Physical Disentanglement in a Container-Based File System

A Presentation for CS854

Lanyeu Lue,
Yupu Zhang,
Thanh Do,
Samer Al-Kiswany,
Andrea C. Arpaci-Dussueau,
Remzi H. Arpaci-Dussueau

Presented by Krishna Vaidyanathan

March 4, 2016

Table of Contents

Introduction

- Isolation is central to increased reliability and improved performance of modern computer systems.
- We say, for example, that two files are *entangled* if their blocks are allocated using the same bitmap.
- Entanglement mainly arises because:
    - Logically-independent file system entities are not physically independent.

Motivation

- Entanglement can cause three main problems:
  1. Global failure
  2. Slow recovery
  3. Bundled performance

Motivation: Global Failure

- Single fault leads to a global failure.
- Current file systems crash entire system or mark whole file system read -only.
- For example:
  - Btrfs crashes entire OS when invariant is violated.
  - ext3 marks whole file-system read-only when it detects corruption in single inode bitmap.

Motivation: Global Failure

- Single fault leads to a global failure.
- Current file systems crash entire system or mark whole file system read -only.
- For example:
  - Btrfs crashes entire OS when invariant is violated.
  - ext3 marks whole file-system read-only when it detects corruption in single inode bitmap.

Global Failures	Ext3	Ext4	Btrfs
Crash	129	341	703
Read-only	64	161	89

Motivation: Global Failure

- Single fault leads to a global failure.
- Current file systems crash entire system or mark whole file system read -only.
- For example:
  - Btrfs crashes entire OS when invariant is violated.
  - ext3 marks whole file-system read-only when it detects corruption in single inode bitmap.

Global Failures	Ext3	Ext4	Btrfs
Crash	129	341	703
Read-only	64	161	89

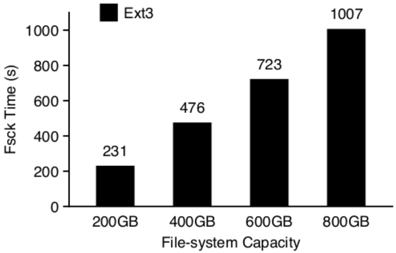
Fault Type	Ext3	Ext4
Metadata read failure	70 (66)	95 (90)
Metadata write failure	57 (55)	71 (69)
Metadata corruption	25 (11)	62 (28)
Pointer fault	76 (76)	123 (85)
Interface fault	8 (1)	63 (8)
Memory allocation	56 (56)	69 (68)
Synchronization fault	17 (14)	32 (27)
Logic fault	6 (0)	17 (0)
Unexpected states	42 (40)	127 (54)

Motivation: Slow Recovery

- After failure, offline file-system checker scans whole file system.
- Checkers are pessimistic: entire file system checked, when only a small piece of corrupted data.
- Not scalable.

Motivation: Slow Recovery

- After failure, offline file-system checker scans whole file system.
- Checkers are pessimistic: entire file system checked, when only a small piece of corrupted data.
- Not scalable.

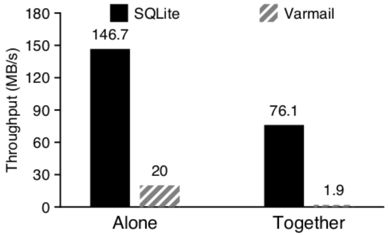


Motivation: Bundled Performance

- File-systems, like ext3, use a journal to keep track of uncommitted changes, committing at periodic intervals.
- All updates within a short period of time are grouped together.
- Performance of independent processes are bundled.

Motivation: Bundled Performance

- File-systems, like ext3, use a journal to keep track of uncommitted changes, committing at periodic intervals.
- All updates within a short period of time are grouped together.
- Performance of independent processes are bundled.



Current Solutions

- Namespaces: defines subset of files and directories to be made visible.

Current Solutions

- Namespaces: defines subset of files and directories to be made visible.
    - Fails to address above problems.
    - Files from different namespaces may still share metadata, system states etc.,
- Static disk partitions: Multiple file systems can be created on separate partitions.

Current Solutions

- Namespaces: defines subset of files and directories to be made visible.
  - Fails to address above problems.
  - Files from different namespaces may still share metadata, system states etc.,
- Static disk partitions: Multiple file systems can be created on separate partitions.
  - Single `panic()` or `BUG_ON()` can still crash entire OS.
  - Not flexible, and number of partitions limited.

Usage Scenarios

- Virtual Machines:

Usage Scenarios

- Virtual Machines:
  - Fault isolation of paramount importance.
  - Single fault triggered by one virtual disk can cause host file system to become read-only.
  - Redeployment and recovery require considerable downtime.

Usage Scenarios

- Virtual Machines:
  - Fault isolation of paramount importance.
  - Single fault triggered by one virtual disk can cause host file system to become read-only.
  - Redeployment and recovery require considerable downtime.
- Distributed File Systems:

Usage Scenarios

- Virtual Machines:
  - Fault isolation of paramount importance.
  - Single fault triggered by one virtual disk can cause host file system to become read-only.
  - Redeployment and recovery require considerable downtime.
- Distributed File Systems:
  - Physical entanglement negatively impacts distributed file systems, especially multi-tenant settings.
  - Specifically, HDFS does not provide fault isolation for applications.
  - Eg: four clients concurrently read different files, and the machine which stores the data blocks crashes.

Cube Abstraction

- Enables logical relation between files and directories.
- Safely combine performance and reliability properties of groups of files and their metadata.
- Each cube is completely independent at the file-system level.
- Cubes are:
  1. Isolated
  2. Transparent
  3. Flexible
  4. Elastic
  5. Customized
  6. Lightweight

Principles: No Shared Physical Resources

- Storage space is divided into fixed-size block groups.
- Each block group has its own metadata.
- Files and directories are allocated to particular block groups.
- Any block group and its corresponding metadata blocks can be shared across any set of files.

Principles: No Access Dependency

- Cubes must not contain references/need access to other cubes.
- Data structures that violate this: linked list, trees etc.,
- One failed entry can affect all entries above or below it.

Principles: No Bundled Transactions

- To guarantee consistency and metadata, existing file-systems use journaling.
- A transaction for this contains temporal updates from many files.
- Problems:
    - Updates to all files in transition fails.
    - Performance of independent files and workloads coupled.

- Cubes as basic new abstraction.
- Cube implemented as a special directory.
- All files and sub-directories belong to same cube.
- To create a cube, `mkdir()` must be called with a cube flag.
- To delete, `rmdir()`.

Physical Resource Isolation

- Leverage existing concept of a block group.
- Block group can be assigned to only one cube at any time.
- All metadata associated with a block group belongs to only one cube.

Access Independence

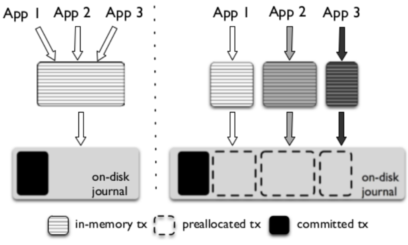
- No cube can reference another cube.
- One sub-super block for each cube, stored after the super block.
- Sub-super block refers its own orphan inode.
- Direction indirection:
  - Each cube records its top directory.
  - Filesystem performs longest prefix match of pathname among cube's top directory paths.
  - Only remaining pathname wihtin the cube is traversed in traditional manner.
  - Eg., for access to `/home/bob/research/paper.tex`, IceFS skips directly to parsing `paper.tex` within the cube if `/home/bob/research` designates the top.

Transaction Splitting

- Each cube has its own running transaction to buffer writes.
- Transactions committed to disk in parallel without waiting or dependencies.
- Any failure can be attributed to faulty cube and recovery action can be taken only on it.

Transaction Splitting

- Each cube has its own running transaction to buffer writes.
- Transactions committed to disk in parallel without waiting or dependencies.
- Any failure can be attributed to faulty cube and recovery action can be taken only on it.



(a) Ext3/Ext4 (b) IceFS

Localized Reactions to Failures

- Fault detection:
  - IceFS modifies existing detection techniques to make them cube-aware.
  - Instruments fault-handling and crash-triggering functions to include the ID of responsible cube.
- Localized Read-Only:
  - Individually aborts transactions for a single cube.
  - Faulty cube alone is made read-only.
- Localized Crashes:
  1. Fails the crash-triggering thread.
  2. Prevent new threads.
  3. Evacuates running threads.
  4. Clean up the cube.

Localized Recovery

- Offline Checking:
  - Supports partial checking of a file system by examining only faulty cubes.
  - ice-fsck identifies, checks, and repairs the faulty cube alone.
- Online Checking:
  - Offline checking implies data will be unavailable, which is not acceptable for many applications.
  - IceFS enables on-line checking of faulty cubes alone.
  - Online ice-fsck is a user-space program that takes the ID of the faulty cube.

Specialized Journaling

- Since there are no dependencies across cubes, different consistency modes for cubes.
- Five consistency modes:
  1. no fsync
  2. no journal
  3. writeback journal
  4. ordered journal
  5. data journal

Implementation

- Ext3/JBD in Linux 3.5 was modified for data structures and journaling isolation.
- VFS for directory indirection.
- e2fsprogs 1.42.8 for file system creation and checking.

Overall Performance

Workload	Ext3 (MB/s)	IceFS (MB/s)	Difference
Sequential write	98.9	98.8	0%
Sequential read	107.5	107.8	+0.3%
Random write	2.1	2.1	0%
Random read	0.7	0.7	0%
Fileserver	73.9	69.8	-5.5%
Varmail	2.2	2.3	+4.5%
Webserver	151.0	150.4	-0.4%

Evaluation

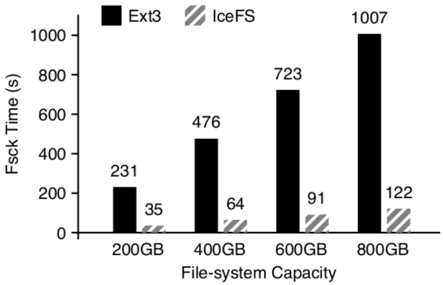


Figure : Performance of IceFS Offline Fsck.

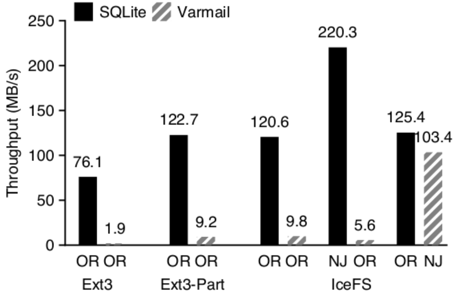


Figure : IceFS under different journaling modes.

Limitations

- IceFS uses separate journal commit thread for every cube.

Limitations

- IceFS uses separate journal commit thread for every cube.

Device	Ext3 (MB/s)	Ext3-Part (MB/s)	IceFS (MB/s)
SSD	40.8	30.6	35.4
Disk	2.8	2.6	2.7

Limitations

- IceFS uses separate journal commit thread for every cube.

Device	Ext3 (MB/s)	Ext3-Part (MB/s)	IceFS (MB/s)
SSD	40.8	30.6	35.4
Disk	2.8	2.6	2.7

- Cache flush time accounts for more in SSD, while seek times dominate in hard drives.

References

Lu, L., Zhang, Y., Do, T., Al-Kiswany, S., Arpaci-Dusseau, A. C., & Arpaci-Dusseau, R. H. (2014). Physical disentanglement in a container-based file system. In 11th USENIX Symposium on Operating Systems Design and Implementation (OSDI 14) (pp. 81-96).