The Btrfs Filesystem

Chris Mason

ORACLE

The Btrfs Filesystem

- Jointly developed by a number of companies
  - Oracle, Redhat, Fujitsu, Intel, SuSE, many others
- All data and metadata is written via copy-on-write
- CRCs maintained for all metadata and data
- Efficient writable snapshots
- Multi-device support
- Online resize and defrag
- Transparent compression
- Efficient storage for small files
- SSD optimizations and trim support
- Used in production today in Meego devices

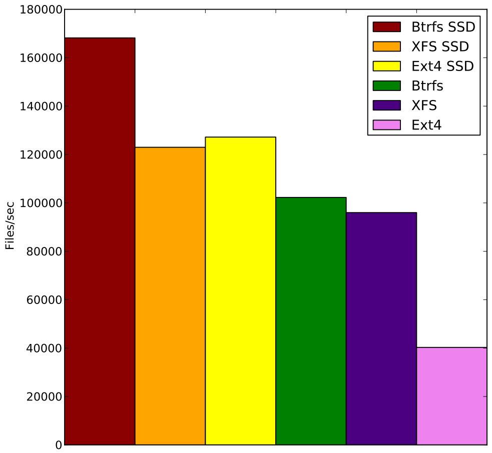
Btrfs Progress

- Many performance and stability fixes
- Significant code cleanups
- Efficient free space caching across reboots
- Improved inode number allocator
- Delayed metadata insertion and deletion
- Multi-device fixes, proper round robin allocation
- Background scrubbing
- New LZO compression mode
- Batched discard (fitrim ioctl)
- Per-inode flags to control COW, compression
- Automatic file defrag option

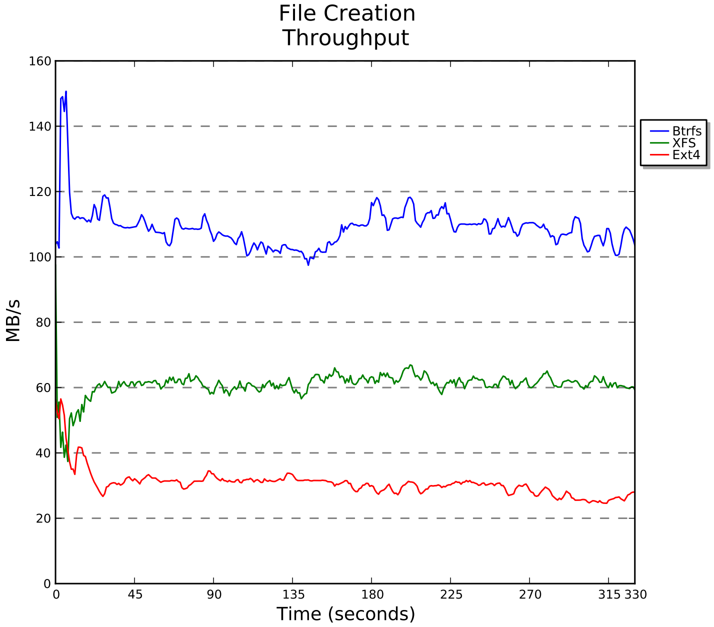
Billions of Files?

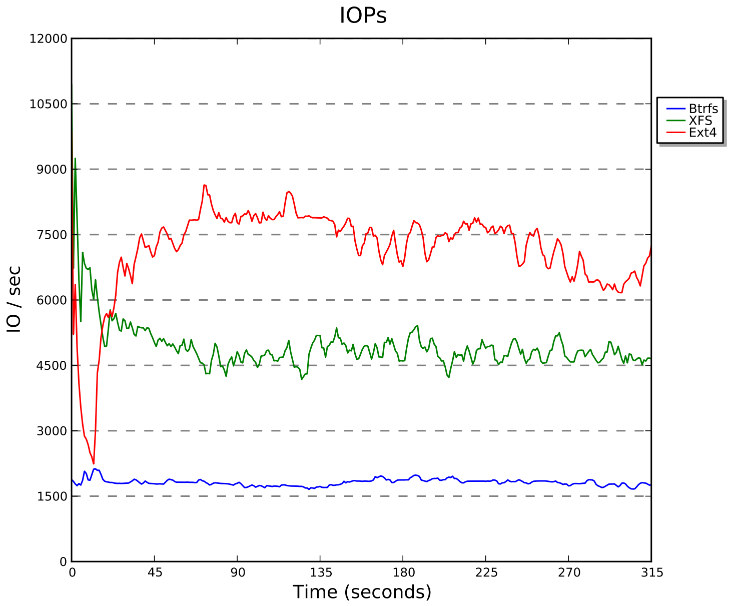
- Ric Wheeler’s talk includes billion file creation benchmarks
- Dramatic differences in filesystem writeback patterns
- Sequential IO still matters on modern SSDs
- Btrfs COW allows flexible writeback patterns
- Ext4 and XFS tend to get stuck behind their logs
- Btrfs tends to produce more sequential writes and more random reads

File Creation Benchmark Summary



- Btrfs duplicates metadata by default
  - 2x the writes
- Btrfs stores the file name three times
- Btrfs and XFS are CPU bound on SSD


File Creation
Throughput
MB/s
0
20
40
60
80
100
120
140
160
Time (seconds)
0
45
90
135
180
225
270
315
330
Btrfs
XFS
Ext4



IOPs
IO / sec
Time (seconds)
Btrfs
XFS
Ext4
0
1500
3000
4500
6000
7500
9000
10500
12000
0
45
90
135
180
225
270
315

IO Animations

- Ext4 is seeking between a large number of disk areas
- XFS is walking forward through a series of distinct disk areas
- Both XFS and Ext4 show heavy log activity
- Btrfs is doing sequential writes and some random reads

Metadata Fragmentation

- Btrfs btree uses key ordering to group related items into the same metadata block
- COW tends to fragment the btree over time
- Larger blocksizes lower metadata overhead and improve performance
- Larger blocksizes provide limited and very inexpensive btree defragmentation
- Ex: Intel 120GB MLC drive:
  - 4KB Random Reads – 78MB/s
  - 8KB Random Reads – 137MB/s
  - 16KB Random Reads – 186MB/s
- Code queued up for Linux 3.1 allows larger btree blocks

Scrub

- Btrfs CRCs allow us to verify data stored on disk
- CRC errors can be corrected by reading a good copy of the block from another drive
- New scrubbing code scans the allocated data and metadata blocks
- Any CRC errors are fixed during the scan if a second copy exists
- Will be extended to track and offline bad devices
- (Scrub Demo)

Discard/Trim

- Trim and discard notify storage that we're done with a block
- Btrfs now supports both real-time trim and batched
- Real-time trims blocks as they are freed
- Batched trims all free space via an ioctl
- New GSOC project to extend space balancing and reclaim chunks for thinly provisioned storage

Future Work

- Focus on stability and performance for desktop and server workloads
- Reduce lock contention in the Btree and kernel data structures
- Reduce fragmentation in database workloads
- Finish offline FS repair tool
- Introduce online repair via the scrubber
- RAID 5/6
- Take advantage of new storage technologies
  - High IOPs SSD
  - Consumer SSD
  - Shingled drives
  - Hybrid drives

Future Work: Efficient Backups

- Existing utilities can find recently updated files and extents
- Integrate with rsync or other tools to send FS updates to remote machines
- Don't send metadata items, send and recreate file data instead

Future Work: Tiered Storage

- Store performance critical extents in an SSD
  - Metadata
  - fsync log
  - Hot data extents
- Migrate onto slower high capacity storage as it cools

Future Work: Deduplication

- Existing patches to combine extents (Josef Bacik)
  - Scanner to build DB of hashes in userland
- May be integrated into the scrubber tool
- May use existing crc32c to find potential dups

Future Work: Drive Swapping

- GSOC project
- Current raid rebuild works via the rebalance code
- Moves all extents into new locations as it rebuilds
- Drive swapping will replace an existing drive in place
- Uses extent-allocation map to limit the number of bytes read

Thank You!

- Chris Mason <chris.mason@oracle.com>
- http://btrfs.wiki.kernel.org