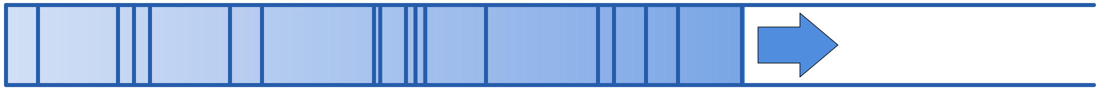
Metrics for RAMCloud Recovery

John Ousterhout
Stanford University

Logging/Recovery Basics

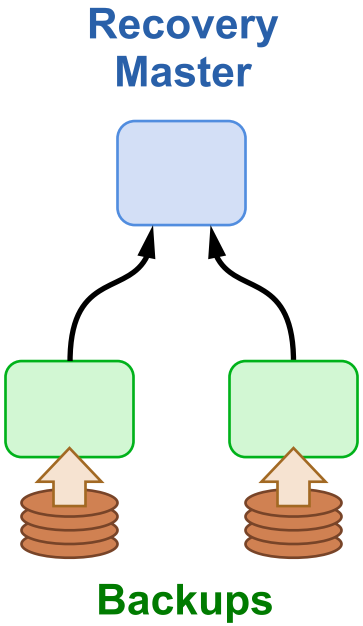
- All data/changes appended to a log:



- One log for each master (kept in DRAM)
- Log data is replicated on disk on 2+ backups
- During recovery:
  - Read data from disks on backups
  - Replay log to recreate data on master
- Recovery must be fast: 1-2 seconds!
  - Only one copy of data in DRAM
  - Data unavailable until recovery completes

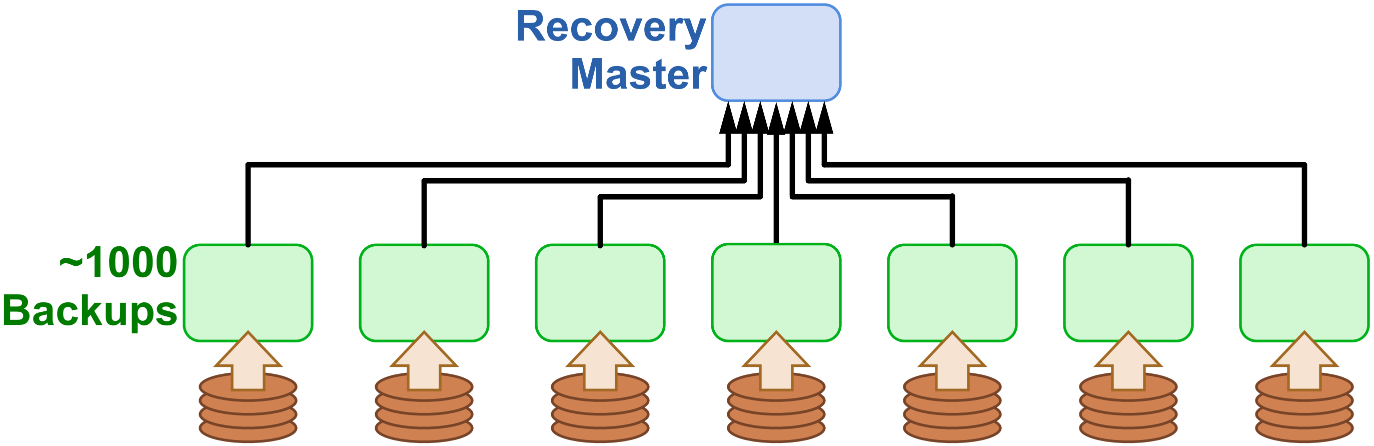
Recovery, First Try

- Master chooses backups statically
  - Each backup stores entire log for master
- Crash recovery:
  - Choose recovery master
  - Backups read log info from disk
  - Transfer logs to recovery master
  - Recovery master replays log
- First bottleneck: disk bandwidth:
  - 64 GB / 2 backups / 100 MB/sec/disk ≈ 320 seconds
- Solution: more disks (more backups)



Recovery, Second Try

- Scatter logs:
  - Each log divided into 8MB segments
  - Master chooses different backups for each segment (randomly)
  - Segments scattered across all servers in the cluster
- Crash recovery:
  - All backups read from disk in parallel
  - Transmit data over network to recovery master

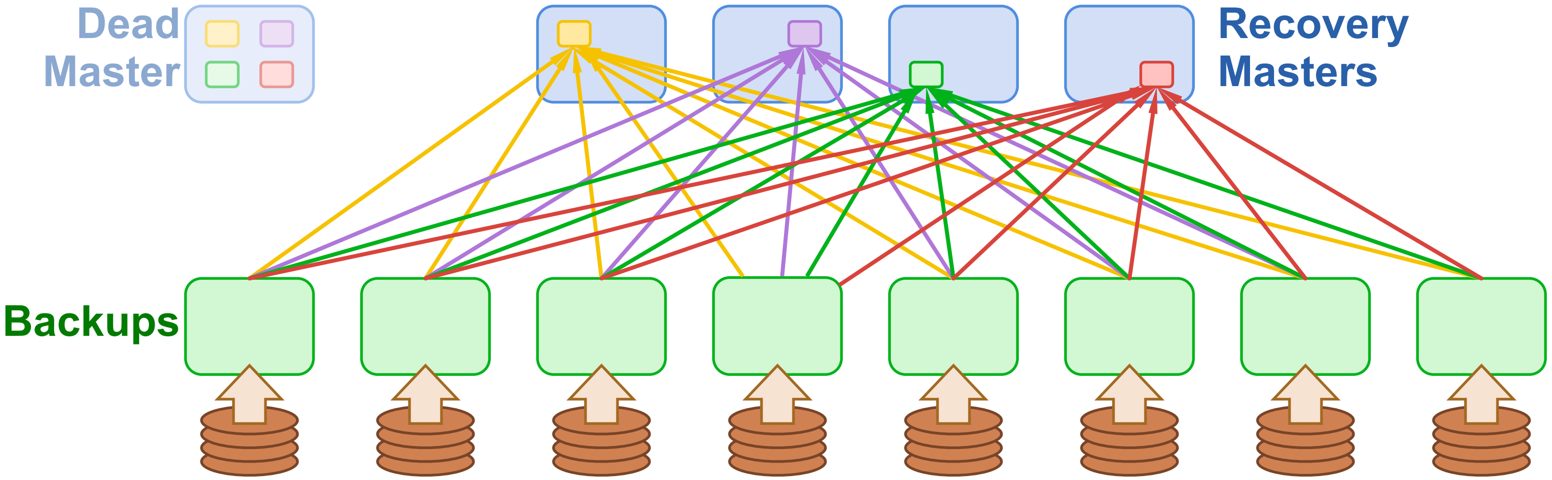


Scattered Logs, cont’d

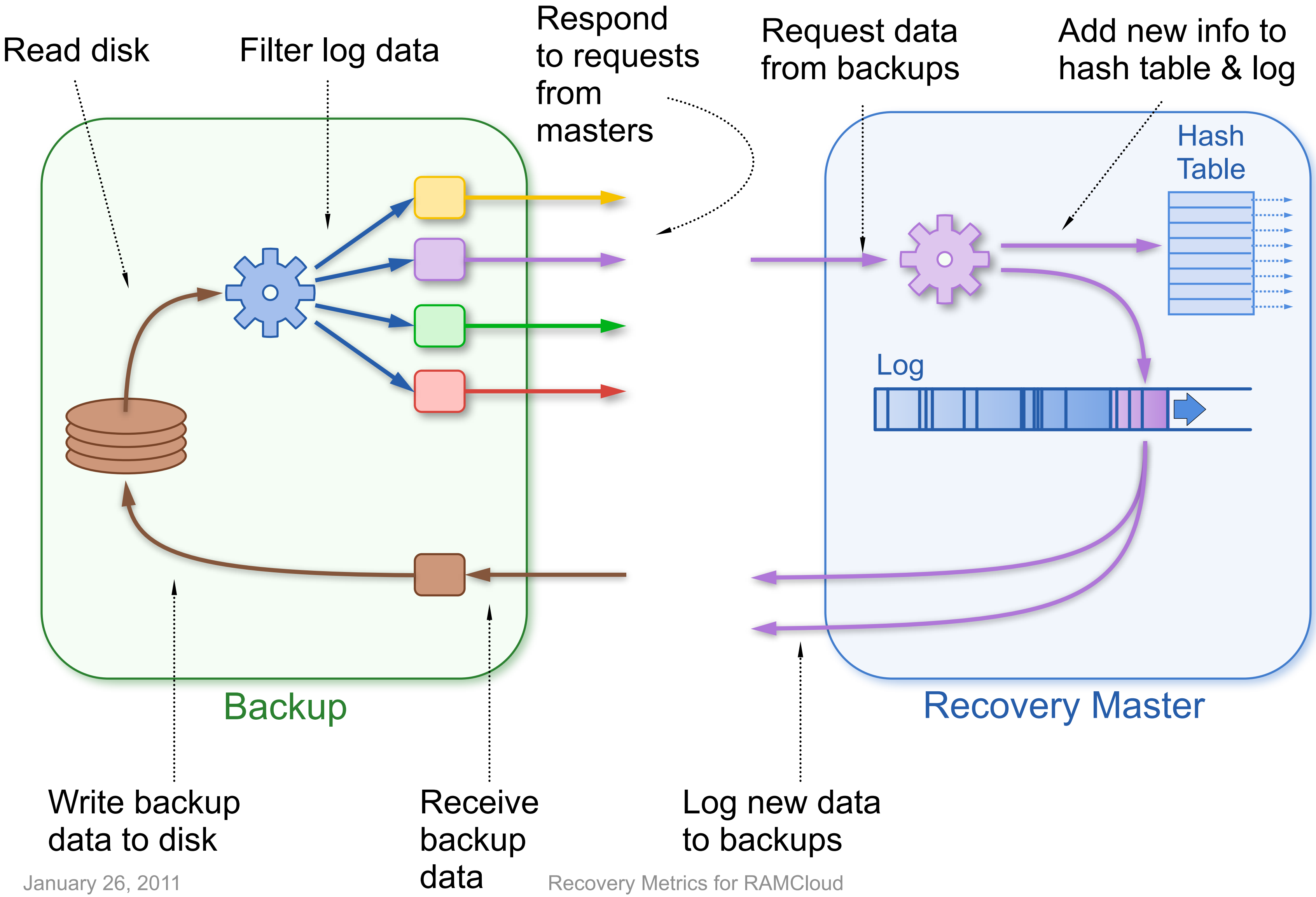
- Disk no longer a bottleneck:
  - 64 GB / 8 MB/segment / 1000 backups ≈ 8 segments/backup
  - 100ms/segment to read from disk
  - 0.8 second to read all segments in parallel
- Second bottleneck: NIC on recovery master
  - 64 GB / 10 Gbits/second ≈ 60 seconds
- Solution: more NICs
  - Spread work over 100 recovery masters
  - 64 GB / 10 Gbits/second / 100 masters ≈ 0.6 second

Recovery, Third Try

- Divide each master's data into partitions
  - Recover each partition on a separate recovery master
  - Partitions based on tables & key ranges, *not log segment*
  - Each backup divides its log data among recovery masters



Parallelism in Recovery



Read disk

Filter log data

Respond to requests from masters

Request data from backups

Add new info to hash table & log

Hash Table

Log

Backup

Recovery Master

Write backup data to disk

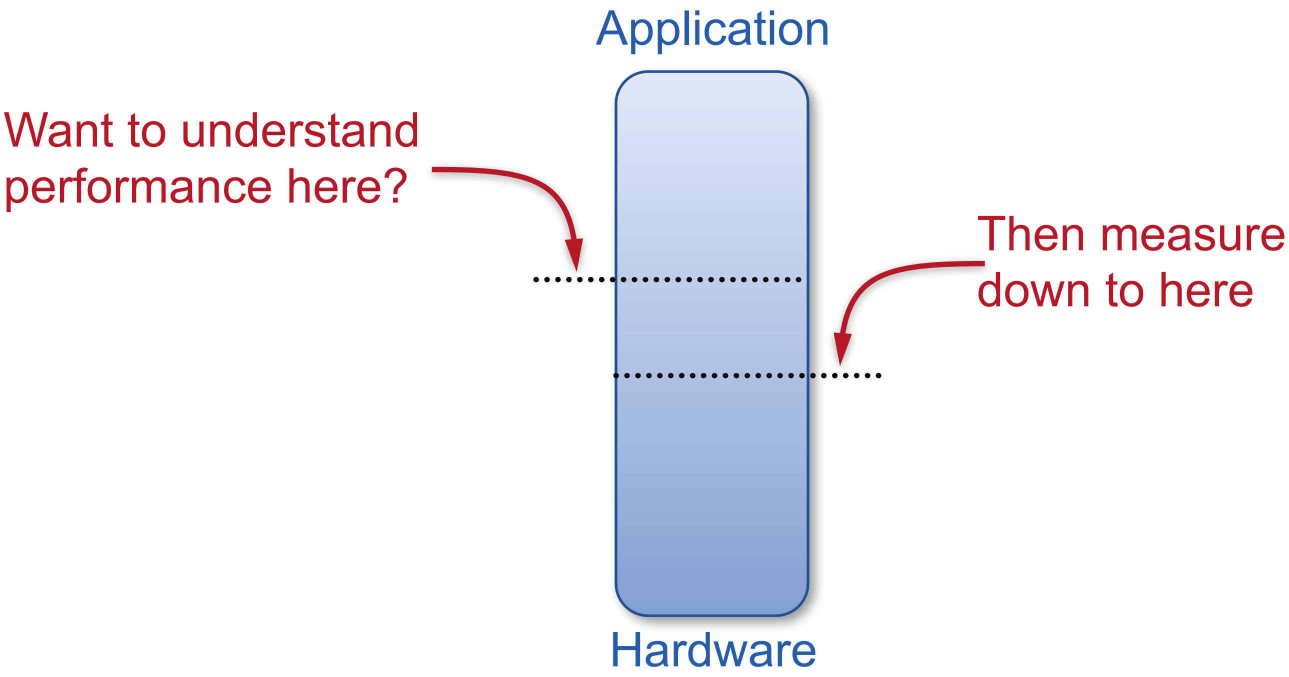
Receive backup data

Log new data to backups

Sample Performance Questions

- How fast is recovery?
- What is the current bottleneck?
- How do we fix it?
- What will the next bottleneck be?
- What is the ultimate limit for performance?
- If the network bandwidth drops, will it hurt recovery performance?
- Someone just made a commit and the behavior/ performance is different; why?

Measure Deeper


Application
Want to understand performance here?
Then measure down to here
Hardware

- Performance measurements often wrong/counterintuitive
- Measure components of performance
- Understand *why* performance is what it is

Utilizations

Where is time being spent?
(% of recovery time in each major activity)

- Backup:
  - Disk: reading log data, writing new backup data
  - CPU: filtering logs
  - Network: transmitting logs to recovery masters, receiving new backup data
- Recovery masters:
  - CPU: processing incoming log data
  - Network: receiving log data, writing new backup data

Utilizations, cont’d

- Stalls:
  - Master waiting for incoming log data
  - Master waiting to write new backup data
  - Backup delaying RPC response because segment not yet filtered
  - Disk reads delayed because disk busy writing
  - Can’t send RPC because NIC busy
- Break out low-level factors if significant:
  - Memcpy for master copying data into log
  - Memcpy for backup copying filtered segment data
  - …

Efficiency

How efficient are individual components/algorithms?

- Backups:
  - Disk read/write bandwidths
  - Network in/out bandwidths
  - RPCs/sec.
  - Filtering: bytes/sec., objects/sec.
- Masters:
  - Network in/out bandwidths
  - RPCs/sec.
  - Efficiency of concurrent RPCs
  - Log processing: bytes/sec., objects/sec.
- Miscellaneous:
  - memcpy bandwidth

Miscellaneous

- Total times:
  - For entire recovery
  - For each master to complete its recovery
- Total data volumes:
  - Read by each backup from disk
  - New data received by each server
  - Total new data written to server log
  - Total new live data after recovery
- Uniformity of filtering: how evenly does segment data divide between servers?
- Unexpected events:
  - Reading secondary copies of segments
  - Crashes of recovery masters/backups

Aggregation

- Collect info separately on each master/backup
  - Information applies to a single recovery
  - Write to local log at end of recovery
- Coordinator collects data from each master/backup at the end of recovery
  - Log raw data on coordinator
  - Aggregate to generate overall statistics, such as:
    - Overall utilizations
    - Effective disk bandwidth (read/write) for the entire cluster
    - Effective network bandwidth for entire cluster
    - Avg. disk utilization for backups reading log data
    - Disk bandwidth on individual backups: avg., best, worst, std dev
  - Log aggregate info