Data integrity in the Cloud

Christoph Hellwig
Nebula, Inc
Data integrity in brief:

- Writing data reliably to persistent storage
- Retrieving the same data again later
Application writes

- Applications only write to the OS cache
- Reliable ACKs lost
- Need fsync to:
  - Transfer data to disk
  - Get a reliable ACK
Operating System writes

- Often the OS writes to disk cache only
- Reliable ACKs lost
- Need a cache flush to:
  - Transfer data to disk
  - Get reliable ACK
Byte-level writes are not atomic

- OS caches operate on page granularity (usually 4096 bytes)
- Disk drives operate on sectors (usually 512 bytes)
- Striped RAID operates on much larger granularities *(but pretends not to)*
Write granularity issues

- Too small writes need read-modify-write cycles
  - May cause untouched data to be corrupted
- Too large writes can be torn
  - Half-updated data may be on disk
Reading data

- As long as the disk and data is there that's easy, right?
  - Silent data corruption happens (a lot)
  - Disk might fail entirely and go away
Data protection

- Disks use error correction to deal with bit flips
  - Reads that succeed should return the right data

- To protect against disk failure data is stored in multiple places
  - N-way mirroring (e.g. RAID1)
  - Erasure encoding (e.g. RAID5/6)
I/O architecture complexity

- **Personal Computer**
  - Disk
  - Driver
  - File System
  - Multipathing
- **Enterprise Server**
  - RAID / SAN
  - Driver
  - Volume Manager
  - Multipathing
- **Enterprise Virtualization**
  - Guest
  - Hypervisor
  - File System
  - Multipathing

Increasing complexity
I/O architecture complexity

The Cloud!
The Cloud?

- Cloud computing in I/O terms:
  - Virtualization as abstraction
  - Massive scale distributed systems
  - (Object Storage)
Cloud I/O architectures (1)

The “Enterprise” I/O stack

- Just treats “cloud” as a management layer
- Scale-out of compute and disk is handled independently
“Instance” storage
- Local non-fault tolerant storage
- Used for instance-local temporary data
- Part of the AWS and Openstack models
Cloud I/O architectures (3)

- Distributed storage
  - Access by internal networking
  - Data stored on a large number of nodes
Cloud data integrity issues

- Mapping between layers
  - Complicated I/O stacks
  - Mapping block I/O semantics on file system semantics

- Distributed systems
  - Replication needs to be location-aware
  - Failure is the norm
Running the SQLite test produced significantly better results when running under the virtual machine compared to the host. The SQLite performance we have seen vary substantially between kernel releases and file-systems, but with the virtualization performance, it's doing much better this way compared to the host OS with both using the Linux 2.6.31 kernel and EXT4 file-system.
I/O layering issues

- Caching semantics need to be preserved over all layers
  - Including mapping them from block device layers to file systems and back
  - On distributed systems multiple writers complicate semantics a lot
    - This may include live migration
Data placement
Data placement

Good placement?
Data placement

Better placement?

Rack
Switch
Node
Node
Node
Node
Node

Rack
Switch
Node
Node
Node
Node
Node

Rack
Switch
Node
Node
Node
Node
Node
Node

Better placement?
Data placement

Or maybe this?
In-flight data protection

- Data might get corrupted in-flight
  - Lots of potentially buggy I/O layers
  - Large clusters → multiple hops through switches

- Various protocols offer error checking or correction
  - E.g. iSCSI crc32c
  - But we'd really like to be able to verify integrity through the whole I/O stack
T10 DIF/DIX

- **End-to-end data protection**
  - Stores application checksum in extended disk sectors
  - Standard format that can be validated by all intermediate layers

- **Issues:**
  - Requires disk support, but not part of the ATA standard
Questions?