HY590.45
Modern Topics in Scalable Storage Systems

Kostas Magoutis
magoutis@csd.uoc.gr
http://www.csd.uoc.gr/~hy590-45
Support for backup

- **Snapshots**
  - Petal can quickly create an exact copy of a virtual disk at a specified point in time by using copy-on-write techniques.
  - A snapshot is like any other vdisk, but cannot be modified.
  - VDir: vdiskID → <global-map-identifier, epoch-number>

- **Crash-consistent snapshot**
  - Similar to disk image left after an application crash.
Reconfiguration

• Dynamic change in vdisk redundancy, # of servers

• How is it performed
  – Create new GMap with desired redundancy, server mapping
  – Change VDir entries that refer to old GMap to new one
  – Redistribute data to the servers according to new GMap, requiring substantial amounts of network and disk traffic
  – Read requests will be tried on new GMap first, then the old GMap if the translation has not yet been transferred
  – Writes are always performed on the new GMap

• Improve efficiency via fencing
Chained de-clustering
Data access and recovery

- How are reads performed
- How are writes performed
Data access and recovery

Block (64KB)

Read protocol
1. Try primary; if live, get read lock
2. If down, try secondary; if live, get read lock
3. Return block contents, release read lock

Write protocol
1. Contact primary
2. If alive, mark block busy in stable storage
3. Primary apply request locally and simultaneously send write to secondary
4. When both complete, clear busy bit, respond to client
5. If primary crashes during request, busy bit is used to recover later on
6. If primary dead, start from secondary
7. (Secondary checks primary indeed crashed)

If primary or secondary is down, on live node:
1. Mark data as *stale* before writing to disk
2. During recovery, make replicas consistent by exchanging *dirty-region log*
a read will see a state at least as recent as that produced by the most recently completed write that completed before the read started
if some reader sees the results of a particular write, then any reader that starts after that reader finishes will also see a result at least that recent.
Recovery from primary crash

write(b,6)

write

write(b,6)

primary

Restart

check status

reconcile

clear busy bit

ack

write

busy

write

busy

write

ack

write
Recovery from primary crash

- **Primary**
  - `write(b,6)`
  - `write(b,5)`
  - `set stale`
  - `clear busy bit`

- **Secondary**
  - `write(b,5)`
  - `primary dead??`
  - `set stale`
  - `write`
  - `ack`

- **Replay write**
  - `reconcile`
  - `get dirty-region log`
  - `write`
  - `ack`
  - `write(b,6)`
  - `write`
  - `busy`
  - `crash`
Recovery from secondary crash

write(b,6) -> Primary
write(b,6) -> Secondary
set stale
write
clear busy bit
ack
backup dead??
crash
write
write
busy
write
clear busy bit
ack
get dirty-region log
reconcile
replay write
write(b,5)
write
write
ack
ack
write
clear busy bit
restart
Petal cannot handle partitions

write(b,5)

partition

write(b,6)

set busy

backup dead??

set stale

write

clear busy bit

ack

write

primary dead??

set stale

write

ack
Prototype
Latency, throughput results

<table>
<thead>
<tr>
<th>Client Request Latency (ms)</th>
<th>Local Disk</th>
<th>Petal RZ29 Log</th>
<th>Petal NVRAM Log</th>
</tr>
</thead>
<tbody>
<tr>
<td>512 byte Read</td>
<td>9</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>8 Kbyte Read</td>
<td>11</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>64 Kbyte Read</td>
<td>21</td>
<td>28</td>
<td>28</td>
</tr>
<tr>
<td>512 byte Write</td>
<td>10</td>
<td>19</td>
<td>12</td>
</tr>
<tr>
<td>8 Kbyte Write</td>
<td>12</td>
<td>22</td>
<td>16</td>
</tr>
<tr>
<td>64 Kbyte Write</td>
<td>20</td>
<td>40</td>
<td>33</td>
</tr>
</tbody>
</table>

Table 1: Latency of a Chained-Declustered Virtual Disk

<table>
<thead>
<tr>
<th>Aggregate Throughput (RZ29 Log)</th>
<th>Normal</th>
<th>Failed</th>
<th>% of Normal</th>
</tr>
</thead>
<tbody>
<tr>
<td>512 byte Read</td>
<td>3150 req/s</td>
<td>2310 req/s</td>
<td>73 %</td>
</tr>
<tr>
<td>8 Kbyte Read</td>
<td>20 Mbytes/s</td>
<td>14.6 Mbytes/s</td>
<td>73 %</td>
</tr>
<tr>
<td>64 Kbyte Read</td>
<td>43.1 Mbytes/s</td>
<td>33.7 Mbytes/s</td>
<td>78 %</td>
</tr>
<tr>
<td>512 byte Write</td>
<td>1030 req/s</td>
<td>1055 req/s</td>
<td>102 %</td>
</tr>
<tr>
<td>8 Kbyte Write</td>
<td>6.6 Mbytes/s</td>
<td>6.6 Mbytes/s</td>
<td>100 %</td>
</tr>
<tr>
<td>64 Kbyte Write</td>
<td>12.3 Mbytes/s</td>
<td>12.5 Mbytes/s</td>
<td>101 %</td>
</tr>
</tbody>
</table>

Table 2: Normal and Failed Throughput of a Chained-Declustered Virtual Disk
Scaling

![Graph showing relative throughput vs number of servers for different data sizes (512 byte Read, 8 Kbyte Read, 64 Kbyte Read, 512 byte Write, 8 Kbyte Write, 64 Kbyte Write).](image)