Slide 1

Case study: ext3 FS

1

Slide 2

Brief Journaling Intro
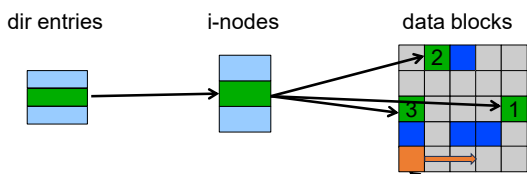
dir entries   i-nodes   data blocks

- Example: deleting a file
  1. Remove the directory entry
  2. Mark the i-node as free
  3. Mark disk blocks as free

2

Slide 3

Brief Journaling Intro

dir entries   i-nodes   data blocks

1. Remove the directory entry
2. Mark the i-node as free
3. Mark disk blocks as free

1. Write to journal
2. Perform updates
3. Remove journal entry

3

Slide 4

The ext3 file system

- Design goals
  - Add journaling capability to the ext2 FS
  - Backward and forward compatibility with ext2
    - Existing ext2 partitions can be mounted as ext3
  - Leverage the proven ext2 performance
  - Reuse most of the ext2 code base
  - Reuse ext2 tools, including e2fsck

4

Slide 5

The ext3 journal

Option1: Journal FS data structure updates

- Example:
  - **Start transaction**
  - Delete dir entry
  - Delete i-node
  - Release blocks 32, 17, 60
  - **End transaction**

Option2: Journal disk block updates

- Example:
  - **Start transaction**
  - Update block #n1 *(contains the dir entry)*
  - Update block #n2 *(i-node allocation bitmap)*
  - Update block #n3 *(data block allocation bitmap)*
  - **Add transaction**

Question: which approach is better?

5

Slide 6

The ext3 journal

Option1: Journal FS data structure updates

- ✓ Efficient use of journal space; hence faster journaling
- ✗ Individual updates are applied separately
- ✗ The journaling layer must understand FS semantics

**Option2: Journal disk block updates**

- ✗ Even a small update adds a whole block to the journal
- ✓ Multiple updates to the same block can be aggregated into a single update
- ✓ The journaling layer is FS-independent (easier to implement)

Ext3 implements Option 2
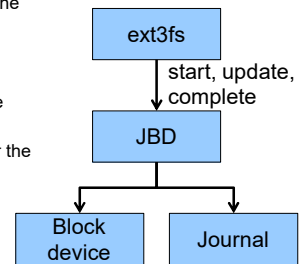
6

Journaling Block Device (JBD)

- The ext3 journaling layer is called Journaling Block Device (JBD)
- JBD interface
  - Start a new transaction
  - Update a disk block as part of a transaction
  - Complete a transaction
    - Completed transactions are buffered in RAM

```
        ext3fs
          |  start, update,
          ↓  complete
         JBD
        /     \
   Block       Journal
   device
```
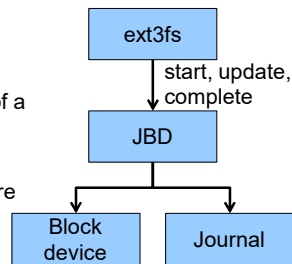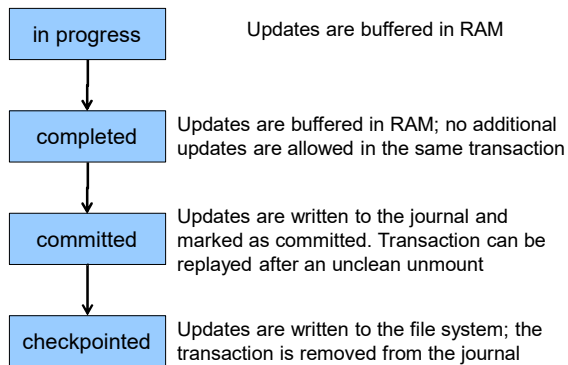
7

Journaling Block Device (JBD)

- JBD interface (continued)
  - Commit: write transaction data to the journal (persistent storage)
    - Multiple FS transactions are committed in one go
  - Checkpoint: flush the journal to the disk
    - Used when the journal is full or the FS is being unmounted

```
        ext3fs
          |  start, update,
          ↓  complete
         JBD
        /     \
   Block       Journal
   device
```

8

Transaction lifecycle

```
in progress        Updates are buffered in RAM
    ↓
completed          Updates are buffered in RAM; no additional
                   updates are allowed in the same transaction
    ↓
committed          Updates are written to the journal and
                   marked as committed. Transaction can be
                   replayed after an unclean unmount
    ↓
checkpointed       Updates are written to the file system; the
                   transaction is removed from the journal
```

9

Journaling modes

- Ext3 supports two journaling modes
  - Metadata+data
    - Enforces atomicity of all FS operations
  - Metadata journaling
    - Metadata is journalled
    - Data blocks are written directly to the disk
    - Improves performance
    - Enforces file system integrity
    - Does not enforce atomicity of `write`'s
      - New file content can be stale blocks

10

JBD

- JBD can keep the journal on a block device or in a file
  - Enables compatibility with ext2 (the journal is just a normal file)
- JBD is independent of ext3-specific data structures
  - Separation of concerns
    - The FS maintains on-disk data and metadata
    - JBD takes care of journaling
  - Code reuse
    - JBD can be used by any other FS that requires journaling

11