Web Technologies: RAMCloud and Fiz

John Ousterhout
Stanford University
The Web is Changing Everything

Discovering the potential:
- New applications
- 100-1000x scale
- New development style
- New approach to deployment

Realizing the potential:
- New models of computation (EC2)
- New storage (Bigtable, Dynamo)
- New algorithms (MapReduce)
- New languages
- New frameworks
- New approaches to software development
RAMCloud Introduction

- New research project at Stanford
  (Kozyrakis, Mazières, Mitra, Ousterhout, Parulkar, Prabhakar, Rosenblum)

- Create large-scale storage systems entirely in DRAM

- Interesting combination: scale, low latency

- The future of datacenter storage?

- Topics for this talk:
  - Overview of RAMCloud
  - Motivation
  - Research challenges
RAMCloud Overview

- Storage for datacenters
- 1000-10000 commodity servers
- 64 GB DRAM/server
- All data always in RAM
- Durable and available
- High throughput: 1M ops/sec/server
- Low-latency access: 5-10µs RPC
## Example Configurations

<table>
<thead>
<tr>
<th></th>
<th>Today</th>
<th>5-10 years</th>
</tr>
</thead>
<tbody>
<tr>
<td># servers</td>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td>GB/server</td>
<td>64GB</td>
<td>1024GB</td>
</tr>
<tr>
<td>Total capacity</td>
<td>64TB</td>
<td>1PB</td>
</tr>
<tr>
<td>Total server cost</td>
<td>$4M</td>
<td>$4M</td>
</tr>
<tr>
<td>$/GB</td>
<td>$60</td>
<td>$4</td>
</tr>
</tbody>
</table>
RAMCloud Motivation

- Relational databases don’t scale

- Every large-scale Web application has problems:
  - Facebook: 4000 MySQL servers + 2000 memcached servers

- New forms of storage starting to appear:
  - Bigtable
  - Dynamo
  - PNUTS
  - H-store
  - memcached

- Many apps don’t need all RDBMS features, can’t afford them
**RAMCloud Motivation, cont’d**

Disk access rate not keeping up with capacity:

<table>
<thead>
<tr>
<th></th>
<th>Mid-1980’s</th>
<th>2009</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disk capacity</td>
<td>30 MB</td>
<td>200 GB</td>
<td>6667x</td>
</tr>
<tr>
<td>Max. transfer rate</td>
<td>2 MB/s</td>
<td>100 MB/s</td>
<td>50x</td>
</tr>
<tr>
<td>Latency (seek &amp; rotate)</td>
<td>20 ms</td>
<td>10 ms</td>
<td>2x</td>
</tr>
<tr>
<td>Capacity/bandwidth</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(large blocks)</td>
<td>15 s</td>
<td>2000 s</td>
<td>133x</td>
</tr>
<tr>
<td>Capacity/bandwidth</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(200B blocks)</td>
<td>3000 s</td>
<td>115 days</td>
<td>3333x</td>
</tr>
</tbody>
</table>

- Disks must become more archival
- Can’t afford small random accesses
- RAM cost today = disk cost 10 years ago
Why Not a Caching Approach?

- **Encourages bad habits:**
  - “A few misses are OK” …. NOT!
  - 1% misses → 10x performance degradation

- **Changes disk layout issues:**
  - Optimize for reads, vs. writes & recovery

- **Won’t save much money:**
  - Already have to keep information in memory
  - Example: Facebook caches 75% of data
Does Latency Matter?

- Yes! Latency historically undervalued
- Web applications becoming more data intensive (100’s of storage requests per Web page)
- Low latency enables richer query models, stronger consistency
Is RAMCloud Capacity Sufficient?

- **Facebook:** 200 TB of (non-image) data today

- **Amazon:**
  - Revenues/year: $16B
  - Orders/year: 400M? ($40/order?)
  - Bytes/order: 1000-10000?
  - Order data/year: 0.4-4.0 TB?

- **United Airlines:**
  - Total flights/day: 4000? (30,000 for all airlines in U.S.)
  - Passenger flights/year: 200M?
  - Data/passenger-flight: 1000-10000?
  - Order data/year: 0.2-2.0 TB?

- **Ready today for all online data; media soon**
Data Durability/Availability

- Data must be durable when write RPC returns
- Non-starters:
  - Synchronous disk write (100-1000x too slow)
  - Replicate in other memories (too expensive)
- One possibility: buffered logging

![Diagram showing storage servers with DRAM, disk, and log for asynchronous, batch operations.]
Durability/Availability, cont’d

- Buffered logging supports ~50K writes/sec./server (vs. 1M reads)

- Need fast recovery after crashes:
  - Read 64 GB from disk? 10 minutes
  - Shard backup data across 100’s of servers
  - Reduce recovery time to 1-2 seconds

- Other issues:
  - Power failures
  - Cross-datacenter replication
Other RAMCloud Research Issues

- Low-latency RPCs
- Data model
- Concurrency/consistency model
- Data distribution, scaling
- Automated management
- Multi-tenancy
- Client-server functional distribution
- Node architecture
Status and Plans

- **Project plan:** build *production-quality RAMCloud implementation*
- **Just beginning detailed design/implementation**
- **Current students:**
  - Ryan Stutsman
  - Steve Rumble
  - Aravind Narayanan
- **Possibly room for one first-year student**
  - Must love building real software
  - See me if interested
Many interesting research issues

Exciting combination of scale and latency:
  - 50-500 TBytes
  - 10 microsecond access time

Enable new forms of data-intensive applications
Fiz Overview

- **The problem:**
  - Too hard to develop interactive Web applications
  - Existing frameworks too low-level

- **The solution:**
  - Raise the level of programming: don’t write HTML!
  - Create applications from high-level reusable components

- **Fiz:**
  - Framework for creating components for Web applications
  - Library of built-in components
  - Goal: create community around component set
Questions/Comments
Why not Flash Memory?

- Many candidate technologies besides DRAM
  - Flash (NAND, NOR)
  - PC RAM
  - ...

- DRAM enables lowest latency:
  - 5-10x faster than flash

- Most RAMCloud techniques will apply to other technologies

- Ultimately, choose storage technology based on cost, performance, energy, not volatility
Low-Latency RPCs

Achieving 5-10µs will impact every layer of the system:

- **Must reduce network latency:**
  - Typical today: 10-30 µs/switch, 5 switches each way
  - Arista: 0.9 µs/switch: 9 µs roundtrip
  - Need cut-through routing, congestion mgmt

- **Tailor OS on server side:**
  - Dedicated cores
  - No interrupts?
  - No virtual memory?
Low-Latency RPCs, cont’d

- **Client side: need efficient path through VM**
  - User-level access to network interface?

- **Network protocol stack**
  - TCP too slow (especially with packet loss)
  - Must avoid copies

- **Preliminary experiments:**
  - 10-15 µs roundtrip
  - Direct connection: no switches