Garbage Collection in LogFS

Jörn Engel, Dirk Bolte & Robert Mertens

Lazybastard.org, IBM & University of Osnabrück

January 16, 2007
Flash properties

For all flashes:
- Medium split into eraseblocks (16-128 KiB)
- Eraseblocks can be written in any order
- Eraseblocks must be erased before being written
- Eraseblocks can be partially written

For some flashes:
- Eraseblock split into pages (8-2048 Bytes)
- Partial pages cannot get written
- Writes within eraseblocks must happen in order
LogFS device abstraction

For all media:
- Medium split into segments
- Segments can be written in any order
- Segments must be erased before being written
- Segments can be partially written

For all media:
- Eraseblock split into blocks
- Partial blocks cannot get written
- Writes within segments must happen in order
Mapping flash to LogFS

- One or more eraseblocks form a segment
- One or more flash pages form a block
Three storage areas

- Superblock (1 Segment)
- Journal (2-8 Segments)
- Object store

<table>
<thead>
<tr>
<th>Superblock</th>
</tr>
</thead>
<tbody>
<tr>
<td>Journal 0</td>
</tr>
<tr>
<td>Journal 1</td>
</tr>
<tr>
<td>Object Store 0</td>
</tr>
<tr>
<td>Object Store N</td>
</tr>
</tbody>
</table>
Object store

- Object store split into segments
- Segments split into blocks
- Last block of each segment contains summary

<table>
<thead>
<tr>
<th>Superblock</th>
</tr>
</thead>
<tbody>
<tr>
<td>Journal 0</td>
</tr>
<tr>
<td>Journal 1</td>
</tr>
<tr>
<td>Object Store 0</td>
</tr>
<tr>
<td>Object Store N</td>
</tr>
</tbody>
</table>

| Data | Data | Data | Data | Data | Data | Data | Data | Sum |
Segment summary

Summary contains for each block:
- Inode number (ino)
- Logical position in a file (pos)
- Physical offset of the block (ofs)

Plus some segment-global information:
- Erase count
- Write time
- ...

Block verification

- No bitfields to track block usage
- A block is used iff an inode points to it
- Summary contains (ino,pos) pair as a back pointer
Regular Unix format with inode, indirect blocks and data blocks:

- **Regular file/directory/**...
- **Inode**
- **Level 0**
  - 1x indirect blocks
  - Data
- **Level 1**
  - 1x indirect blocks
  - Data
  - 1x indirect blocks
  - Data
- **Level 2**
  - 2x indirect blocks
  - Data
  - 2x indirect blocks
  - Data
- **Level N**
  - Nx indirect blocks
  - Data
Inode file

- No reserved areas for inodes
- Inodes stored in inode file (i-file)
- Ifile’s inode stored in journal
Inode file

Inode file

Level N

Nx indirect blocks

Level 2

2x indirect blocks

Level 1

1x indirect blocks

Level 0

Inode

Regular file/directory/...

Level N

Nx indirect blocks

Level 2

2x indirect blocks

Level 1

1x indirect blocks

Level 0

Data

Jörn Engel, Dirk Bolte & Robert Mertens

Garbage Collection in LogFS
Writes

- Flash must be erased before being written
- Eraseblocks too large to be practical
- Writes cannot happen in-place
- Solution: Wandering tree
The red block is overwritten with new data ...

Regular file/directory/...

Inode

1x indirect blocks

Data Data Data Data Data Data

2x indirect blocks

1x indirect blocks

1x indirect blocks

Data

2x indirect blocks

1x indirect blocks

1x indirect blocks

Data

Nx indirect blocks

Level N

Level 2

Level 1

Level 0

Data
... and has to be written to a different location.

```
Regular file/directory/...

Inode
1x indirect blocks
2x indirect blocks
1x indirect blocks
1x indirect blocks
1x indirect blocks

Data Data Data Data Data Data Data Data Data...

Level 0
Level 1
Level 2
Level N

Nx indirect blocks
2x indirect blocks
2x indirect blocks
```

Jörn Engel, Dirk Bolte & Robert Mertens

Garbage Collection in LogFS
Since the indirect block still points to the old location ...
Writes

... it has to be changed as well ...

Regular file/directory/...

Inode

1x indirect blocks

2x indirect blocks

Nx indirect blocks

Data

Level N

Level 2

Level 1

Level 0

Data

Data

Data

Data

Data

Data

Data

Data

Data

Data

Data

Data

Data

Data

Data

...
... along with all other indirect blocks further up.
Why Garbage Collection?

- Writes happen out-of-place
- Blocks obsoleted by writes
- Near-empty segments are useless
Why Garbage Collection?

A single valid block in a segment prevents reuse
Simple Garbage Collection

- GC has to move each valid block away ...

```
+---+---+---+---+---+---+---+---+
| Data | Data | Data | Data | Data | Data | Data | Sum |
+---+---+---+---+---+---+---+---+
| Data |   |   |   |   |   |   | Sum |
+---+---+---+---+---+---+---+---+
```

Jörn Engel, Dirk Bolte & Robert Mertens
Simple Garbage Collection

... thereby obsoleting the old block ...

<table>
<thead>
<tr>
<th>Data</th>
<th>Data</th>
<th>Data</th>
<th>Data</th>
<th>Data</th>
<th>Data</th>
<th>Sum</th>
</tr>
</thead>
</table>

| Data | Data | Data | Data | Data | Data | Data | Data | Sum |

| Data | Data | Data | Data | Data | Data | Data | Data | Sum |

Jörn Engel, Dirk Bolte & Robert Mertens
Garbage Collection in LogFS
Simple Garbage Collection

... so it can be deleted.
Simple Garbage Collection

- GC should free more segments than it uses.
Simple Garbage Collection

- GC *must* free more segments than it uses!
Simple Garbage Collection

- GC *MUST* free more segments than it uses!!!
- GC *MUST* free more segments than it uses!!!
- GC *MUST* free more segments than it uses!!!
- GC *MUST* free more segments than it uses!!!

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>Sum</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>D</td>
<td>E</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F</td>
<td>Sum</td>
</tr>
</tbody>
</table>

| A' | B' | C' | D' | E' | F' |

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>Sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>E</td>
<td>F</td>
<td>Sum</td>
</tr>
<tr>
<td>A'</td>
<td>B'</td>
<td>C'</td>
<td>D'</td>
</tr>
</tbody>
</table>
Garbage Collection with a Wandering Tree

Wandering tree?
Garbage Collection with a Wandering Tree

Didn't we have a wandering tree?
Garbage Collection with a Wandering Tree

Oh dear!

Before: A0 B0 A1 A2 A3 C0 Sum

After: A0' A1' A2' A3' C0 Sum
Garbage Collection with a Wandering Tree

Mustn’t GC free more segments than it uses?
Garbage Collection with a Wandering Tree

Oooh!
Introducing Levels

Definition: Two nodes are on the same level, if their distance from the root node is identical.

Inode

Regular file/directory/...

Level N

Level 2

Level 1

Level 0

Data Data Data Data Data Data Data Data Data Data Data Data

2x indirect blocks 2x indirect blocks

1x indirect blocks 1x indirect blocks

Nx indirect blocks

Jörn Engel, Dirk Bolte & Robert Mertens

Garbage Collection in LogFS
Garbage Collection with Levels

- Blocks on the same level are written to the same segments
- Blocks on different levels are written to different segments
Garbage Collection with Levels

Jörn Engel, Dirk Bolte & Robert Mertens
Garbage Collection with Levels

Aaaah!
Thanks!

Arnd Bergmann
Martin Schwidefsky
Dirk Bolte
Robert Mertens
You, the audience
Questions?
Suggestions?
Sponsors?