飞天技术汇
APSARA USER GROUP

LINUXCON containercon CLOUDOPEN CHINA 中国

MongoShake -- Multi Active-Active and Cross-Region Disaster Recoverable MongoDB Service

陈星CHENXING(花名：烛昭ZHUZHAO)

MONGODB ENGINEER

MONGODB TEAM, ALIBABA CLOUD

阿里云

Agenda

- Background Knowledge
- Redundant & Replication
- Active-Active Replication
- MongoShake
- Business User Case
- Performance
- Open Source Plan
- Q&A

Background Knowledge

Background Knowledge

- MongoDB utilization in Alibaba cloud service
  - External cloud developer from aliyun.com
  - Internal business such as Taobao, Tmall, GaoDe
- Various toolkit around MongoDB
  - Automatic failover with Management and Control system
  - Audit logging for inspection
  - Monitoring and alarm system
  - Multi-region replication among ReplicaSet
  - ……

Basic Concept of MongoDB

- Mongod

  mongod is
  requests, m
  les data
  perations.

- Oplog

  As a cappe
  documents
  reproduced
  operation of
  could be

- ReplicaSet

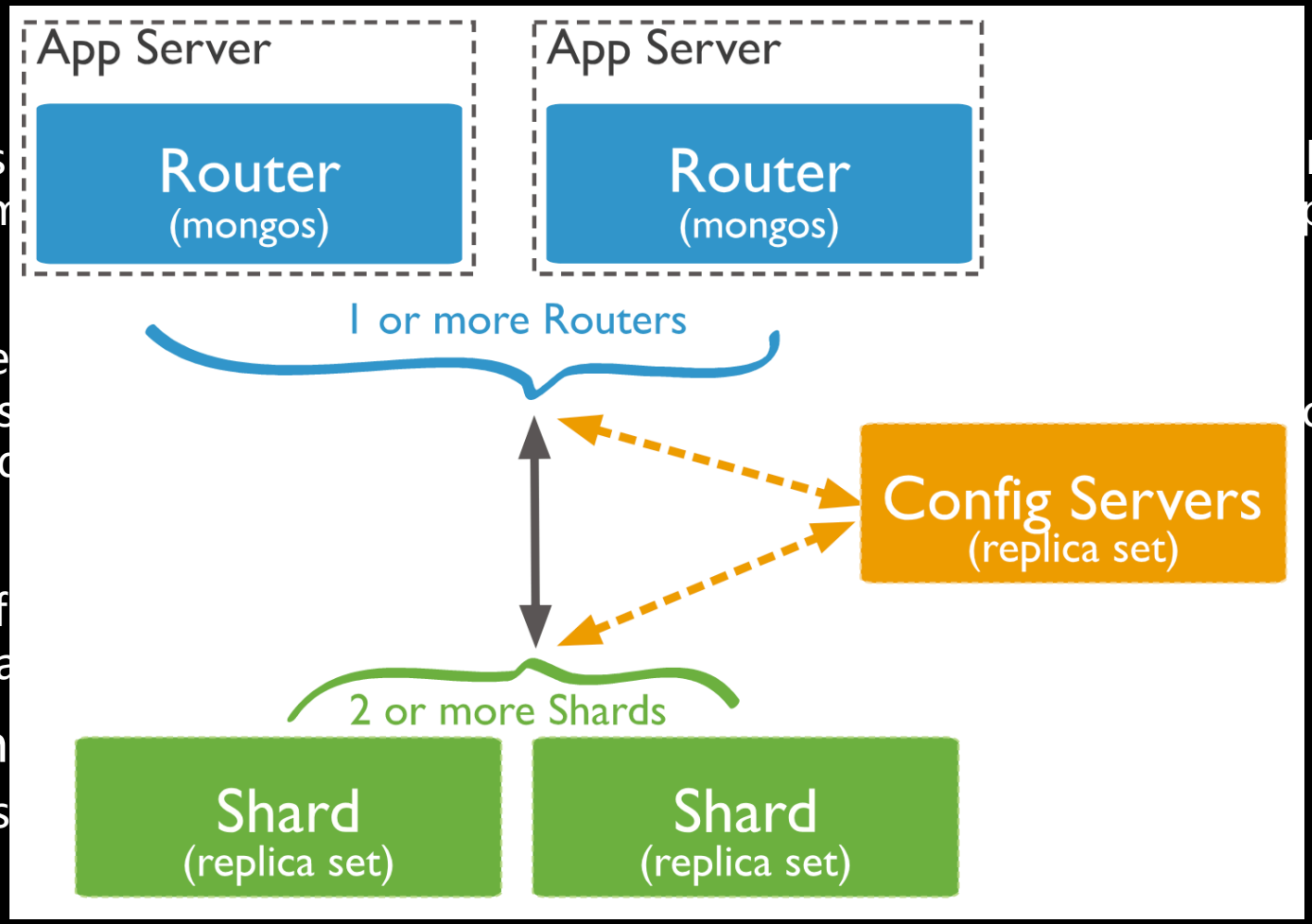
  A group of
  Its seconda
  hidden or Arbiter.

- Replication

  The proces

- Sharding

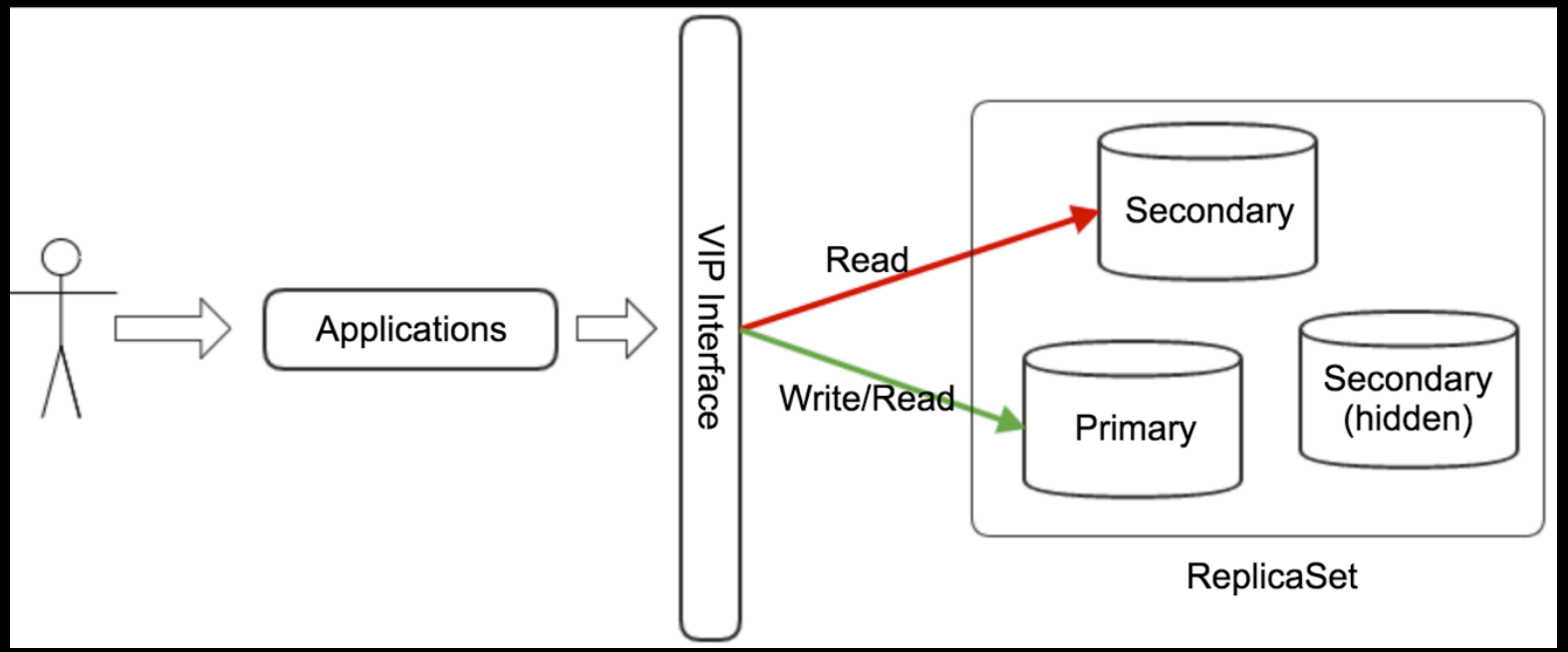
  Sharding is a method for distributing data across multiple machines. MongoDB uses sharding to support deployments with very large data sets and high throughput operations.



MongoDB as Cloud Service

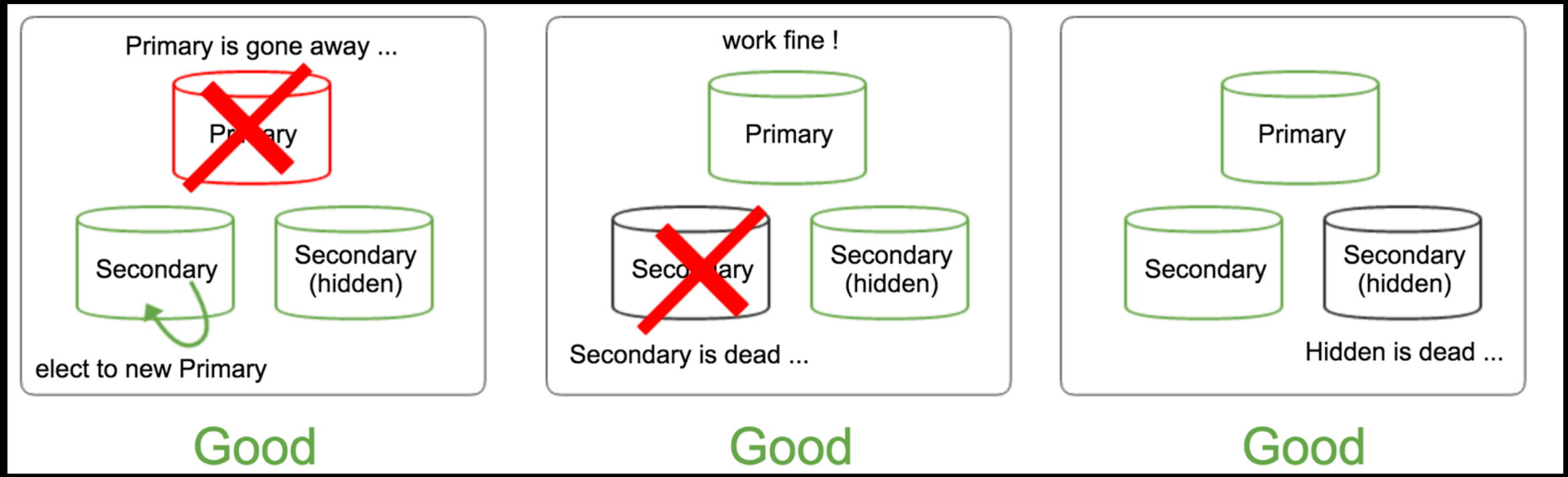
ReplicaSet typical instance deployment in service.

- All instances (include sharding) are based on ReplicaSet
- Consist of : 1 primary，1 secondary，1 secondary(hidden)



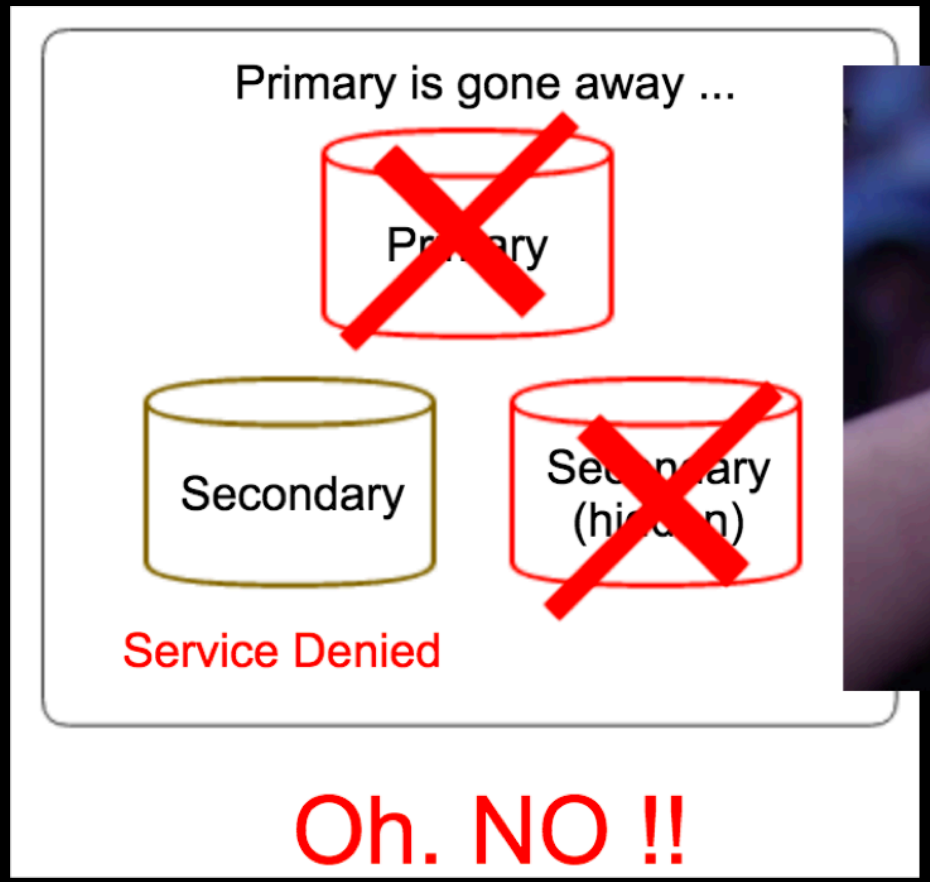
High Available

- MongoDB HA with ReplicaSet



High Available

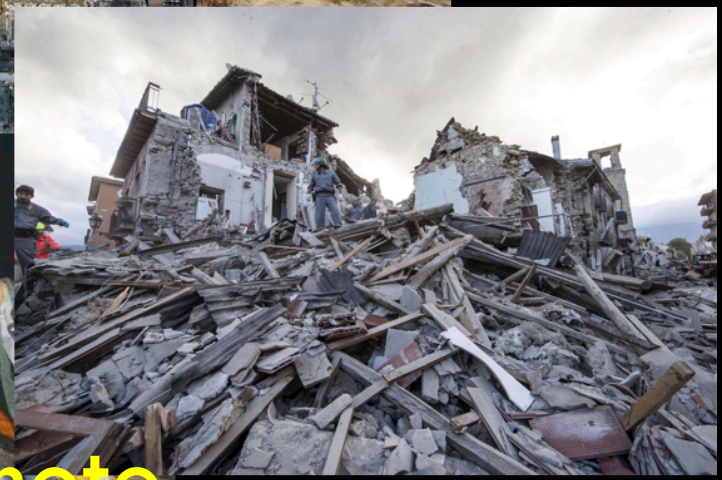
- Very small probability (~ 0.01 %).
  - More than 1/2 nodes were crash



High Available

- What we will do if …
  - Power outage
  - Data Center malfunction
  - Network device is broken
  - Earthquake or Tsunami

How can we recovery from disaster ?



It is necessary to have another remote duplication in somewhere for RECOVERY !

Redundant and Replication

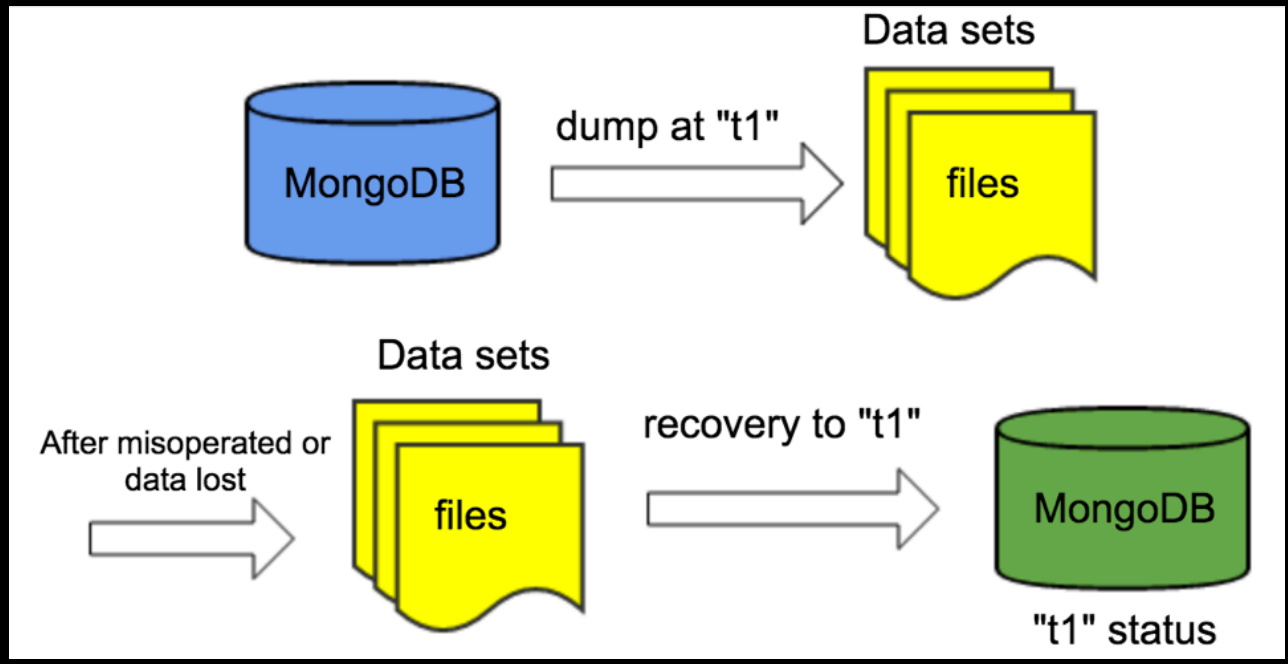
Redundant and Replication — Why we need ?

- The consequence after disaster ….
  - Service will be suspend for a few minutes but comes back online soon.
  - Data would be lost in a very short time.
  - Persistent storage of the data center were completely broken. data is lost ! (really bad)

- Take action before disaster …
  - Make more duplication. makes the data safety …
  - Two ways : backup and replication

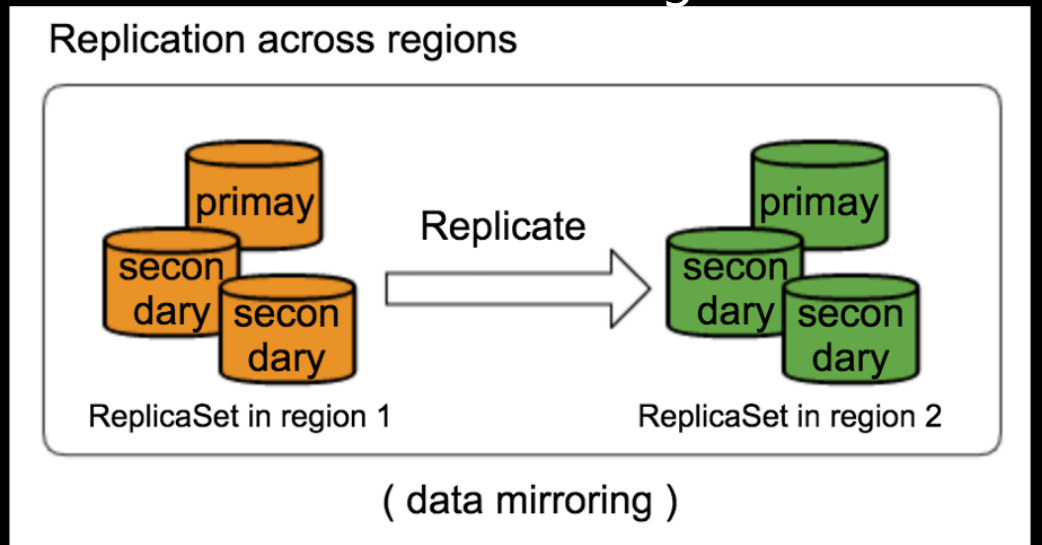
Redundant & Replication

- Backup
  - Logical backup
    - Full backup + incremental backup
    - Mongodump with -oplog and mongorestore
    - Dump every day
  - Hot physical backup
    - From wiredtiger “backup” cursor



Redundant & Replication

- Replicate in ReplicaSet
  - Default data syncing from Primary to Secondary
  - Need to be configured as one ReplicaSet. NOT cross Region. In normal all nodes will be deployed in same data center
- Replicate across Regions
  - Replay the source regions oplog into another region.
  - Both the source and destination MongoDB can be written.



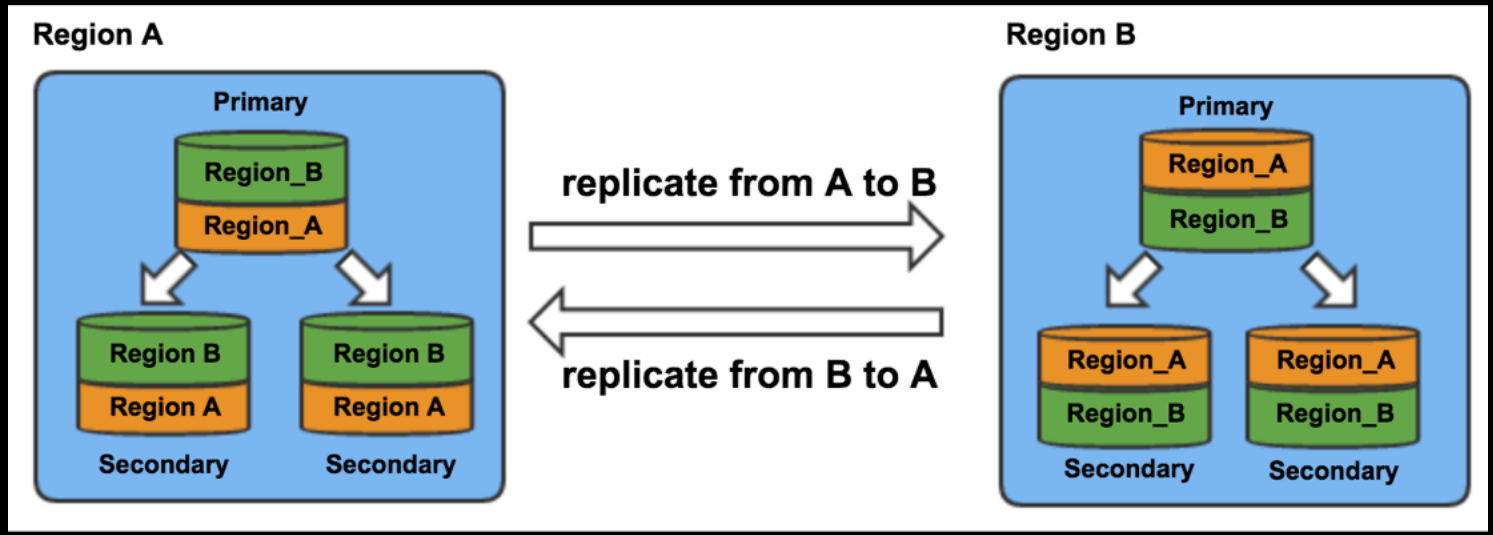
Redundant & Replication

	Backup	Master/Slave Sync	Master/Slave Async
Consistent	Bad	Wonderful	Bad
Failure Tolerance	Good	Bad	Wonderful
Scaleout	Bad	Bad	Good
Overhead	Low	Middle	Low
Data May Loss	Yes	No	Yes

Active-Active Replication

Active-Active Replication

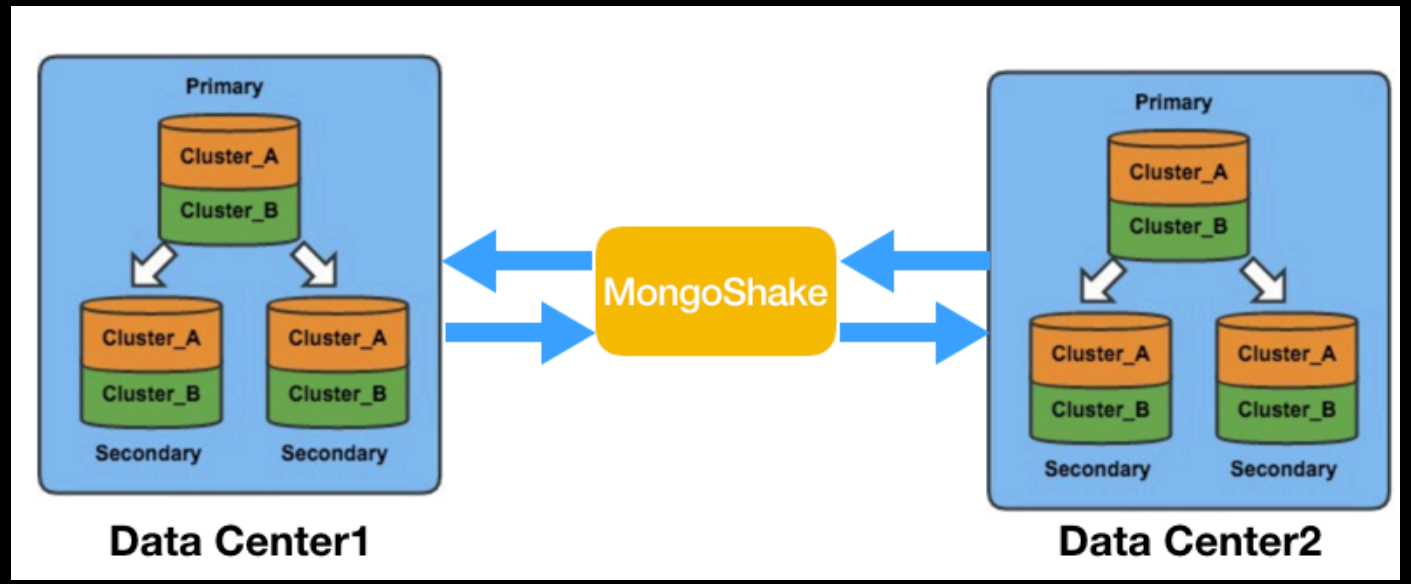
- Inspired by Master/Master and asynchronous
  - Multi regions are all writable
  - Regions replicated each other
  - Any of these regions has the overall data set via replicating



- Relationship with replication
  - 2 Regions : (relationship = 2)
    A->B, B->A
  - 3 Regions : (relationship = 6)
    A->B, A->C, B->A, B->C, C->A, C->B
  - Count(link) = N * (N - 1).
    (N is number of regions)

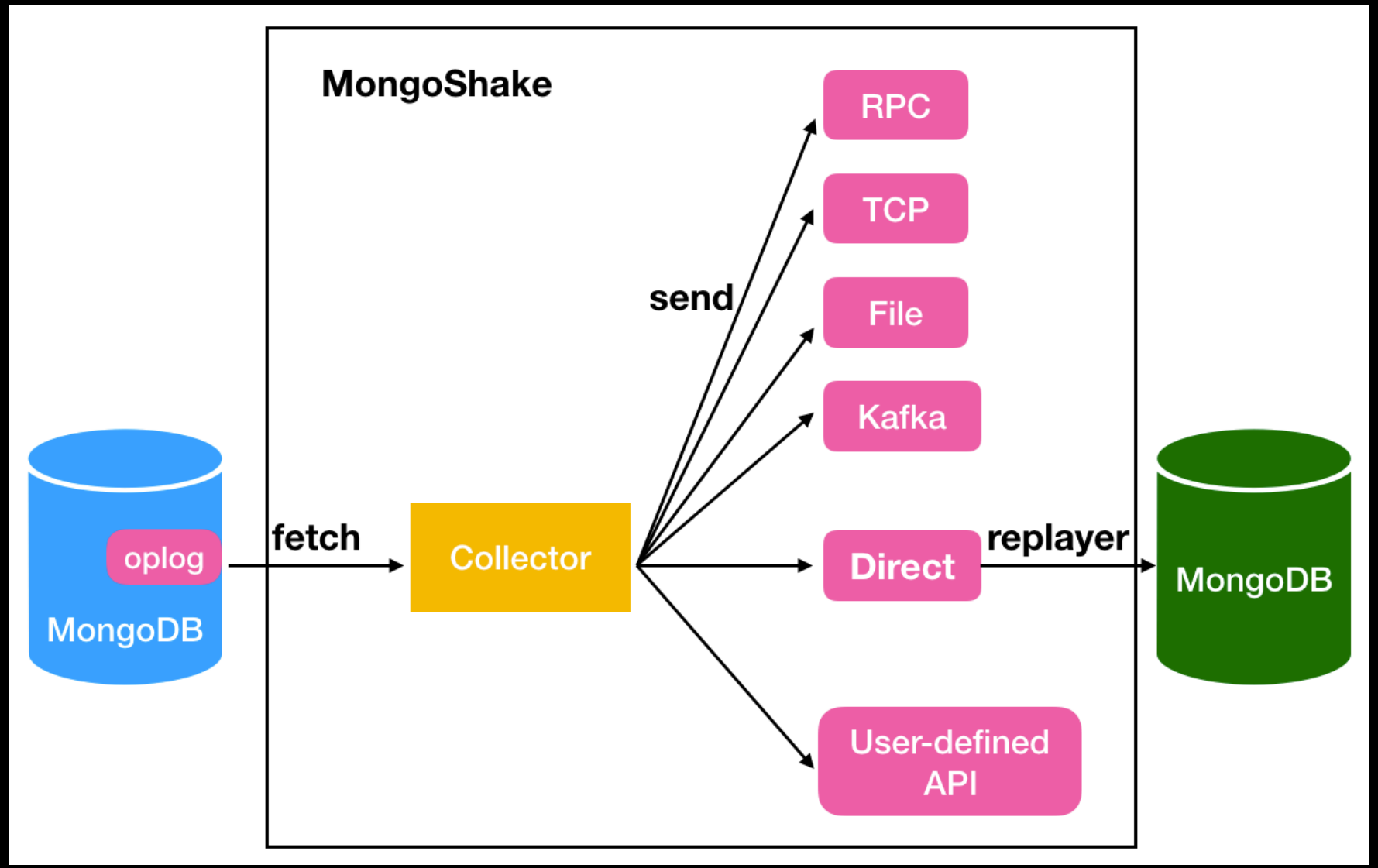
Active-Active Replication

- Pros
  - There is no central node (the most significant). no Single Of Failure ！
  - Every region is under replication can be replaced by any others to serve the user request. (because of mirroring)
- Cons
  - Double or triple data size by replication. (decided by number of regions)
  - Network traffic will be raised

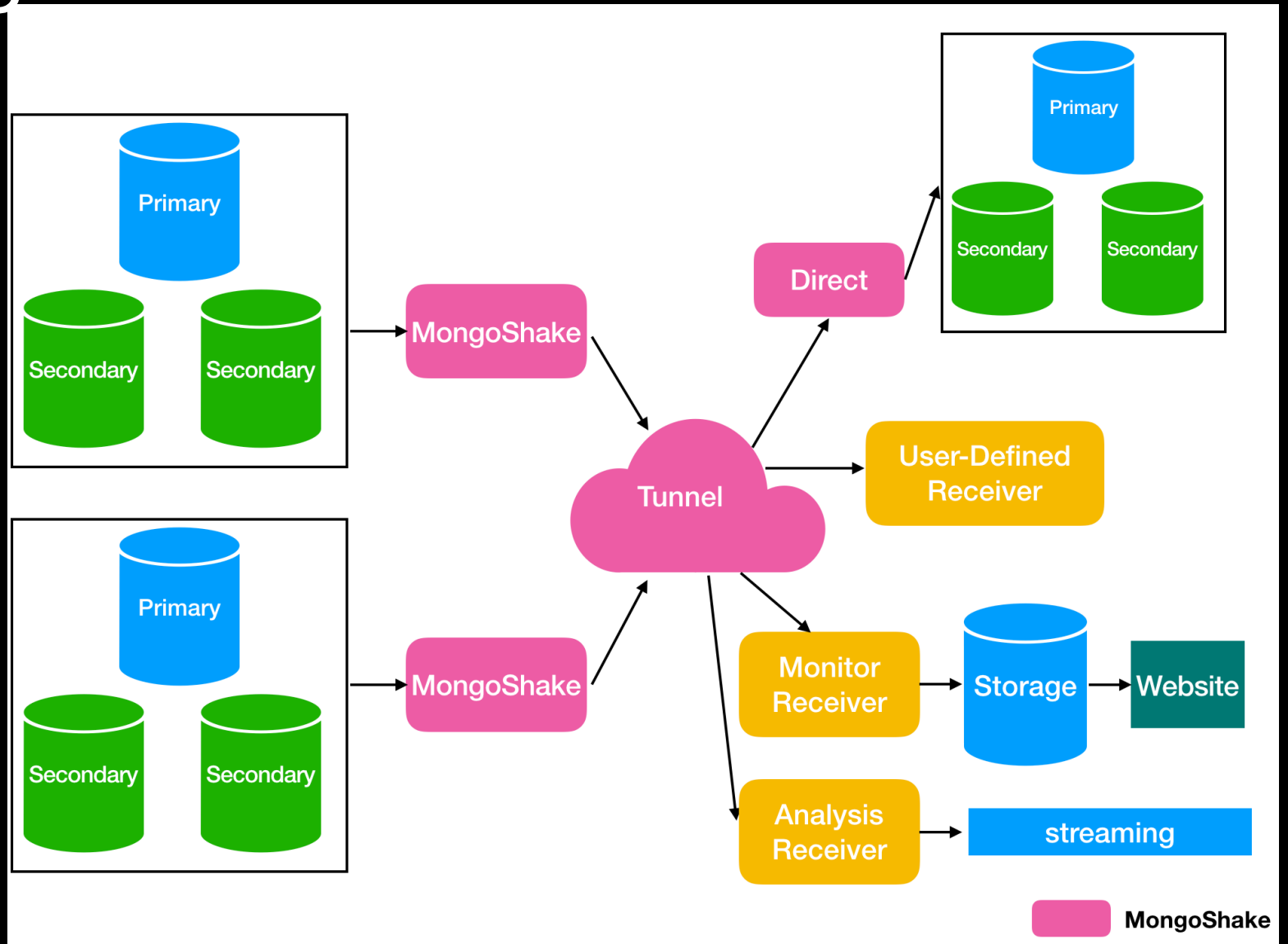


MongoShake Architecture and Design

MongoShake Architecture



MongoShake Architecture-Dataflow

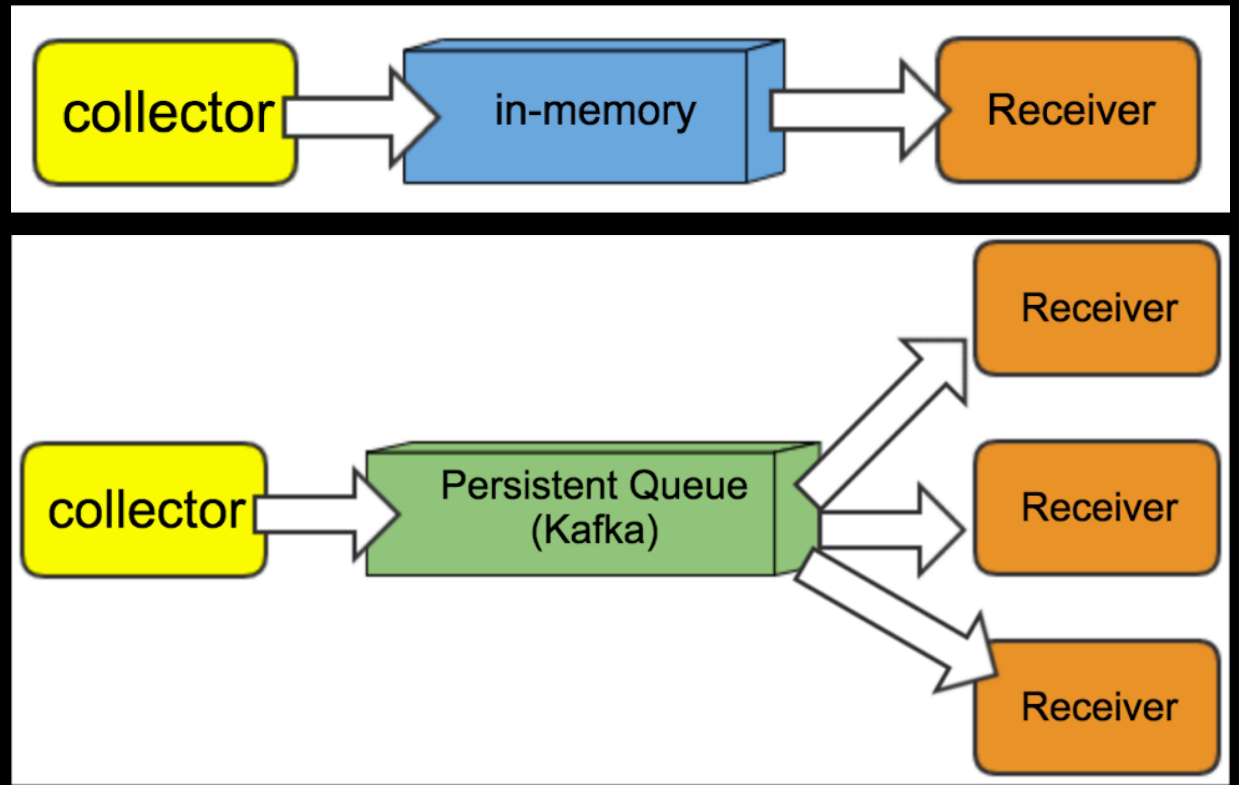


MongoShake Architecture

- Collector
  - Tail and fetch oplog from source MongoDB ReplicaSet. Both DML and DDL
  - Role “Secondary” is suggested to reduce the influence to other operations.
  - Transfer oplogs into tunnel in parallel.

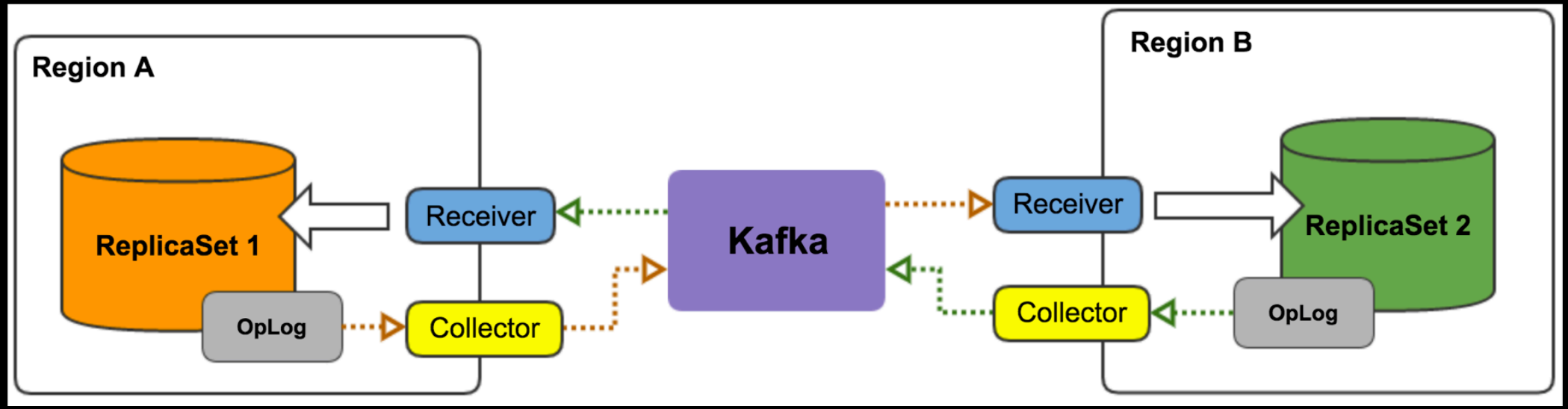
MongoShake Architecture

- Tunnel
  - An abstraction of Queue with two type : persistent storage queue or in-memory queue
  - Different : the oplogs message in persistent queue can be broadcast to group of receivers and been isolated asynchronously.



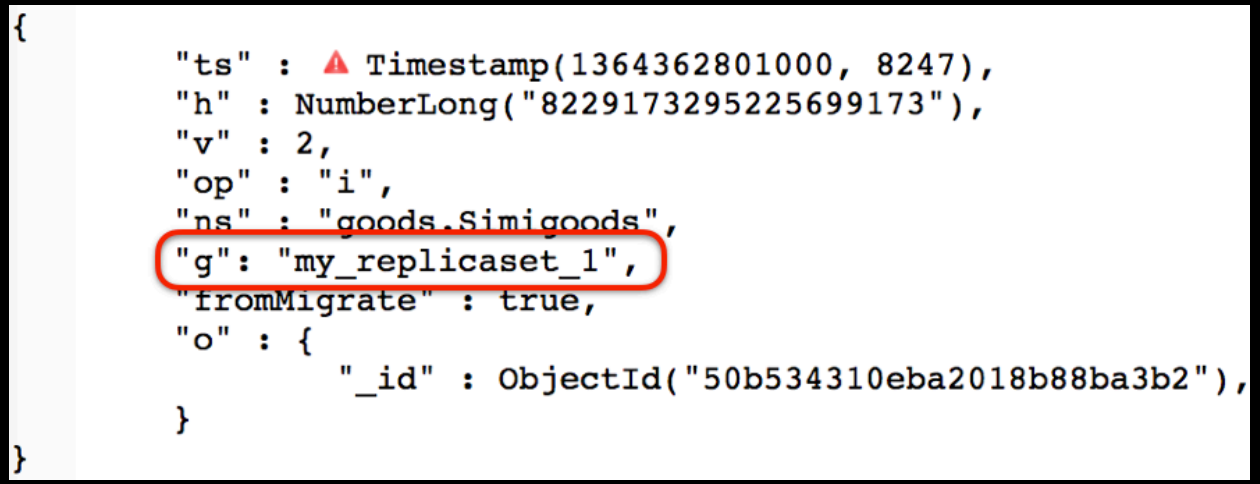
MongoShake Architecture and Design

- MongShake Active-Active deployment
  - 2 ReplicaSets in 2 regions and replicated each other
  - 2 pipelines (relationship), include two collectors and two receivers.

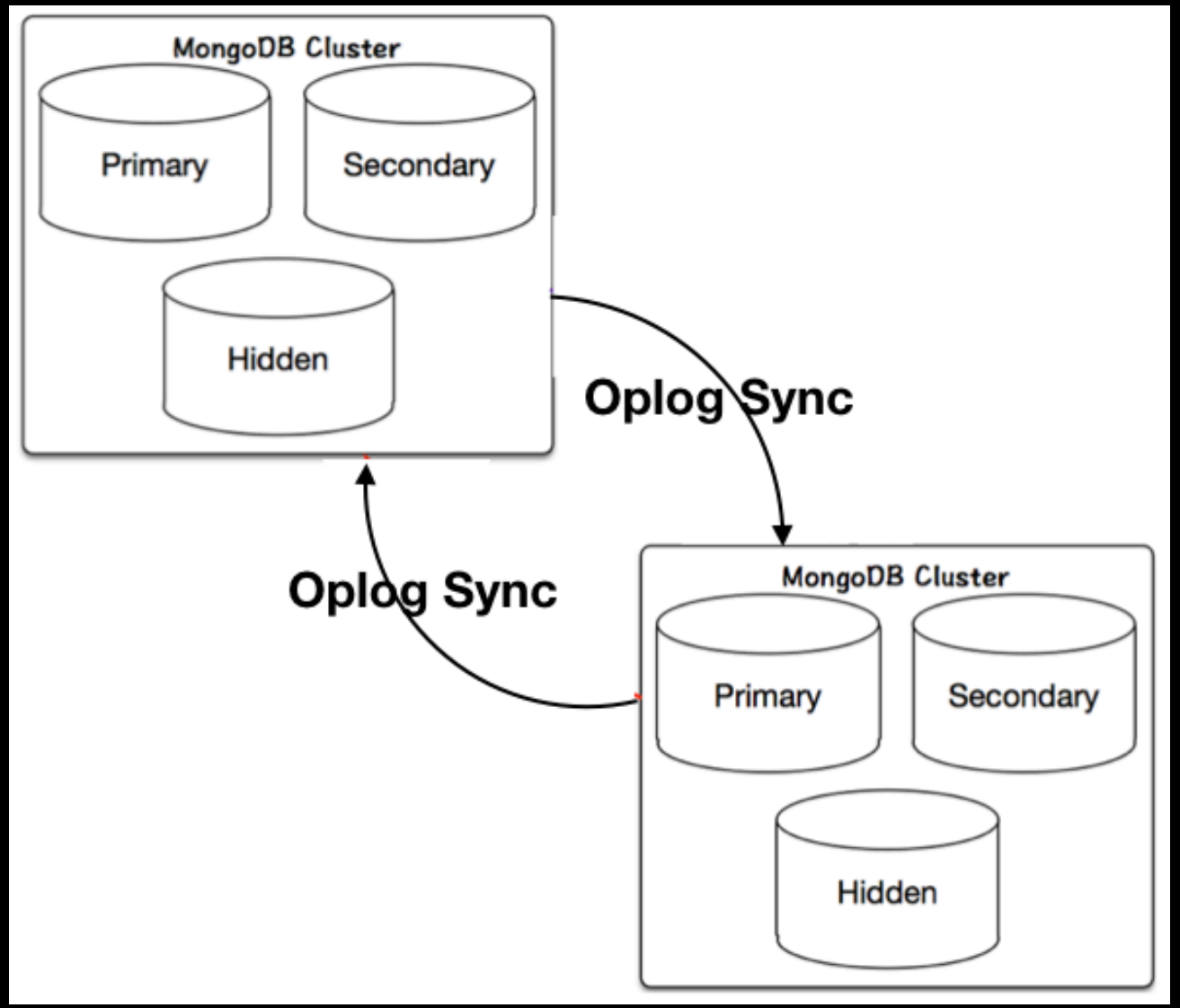


MongoShake Architecture-Loop

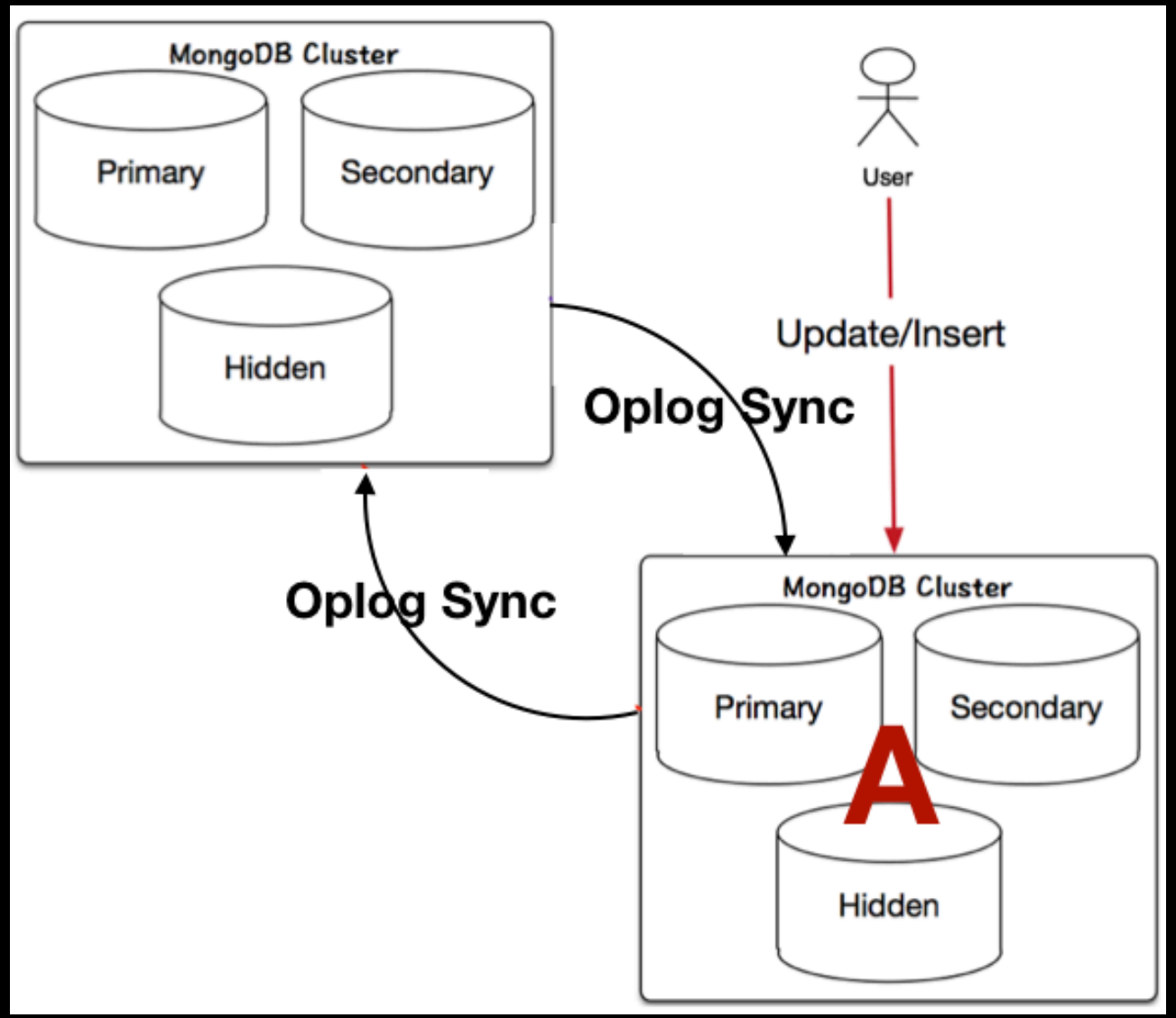
- Avoid oplog circular replication
  - Add extra flags in oplog entry => “g” (global id)
    field indicated where this entry from
  - A namespace will be configured to a specific
    collector. its value is similar to “g”
  - Collector is only fetching the oplog entries that tagged with ${g} == ${namespace}. Others entries will be filtered. so never broadcast again



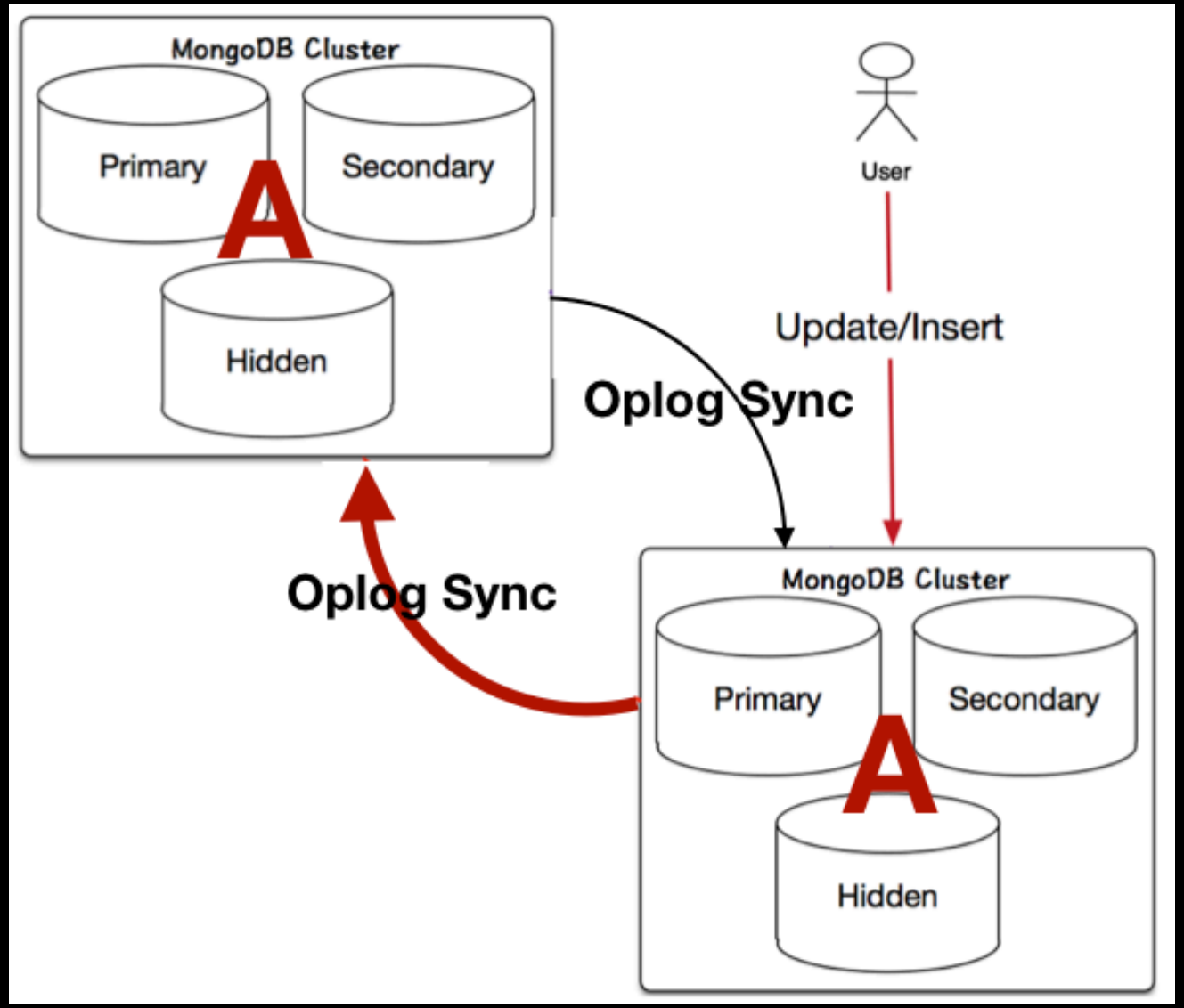
MongoShake Architecture-Loop



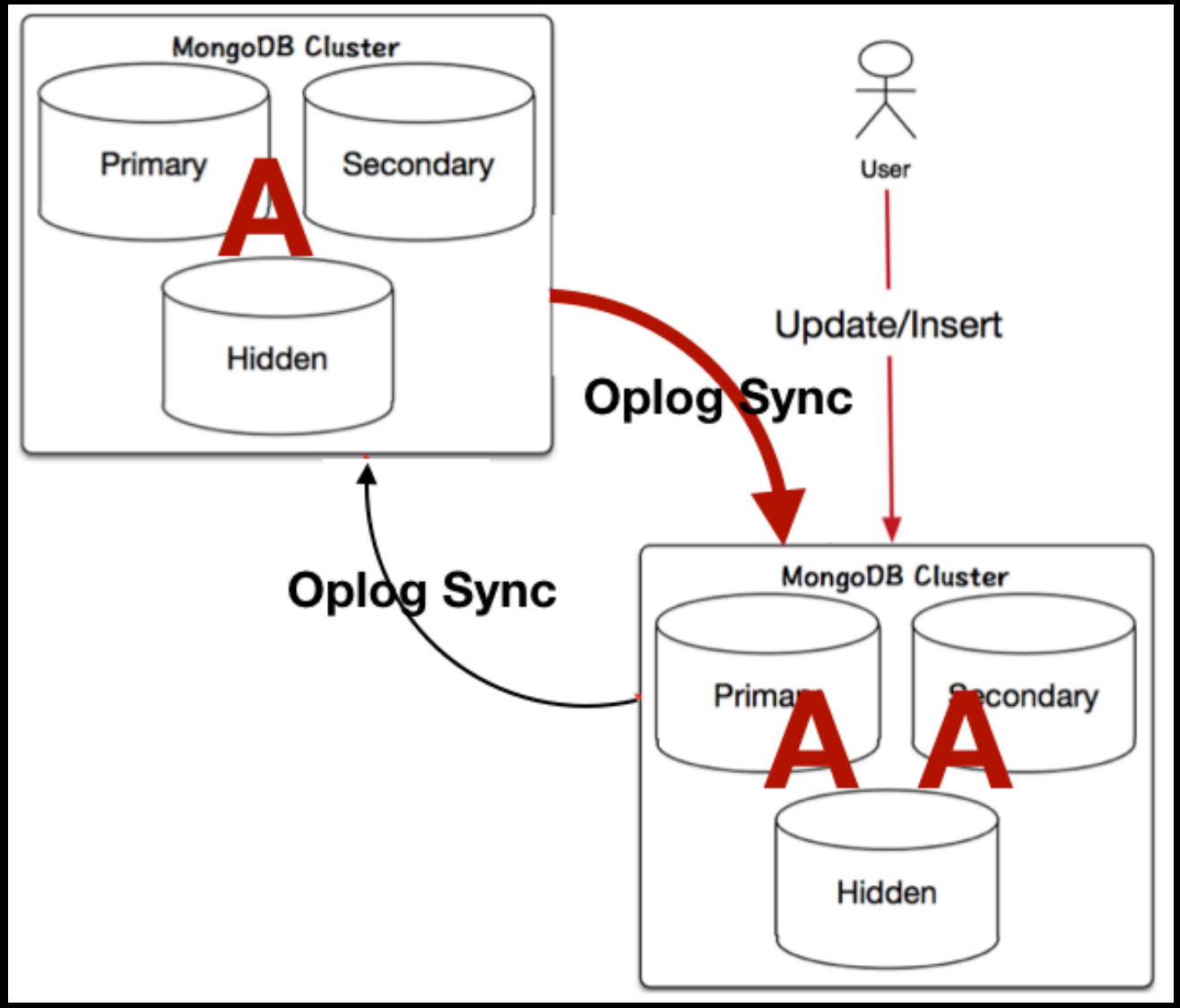
MongoShake Architecture-Loop



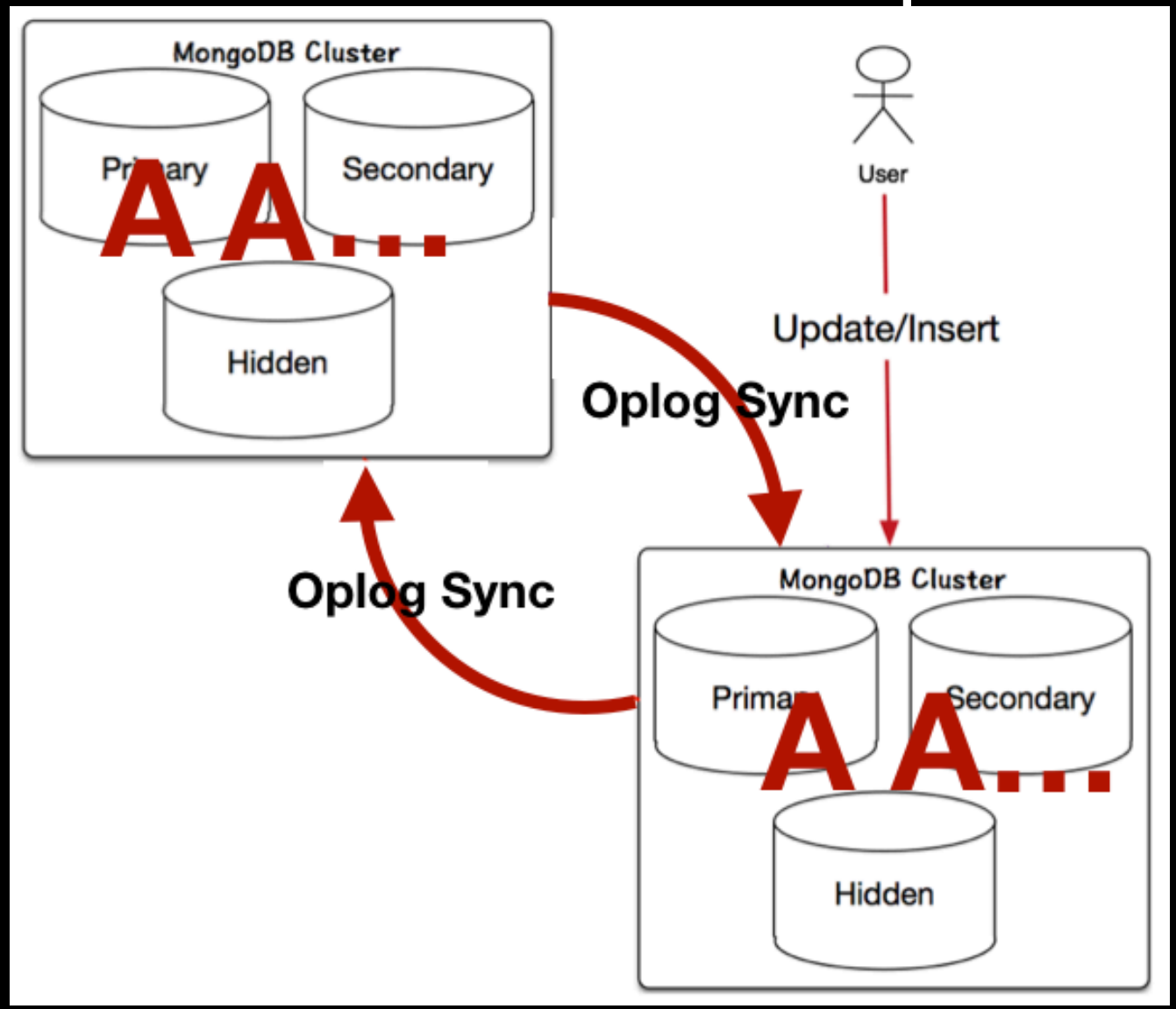
MongoShake Architecture-Loop



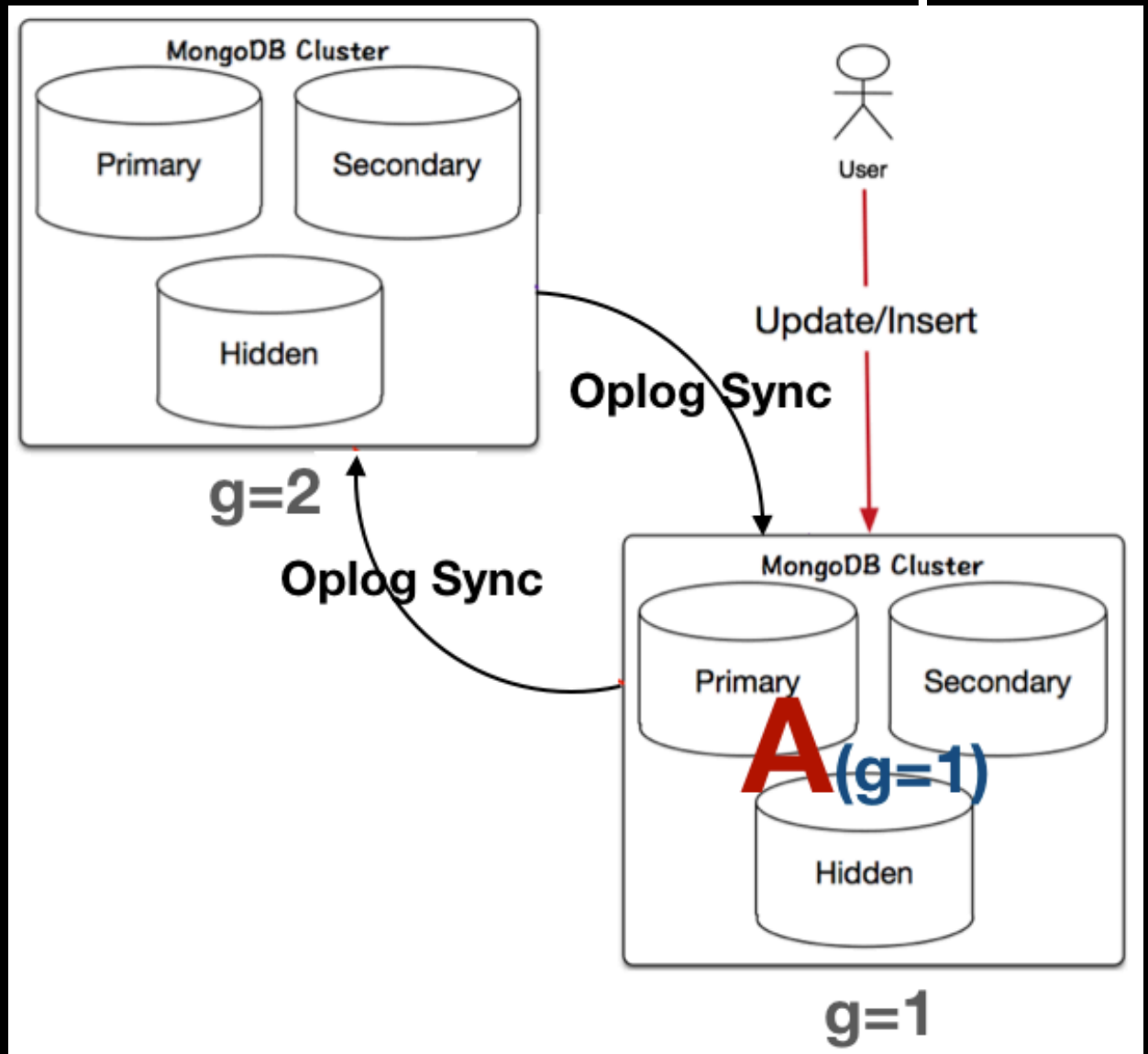
MongoShake Architecture-Loop



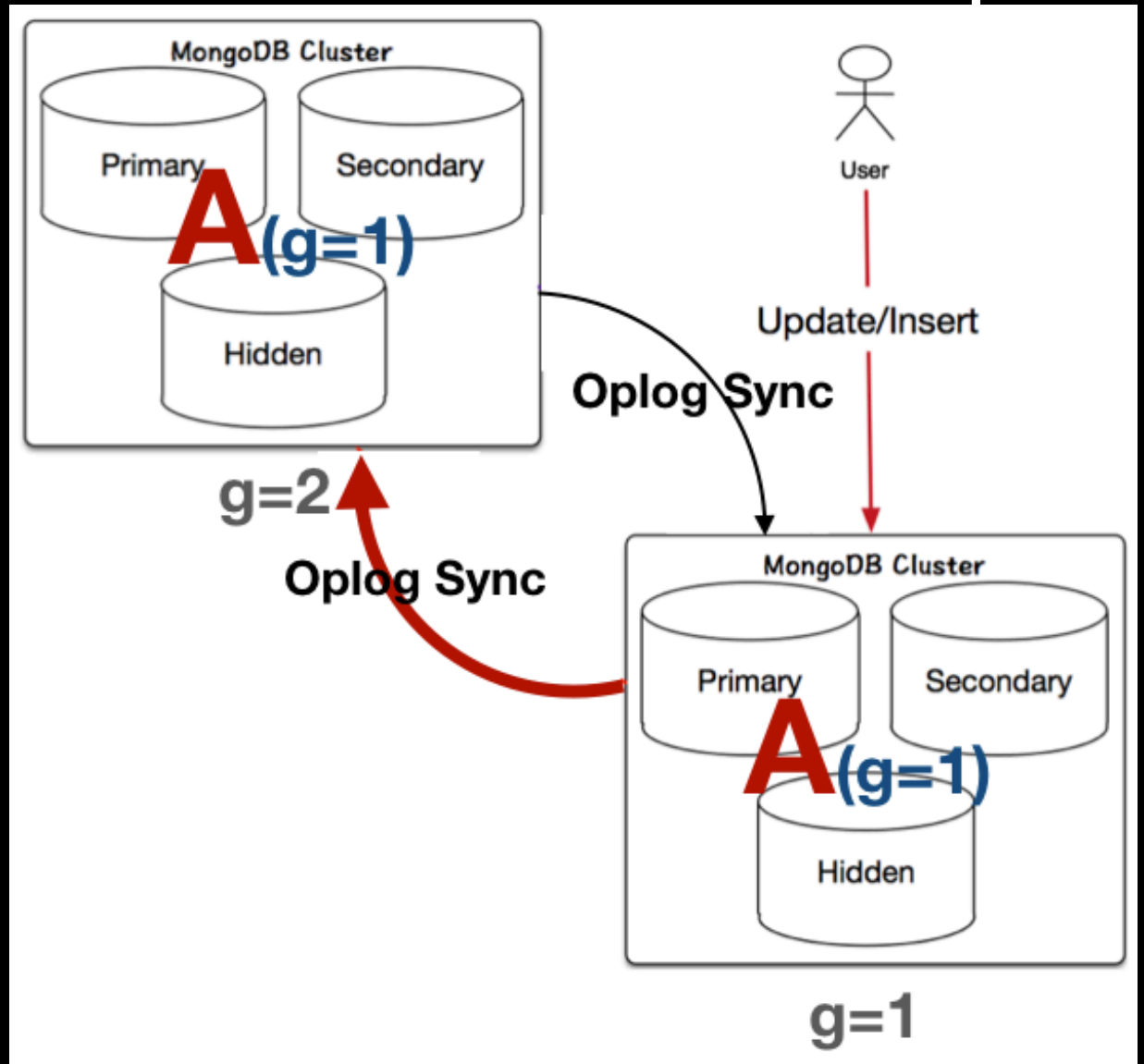
MongoShake Architecture-Loop



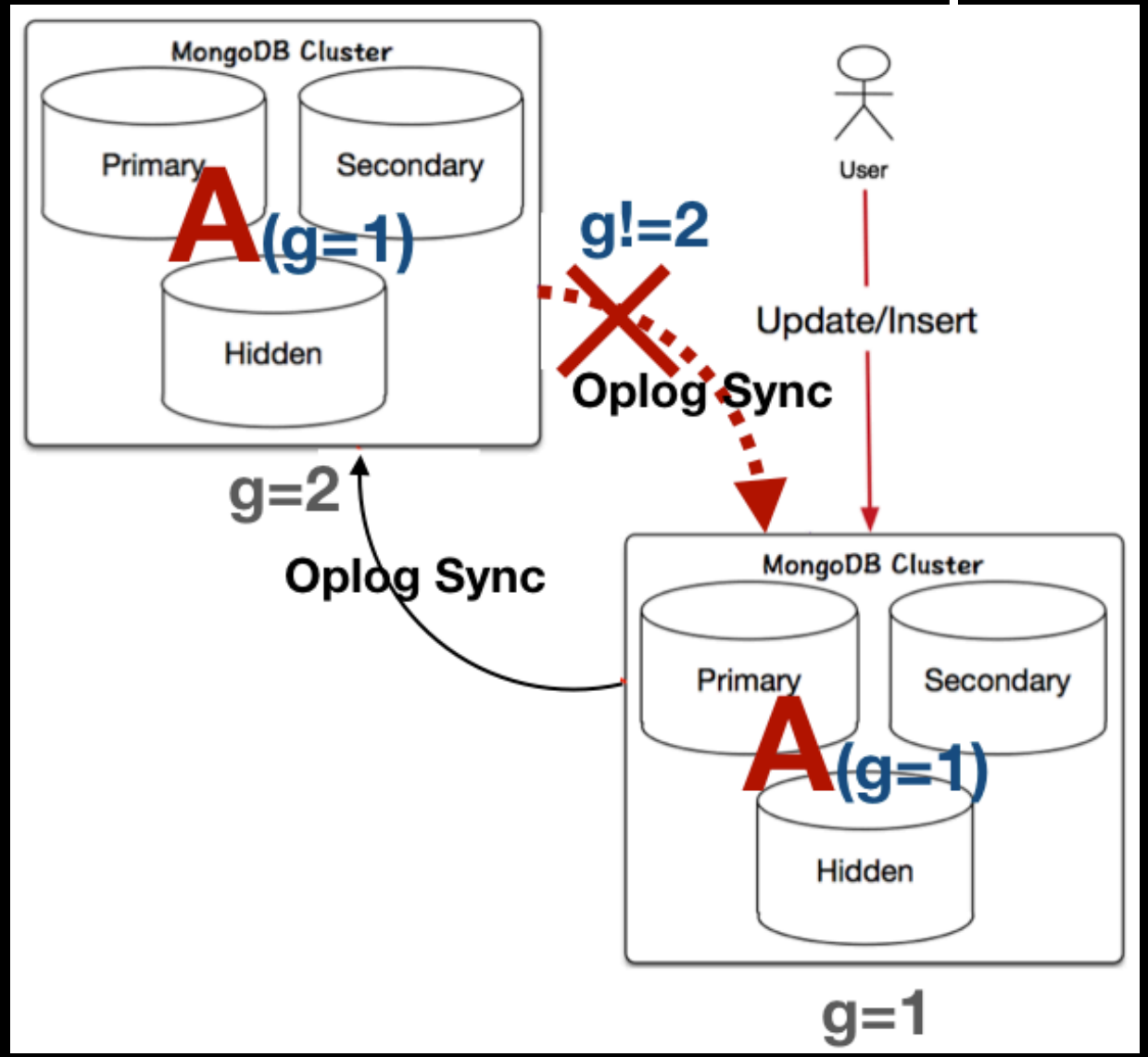
MongoShake Architecture-Loop



MongoShake Architecture-Loop

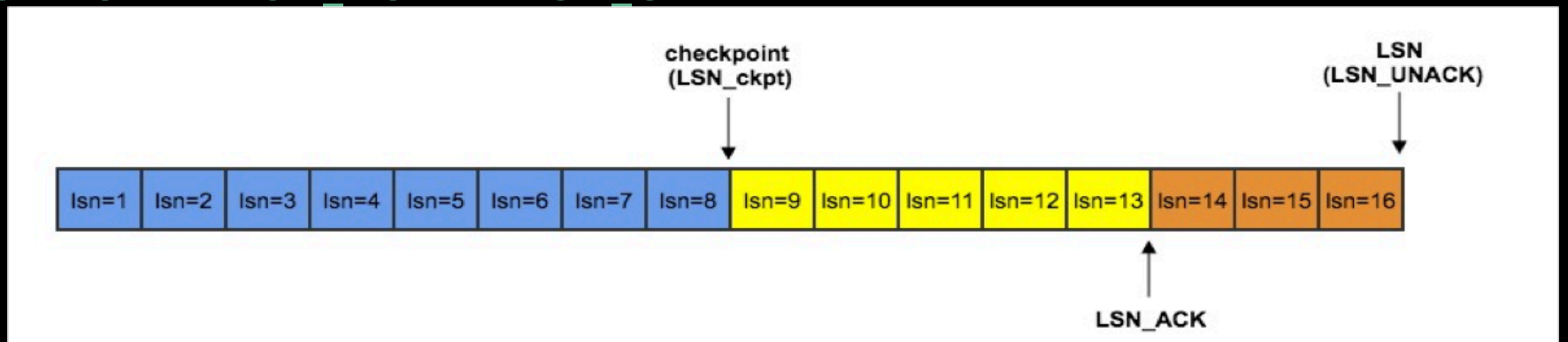


MongoShake Architecture-Loop



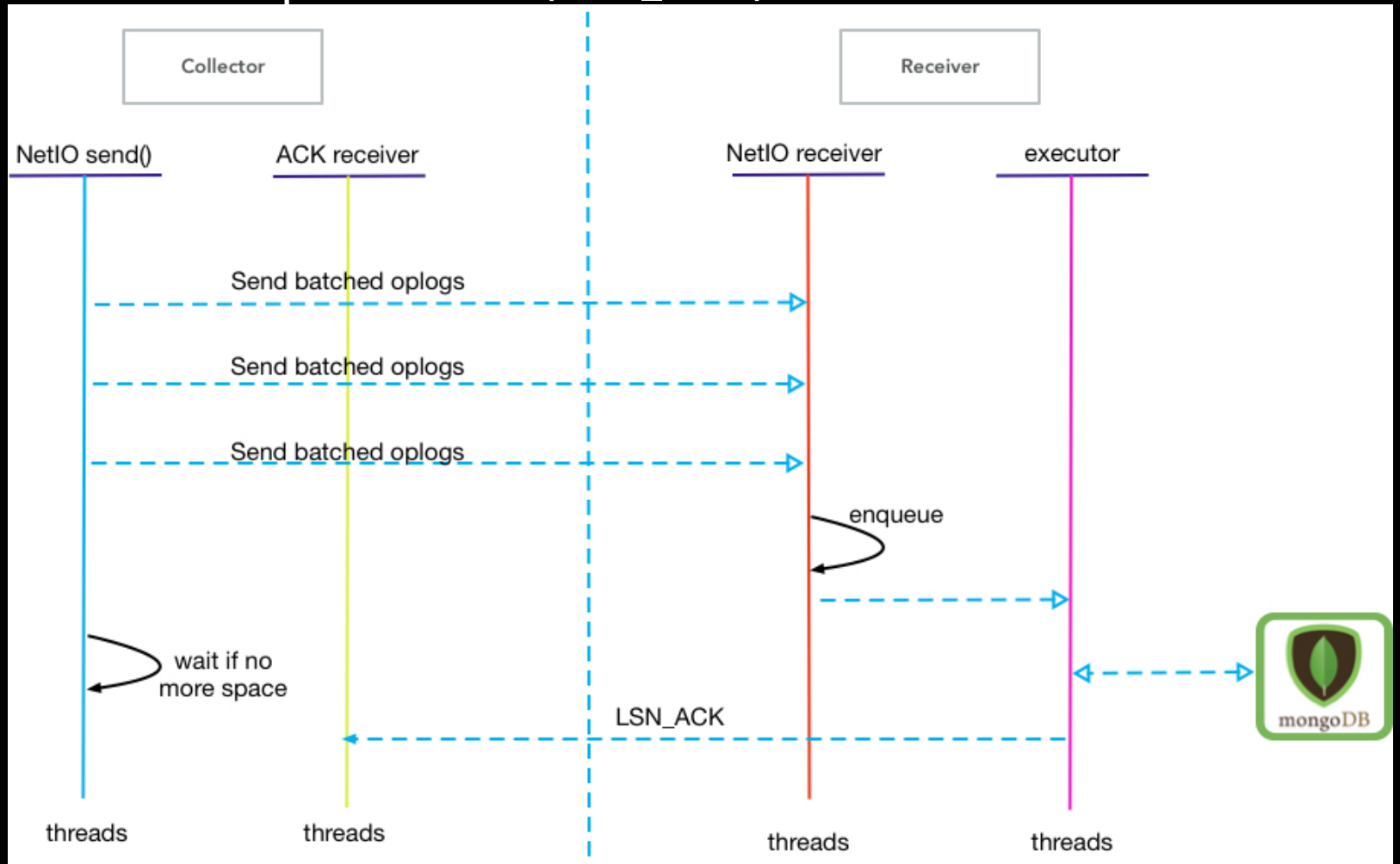
MongoShake Architecture-LSN

- Checkpoint (LSN_CKPT)
  - On crash or failure restarted, we need a offset point as the new beginning (checkpoint)
  - Idempotent. the result of data replaying will be looked as the same. It is no matter how much times the oplogs replayed or being replayed.
  - For the performance reasons and feature of idempotent. we would not save the offset positions on every updates. store them periodic (usually more than 30s)
- LSN related values
  - LSN is Log Sequence Number. increased only of value from oplog entry {ts: Timestamp}
  - LSN_ACK offset that we have executed successfully into destination.
  - LSN_UNACK offset that we has have not executed successfully into destination MongoDB but transferred
    - In sometimes. we would retransmit the segments from LSN_ACK to LSN_UNACK
  - LSN_UNACK >= LSN_ACK >= LSN_CKPT



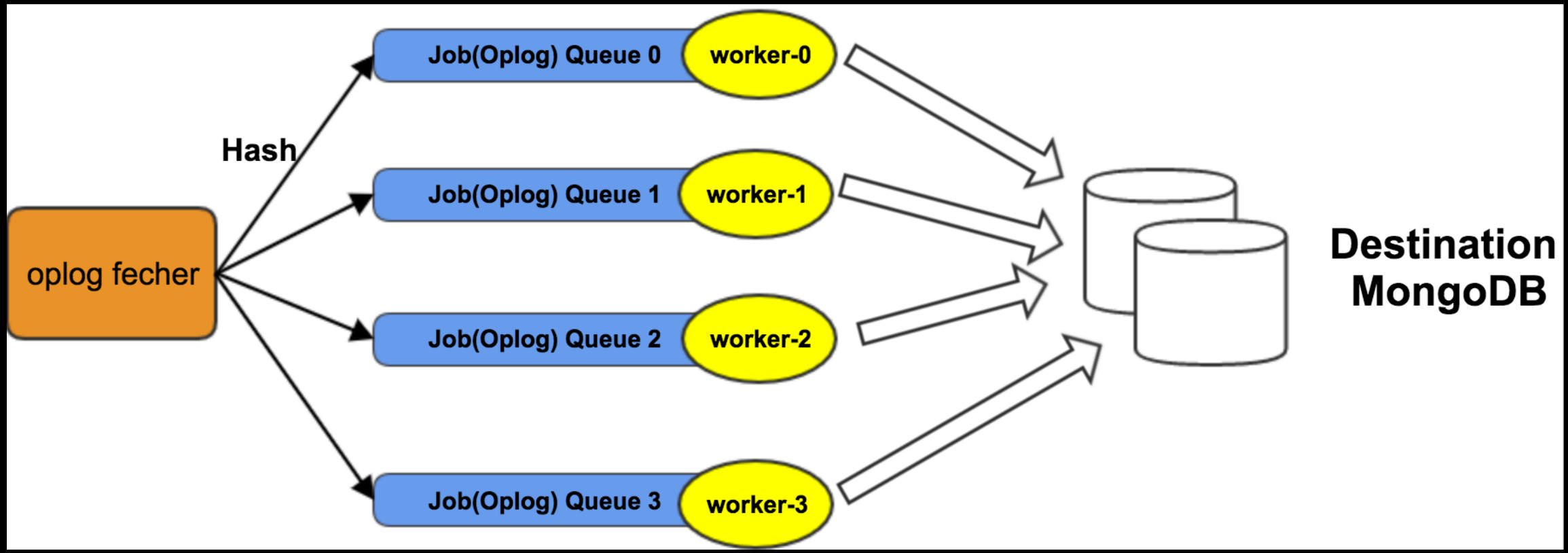
MongoShake Architecture-LSN ACK

- Async transfer & separate ACK (LSN_ACK)



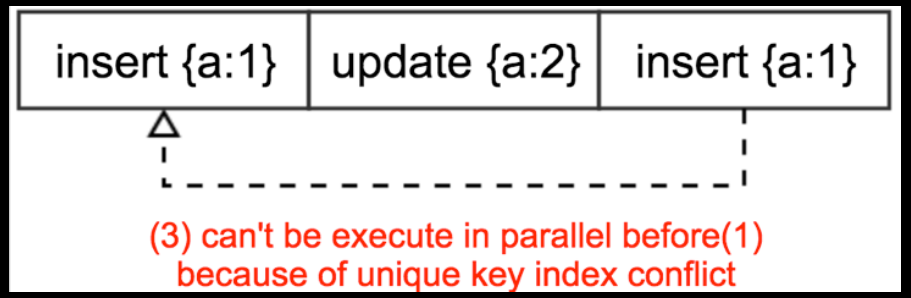
MongoShake Improvements — Parallel

- Oplogs executed in parallel



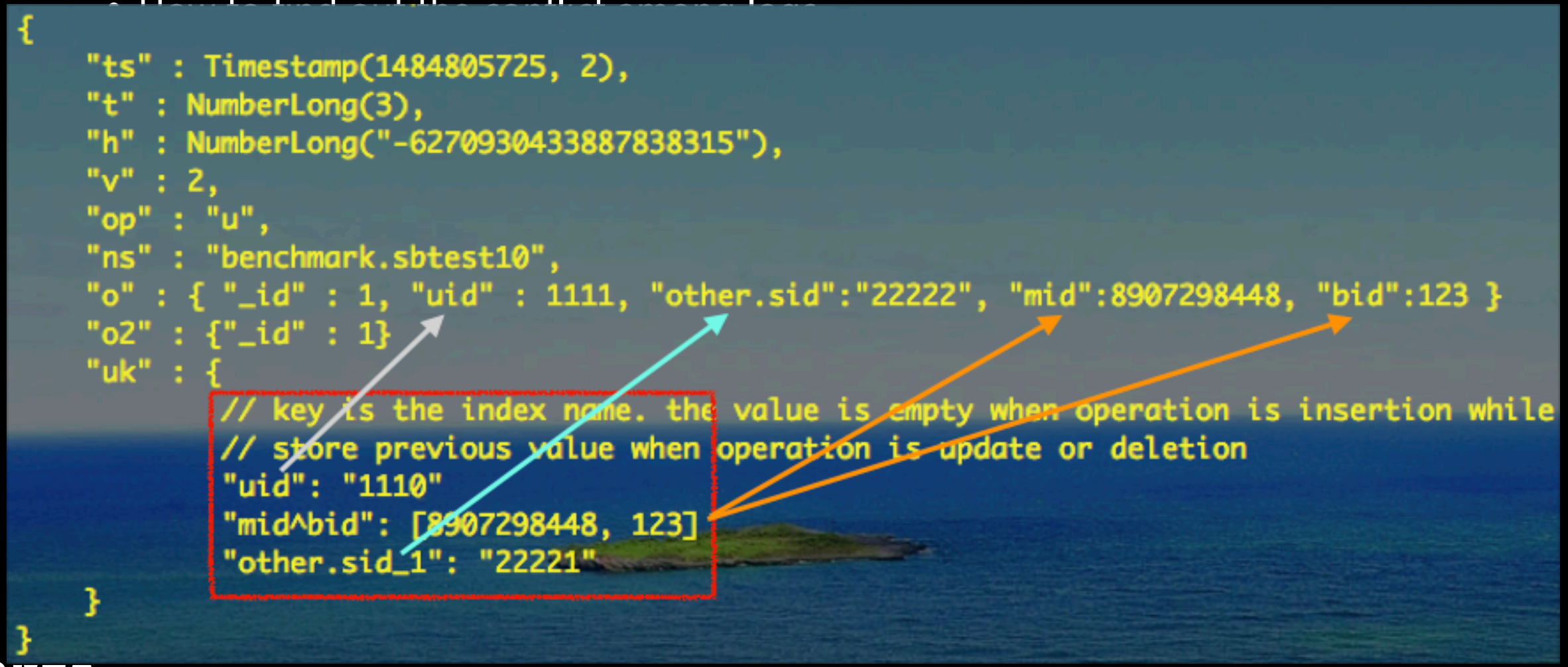
MongoShake Improvements — Parallel

- By collection name in Oplog.ns
  - crc32(ns) % n
  - Pros
    - Order of documents execution in same collection is guaranteed
    - Better concurrency with more collections created.
  - Cons
    - Performance will be dropped with only few collections exist
- By document primary id in Oplog._id
  - _id % n
  - Pros
    - Equally distributed in each queue
    - No jumbo queue normally
    - Single document’ s operation order is guaranteed
  - Cons
    - The order between different documents is not guaranteed.
    - May be phantom read (transaction is not supported in MongoDB)
  - MUST NOT include unique index (may be conflicted by unique indices)



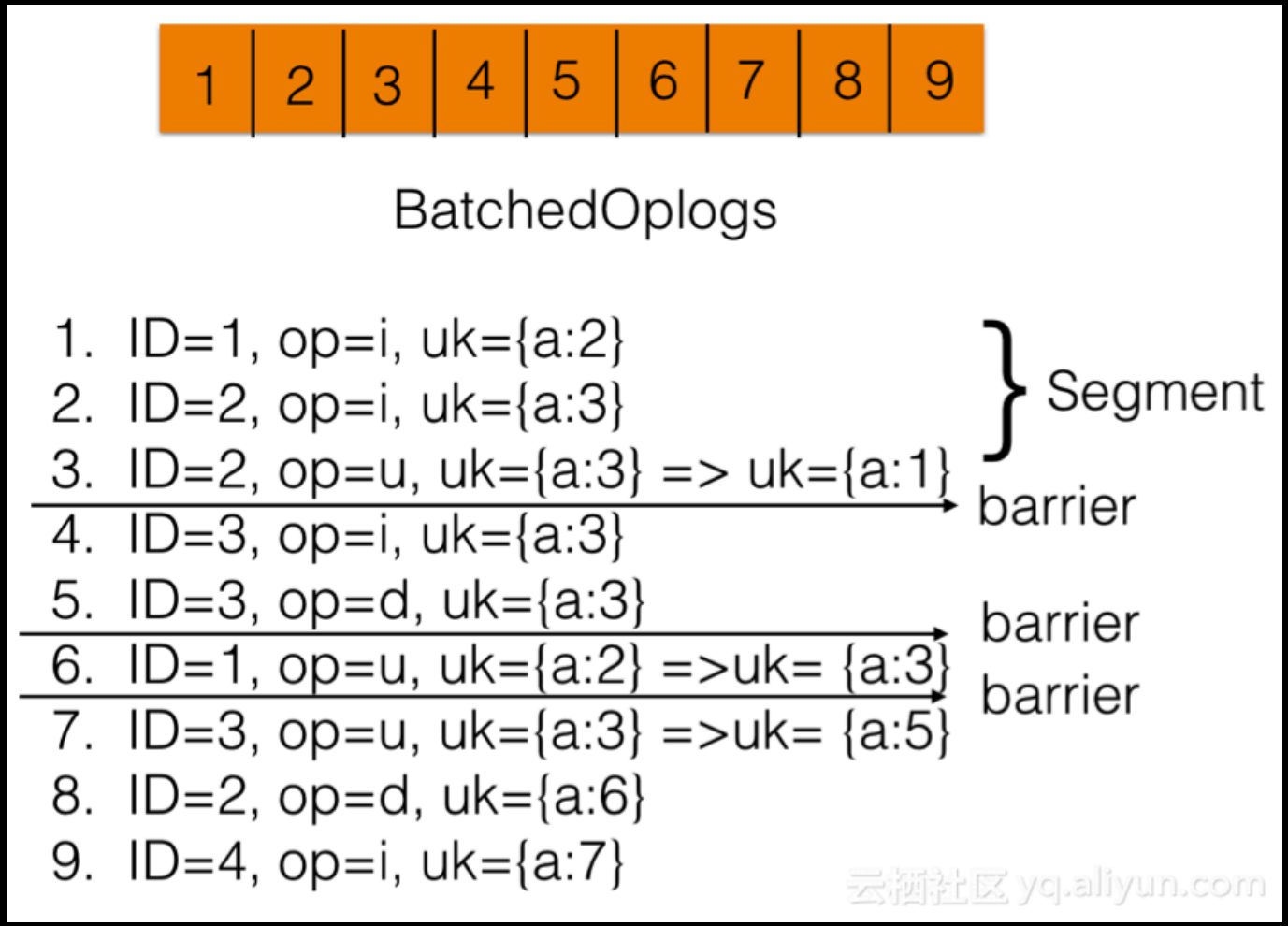
MongoShake Improvements — Parallel

- How to find out the conflict among logs



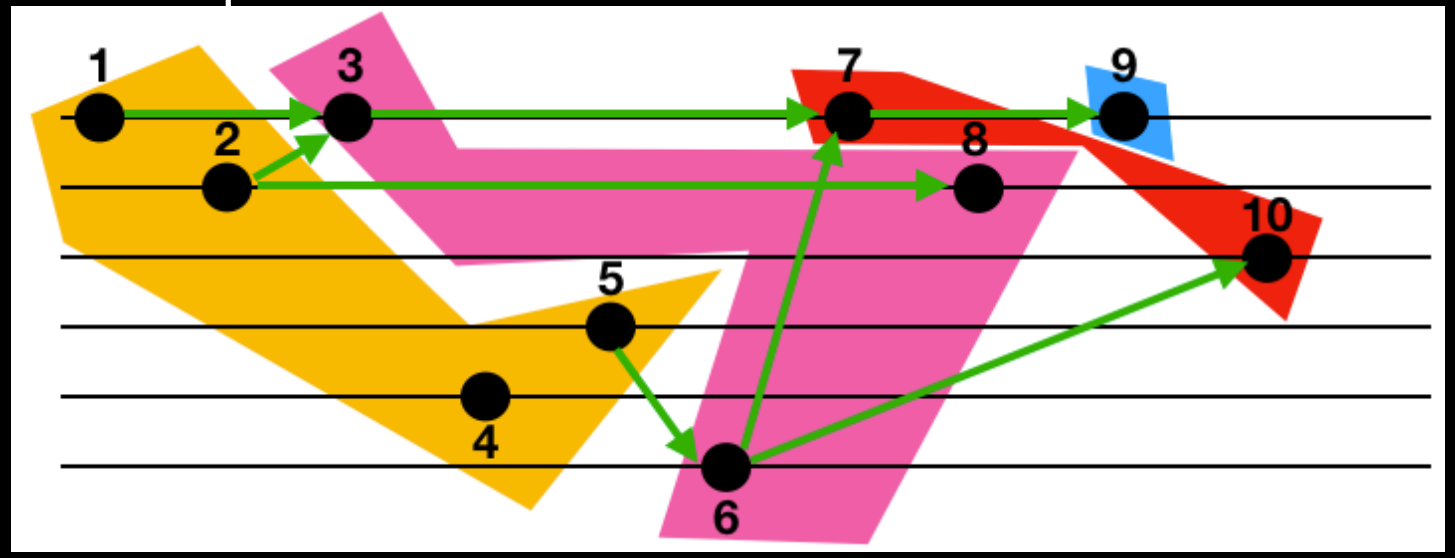
MongoShake Improvements — Parallel

Solution 1: Example: hashed by collection; 10 oplogs in this collection.



MongoShake Improvements — Parallel

Solution 2: 10 oplogs. from 1~10 in order of time. 2 and 3, 5 and 6, 6 and 7, 6 and 10 operates the same unique index.



Rules:

- If node M and node N operate the same unique key with same value no matter before or after, and the seq_id of M is less than N, then we build a directed edge from M to N.
- If node M and node N has the same document ID and the seq_id of M is less than N, then also build a directed edge from M to N.

Result: 1,2,4,5 -> 3,6,8 -> 7,10 -> 9

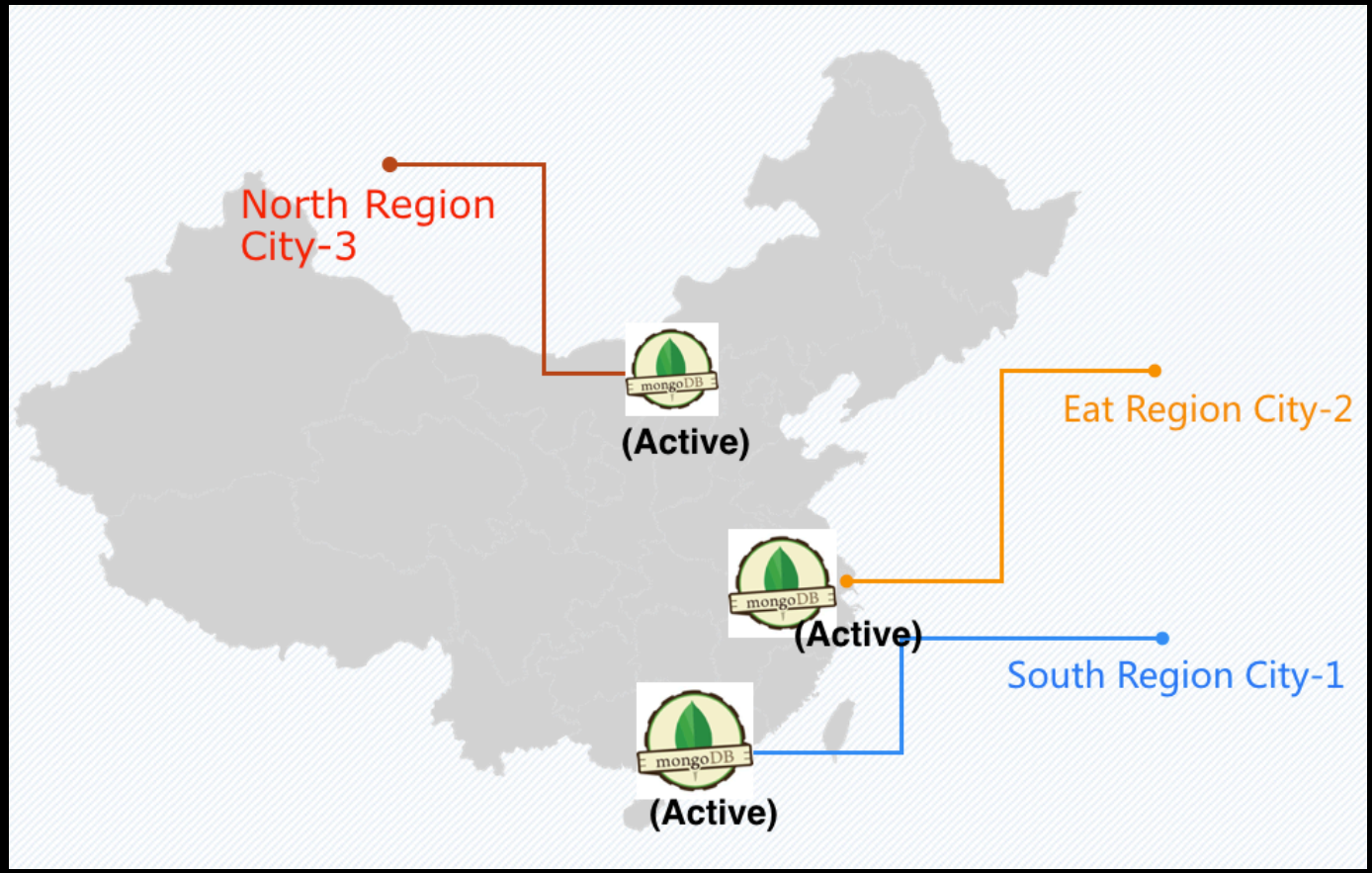
MongoShake Improvements — More features

- Oplog checksum
  - crc32() and verification
- Compress
  - Support snappy, gzip, LZO
  - Saved bandwidth on long distance transmission. it take more CPU but less network I/O
- Operation merge
  - Single document’ s adjacent updates will be merged
    - Adjacent (update docA to {a:1} + update docA to {b:1} => update {a:1, b:2})
  - Insertion followed by a deletion => Nothing
- Filter
  - Blacklist & Whitelist

Business User Case

Business User Case – 高德地图

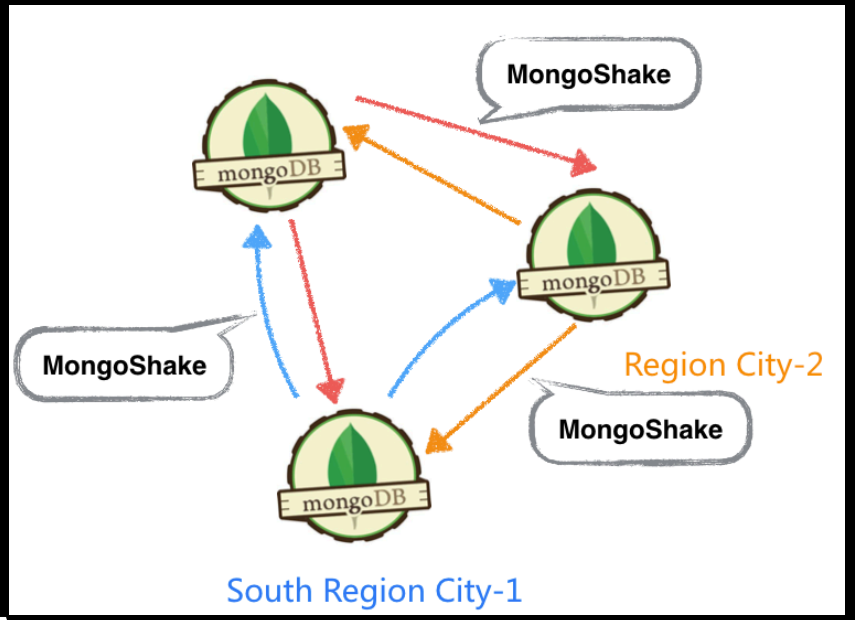
- Have deployed 3 regions in cities

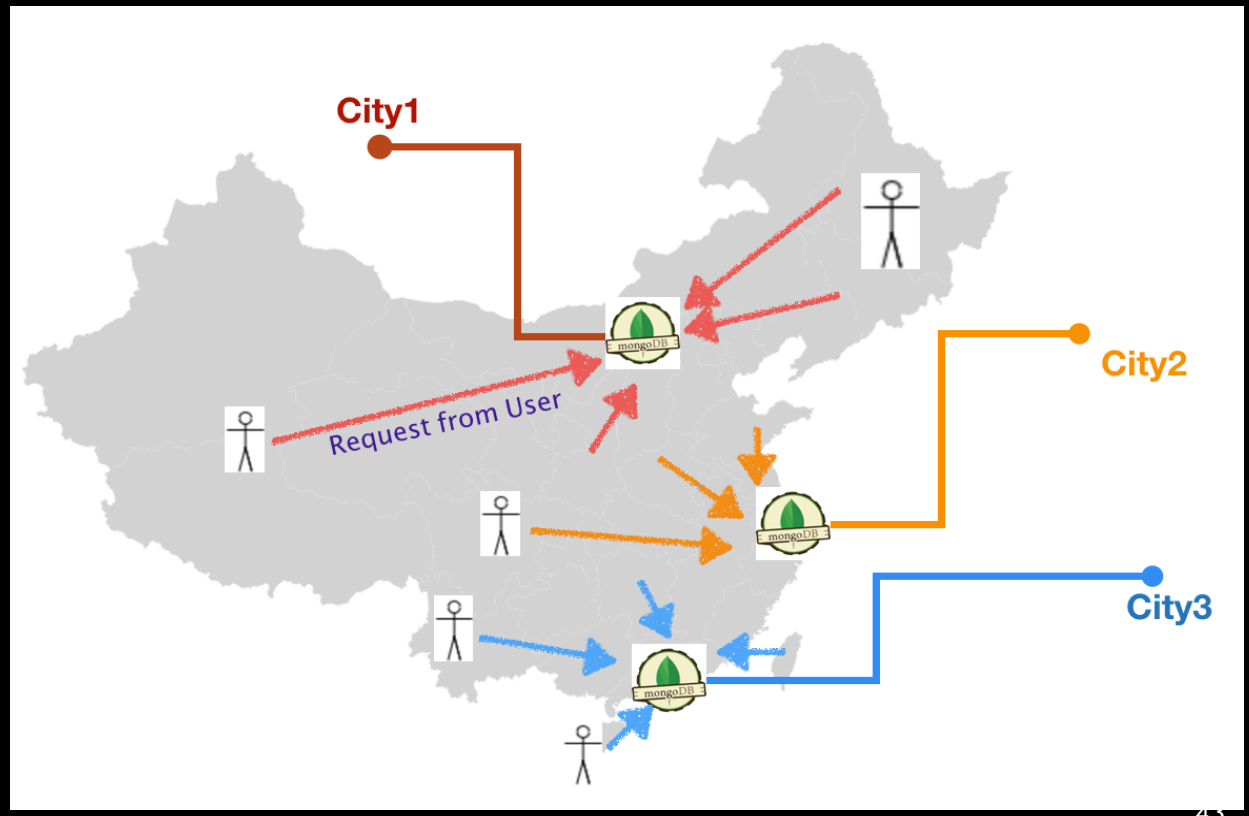


Active MongoDB Region with 1 ReplicaSet
Primary/Secondary/Hidden

Business User Case – 高德地图

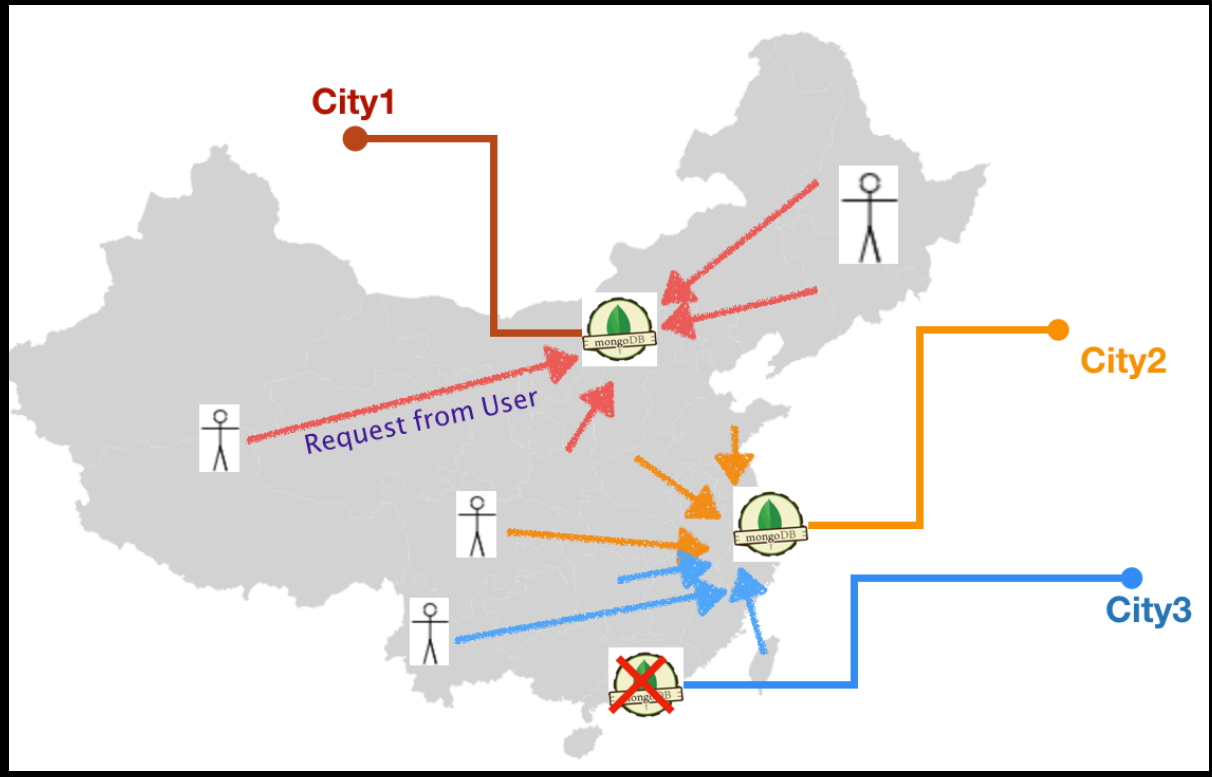
- Triple Active-Active, all are writable
- Geographical Routing by smart DNS





Business User Case – 高德地图

- Reroute the requests to the another closest “Active” MongoDB by DNS.
- All of the remain MongoDB are still writable
- Requests can be easily switch back to previous crashed region while the region recovers
- RPO < 1s and RTO < 60s

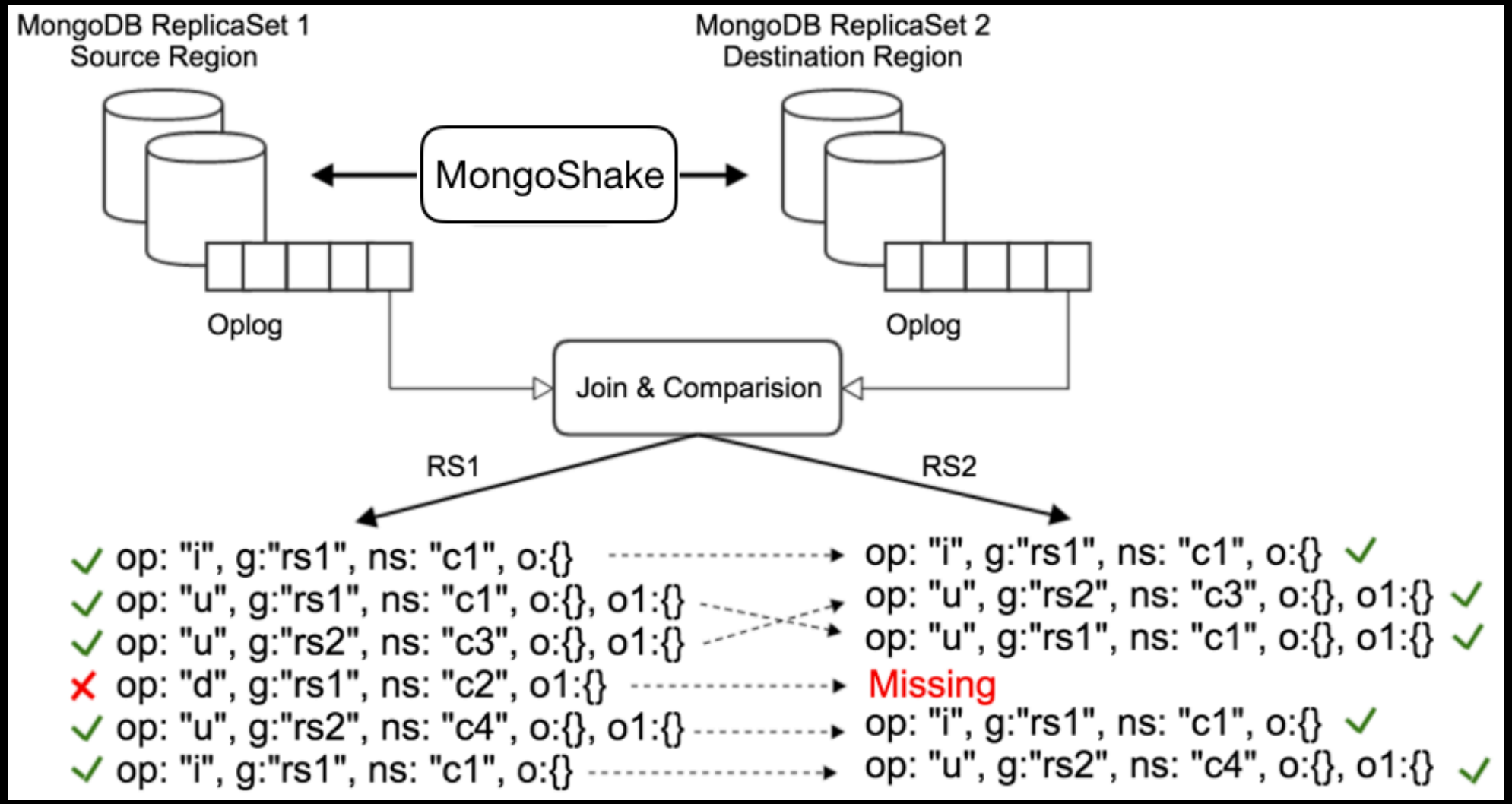


Data Quality Insurance

- Large scale size of all MongoDB data
  - 2 billion+ oplogs / day，incremental data size > 10G，70% is updates and 20%+ is insert and delete
- While serving with queries. full scan is impossible
  - Collection full scan > 5 hours
  - Document has modification and queries are executing while inspecting

	MetaData	Sampling	Full Comparison	Oplog Comparison
Found the problem	Fast	Slow	Slow	Fast
Locate the problem	Impossible	Slow	Fast	Fast
Cost	Low	Medium	High	Medium
Performance	Good	Good	Bad	Medium

Data Quality Insurance



Performance

Performance-QPS

- Single ReplicaSet
  - Mock tunnel: 55w
  - Direct tunnel, Write into DB, not conducive to merge: 19w
  - Direct tunnel, Write into DB, conducive to merge: 46w

Open Source

https://github.com/aliyun/mongo-shake

Q&A

Thank you!

Github address: https://github.com/aliyun/mongo-shake

We are hiring: zhuzhao.cx@alibaba-inc.com

Detailed document: https://yq.aliyun.com/articles/603329

MongoShake WeChat Group

MongoShake开源交流群