# ABISS BOF

## Active Block I/O Scheduling System

http://abiss.sourceforge.net/

Werner Almesberger
werner@almesberger.net

July 20, 2006

# What is ABISS ?

- Real-time file I/O for streaming applications
- Currently supports FAT, VFAT, ext2, and ext3 without journaling
- By Philips Research

# Quick overview

- Opening a file
  - Application tells kernel what and how it will read
  - Kernel caches file data location
  - Admission control by abissd, a user-space daemon
- Reading a file
  - Kernel prefetches data and locks it in memory
  - Application tells kernel what it has read
  - Data rate (of prefetch) is limited
- Writing is similar, but needs delayed allocation

# Real-time philosophy

- Bound duration of I/O operations
- Isolate real-time I/O from best-effort (with priorities)
- Move slow things out of the application's critical path
- Don't worry about other tight low-level limits
  - Other people are already hunting the latency hogs
  - We have plenty of time (many milliseconds)

# Changes since last year

- Dynamic resizing of the playout buffer
  (through abissd, e.g., after admitting more RT traffic)
- Jens Axboe added I/O priorities to mainline
  - Includes the CFQ I/O scheduler
  - Proof of concept use of CFQ with ABISS
- Removal of the location map
  - Eliminates ugly `getblk` hooks
  - Avoids unbounded data structure/message
  - Meta-data problems strike

# Meta-data

- Some is shared among files
- Access happens now and then when accessing data
- Location data is implicitly cached with file pages
  (page cache is indexed per file)

Journals are even worse.

# I/O priority vs. locks

Scenario:

- Lock is held by process waiting for I/O to begin
  - Wait for request queue slot
  - Wait for request to get scheduled
- There is another process with higher I/O priority that wants
  - the same data
  - the same lock
- "High-priority" process should be allowed to accelerate access

# I/O locks (Under construction)

- Combine existing lock with I/O priority, e.g.,
  - Page lock, page data
  - FAT lock
- Can "boost" lock holder:
  - Tell waiting process to raise its I/O priority
  - Upgrade requests in the elevator queue
    (We already had this for data)

# Remaining issues

- Make I/O locks really work
- Protection from memory pressure
- Delayed allocation
- Upgrades in non-ABISS elevators
- Make barriers softer

(Who else cares about them ?)