Bit Flips in Distributed Systems

- Distributed systems typically tolerate crash faults
- Arbitrary faults, e.g., bit flips, occur surprisingly often
- Can disrupt large services such as Amazon S3

Problem:
- Cope with bit flips
- Without expensive Byzantine-fault tolerance

A Framework for Distributed Protocols

- Automatically improves the fault coverage of protocols for crash-fault model
- Uses encoding compiler by Schiffel et al.
- Encoded protocols tolerate non-malicious arbitrary faults, e.g., bit flips and hardware design faults

Virtualizing Non-malicious Arbitrary Faults

- Errors within shaded area abort process
- Propagated errors invalidate encoding
- Invalid encoding treated as omission/performance failures

Open Questions – Future Work

Current work:
- Overhead of encoding atomic broadcast protocols
- Stronger encoding: use ANB-encoding
- Fault injection: how dangerous are bit flips?

Encoding in a Nutshell

- Each 32-bit value encoded in a 64-bit value
- Domain divided in
  - a few valid values (code words)
  - many invalid values (non-code words)
- All operations transformed to support encoding

Overhead Analysis: AN-encoded Elections

- Election time overhead less than 2%
- Doubled CPU overhead, 10 ms heart-beat period
- Absolute CPU utilization less than 5%
- Low cost for elections

Assumptions and Fault Models

- No process identifies itself as another process
- No reverse engineering of encoding
- Non-synchronized local clocks available
- Clocks do not fail

Encoding flavors:

<table>
<thead>
<tr>
<th>Error</th>
<th>no encoding</th>
<th>AN</th>
<th>ANB</th>
<th>ANBDmem</th>
</tr>
</thead>
<tbody>
<tr>
<td>crash</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>corruption on transmission</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>faulty operation</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>data corruption</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>control flow</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>address bus on read</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>address bus on write</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>