Recon:
Verifying File System Consistency at Runtime

Angela Demke Brown
with Daniel Fryer, Jack Sun,
Rahat Mahmood, Shaun Benjamin, Tim Cheng
and Dr. Ashvin Goel

University of Toronto

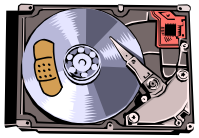
October 21, 2011

Problem in a Nutshell

- File systems store valuable data
  - Both business value and personal value
- Tension between performance and reliability
  - And availability, scalability, new features, backwards compatibility…
- Metadata describes content of file system
  - Complex relationships exist
  - Small errors can result in significant corruption

➢ Goal is to ensure metadata consistency

Causes of Metadata Inconsistency

1. Crash failures 
   - For performance, update metadata asynchronously
   - Badly timed crash can leave inconsistent updates
   - ✓ Enable recovery to consistent state after crash
     - ✓ Journaling, soft updates, copy-on-write systems
2. Hardware errors 
   - Latent sector errors (aka bitrot), lost or torn writes
   - ✓ Checksums + redundancy

➢ We assume these solutions are available

Remaining Cause of Inconsistency

3. Bugs in file systems

High Severity ext3 bug reports	Closed
ext3 corruption fix	2002-06
Linux: Data corrupting ext3 bug in 2.4.20	2002-12
panic/ext3 fs corruption with RHEL4-U6-re20070927.0	2007-11
Re: [2.6.27] filesystem (ext3) corruption (access beyond end)	2008-06
linux-2.6: ext3 filesystem corruption	2008-09
linux-image-2.6.29-2-amd64: occasional ext3 filesystem corruption	2009-06
ENOSPC during fsstress leads to filesystem corruption on ext2, ext3, and ext4	2010-03
ext3: Fix fs corruption when make_indexed_dir() fails	2011-06

- New bugs still being found in mature file systems

Current Approaches

- Offline consistency check and repair program (e2fsck)
  - ✘ Slow
  - ✘ Repairs may be incorrect
- Restore from last consistent backup
  - ✘ Loss of all data more recent than backup

➢ Can we protect file systems from themselves?

Our Approach

- Verify that file system always preserves metadata consistency
  - Observe file system behavior at runtime
  - Guard against updates that show symptoms of bugs
  - Prevent corruption from propagating to media
- Aim to handle arbitrary file system bugs, memory corruption
  - Assume system already handles crash failures, latent sector errors

➢ Recon makes *silent failures* detectable

Key Challenges

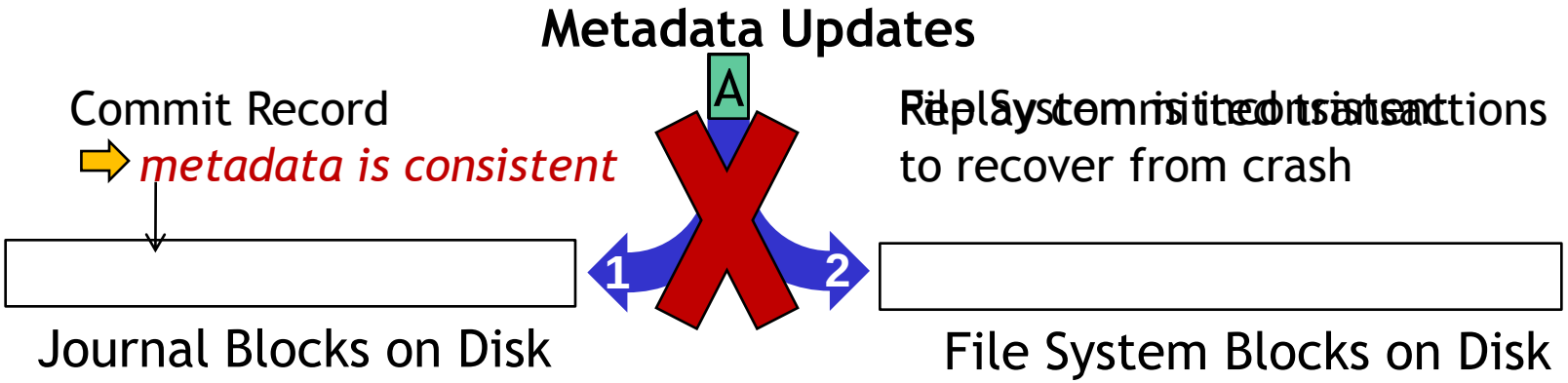
1. What consistency properties to check?
   - Same as offline checker
   - ✘ Global properties require full disk scan
   - ✓ Define *consistency invariants*, use local checks
2. When should consistency be checked?
   - When file system is supposed to be consistent
3. How should the properties be checked?
   - Must be independent of the file system

Consistency Invariants

- Each global consistency property can be converted to a local invariant
- Consistency property:
  - all in-use data blocks marked in block bitmap
- Corresponding consistency invariant
  - If transaction makes a data block live (add pointer to the block), it must also flip a corresponding bit (from 0 to 1) in block bitmap
  - Invariant can be checked locally, by examining updated pointer block and updated block bitmap

When to Check Consistency?

- In-memory metadata may be inconsistent
  - Can't check at arbitrary times
- Modern file systems use transactional updates
  - e.g., journaling, copy-on-write



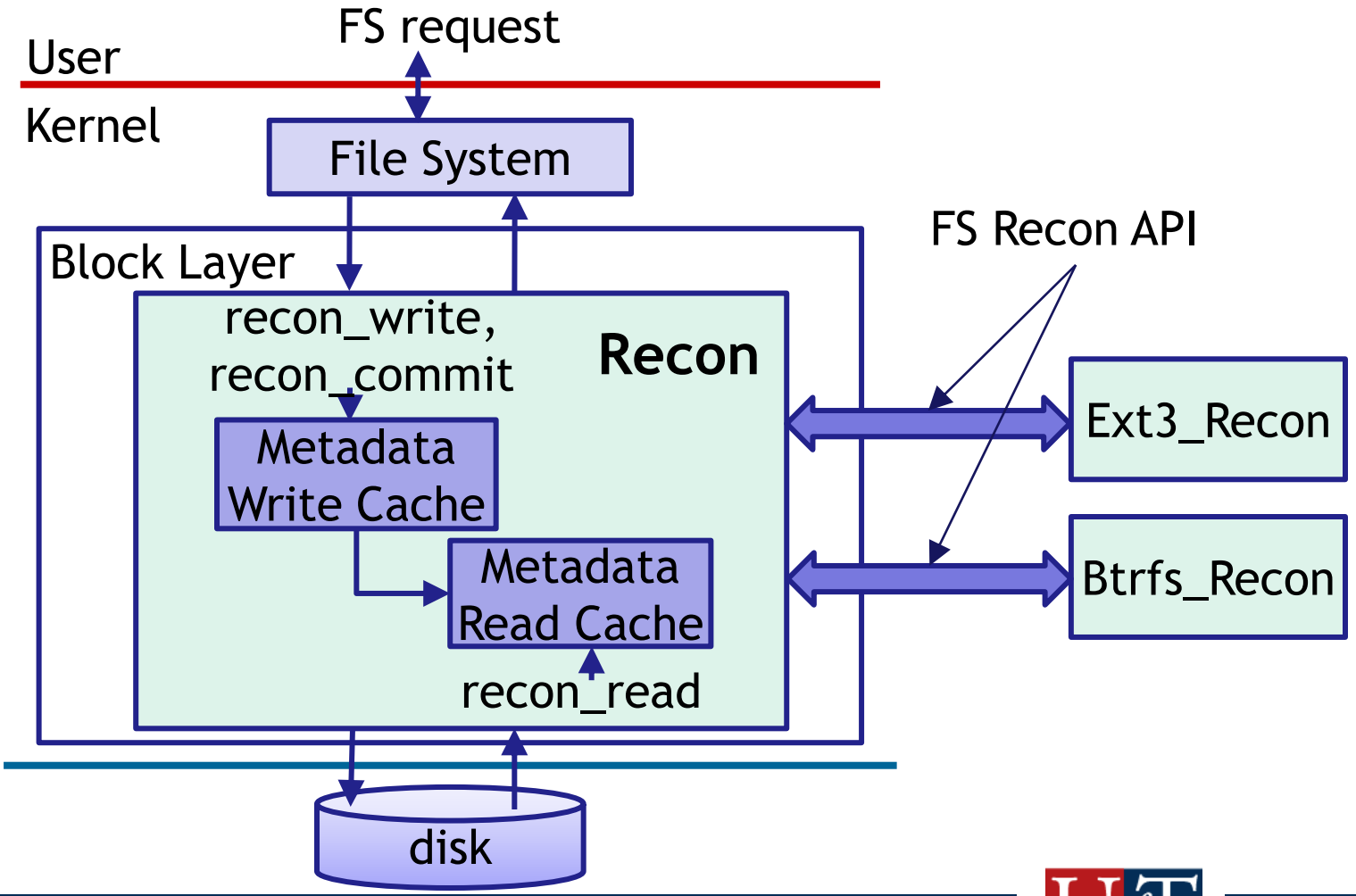
➢ Check consistency at commit points

How to Check Invariants?

- Conflicting requirements
  - Need to interpret file-system format, transaction semantics to make consistency assertions
  - Need to interpret semantics from outside because file system may be buggy
- Recon is a framework for file-system specific
  - metadata interpretation
  - invariant checking

➢ Relies on *pointer-before-block* assumption

Recon System Architecture



Ext3 Invariant Checking

Invariant:

block pointer set from 0 to N $\Longleftrightarrow$ *bit N set in bitmap*

Change Records

[Type, Identity, Field, Old, New]

[Inode, 12, block[1], 0, 22717]
[BBM, 22717, 0, 0, 1]
[BGD, 0, free_blocks, 1500, 1499]
[Inode, 12, i_size, 4052, 7249]
[Inode, 12, i_blocks, 8, 16]

Key	New Ptr	BBM bit set
…		
22717	1	1
…		

✓ Ok

- Each invariant is checked independently

Handling Violations

Several options exist:

- Log warning and continue
  - Obviously risky since we know something is inconsistent!
- Force unmount of file system, prevent writes
  - Reduces availability, loses most recent data
- Take snapshot of filesystem and continue
  - And log warning message
- Micro-reboot file system a la Membrane

That's All Folks

- Recon detects metadata corruption as well as the offline checker
  - But does so *before* the damage reaches disk
- The performance impact is reasonable

➢ Questions?