Scalability in the Linux Storage Subsystem

Overview

- Not an overview of what we can do
- Random thoughts on where I think we have problems
- A few words on the biggest challenge I think we face in terms of scalability

Random Thought #1

- Direct I/O scalability is a solved problem
- We're into micro-optimisations now
  - controlling submission/completion CPUs to minimise cacheline bouncing
  - Need to be careful optimising direct I/O doesn't regress other types of I/O
- Implies block and driver layer scalability is good
- Large single I/Os can be easily issued

Random Thought #2

- Need to expose geometry and status of the underlying storage topology
  - to both userspace and kernel interfaces
  - independent failure domains
  - stripe/concat geometry
  - load feedback
- Needs to be dynamic
  - automatic filesystem grow
  - loss of redundancy -> data redistribution

Random Thought #3

- writeback doesn't scale
  - inefficient within a filesystem
    - optimised for filesystems that don't do I/O on inode writeback in ->write_inode
  - single threaded within a filesystem
  - not NUMA aware
  - Different filesystems use different amounts of CPU time in writeback
    - delayed allocation consumed more CPU
  - Lacks alignment of I/O to underlying storage

Random Thought #4

- Error handling is suboptimal
  - can't scale effectively if error paths are not robust
  - In complex topologies, errors are common and not the exception
  - Filesystems are often blamed for “hanging” when the real culprit if an undelivered error from the lower layers

Random Thought #5

- DM/MD are not really useful on common hardware RAID configurations
  - power of 2 chunk size prevents effective use of non-power of 2 RAID3/4/5/6 back end data disks
    - 8/12/14/16/24/48 disk trays don't match at all
  - complex toplogies with different geometries are difficult to express and expose to the filesystem correctly

The IOPS Challenge

- SSDs
  - Ready for 50,000 IOPS/s per disk?
    - >200,000 ctxsw/s per disk
    - 50,000 intr/s per disk
    - Does not scale to many disks
  - Raw IOP capacity per HBA
    - will be a limiting factor
    - driver design will need to focus on IOPS optimisations, not achieving max bandwidth
  - CPU overhead will be high

The IOPS Challenge

- Looks more like the network problem
  - similar “packet” rates to gigabit ethernet per disk
  - many, many more “interfaces” than a typical network stack
  - HBAs with multiple disks will have to handle packet rates closer to 10Gb ethernet
  - similar interrupt scaling tricks will be needed
    - MSI-X directed interrupts
      - one vector per disk behind the HBA?
    - polling rather than interrupt driven

The IOPS Challenge

- Will require both hardware and software to evolve
- Not going to happen overnight
- Two orders of magnitude increase in performance is a big disconnect
- Optimisations being made for current (cheap) SSDs have a short life
  - random write performance is not a limiting factor at the high end....