Regensburg HPC Workshop 2009

Andreas Dilger
Sun Microsystems

Overview

- Ext4 features
- Multi Mount Protection
- RAID tuning
- OST Pools
- File size and Glimpse
- Timeouts and Eviction

Ext4 Features

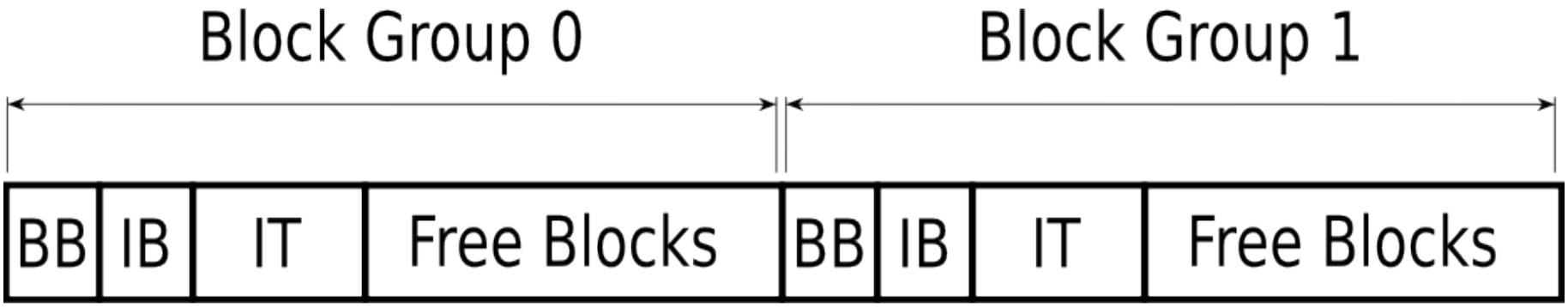
- Fast extended attributes (1.4)
- Extent support (1.4)
- Multiple block allocator (1.4)
- Uninitialized groups for faster fsck (1.6.5)
- Inode versioning (VBR,1.8)
- Delayed block allocation (all)
- File extent map (FIEMAP, 1.8)
- Nanosec timestamp (1.6.5)
- Flexible Inode Placement
- Larger Files (> 2TB)
- Persistent file preallocation (sys_fallocate)
- Larger file system (>16TB)

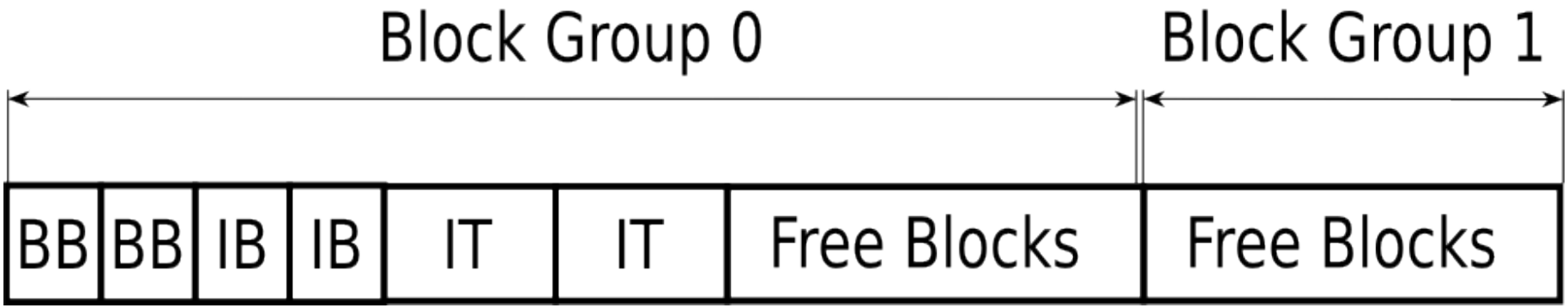
Ext4 - Flexible block groups

- Can place bitmaps and inode table anywhere
- Co-locate group bitmaps and inode tables to provide larger contiguous free spaces
- Avoid costly seeks for both data/metadata
- Allow for new allocation strategies that exploit the new meta-data allocation
- Avoid using all group metadata (with uninit_bg) to keep fsck times low
- Format time option only

```
mke2fs -O flex_bg -G nr_merged_groups
mkfs.lustre --mkfsoptions "-O flex_bg
  -G nr_merged_groups" ...
```

Ext4 – Flexible Block Group Layout





Ext4 - 16 Threads FFSB results

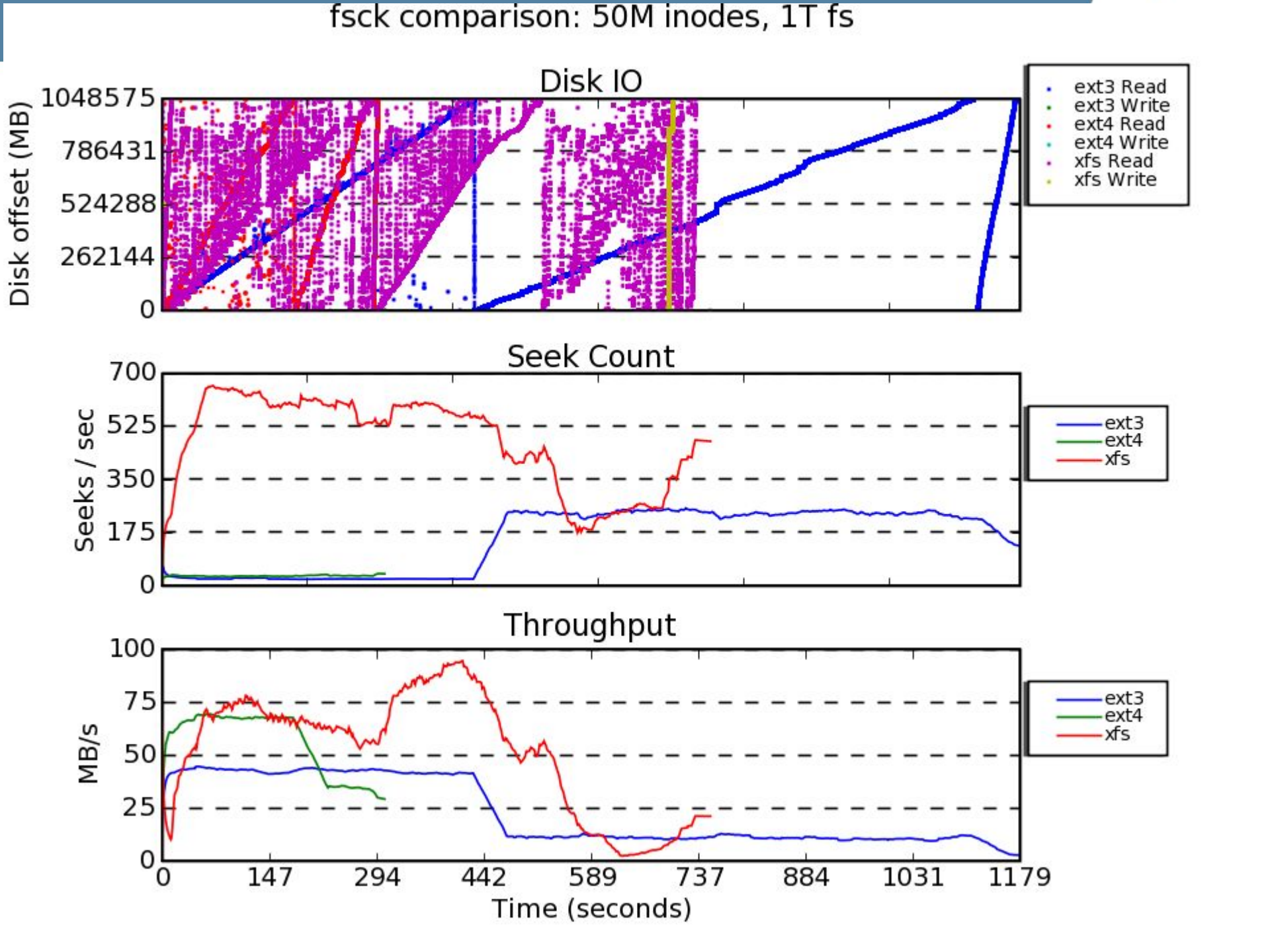
Op	Ext4	Ext4(flex_bg)	% change
read	96778	119135	18.76%
write	143744	174409	17.58%
create	1584997	1937469	18.19%
append	46735	56409	17.14%
delete	93333	113598	17.83%
Total	6514.51	7968.24	18.24%

Ext4 – e2fsck improvements

- Skip uninitialized bitmaps
- Skip unused inodes
- Checksum group descriptors
- SOON: support for > 16TB filesystems

```
mke2fs -O uninit_bg ... /dev/sda

tune2fs -O uninit_bg ... /dev/sda
e2fsck -fy /dev/sda
```


fsck comparison: 50M inodes, 1T fs
Disk IO
Disk offset (MB)
1048575
786431
524288
262144
0
ext3 Read
ext3 Write
ext4 Read
ext4 Write
xfs Read
xfs Write
Seek Count
Seeks / sec
700
525
350
175
0
ext3
ext4
xfs
Throughput
MB/s
100
75
50
25
0
0
147
294
442
589
737
884
1031
1179
Time (seconds)
ext3
ext4
xfs

Ext4 – Multi Mount Protection

- Important for failover devices
- Delays mount/e2fsck by ~10s seconds

```
mke2fs -O mmp /dev/sda
tune2fs -O mmp /dev/sda

tune2fs -F -O ^mmp /dev/sda
```

LDISKFS-fs warning (device sda): Device is already active on another node.

LDISKFS-fs warning (device sda): MMP failure info: last update time: 1239822128, last update node: lin-cli1, last update device: sdb

Ext4 – current state of affairs

- Available in RHEL5.3 update
- Available in SLES11
- Will be available in RHEL6
- Lustre ldiskfs ported to ext4 baseline
- Used by SLES11 in 1.8.1
- Optional for RHEL5 in 1.8.1

- MMP not yet accepted upstream

RAID tuning for ext3/4

- Better layout for ext3/4 (ignore for flex_bg)
- Tunes mballoc to RAID geometry
- Keep RAID stripe-width below 1MB

e.g. 64kB RAID6 6+2 (4kB blocks)

64kB/stride / 4kB/block = 16 block/stride
64kB * 6 stripes / 4kB/block = 96 blocks

```
mke2fs -E stride-size=16 -E stripe-width=96
tune2fs -E stripe-width=96

lctl conf_param lustre.osc.max_pages_per_rpc=192
```

OST Pools

- Named groups of OSTs
- Useful for heterogeneous storage
- Advisory OST selection only

```
mgs# lctl pool_add lustre.slow lustre-OST[0-13]
mgs# lctl pool_add lustre.fast lustre-OST[14-22]

lfs setstripe -c 2 -p fast /mnt/lustre/mydir
lfs setstripe -c 2 -p slow /mnt/lustre/yourdir

lctl get_param -N lov.lustre*.pools.*
lctl get_param -n lov.lustre*.pools.fast
```

OSS Read Cache

- Cache read/write data on OSS
- Benefits repeat reads/read-modify-write
- Can limit to smaller files (default all files)

```
lctl set_param obdfilter.*.read_cache_enable=0
lctl set_param
  obdfilter.*.readcache_max_filesize=32M
```

Recovery Improvements

- Version Based Recovery (in 1.8)
  - Independent recovery stream per file
  - Isolate recovery domain to dependent ops
- Commit on Share (in 2.0)
  - Avoid client getting any dependent state
  - Avoid sync for single client operations
  - Avoid sync for independent operations
- Adaptive Timeouts (in 1.8)

Adaptive Timeouts (AT)

- Avoid need to specify RPC timeout
- Handle different scale systems
- Handle different IO loads
- Reduce recovery time to minimum

- Per-client negotiated timeout
- Per-service negotiated timeout
- Communicated with every RPC

Mechanisms - server

- Servers track the estimated RPC completion times over a limited period of time
- Servers report the latest RPC service time estimate in each RPC reply, along with actual service time for this particular RPC
- Server sends an *early reply* if RPC deadline approaches
- Watchdogs adapt too

Mechanisms - client

- Clients remember server estimates (per portal, per import)
- Clients set RPC timeout based on server estimate
- Clients include deadline information in RPC request
- If client receives an early reply to an RPC, it adjusts its timeout
- Client timeout is server estimate plus measured network latency

AT status

- Adaptive timeout information can be read from /proc/fs/lustre/*/timeouts files, for each service and for each client.
- Service

```
cfs21:~# cat /proc/fs/lustre/ost/OSS/ost_io/timeouts
service : cur  33  worst  34 (at 1193427052, 0d0h26m40s ago)   1   1  33   2
```

>the ost_io service on this node is currently reporting an estimate of 33 seconds. The worst RPC service time was 34s, and this happened 26 minutes ago. Finally, there is a history of service times -- there are 4 "bins" of adaptive_timeout_history/4 seconds each, and these are the maximum RPC times that took place in each of those bins. So in the last 150s, the max RPC time was 1, same with 150-300s, from 300-450s the worst was 33s, and from 450-600s the worst was 2s. The current estimate is the max of the 4 bins.

AT status – con't

- Client

> The times as reported by the servers are also tracked in the client obd's:

```
cfs21:~# cat /proc/fs/lustre/osc/lustre-OST0001-osc-ce129800/timeouts
last reply : 1193428639, 0d0h00m00s ago
network    : cur   1  worst   2 (at 1193427053, 0d0h26m26s ago)   1   1   1   1
portal 6   : cur  33  worst  34 (at 1193427052, 0d0h26m27s ago)  33  33  33   2
portal 28  : cur   1  worst   1 (at 1193426141, 0d0h41m38s ago)   1   1   1   1
portal 7   : cur   1  worst   1 (at 1193426141, 0d0h41m38s ago)   1   0   1   1
portal 17  : cur   1  worst   1 (at 1193426177, 0d0h41m02s ago)   1   0   0   1
```

> In this case, RPCs to portal 6, the OST_IO_PORTAL (see lustre/include/lustre/lustre_idl.h), shows the history of what the ost_io portal has been reporting as the service estimate

2.1: ZFS OST/MDT Storage

Capacity

- Single filesystem 100TB+ (2^{64} LUNs * 2^{64} bytes)
- Trillions of files in a single file system (2^{48} files)
- Dynamic addition of capacity/performance

Reliability and resilience

- Transaction based, copy-on-write
- Internal data redundancy (double parity, 3 copies)
- End-to-end checksum of all data/metadata
- Online integrity verification and reconstruction

Functionality

- Snapshots, filesets, compression, encryption
- Online incremental backup/replication
- Hybrid storage pools (HDD + SSD)

2.x: Imperative Recovery

Server driven notification of failover

- Server notifies client of failover completed
- Client replies immediately to server
- Avoid client waiting on RPC timeouts
- Avoid server waiting for dead clients

Can tell between slow/dead server

- No waiting for RPC timeout start recovery
- Can use external or internal notification

2.x: SMP Scalability

- Future nodes will have 100s of cores
  - Need excellent SMP scaling on client/server
  - Need to handle NUMA imbalances
- Remove contention on servers
  - Per-CPU resources (queues, locks)
  - Fine-grained locking
  - Avoid cross-node memory access
    - Bind requests to a specific CPU deterministically
      - Client NID, object ID, parent directory
- Remove contention on clients
  - Parallel copy_{to,from}_user, checksums

Thank you

Andreas Dilger
<adilger@sun.com>