LECTURE - 32

Lecture Outline

- Log-Structured File System (LFS) [RO91]
- RAID
-
- ***Scribe for today?***

Log-Structured File System

- Technological under-pinnings:
  - Disk I/O becoming bottleneck since CPUs are getting faster
  - Disk I/O dominated by writes, since reads mostly served by main memory caching
- Characteristics of application workloads:
  - Lots of accesses to small files
  - Random disk I/Os
  - Synchronous meta-data update in FFS => slow
  - FFS could use only about 5% disk bandwidth

The Log as the Structure

- Large asynchronous writes (0.5-1MB) to the end of the log
- How to retrieve information from the log?
  - Sequential search would be too slow
- I-node structure is same as in FFS
- Getting to i-node given the i-node number uses *i-node map* (level of indirection)
- I-node map is small enough to be in memory

Free Space Management

- What if log fills up disk?
  - Threading vs. copying
- Intermediate solution: segments
  - Thread across segments
  - Copy within segments
- Segment cleaning: copy live-data out of segment, to create free segments
  - Segment with long-lived copy ==> can ignore while *cleaning*

Segment Cleaning

- Read a set of segments
- Copy live data to new segments, create free segments
- Need to identify:
    - Which blocks are live
    - Which block belongs to which file
    - *Segment summary information*
    - Notion of file/inode version

Segment Cleaning Policies

- When should the cleaning be done?
  - Periodically; after threshold disk utilization
- How many segments to clean at a time?
  - Fixed; until achieving some number of clean segments
- Which segments to clean?
  - Most fragmented; having the least utilization
- How should the blocks be grouped when writing out?
  - All files in a dir in one place; age sort

Crash Recovery

- Checkpoint
  - Checkpoint region is fixed!
- What to checkpoint?
  - I-node map blocks, segment usage table, pointer to last segment written
- Roll-forward
  - Read from last segment onwards
  - Update i-node map, segment usage table
  - Directory operation log, for consistency between directory entries and i-nodes

RAID

- Raid-1: Mirroring
- Raid-2: Hamming code ECC
- Raid-3: Bit-level parity
- Raid-4: Block-level parity
- Raid-5: Block-level distributed parity