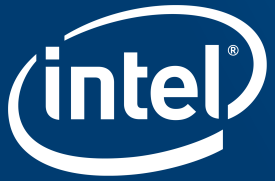


ZFS Improvements for Lustre*

Andreas Dilger, Intel High Performance Data Division

LAD'17

ZFS Enhancements of Interest to Lustre

Changes included in ZFS 0.7.x

- Multi-modifier protection (MMP) for improved HA safety (LLNL)
- Dynamic dnode size (large dnodes) (LLNL)
- Native dnode quota accounting (Intel)
- Multi-threaded dnode allocation, locking, APIs (Delphix, Intel, LLNL)
- CPU-optimized checksums, RAID parity (Uni Hamburg, Intel)
- QAT hardware-assisted checksums and compression (Intel)
- Improved kernel memory allocation for IO buffers (ABD) (others, Intel)
- Improved JBOD fault detection, handling, enclosure LEDs (LLNL)
- Fault Management Architecture (FMA)/ZED integration (others, Intel)

Planned for ZFS 0.8.x

- On-disk data/metadata encryption (Datos)
- Project quota accounting for ZFS (Intel)
- Declustered parity RAID (dRAID) (Intel)
- Metadata Allocation Class (Intel)
- Sequential scrubbing/resilvering (Nexenta)
- More on the way ...

Multi-Modifier Protection (PR#6073 LLNL, 0.7)

MMP prevents multiple nodes importing the same pool

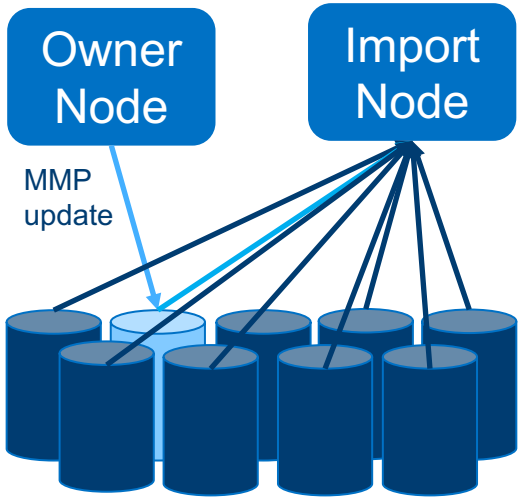
- Significant risk if HA software or STONITH fails
- ZFS not robust against this kind of corruption
  - Inconsistent blocks written with valid checksums by peer

Owner writes periodic update to one random VDEV label

- Update only timestamp in reserved MMP überblock
- No extra überblock written if normal TXG written
- Compatible with older ZFS versions

Import node checks *all* überblocks for modifications after delay

- Won't import pool if it detects any modified timestamp



Enabled by default in 2.10.1+, or with "`zpool property multihost=on`" for existing pools

Dynamic (large) dnode Size (PR#3542 LLNL, 0.7)

ZFS 0.6.x and earlier supported only 512-byte dnodes

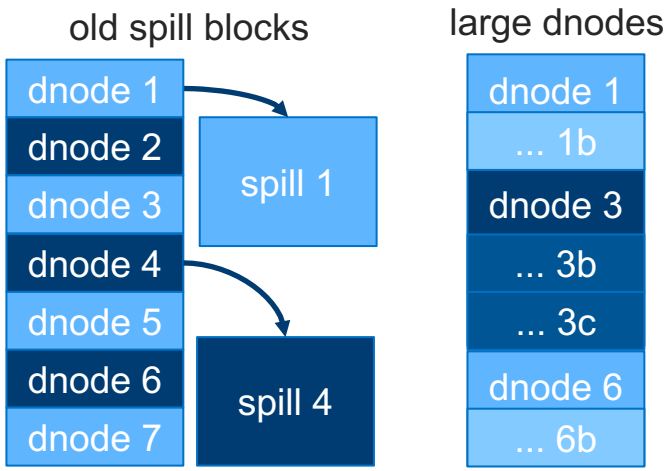
Lustre xattrs (LMA, LinkEA, LOV + PFL, ...) didn't all fit within 512-byte dnode

- Each dnode allocates two extra 4K blocks (*spill block* + mirror copy) for xattrs
- Over 9 GB mirrored writes to create 1M files

Large dnode feature included ZFS in 0.7 release

- Variable dnode size from 0.5KB-16K
- Only 2 GB mirrored writes to create 1M files
- Reduce seeks by 50% (no seek for spill block)

Enable with "`dnodesize=auto`" in **0.7.1+ ONLY**



User/Group dnode Accounting (PR#3500 Intel, 0.7)

- ZFS didn't support native dnode accounting, only block accounting
- Lustre implemented own primitive schema to support file quota
  - Didn't scale well - two files updated on every file creation
- Add native dnode accounting to ZFS in same manner as block accounting
  - Accounting is implemented in the syncing thread
  - Updates single accounting file for all quotas

File Creation Performance (Delphix, Intel, 0.7)

Multi-threaded transaction group (TXG) syncing (PR#5752)

- Flush dirty dnode blocks to multiple devices in parallel

Improved object allocation (PR#6564, PR#6611, PR#6117)

- Multi-threaded dnode allocation to avoid lock contention

Batched quota updates (PR#4642)

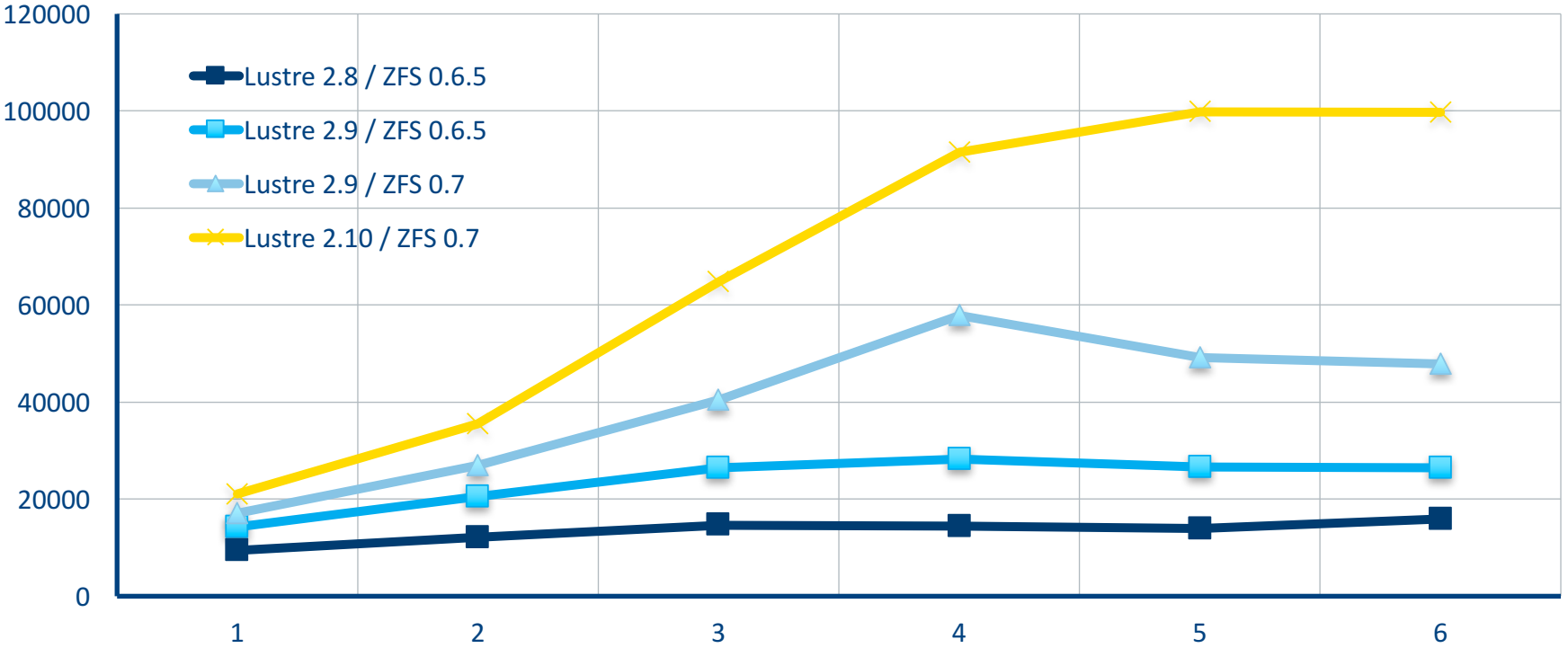
- Modify quota updates once per TXG (+/-n), not once per block (+/-1)

Reduced dnode lookups to avoid needless overhead (PR#5534, PR#5894)

- Add `*_by_dnode()` APIs after initial dnode lookup is done

Reduce unnecessary allocations during create (PR#6048)

Lustre file creation: step by step (mds-survey)



Hardware specification on slide 16

QAT Hardware Compression (PR#5846 Intel, 0.7)

Compression *improves* performance

Intel Quick Assist Technology (QAT)

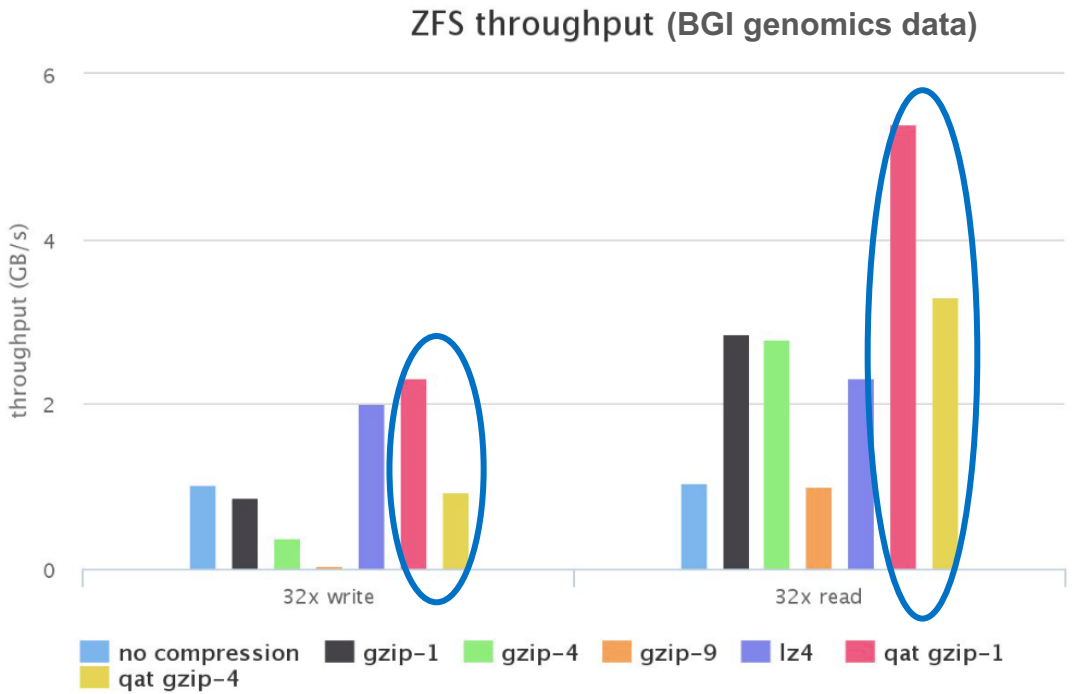
- PCI card/chipset accelerator

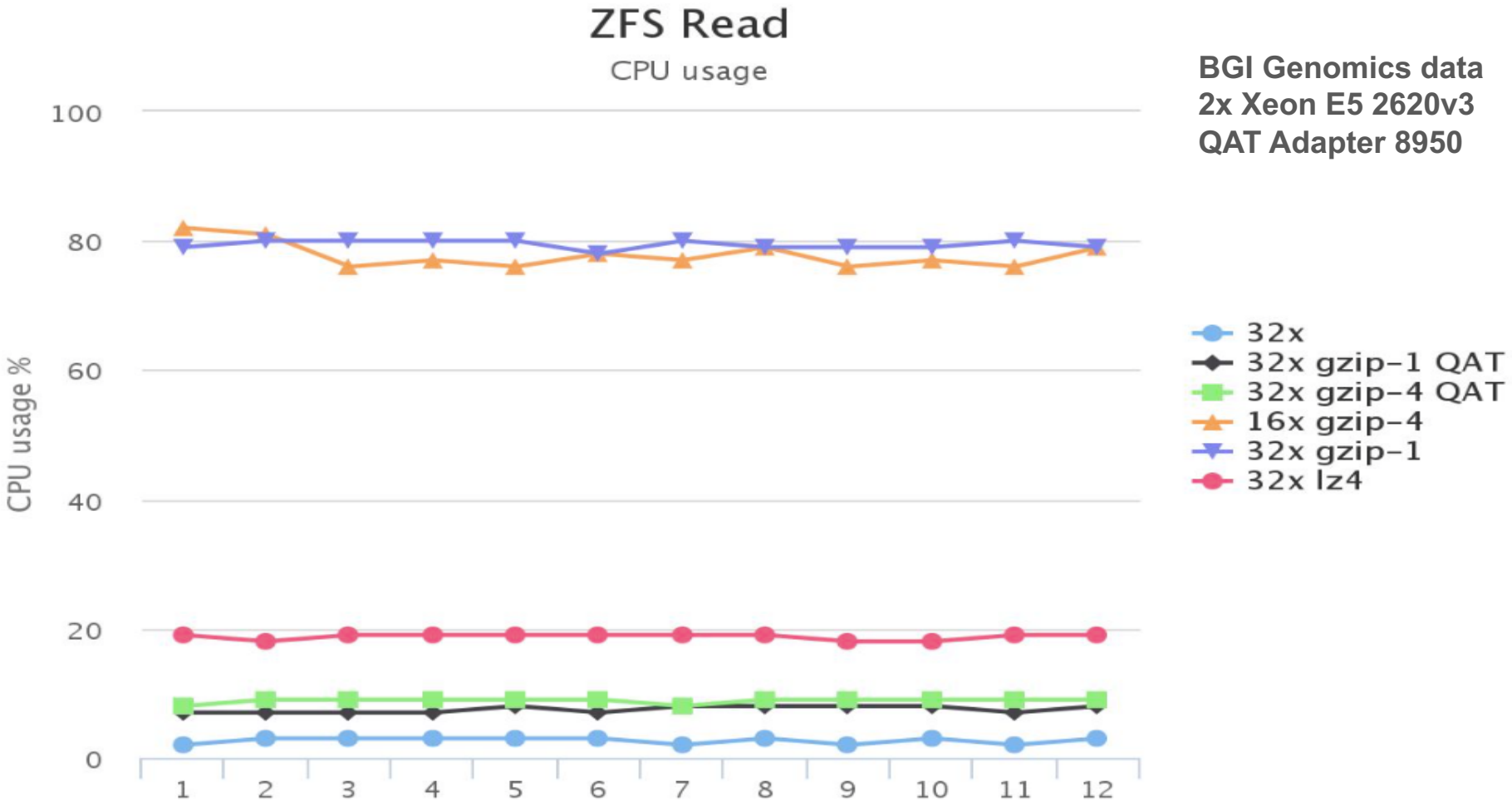
QAT integrated with ZFS 0.7.0

- Not built unless QAT libraries installed

Benchmark shows ZFS local perf

- Data from Beijing Genomics Institute
- 2 Intel® Xeon E5 2620v3 + QAT 8950





BGI Genomics data
2x Xeon E5 2620v3
QAT Adapter 8950

Fault Management Architecture ZED Integration (PR#4673 Intel 0.7)

Fault Management Architecture (FMA) correlates events

- Repeated checksum errors, or totally failed drives, ...

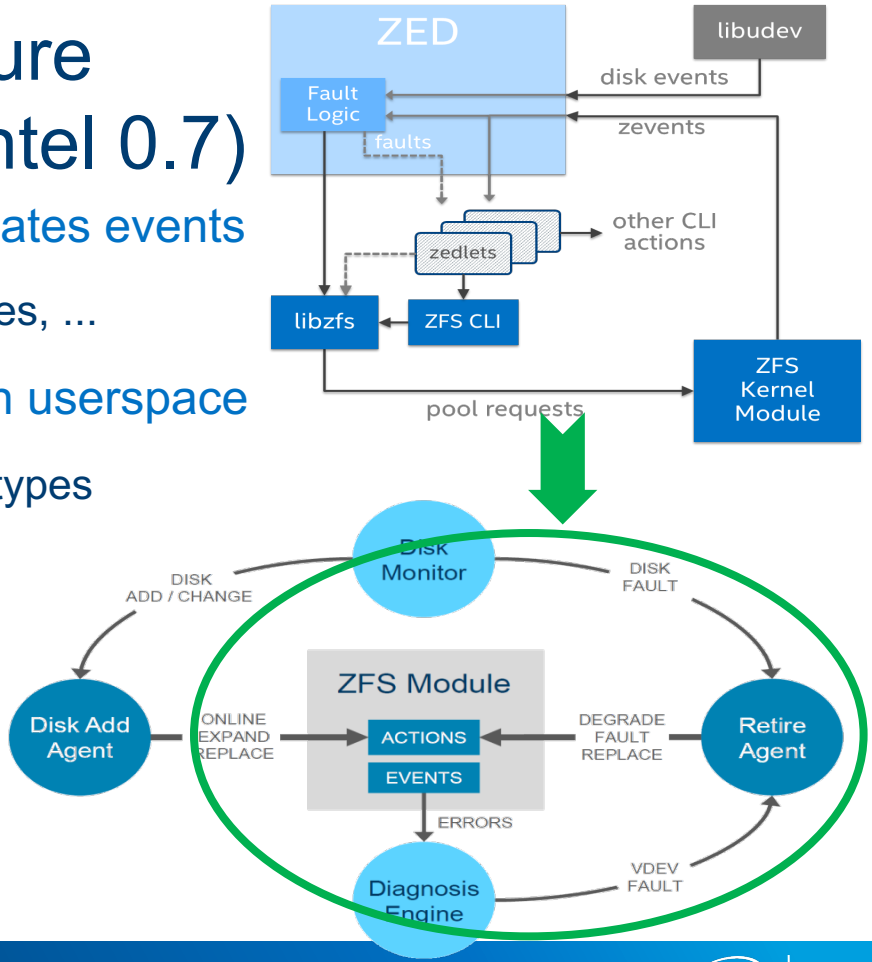
ZFS Event Daemon (ZED) handles actions in userspace

- zedlets (scripts) run based on registered event types
- Alert admins of failures, mark disks offline
- Auto-replace bad drive with hot spare

SCSI Enclosure Services handles JBODs

- Illuminate enclosure LED blinkenlights

ZED/FMA integration with IML in 4.1 release



Declustered Parity RAID (PR#3497 Intel, 0.8)

dRAID with Distributed Hot Space

RAID Data+Parity separated from drive count

- RAID stripes use pseudo-random permutation across drives
- Each permutation is repeated, but doesn't cross block boundary

Hot spare drive(s) mixed with D+P drives

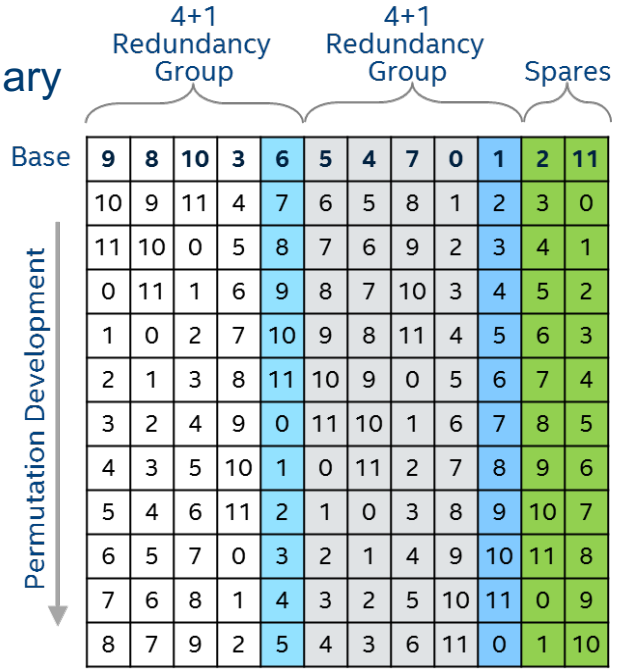
- Add bandwidth/IOPS of spares, use space from other drives

Resilver across all drives in zpool

- Potentially improve performance by `O(num_vdevs)`
- Shorter risk window with failed drive before rebuild finished

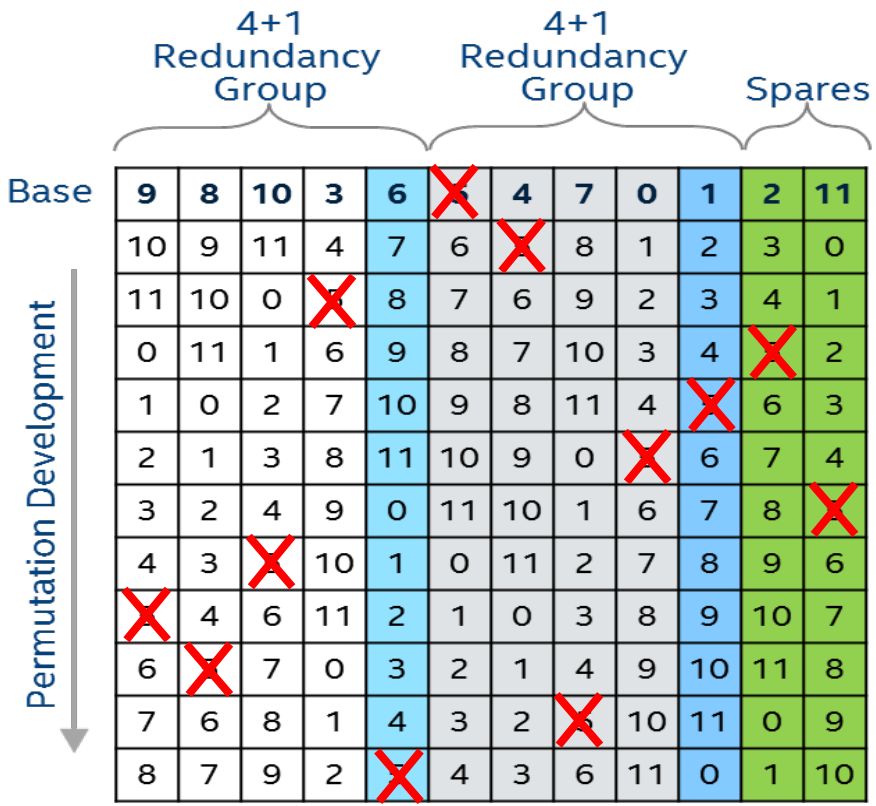
Sequential rebuild scans metaslabs for free space

- Fixed alignment of RAID chunks allows parity reconstruction
- Sequential drive access speeds rebuild, unlike RAID-z
- Skipping free space speeds rebuild, unlike traditional RAID



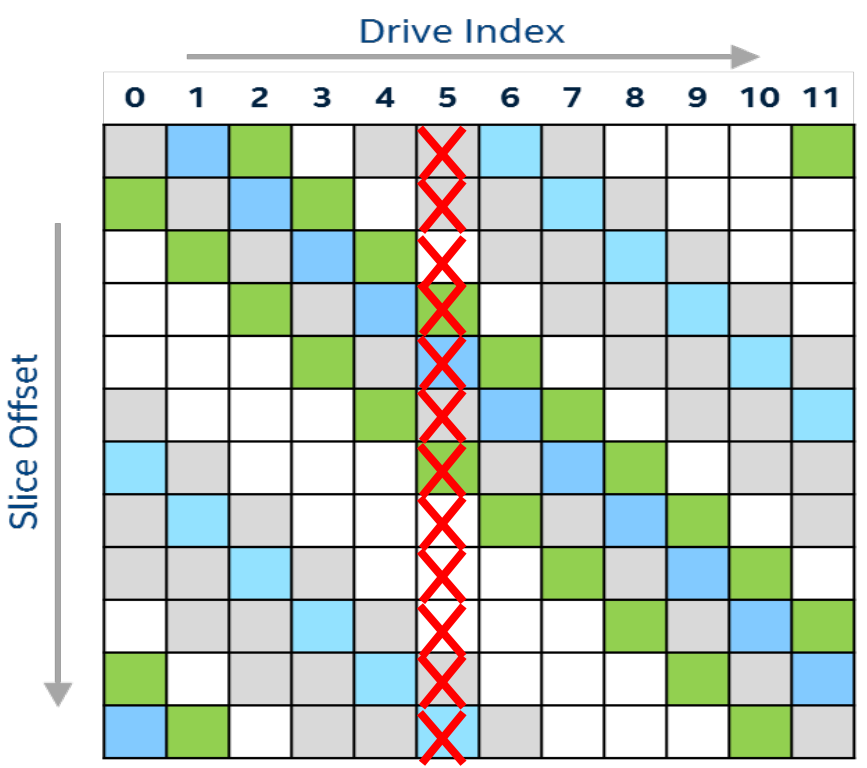
Base											
9	**8**	**10**	**3**	**6**	**5**	**4**	**7**	**0**	**1**	**2**	**11**
10	9	11	4	7	6	5	8	1	2	3	0
11	10	0	5	8	7	6	9	2	3	4	1
0	11	1	6	9	8	7	10	3	4	5	2
1	0	2	7	10	9	8	11	4	5	6	3
2	1	3	8	11	10	9	0	5	6	7	4
3	2	4	9	0	11	10	1	6	7	8	5
4	3	5	10	1	0	11	2	7	8	9	6
5	4	6	11	2	1	0	3	8	9	10	7
6	5	7	0	3	2	1	4	9	10	11	8
7	6	8	1	4	3	2	5	10	11	0	9
8	7	9	2	5	4	3	6	11	0	1	10

Permutation Layout



Base												
0	9	8	10	3	6	5	4	7	0	1	2	11
1	10	9	11	4	7	6	5	8	1	2	3	0
2	11	10	0	5	8	7	6	9	2	3	4	1
3	0	11	1	6	9	8	7	10	3	4	5	2
4	1	0	2	7	10	9	8	11	4	5	6	3
5	2	1	3	8	11	10	9	0	5	6	7	4
6	3	2	4	9	0	11	10	1	6	7	8	5
7	4	3	5	10	1	0	11	2	7	8	9	6
8	5	4	6	11	2	1	0	3	8	9	10	7
9	6	5	7	0	3	2	1	4	9	10	11	8
10	7	6	8	1	4	3	2	5	10	11	0	9
11	8	7	9	2	5	4	3	6	11	0	1	10

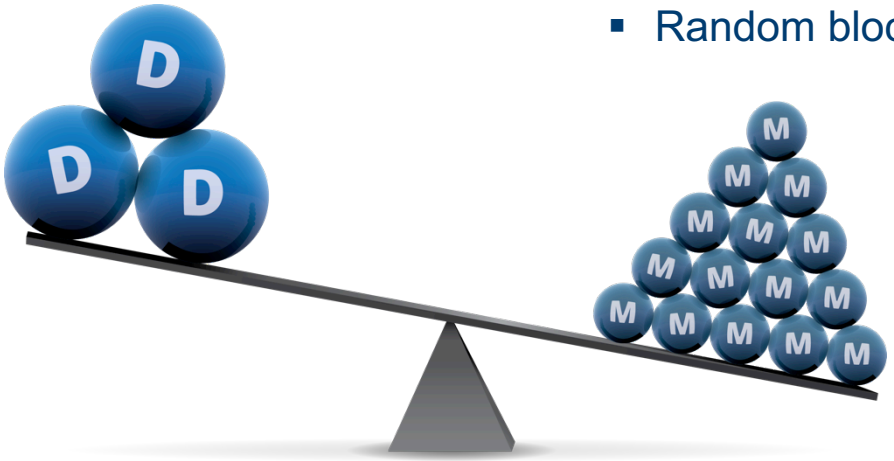
Drive Layout



Metadata Isolation from Data

File Data

- Large blocks (up to 16MB)
- Free space fragmentation
- High throughput
- Large capacity
- Sequential
- RAID-Z2
- Typically HDD



Metadata

- Transient lifetime (especially COW)
- Need fast performance for scrub
- Random block access
  - Small (<32KB)
  - Higher IOPS
  - Lower latency
  - Mirror
  - SSD preferred

Metadata Allocation Class (PR#3779 Intel, 0.8)

Virtual Devices divided into *Metaslabs*

- Metaslabs belong to an *allocation class*
- Pluggable allocators for data types

Metadata Allocation Class (MAC) uses

- Existing allocation class mechanism
- ZIO allocation stage / block typing
- Allocation policies tied to class

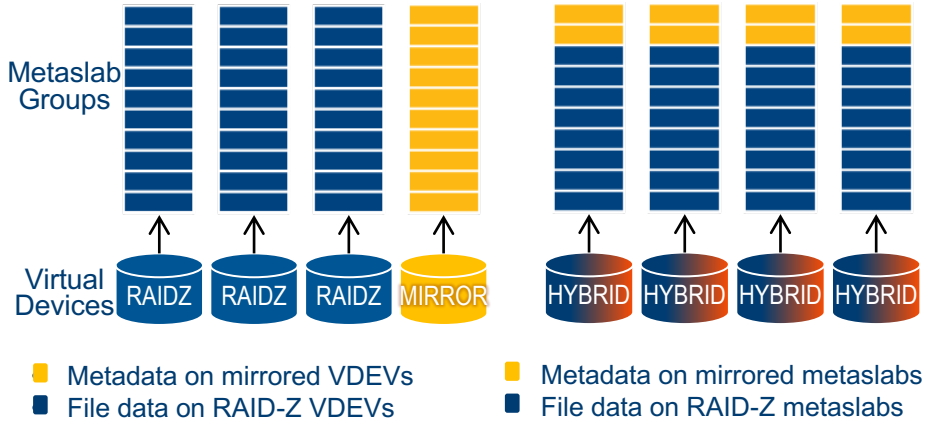
Use dedicated VDEV or hybrid slabs

- Hybrid metaslabs dynamically initialized

Avoids free space fragmentation

Avoid IO contention on dedicated VDEV

Optionally store small file data in MAC



Other ZFS Developments of Interest

ZFS Object Index repair (OI Scrub) (LU-7585, Intel Lustre 2.11)

- MDT Object Index and FID rebuild after corruption/bugs or tar/rsync backup/restore
- MDT/OST migration with tar/rsync ldiskfs backup and restore to ZFS

ZFS on-disk encryption (PR#5769, Datos, ZFS 0.7)

- Tree-based per-block encryption
- Lustre needs a mechanism for managing keys for targets (MGS?, IML?, other external tools?)

Project quota accounting (PR#6260, Intel, ZFS 0.8)

- Project ID for quota accounting independent of file access (UID/GID)
- Project ID inherited from parent dir, compatible with ldiskfs/XFS interfaces

Sequential scrub/resilver (PR#6256, Nexenta, ZFS 0.8)

- Reduce HDD verification/rebuild times by ordering tree traversal to minimize seeks

Miscellaneous

Lustre 2.10.1 and Lustre 2.11 updated from ZFS 0.6.5.9 to ZFS 0.7.1

- Critical bug in ZFS 0.7.0 with `dnodesize=auto`

ZFS 0.8.0 to be released much quicker than 0.7.0 was

- 0.7.0 was released 53 months after 0.6.0 and 21 months after 0.6.5
- Lustre 2.10.x/2.11 will be updated to build with 0.8.x, even if not default

Lots of other interesting work underway, too much to list it all here

Legal Information

All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest Intel product specifications and roadmaps

Tests document performance of components on a particular test, in specific systems. Differences in hardware, software, or configuration will affect actual performance. Consult other sources of information to evaluate performance as you consider your purchase. For more complete information about performance and benchmark results, visit http://www.intel.com/performance.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. No computer system can be absolutely secure. Check with your system manufacturer or retailer or learn more at http://www.intel.com/content/www/us/en/software/intel-solutions-for-lustre-software.html.

Intel technologies may require enabled hardware, specific software, or services activation. Check with your system manufacturer or retailer.

You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.

A "Mission Critical Application" is any application in which failure of the Intel Product could result, directly or indirectly, in personal injury or death. SHOULD YOU PURCHASE OR USE INTEL'S PRODUCTS FOR ANY SUCH MISSION CRITICAL APPLICATION, YOU SHALL INDEMNIFY AND HOLD INTEL AND ITS SUBSIDIARIES, SUBCONTRACTORS AND AFFILIATES, AND THE DIRECTORS, OFFICERS, AND EMPLOYEES OF EACH, HARMLESS AGAINST ALL CLAIMS COSTS, DAMAGES, AND EXPENSES AND REASONABLE ATTORNEYS' FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PRODUCT LIABILITY, PERSONAL INJURY, OR DEATH ARISING IN ANY WAY OUT OF SUCH MISSION CRITICAL APPLICATION, WHETHER OR NOT INTEL OR ITS SUBCONTRACTOR WAS NEGLIGENT IN THE DESIGN, MANUFACTURE, OR WARNING OF THE INTEL PRODUCT OR ANY OF ITS PARTS.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined". Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

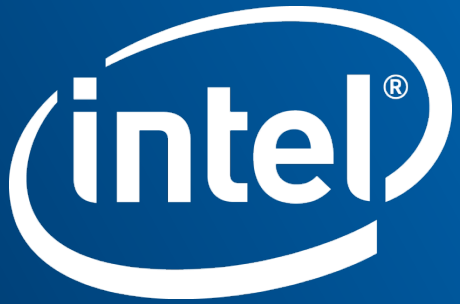
No electrons were harmed during the production of these slides.

Intel and the Intel logo are trademarks of Intel Corporation in the U.S. and/or other countries.

* Other names and brands may be claimed as the property of others.

Hardware Used in mds-survey Benchmarks

- 2 x Intel(R) Xeon(R) CPU E5-2660 v2 @ 2.20GHz – 20 cores
- 64GB RAM
- 3 x 500GB HDD


intel®