

# **Fast Crash Recovery in RAMCloud**

Michał Gregorczyk

Based on "Fast Crash Recovery in RAMCloud" by D. Ongaro, S.M. Rumble, R. Stutsman, J. Ousterhout, and M. Rosenblum

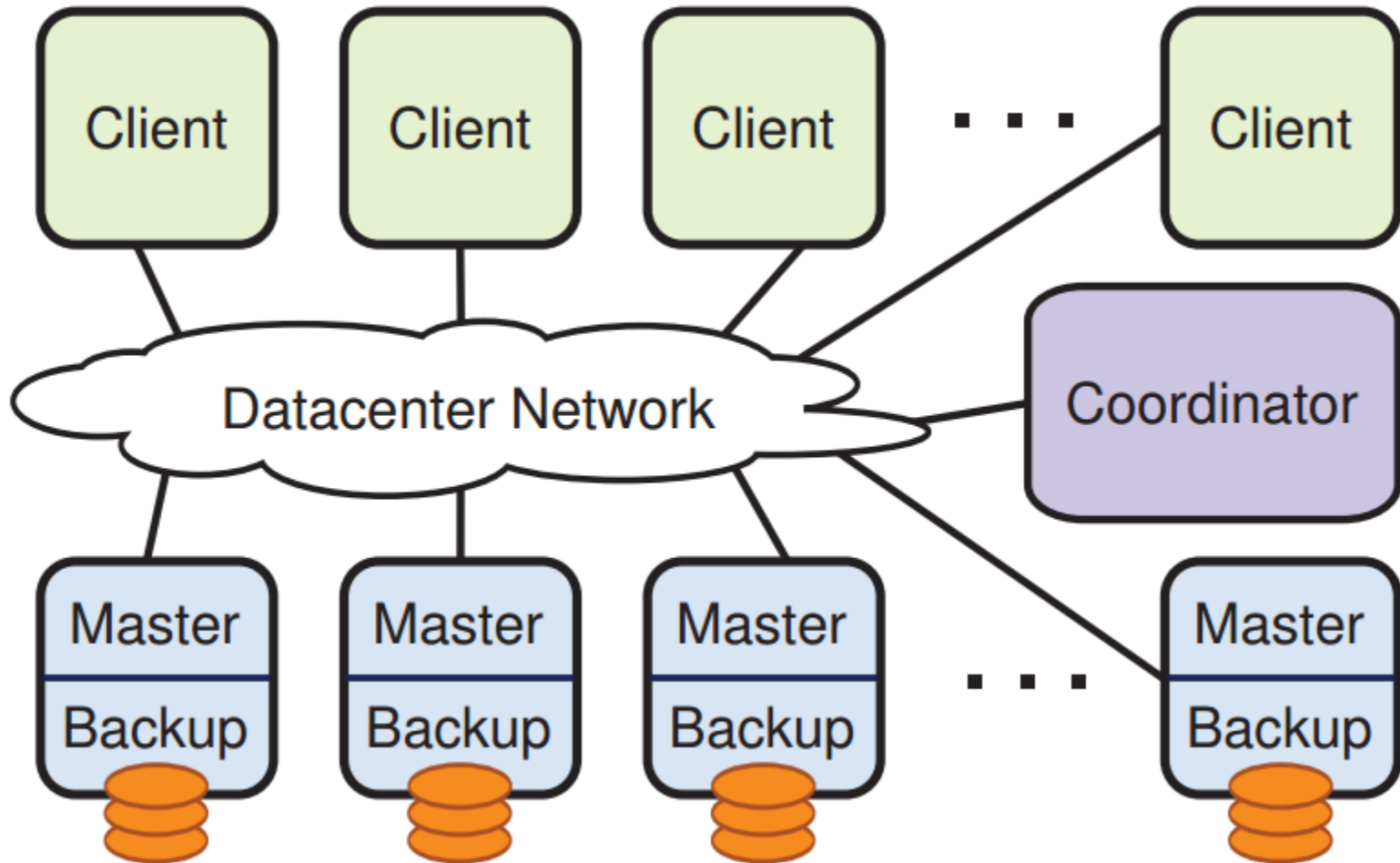
# What is RAMCloud?

- key-value distributed store
- log-structured storage
- data in DRAM
- replicas stored on disks
- high performance - latency of 5-10us
- high reliability - fast crash recovery

# Data Model

- **key-value**
  - key - 64 bits
  - value - byte array up to 1 MB
  - version - 64 bits
- **operations**
  - read
  - write
  - replace if version is equal to

# System Structure



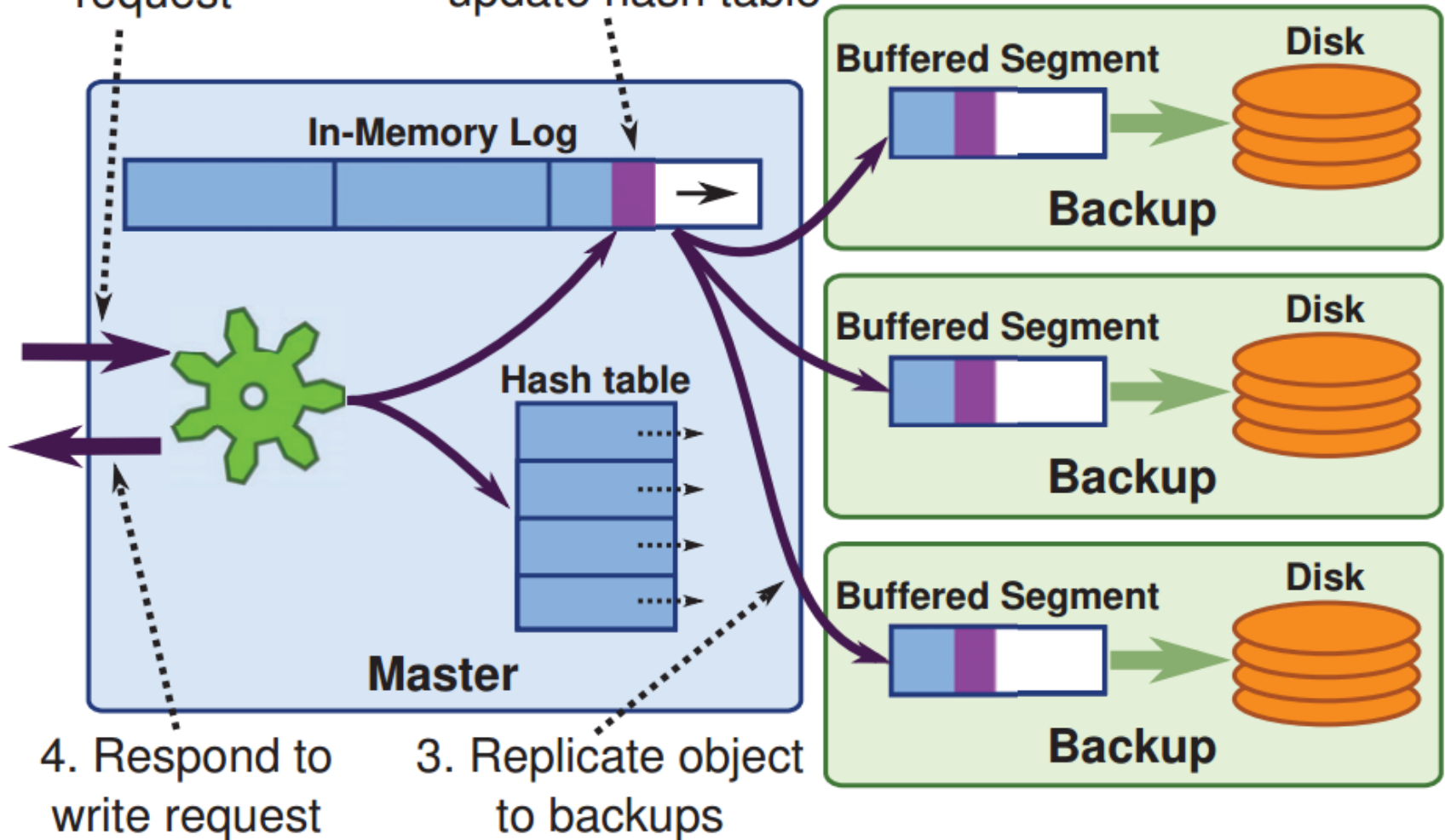
# System structure

- **master**
  - manages key-value pairs in DRAM
- **backup**
  - stores replicas of data from masters
- **coordinator**
  - stores configuration
  - mapping from key to master

- coordinator assigns objects to masters in tablets: key ranges within one table
- coordinator store mapping from tablets and storage servers
- client library caches this mapping

# Log-Structured Storage

1. Process write request
2. Append object to log and update hash table

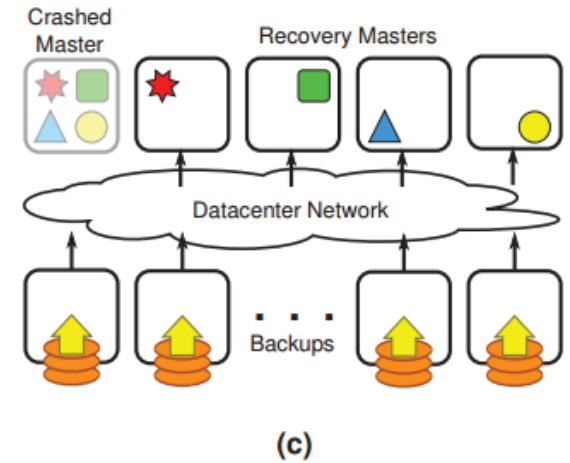
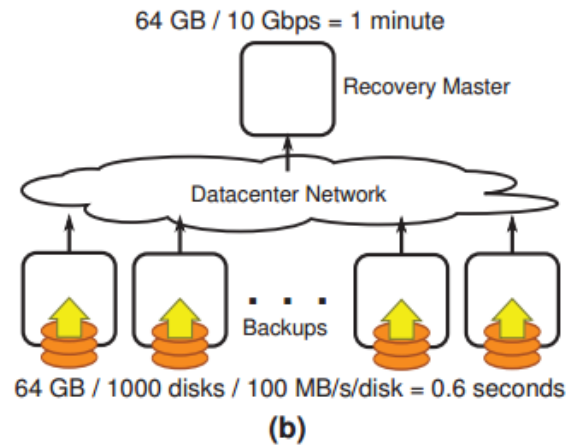
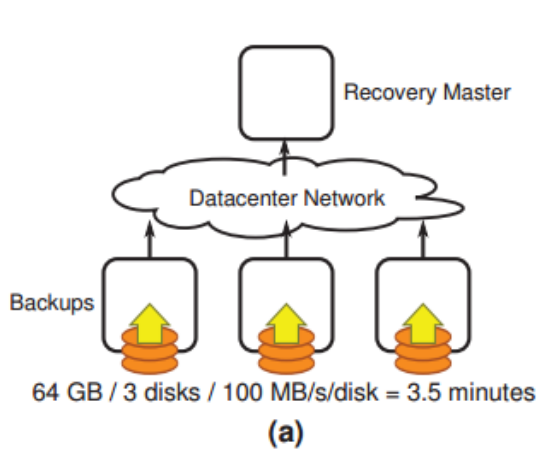


- master forwards new logs to backups
  - backups buffers new logs in memory buffers
  - when buffer is full, backup writes its content to disk
- 
- hash table is used to keep pointers to newest values



- log is split into segments
- segment = 8 MB
- segment is an unit of buffering and disk IO
  
- log cleaner
  - cleaner selects one or more segments to clean
  - segment is scanned and live log entries (hash table) are rewritten at the head of the log
  - old segment is freed

# Recovery



# Recovery

- thousands of backups
- hundreds of recovery masters

## Steps:

- scattering log segments
- failure detection
- recovery

# Scattering Log Segments

- master and backups must reside in different racks
- segments must be distributed so that each backup uses the same amount of time to read data
- avoid overloads of backup servers
- storage servers are continuously entering and leaving

# Scattering Log Segments

Master decides where to put replica:

- select random candidates
- pick best one
  - where are my segments
  - what is disk IO speed
- do not choose backup from the same rack
- allocate buffer on backup server
  - at this point backup server can reject the request

# Failure Detection

- if master fails to respond to RAMCloud client
- RAMCloud servers periodically send random pings to each other
  
- coordinator is informed about problem
- coordinator checks if server is down and starts recovery if the answer is positive

# Recovery Flow

1. Setup
2. Log Reply
3. Cleanup

# Setup

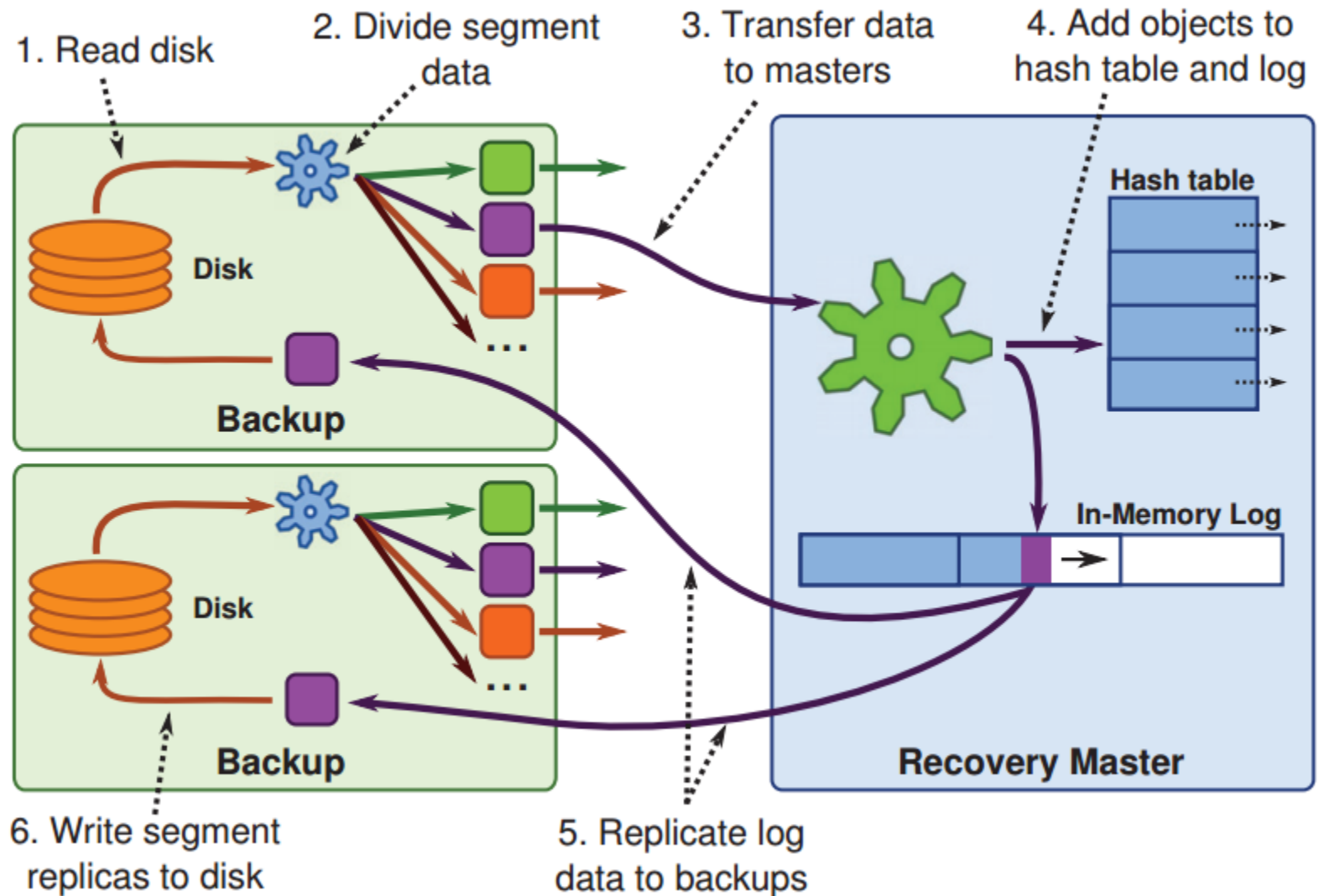
- coordinator reconstructs information about replicas locations by querying all backups in cluster
- coordinator determines if every log segment can be read
  - log digest - list of all segments present at the moment of write
  - only one log segment is marked as active
- data is split according to dead master's will
  - will is periodically uploaded to the coordinator in case of failure



# Setup

Recovery master receives (from coordinator)  
list of backups and list of tablets to recover

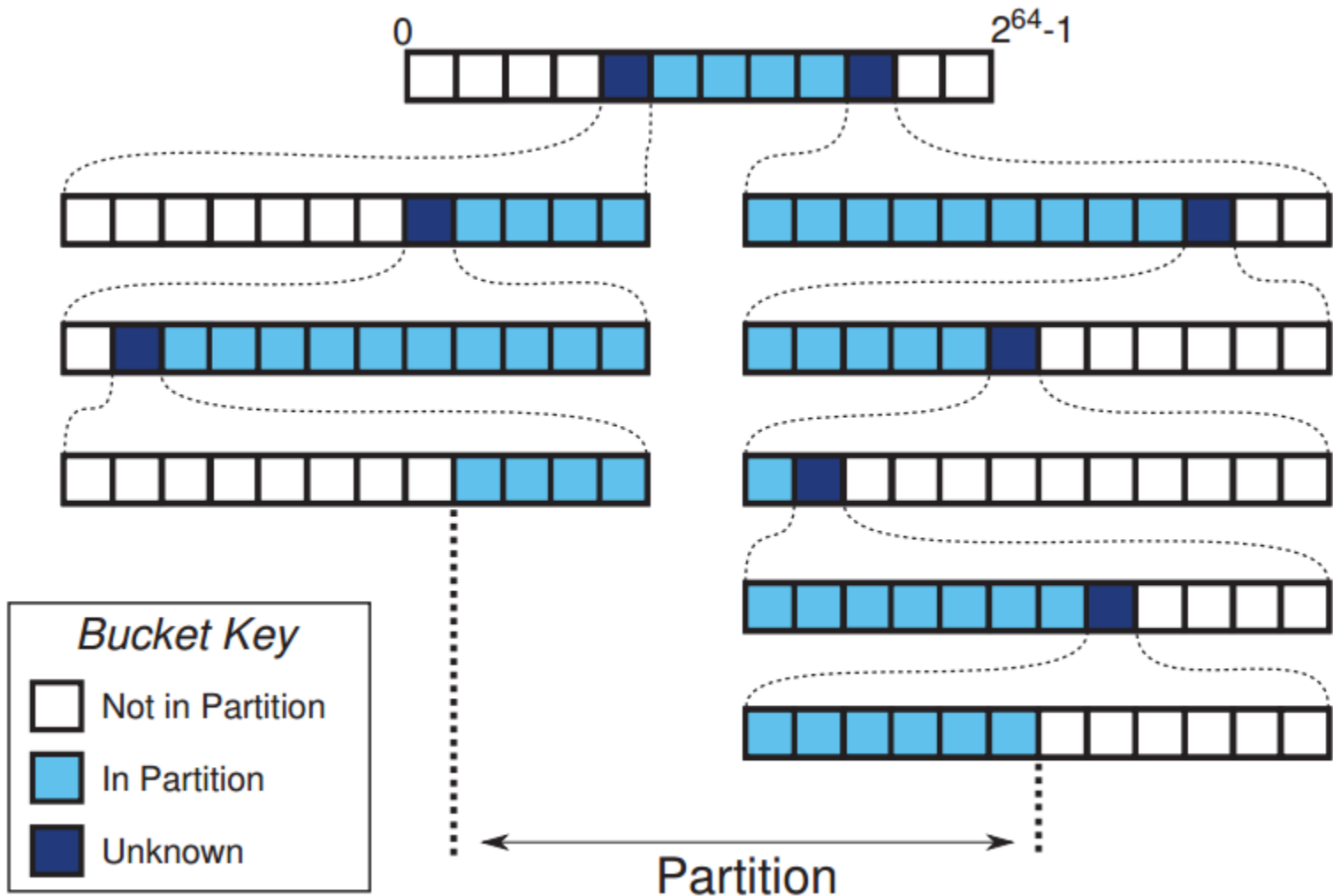
# Reply



# Reply

- data parallelism
- pipelining
  - logs do not have to be replayed in the same order - hash table and version

# Will and Tablet Profiling



# Coordinator Failures

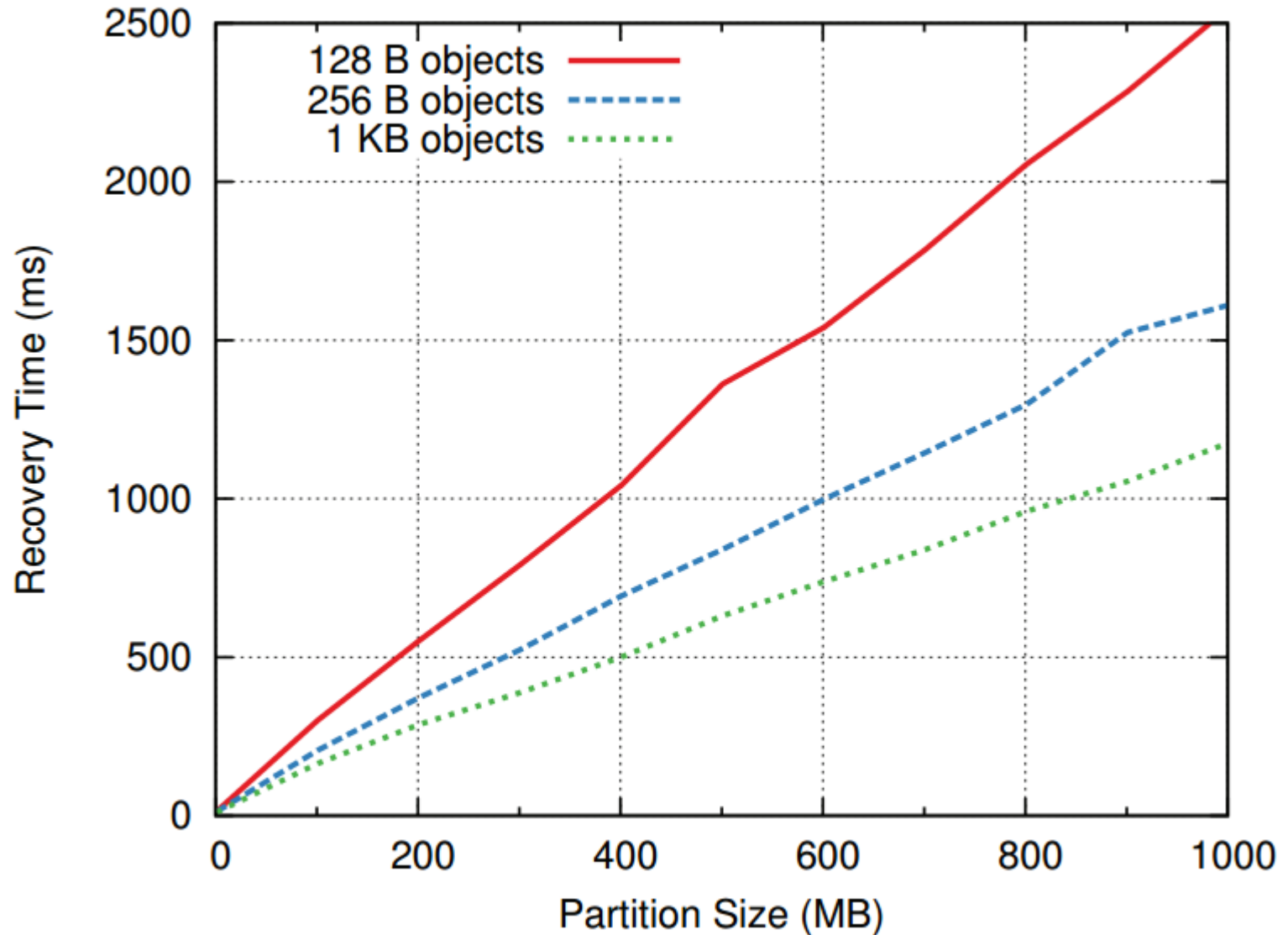
For coordinator recovery RAMCloud uses ZooKeeper and stand by coordinators.



# Evaluation

CPU	Xeon X3470 (4x2.93 GHz cores, 3.6 GHz Turbo)
RAM	16 GB DDR3 at 1333 MHz
Disk 1	WD 2503ABYX (7200 RPM, 250 GB) Effective read/write: 105/110 MB/s
Disk 2	Seagate ST3500418AS (7200 RPM, 500 GB) Effective read/write: 108/87 MB/s
Flash Disks	Crucial M4 CT128M4SSD2 (128GB) Effective read/write: 269/182 MB/s
NIC	Mellanox ConnectX-2 Infiniband HCA
Switches	5x 36-port Mellanox InfiniScale IV (4X QDR)

# Evaluation



Any questions ? No ?  
Thank you.