Long Term Durability with Seagull

Hakim Weatherspoon
(Joint work with Jeremy Stribling and OceanStore group)
University of California, Berkeley
Questions

• Given: wide-area durable storage is complex.
• What is required to convince you to place your data in this system (or a like system)?
  - How do you know that it works?
  - How efficient is it?
    • BW, latency, throughput.
  - Do you trust it?
    • Who do you sue.
  - How much does it cost?
    • BW, Storage, Money.
  - How reliable is it?
    • MTTDL/Fractions of Blocks Lost Per Year (FBLPY).
Relevance to ROC/Sahara/OceanStore

• Components of Communication
  – Heart beating, Fault tolerant routing, etc.

• Correlation
  – Monitoring, Human input, etc.

• Detection
  – Distributed vs. Global.

• Repair
  – Triggered vs. Continuous

• Reliability
  – Continuous restart of communication links, etc.
  – FBLPY (MTTDL).
Outline

• Overview
• Experience.
• Lessons learned
• Required Components
• Future Directions
Deployment

- Planet Lab global network
  - 98 machines at 42 institutions, in North America, Europe, Australia.
  - 1.26Ghz PIII (1GB RAM), 1.8Ghz PIV (2GB RAM)
  - North American machines (2/3) on Internet2
Deployment

- Deployed storage system in November of 2002.
  - ~50 physical machines.
  - 100 virtual nodes.
    - 3 clients, 93 storage serves, 1 archiver, 1 monitor.
  - Support OceanStore API
    - NFS, IMAP, etc.
  - Fault injection.
  - Fault detection and repair.
Path of an OceanStore Update
Path of a Storage Update

• **Erasure codes**
  - redundancy without overhead of strict replication
  - produce $n$ fragments, where any $m$ is sufficient to reconstruct data. If $m < n$, rate $r = m/n$. Storage overhead is $1/r$. 

![Diagram of Path of a Storage Update](image-url)
Durability

• Fraction of Blocks Lost Per Year (FBLPY)*
  - $r = \frac{1}{4}$, erasure-encoded block. (e.g. $m = 16$, $n = 64$)
  - Increasing number of fragments, increases durability of block
    • Same storage cost and repair time.
  - $n = 4$ fragment case is equivalent to replication on four servers.

**Naming and Verification Algorithm**

- Use cryptographically secure hash algorithm to detect corrupted fragments.

- **Verification Tree:**
  - \( n \) is the number of fragments.
  - store \( \log(n) + 1 \) hashes with each fragment.
  - Total of \( n \cdot (\log(n) + 1) \) hashes.

- Top hash is a *block GUID* (*B-GUID*).
  - Fragments and blocks are self-verifying

---

**Diagram:**

```
B-GUID
  /   \
Hd   H14
    /  \
  H12 /    H34
    /  \
  F1   F2   F3   F4
```

**Table:**

<table>
<thead>
<tr>
<th>Fragment 1:</th>
<th>H2</th>
<th>H34</th>
<th>Hd</th>
<th>F1 - fragment data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fragment 2:</td>
<td>H1</td>
<td>H34</td>
<td>Hd</td>
<td>F2 - fragment data</td>
</tr>
<tr>
<td>Fragment 3:</td>
<td>H4</td>
<td>H12</td>
<td>Hd</td>
<td>F3 - fragment data</td>
</tr>
<tr>
<td>Fragment 4:</td>
<td>H3</td>
<td>H12</td>
<td>Hd</td>
<td>F4 - fragment data</td>
</tr>
</tbody>
</table>

**Data:**

```
H14
data
```
Enabling Technology
Tapestry DOLR

GUID
Fragments
Outline

• Overview
• Experience.
• Lessons learned
• Required Components
• Future Directions
Lessons Learned

• Need ability to route to an object if it exists.
  - Hindrance to a long running process.
    • Robustness to node and network failures.
• Need tools to diagnosis current state of network.
• Need ability to run without inner ring.
• Need monitor, detection, repair mechanisms.
  - Avoid correlated failures.
  - Quickly and efficiently detect faults.
  - Efficiently repair faults.
• Need to perform maintenance in distributed fashion.
Outline

• Overview
• Experience
• Lessons learned
• Required Components
• Future Directions
Monitor: Low Failure Correlation Dissemination

- **Model Builder.**
  - Various sources.
  - Model failure correlation.

- **Set Creator.**
  - Queries random nodes.
  - **Dissemination Sets.**
    - Storage servers that fail with low correlation.

- **Disseminator.**
  - Sends fragments to members of set.
Monitor: Low Failure Correlation Dissemination

• Sanity Check
  - Monitored 1909 Web Servers

• Future
  - Simple Network Management Protocol (SNMP)
    • standard protocol for monitoring.
    • Query network components for information about their configuration, activity, errors, etc.
  - Define an OceanStore/Tapestry MIB.
    • Management Information Base (MIB)
Detection

• **Goal**
  - Maintain routing and object state using minimal resources.
    - e.g. less than 1% of bandwidth and cpu cycles.

• **Server Heartbeat's**
  - “Keep-alive” beacon along each forward link.
  - Increasing period (decreasing frequency) with the routing level.

• **Data-Driven Server Heartbeat's**
  - “Keep-alive” Multicast to all ancestors with an object pointer that points to us.
  - Multicast with increasing radius.
Detection

• Republish/Object Heartbeats
  - Object Heartbeat (Republish).
  - Heartbeat period increasing with distance
    • (i.e. Heartbeat frequency decreases with distance)
    • Distance is number of application-level hops

• Distributed Sweep
  - Request object from storage servers.
  - Sweep period increasing with distance
    • (i.e. Sweep frequency decreases with distance)

• Global Sweep (Responsible Party/Client)
  - Request object from storage servers at regular intervals.
  - Period constant. (i.e. frequency constant)
Efficient Repair

- Exploit DOLR’s distributed information and locality.
- Efficient detection and then reconstruction of fragments.
Efficient Repair

- Continuous vs. Triggered
- Continuous (Responsible Party/Client)
  - Request object from storage servers at regular intervals.
  - Period constant. (i.e. frequency constant)
- Triggered (Infrastructure)
  - FBLPY proportional to MTTR/MTTF.
    - Disks: MTTF > 100,000 hours.
    - Gnutella: MTTF = 2.3 hours. (Median 40 minutes).
  - Local vs. Remote Repair.
    - Local stub looks like durable disk.
Efficient Repair

• Reliability vs. Cost vs. Trust.

- Physical Links
- Overlay Edges
Outline

• Overview
• Experience.
• Lessons learned
• Required Components
• Future Directions
Future Directions

- Redundancy, Detection, Repair, Monitoring.
  - None alone is sufficient.
  - Only reliable as weakest link.

- Verify system?
  - System may always be in inconsistent state.
  - How do you know ...
    - Data exists?
    - Data will exist tomorrow?

- Applications/Usage in the long-term.
  - NSF, IMAP, rsynch (back-up), InternetArchive, etc.