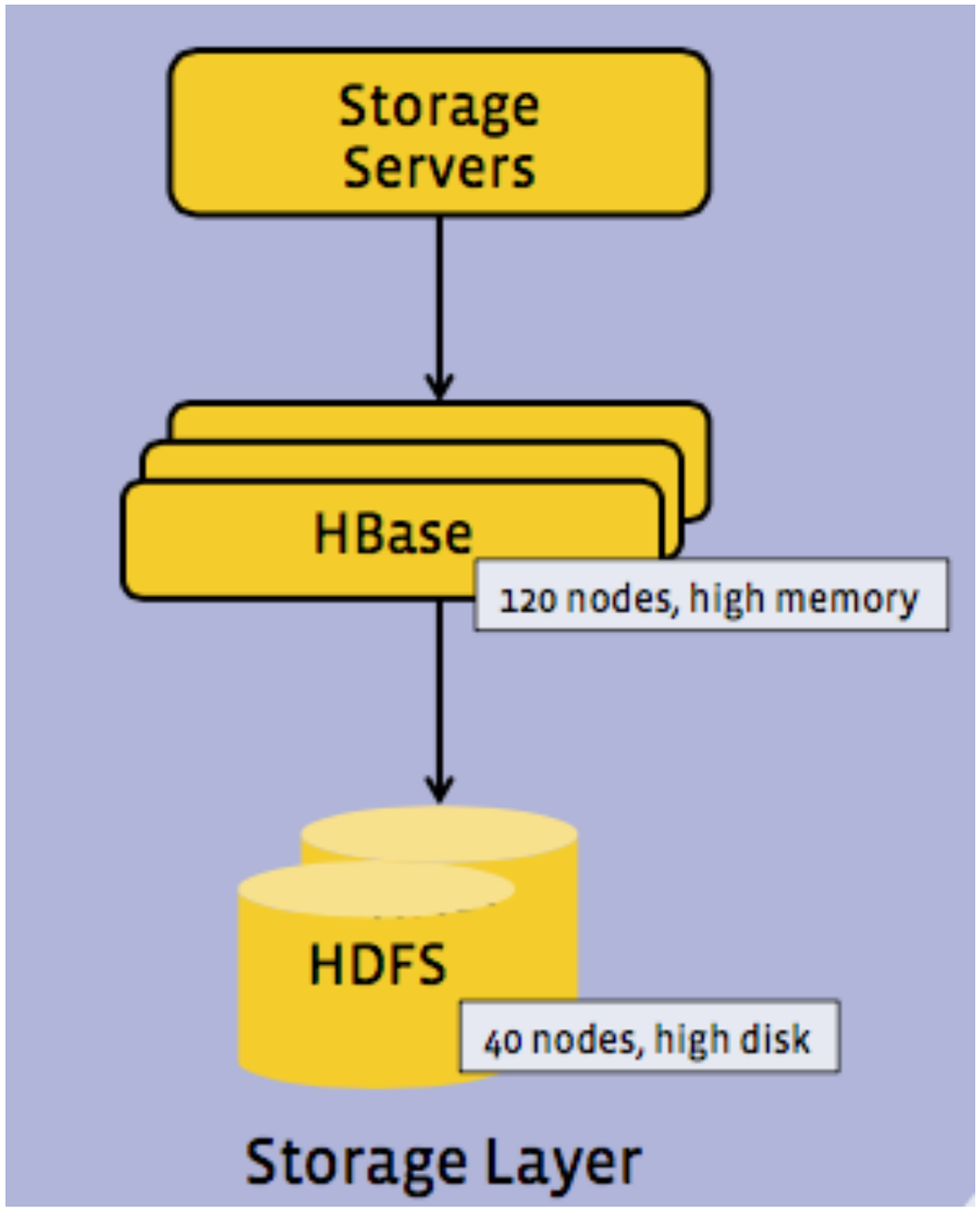
- 시스템의 특성
  - write-heavy workload
  - HA
  - strong consistency within IDC
  - fault isolation

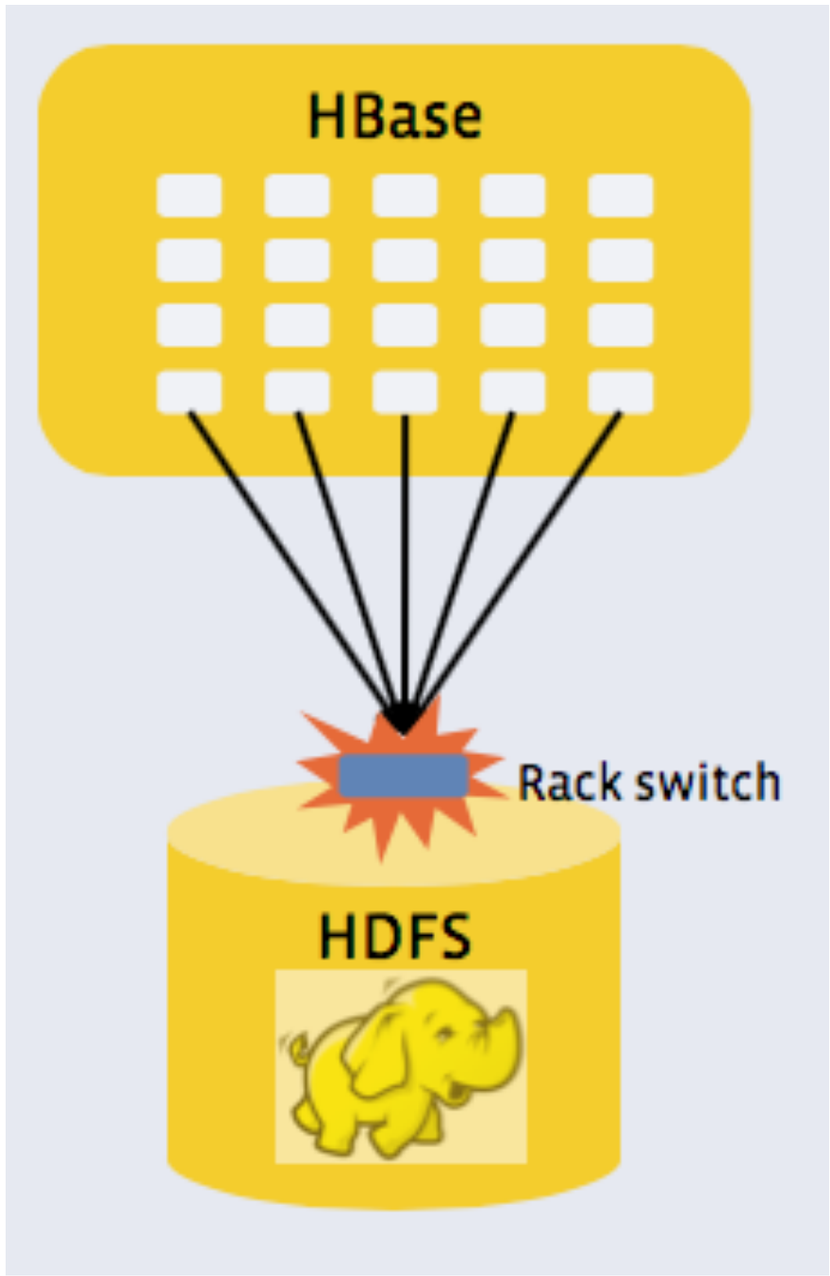
- HBase
  - append-heavy workload에 최적화
  - sorted / column oriented
  - inherent sharding, connection handling, failure recovery
  - base on Hadoop
  - 최적화되어 있지 않아도 적당한 성능을 내줌

ODS - Architecture



ODS - Architecture



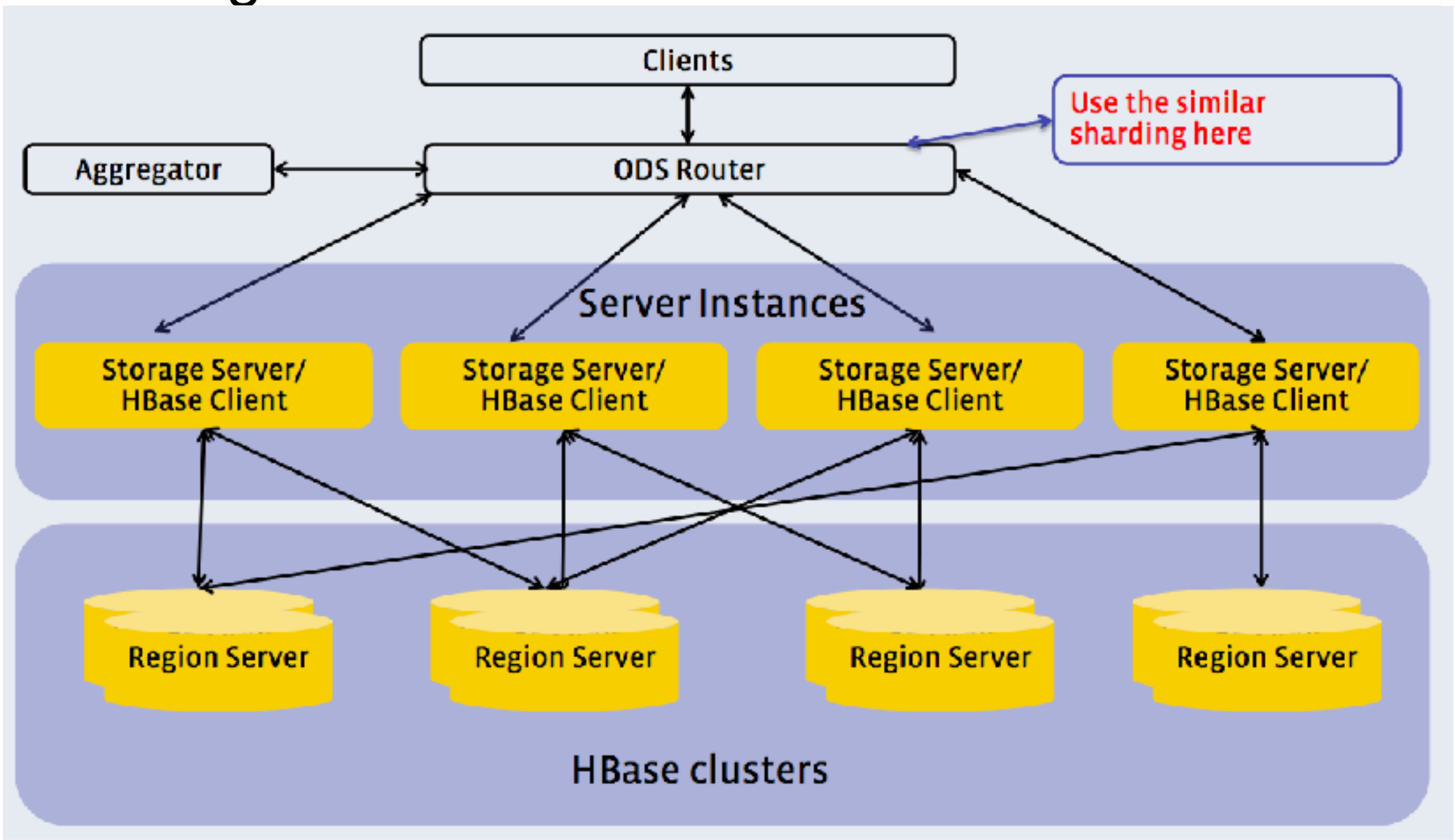


ODS - Migration

- 기존 모니터링 시스템에서 신규 시스템으로 이전하면서 겪은 경험담 공유
  - blah blah …
- 제안
  - 어차피 Hadoop이 중단될 수 있으니 장애 대응 용 클러스터를 한 벌 더 만들자!
  - “double writing”

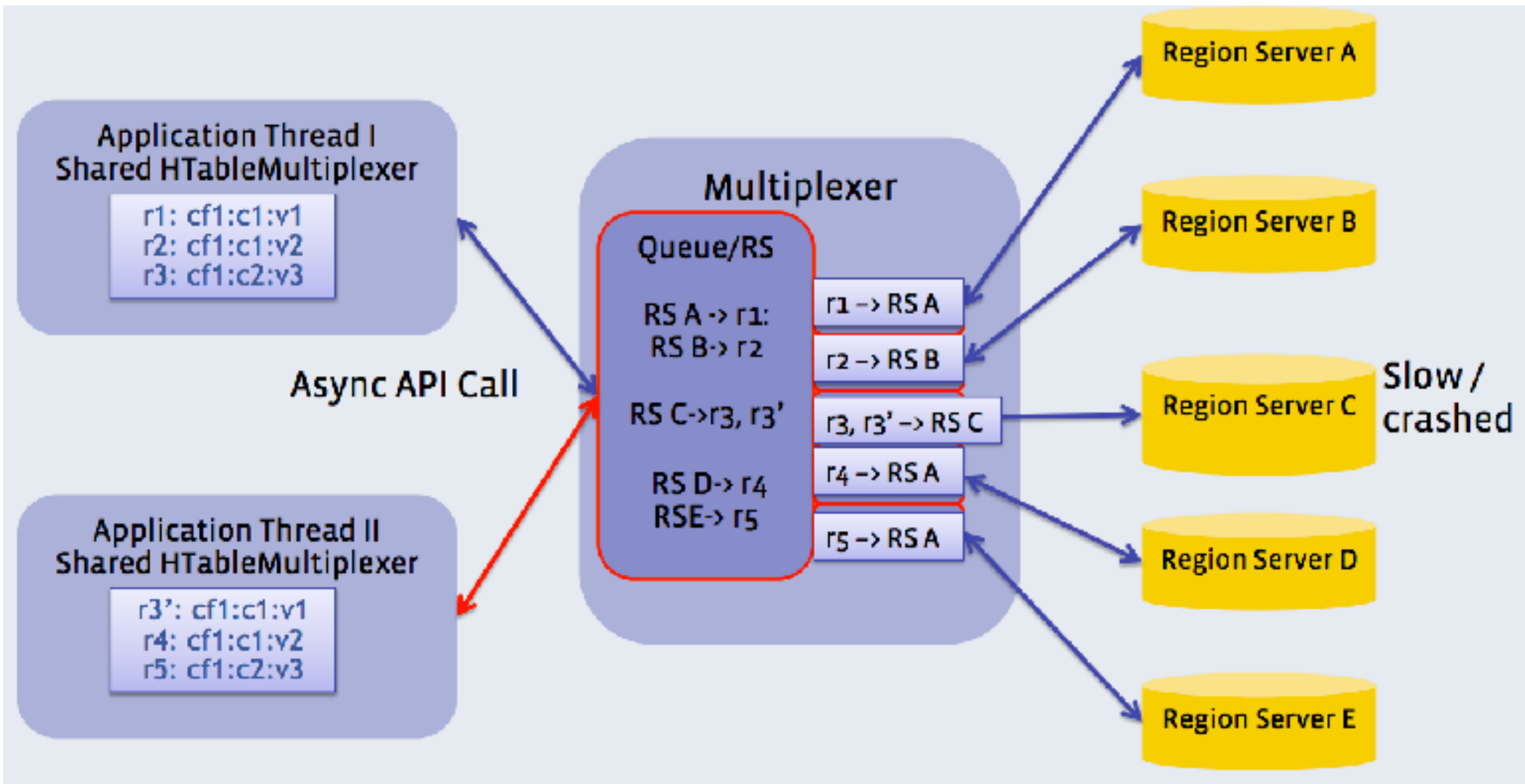
ODS - Performance

- Sharding



ODS - Performance

- Async write for MULTIPUT
  - HtableMultiplexer



Scribe - Introduction

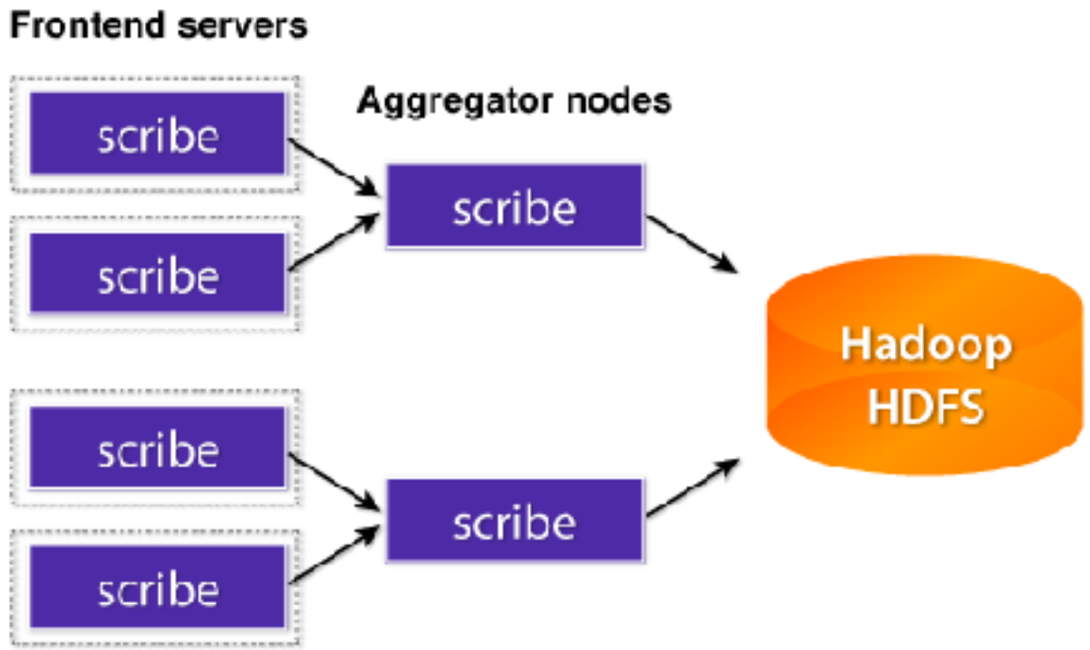
- 확장성있는 분산 로깅 프레임웍
- 실시간으로 서버 로그를 수집하여 중앙 저장소로 전달
- 넓은 범위의 데이터를 로깅하는 데 유용함
- Github에 공개되어 있음

Scribe - Design Goals

- 네트웍 토폴로지를 유연하게 (DAG)
- 신뢰성있는 시스템
- 데이터 모델을 단순하게
- using Thrift

Scribe - Architecture

- scribe server
  - local scribe server에서 central scribe server로 전송
  - local server는 app의 로그를 수집하고 central server는 HDFS에 저장하거나 또 다른 central server layer에 전달
- non blocking C++ server based on Thrift



Scribe - Evaluation

- Reliability
  - 장애에 resilient하지만 transaction을 보장하지 않음
    - central server 장애 시, local server는 디스크에 저장하고 나중에 재전송 시도
      - central server 과부하를 방지하기 위해 즉각적인 재전송을 피함
      - central server의 용량이 부족해지면 미리 local server에 알림
  - central server도 HDFS에 대해 비슷한 메커니즘으로 동작함
- 사용하는 기업
  - Facebook, Twitter, Zynga, Digg, …

Thank You!