Using Multi-Drive Fusion to Scale NVMe Performance

Jinling Chen
chenjinling@derastorage.com
Outline

- NVMe and RAID
- RAID write penalty, write hole
- Hardware RAID
- Software RAID
- Multi-Drive Fusion
- Pros and cons of MDF
- MDF mesh
NVMe: concerns of deploying multiple drives

- To scale performance, as linearly as possible
- To protect data against faulty drive(s)
RAID problem: write penalty

RAID5:
To write to a sector, have to do 2 reads + 2 writes
Write Penalty = 4
### RAID problem: write hole

RAID5 write hole, or double faults, may lead to data corruption eventually.

<table>
<thead>
<tr>
<th></th>
<th>D0</th>
<th>D1</th>
<th>D2</th>
<th>D3</th>
<th>D4</th>
<th>D5</th>
<th>D6</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Write</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D0</td>
<td>D1</td>
<td><strong>D2’</strong></td>
<td>D3</td>
<td>D4</td>
<td>D5</td>
<td>D6</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>D0</td>
<td>D1</td>
<td><strong>D2’</strong></td>
<td>D3</td>
<td>D4</td>
<td>D5</td>
<td>D6</td>
<td>P’</td>
<td></td>
</tr>
</tbody>
</table>

**Power loss hits here**

**Remedy options:**
- A) Write Journaling
- B) Backup Battery against unexpected power loss

**Higher write amplification**

1 more fault source. Adding complexity and cost
Hardware RAID Controller

- Problems:
  - A new single point of faults
  - Adds latency
  - Upward port throttles b/w
  - Hard to contain multiple NVMe drives
Software RAID

- Overhead
  - Host CPU cycles
  - Memory footprint
  - Bus traffic
  - Sync penalties
- Problem as a boot device
Multi-Drive Fusion

- MDF-enabled drives are configured into an autonomous NVMe pool

- Each MDF controller does:
  - Data forwarding to others
  - Smart data placement
  - Localized XOR generating
  - In-drive write journaling primitives
MDF: write flow

Host initiates write to D2 position in Drive2

Drive2 does XOR

MDF forwarding
MDF: write flow (cont.)

- RAID5 write penalty drops from 4 to 2
  - From host view, to 1
- RAID5 write hole: eliminated
  - Drive2 & Drive7 turn writes of D2’ & P’ into a single transaction, so no more degrading the stripe

- The key: cross-drive forwarding
MDF: recovery flow

- Data is recovered by chained XOR within all healthy drives, and finally forwarded to the renewed faulty drive
- XOR can be pipelined across all healthy drives
- Continuously serve host I/O at front side
MDF more advantages

• Releases the host:
  • Host does not read/write parity blocks
  • Host does not compute parity codes
  • Less host CPU cycles and memory footprint, and bus cycles
  • So a CPU-light NVMe box is feasible
MDF brings more possibilities

• Balance workloads globally, including wearing
• Reduce in-drive redundancy
• Global FTL reducing unnecessary mappings
• More: MDF object service, file service
MDF disadvantages

• Extra traffic to PCIe domain
  • Some packets to convey control info across drives
  • Data traffic incurred by data forwarding

• A dedicated interconnect may cure this, MDF Mesh
MDF Mesh

Diagram of MDF Mesh with components such as NIC, CPU, PCIe Switch, and NVMe connected with MDF Link, PCIe, and UPI.
MDF Mesh (cont.)

- A dedicated interconnect for a MDF pool
- Simpler protocol, higher energy efficiency
- Simpler and fault-tolerant topology
- Offloading traffic from host PCIe domain
- More scalable than PCIe complex
Thanks

Welcome to Booth 523