Pond: the OceanStore Prototype

CS 6464
Cornell University

Presented by Yeounoh Chung
Motivation / Introduction

• “OceanStore is an Internet-scale, persistent data store”
• “for the first time, one can imagine providing truly durable, self maintaining storage to every computer user.”
• “vision” of highly available, reliable, and persistent, data store utility model
  - Amazon S3 ?!
## Motivation / Introduction

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>• Universal availability</td>
<td>• High availability</td>
</tr>
<tr>
<td>• High durability</td>
<td>• High durability</td>
</tr>
<tr>
<td>• Incremental Scalability</td>
<td>• High scalability</td>
</tr>
<tr>
<td>• Self-maintaining</td>
<td>• Self-maintaining</td>
</tr>
<tr>
<td>• Self-organizing</td>
<td>• Self-organizing</td>
</tr>
<tr>
<td>• Virtualization</td>
<td>• Virtualization</td>
</tr>
<tr>
<td>• Transparency</td>
<td>• Transparency</td>
</tr>
<tr>
<td>• Monthly access fee</td>
<td>• Pay-per-use fee</td>
</tr>
<tr>
<td>• Untrusted infrastructure</td>
<td>• Trusted infrastructure</td>
</tr>
<tr>
<td>• Two-tier system</td>
<td>• Single-tier system</td>
</tr>
</tbody>
</table>
Outline

• Motivation / Introduction
• **System Overview**
• Consistency
• Persistency
• Failure Tolerance
• Implementation
• Performance
• Conclusion
• Related Work
System Overview
System Overview (Data Object)

<table>
<thead>
<tr>
<th>Name</th>
<th>Meaning</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BGUID</td>
<td>block GUID</td>
<td>secure hash of a block of data</td>
</tr>
<tr>
<td>VGUID</td>
<td>version GUID</td>
<td>BGUID of the root block of a version</td>
</tr>
<tr>
<td>AGUID</td>
<td>active GUID</td>
<td>names a complete stream of versions</td>
</tr>
</tbody>
</table>
System Overview (Update Model)

- Updates are applied atomically
- An array of actions guarded by a predicate
- No explicit locks
- Application-specific consistency
  - e.g. database, mailbox
System Overview (Tapestry)

- Scalable overlay network, built on TCP/IP
- Performs DOLR based on GUID
  - Virtualization
  - Location independence
- Locality aware
- Self-organizing
- Self-maintaining
System Overview (Primary Rep.)

• Each data object is assigned an inner-ring
• Apply updates and create new versions
• Byzantine fault-tolerant
• Ability to change inner-ring servers any time
  - public key cryptography, proactive threshold signature, Tapestry
• Responsible party
System Overview (Achi. Storage)

- Erasure codes are more space efficient

- Fragments are distributed uniformly among archival storage servers
  - BGUID, n_frag

- Pond uses Cauchy Reed-Solomon code
System Overview (Caching)

- Promiscuous caching
- Whole-block caching
- Host caches the read block and publishes its possession in Tapestry
- Pond uses LRU
- Use Heartbeat to get the most recent copy
System Overview (Diss. Tree)
Outline

• Motivation / Introduction
• System Overview
• Consistency
• Persistency
• Failure Tolerance
• Implementation
• Performance
• Conclusion
• Related Work
Consistency (Primary Replica)

- Read-only blocks
- Application-specific consistency
- Primary-copy replication
  - heartbeat $<\text{AGUID}, \text{VGUID}, t, v_{\text{seq}}>$
Persistency (Archival Storage)

• Archival storage
• Aggressive replication
• Monitoring
  - Introspection
• Replacement
  - Tapestry
Failure Tolerance (Everybody)

• All newly created blocks are encoded and stored in Archival servers
• Aggressive replication
• Byzantine agreement protocol for inner-ring
• Responsible Party
  - single point of failure?
  - scalability?
Outline

• Motivation / Introduction
• System Overview
• Consistency
• Persistency
• Failure Tolerance
  • Implementation
• Performance
• Conclusion
• Related Work
Implementation

- Built in Java, atop SEDA
- Major subsystems are functional
  - self-organizing Tapestry
  - primary replica with Byzantine agreement
  - self-organizing dissemination tree
  - erasure-coding archive
  - application interface: NFS, IMAP/SMTP, HTTP
Implementation
Outline

• Motivation / Introduction
• System Overview
• Consistency
• Persistency
• Failure Tolerance
• Implementation
  • Performance
• Conclusion
• Related Work
Performance

- Update performance
- Dissemination tree performance
- Archival retrieval performance
- The Andrew Benchmark
Performance (test beds)

• Local cluster
  - 42 machines at Berkeley
  - 2x 1.0 GHz CPU, 1.5 GB SDRAM,
    2x 36 GB hard drives
  - gigabit Ethernet adaptor and switch

• PlanetLab
  - 101 nodes across 43 sites
  - 1.2 GHz, 1 GB memory
Performance (update)

<table>
<thead>
<tr>
<th>Key Size</th>
<th>Update Size</th>
<th>Archive</th>
<th>Update Latency (ms)</th>
<th>Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>5%</td>
<td>Median</td>
</tr>
<tr>
<td>512</td>
<td>4 kB</td>
<td>off</td>
<td>36</td>
<td>37</td>
</tr>
<tr>
<td></td>
<td></td>
<td>on</td>
<td>39</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>2 MB</td>
<td>off</td>
<td>494</td>
<td>513</td>
</tr>
<tr>
<td></td>
<td></td>
<td>on</td>
<td>1037</td>
<td>1086</td>
</tr>
<tr>
<td>1024</td>
<td>4 kB</td>
<td>off</td>
<td>94</td>
<td>95</td>
</tr>
<tr>
<td></td>
<td></td>
<td>on</td>
<td>98</td>
<td>99</td>
</tr>
<tr>
<td></td>
<td>2 MB</td>
<td>off</td>
<td>557</td>
<td>572</td>
</tr>
<tr>
<td></td>
<td></td>
<td>on</td>
<td>1098</td>
<td>1150</td>
</tr>
</tbody>
</table>

Table 2: Results of the Latency Microbenchmark in the Local Area. All nodes are hosted on the cluster. Ping latency between nodes in the cluster is 0.2 ms. We run with the archive enabled and disabled while varying the update size and key length.

Table 3: Latency Breakdown of an Update. The majority of the time in a small update performed on the cluster is spent computing the threshold signature share over the result. With larger updates, the time to apply and archive the update dominates signature time.
Performance (update)

Table 4: Results of the Latency Microbenchmark Run in the Wide Area. All tests were run with the archive enabled using 1024-bit keys. "Avg. Ping" is the average ping time in milliseconds from the client machine to each of the inner ring servers. UCSD is the University of California at San Diego.

Table 5: Throughput in the Wide Area. The throughput for a distributed ring is limited by the wide-area bandwidth. All tests are run with the archive on and 1024-bit keys.

Figure 5: Throughput in the Local Area. This graph shows the update throughput in terms of both operations per second (left axis) and bytes per second (right axis) as a function of update size. While the ops/s number falls off quickly with update size, throughput in bytes per second continues to increase. All experiments are run with 1024-bit keys. The data shown is the average of three trials, and the standard deviation for all points is less than 3% of the mean.
Performance (archival)

Figure 6: Latency to Read Objects from the Archive. The latency to read data from the archive depends on the latency to retrieve enough fragments for reconstruction.
Performance (dissemination tree)

Figure 7: Results of the Stream Benchmark. The graph shows the percentage of bytes sent over links of different latency as the number of replicas varies.

<table>
<thead>
<tr>
<th>Tokens Passed Using</th>
<th>Latency per Tag (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>OceanStore</td>
<td>329</td>
</tr>
<tr>
<td>Tapestry</td>
<td>104</td>
</tr>
<tr>
<td>TCP/IP</td>
<td>73</td>
</tr>
</tbody>
</table>

Table 6: Results of the Tag Microbenchmark. Each experiment was run at least three times, and the standard deviation across experiments was less than 10% of the mean. All experiments are run using 1024-bit keys and with the archive disabled.
Performance (Andrew benchmark)

<table>
<thead>
<tr>
<th>Phase</th>
<th>LAN</th>
<th>WAN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Linux</td>
<td>OceanStore</td>
</tr>
<tr>
<td>I</td>
<td>0.0</td>
<td>1.9</td>
</tr>
<tr>
<td>II</td>
<td>0.3</td>
<td>11.0</td>
</tr>
<tr>
<td>III</td>
<td>1.1</td>
<td>1.8</td>
</tr>
<tr>
<td>IV</td>
<td>0.5</td>
<td>1.5</td>
</tr>
<tr>
<td>V</td>
<td>2.6</td>
<td>21.0</td>
</tr>
<tr>
<td>Total</td>
<td>4.5</td>
<td>37.2</td>
</tr>
</tbody>
</table>

Table 7: Results of the Andrew Benchmark. All experiments are run with the archive disabled using 512 or 1024-bit keys, as indicated by the column headers. Times are in seconds, and each data point is an average over at least three trials. The standard deviation for all points was less than 7.5% of the mean.
Conclusion

- Pond is a working subset of the vision
- Promising in WAN
- Threshold signatures, erasure-coded archival are expensive
- Pond is fault tolerant system, but it is not tested with any failed node
- Any thoughts?
Related Work

- FarSite
- ITTC, COCA
- PAST, CFS, IVY
- Pangaea