ZFS*: Metadata Performance

LAD'16

Alexey Zhuravlev, HPDD, Intel

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.

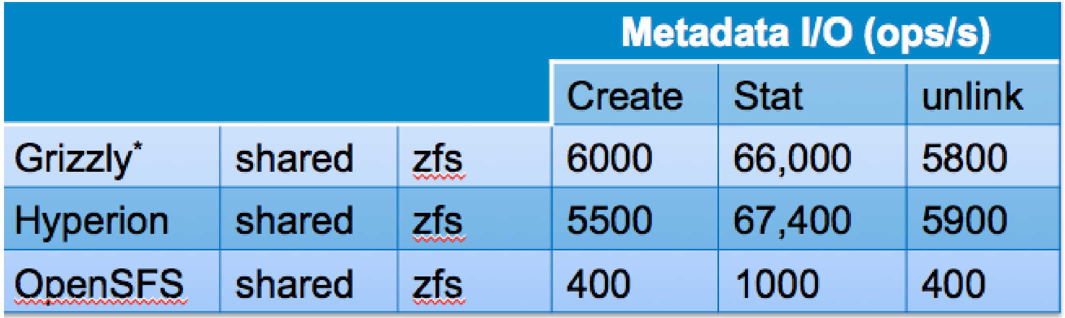
Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined". Intel reserves these for future definition and shall have no responsibility whatsoever 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.

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.

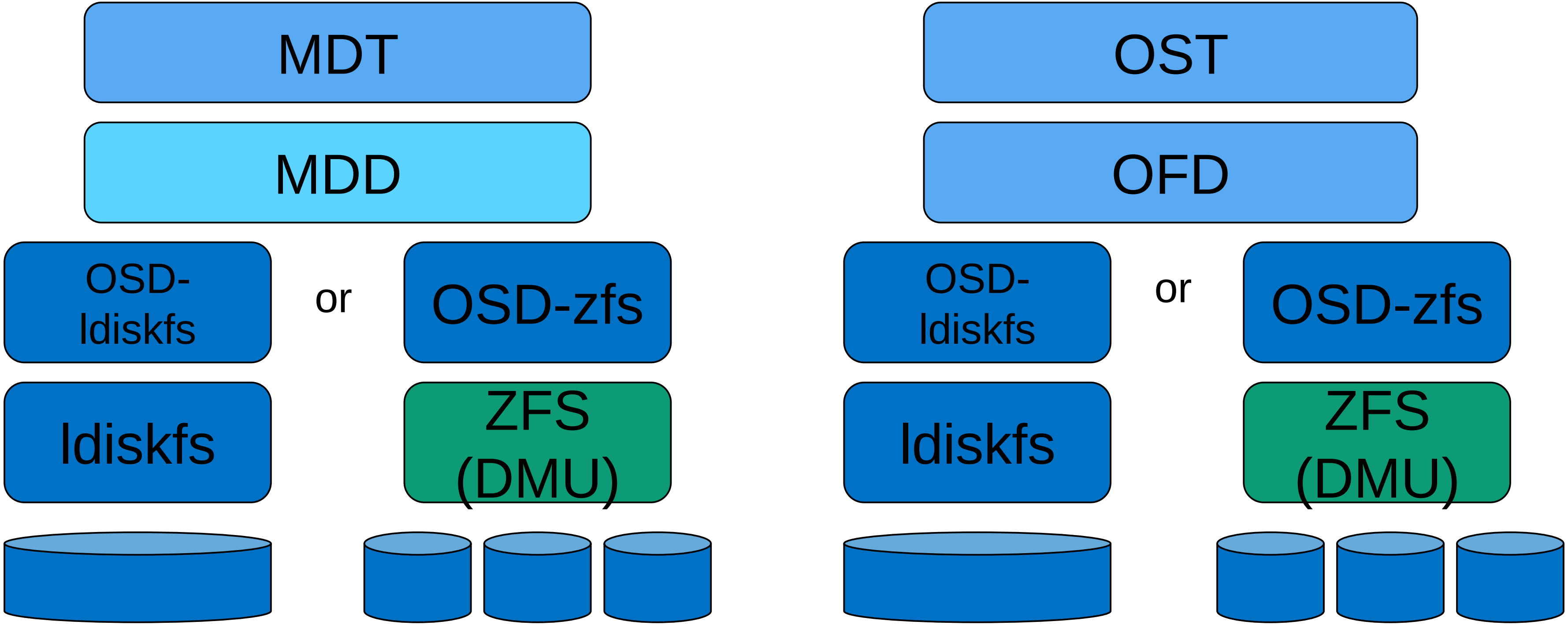
ZFS metadata performance

- Initial focus on correctness, can now focus on performance
- Performance often reported to be below expectations
- Internal benchmarks confirm this



			Metadata I/O (ops/s)		
			Create	Stat	unlink
Grizzly*	shared	zfs	6000	66,000	5800
Hyperion	shared	zfs	5500	67,400	5900
OpenSFS	shared	zfs	400	1000	400

Lustre* on ZFS - Server Layering



MDT

MDD

OSD-ldiskfs or OSD-zfs

ldiskfs

ZFS (DMU)

OST

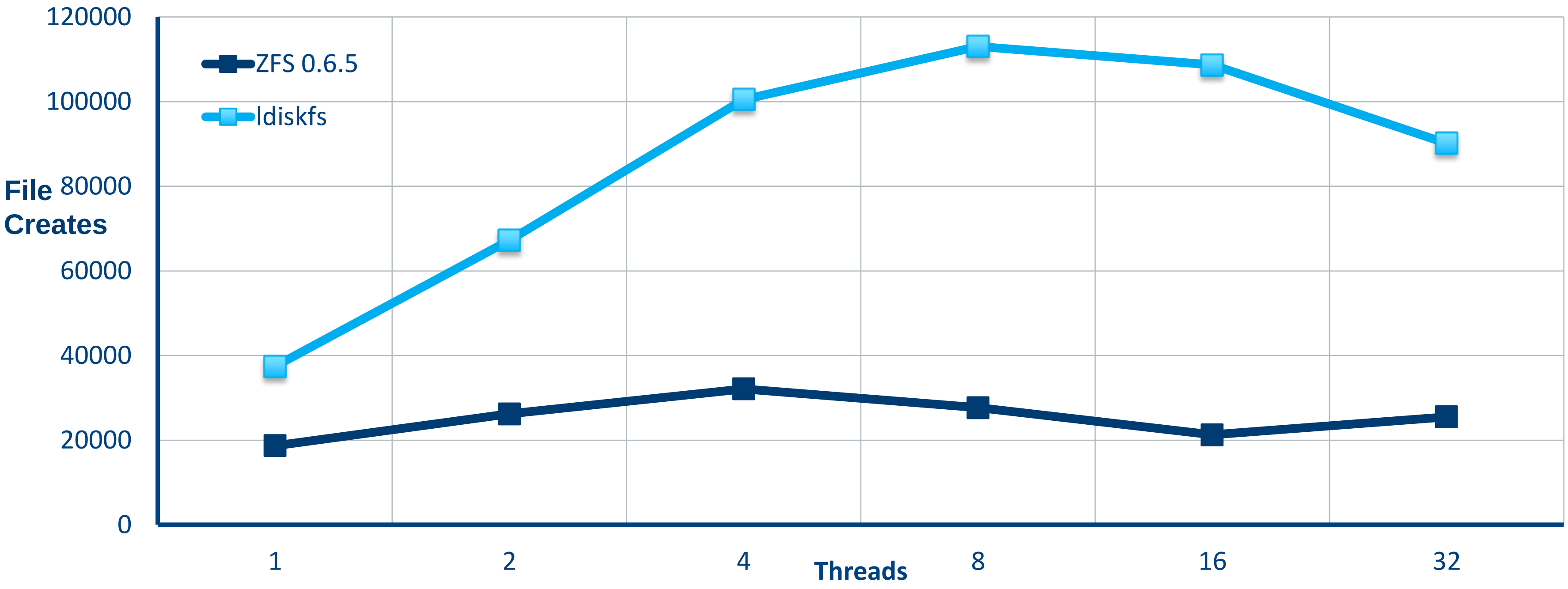
OFD

OSD-ldiskfs or OSD-zfs

ldiskfs

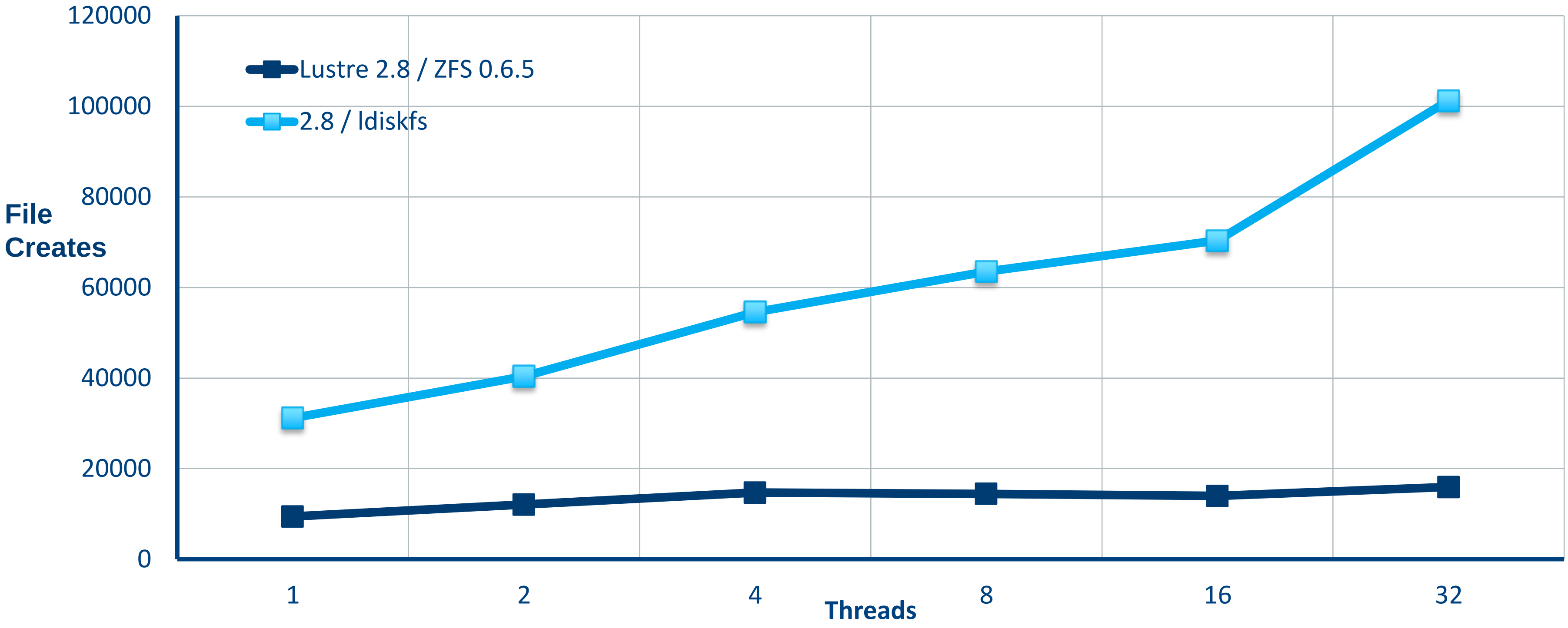
ZFS (DMU)

ZFS vs ldiskfs: 1M creates, directory/thread



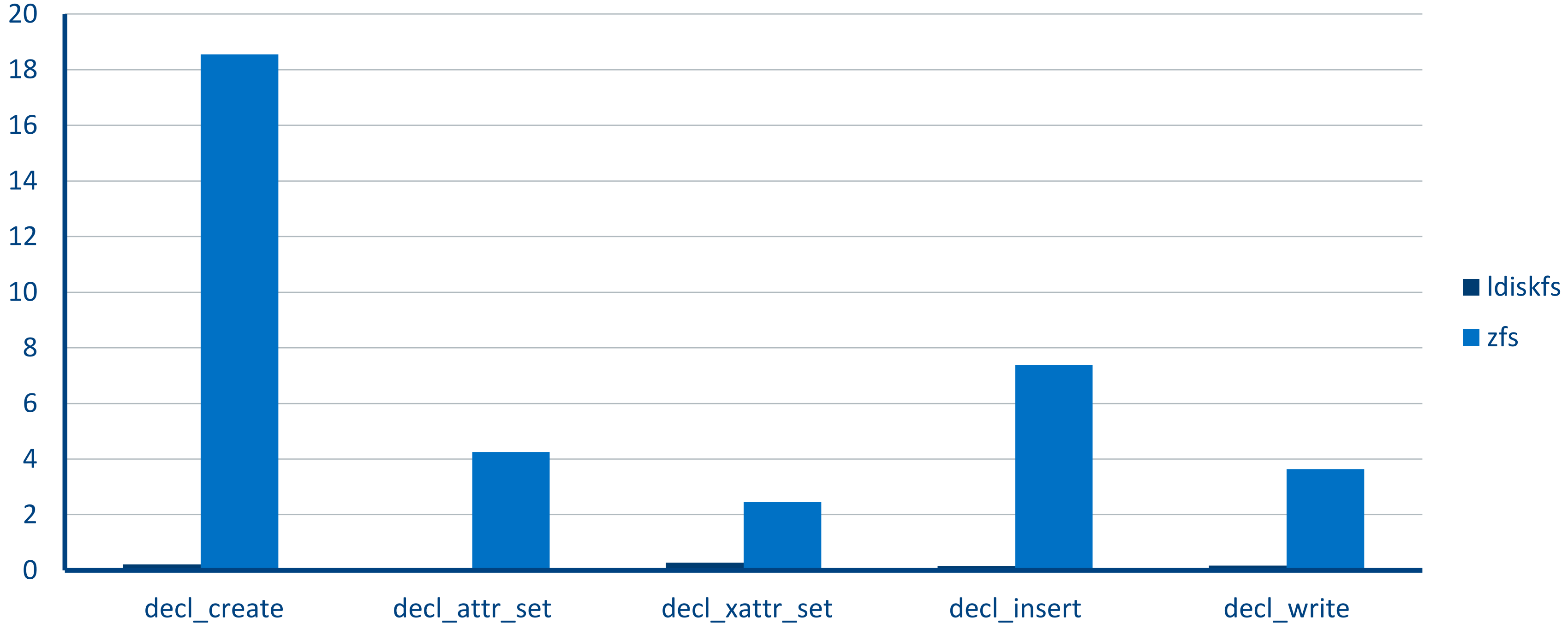
Hardware specification on slide 22

Lustre: mds-survey



Hardware specification on slide 22

Declaration time: ZFS vs ldiskfs

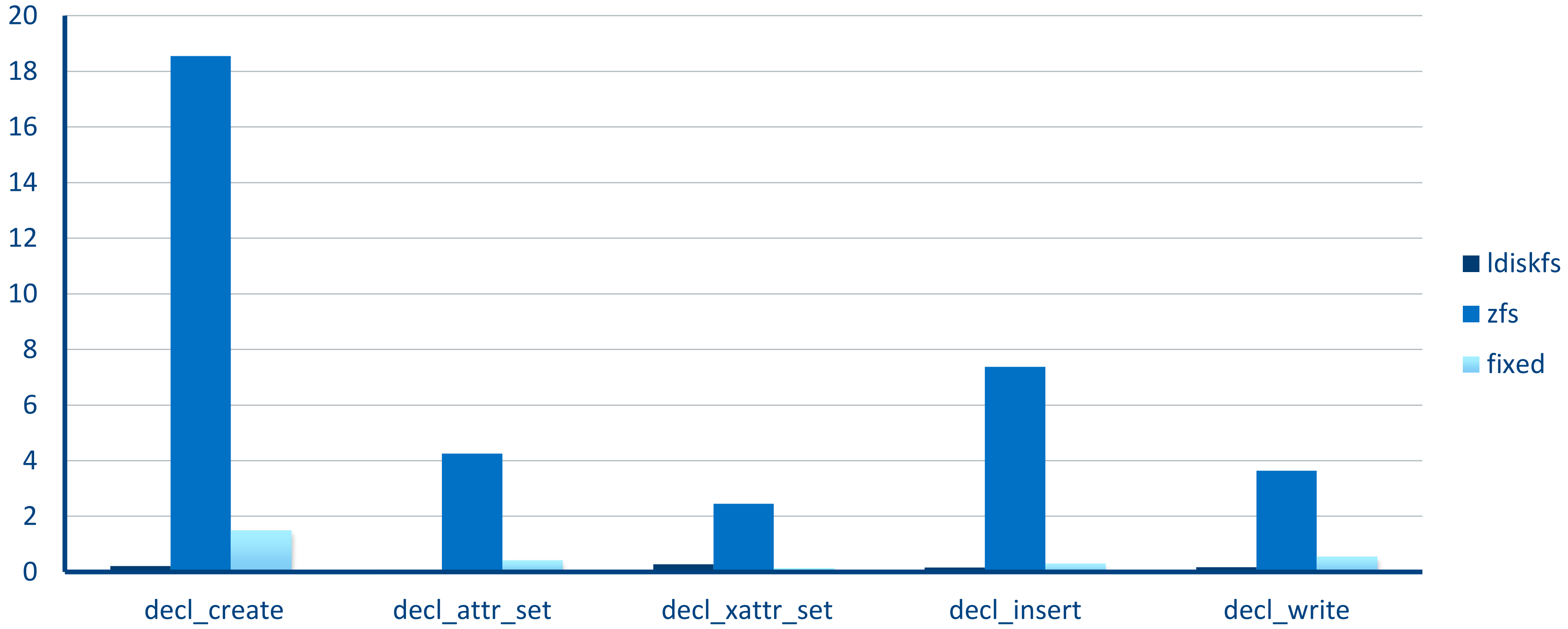


Hardware specification on slide 22

Declarations: expensive

- The more specific, the more expensive
  - dmu_tx_hold_zap() does lookup if name specified
  - dmu_tx_count_write() may check few blocks
  - few calls to dbuf_find()
  - few ZIOs allocated and destroyed with no real need
  - say, 100K / second
- Lustre uses very specific declarations
  - – Which is not required

Declaration time: fixed in LU-7898, landed to 2.9



Hardware specification on slide 22

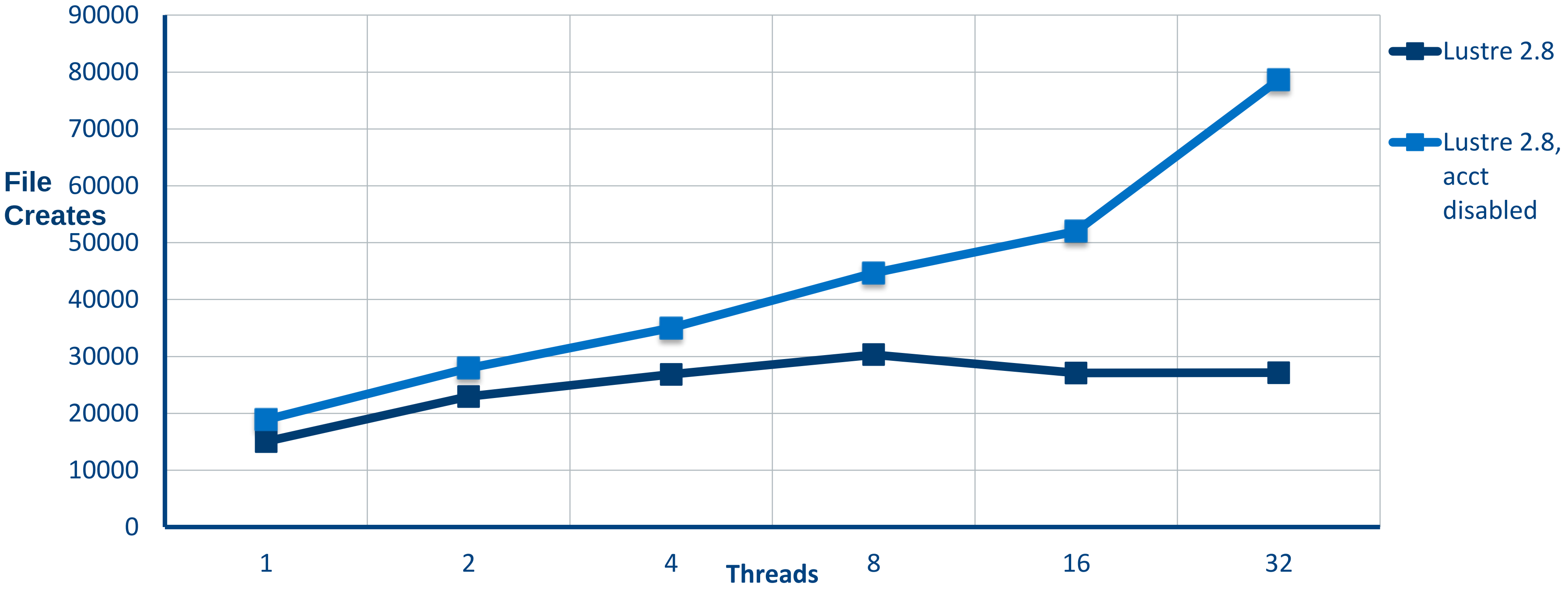
ZFS: large dnodes

- zfs 0.6.5 supports 512 bytes dnodes
- all EAs Lustre need (LMA, LinkEA, LOV, VBR) do not fit
  - Extra 2 4K blocks are allocated (so called *spill block* + redundant copy)
  - 8+ GBs to write to create 1M files
- Large dnode patch landed to ZFS master (for inclusion in 0.7 release)
  - Dnode size can vary: 0.5K to few K
  - 1 GB to write to create 1M files
  - Half head seeks to access files (no need to read spill block)

Dnode accounting (ZFS dnode quota)

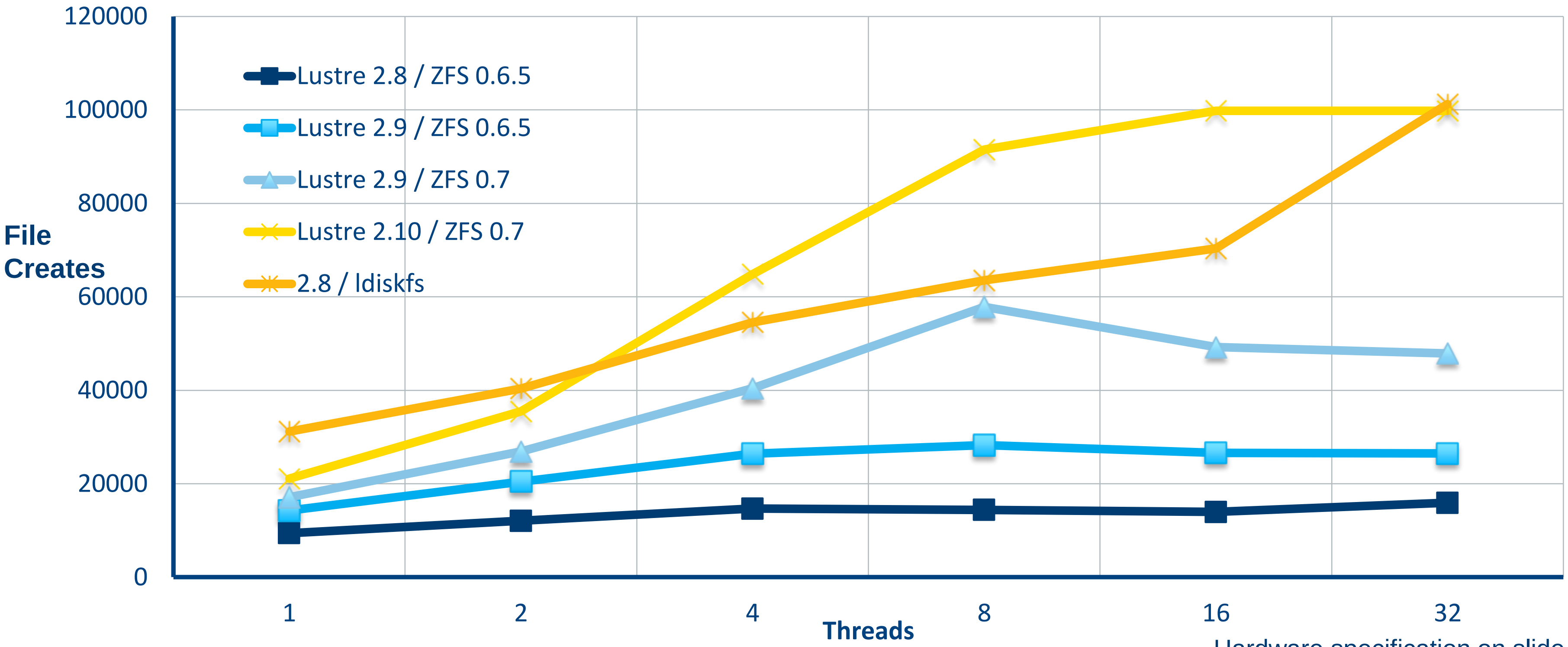
- ZFS didn't support dnode accounting
- Lustre implements own primitive schema to support file quota
  - Doesn't scale well
  - Update 2 accounting files on every file creation – very expensive
- Intel created patch to add native dnode accounting to ZFS (LU-2435)
  - Accounting is implemented in the syncing thread
  - Updates accounting file once
  - Almost ready for landing

Dnode accounting: performance estimation



Hardware specification on slide 22

Lustre: step by step (mds-survey)



Hardware specification on slide 22

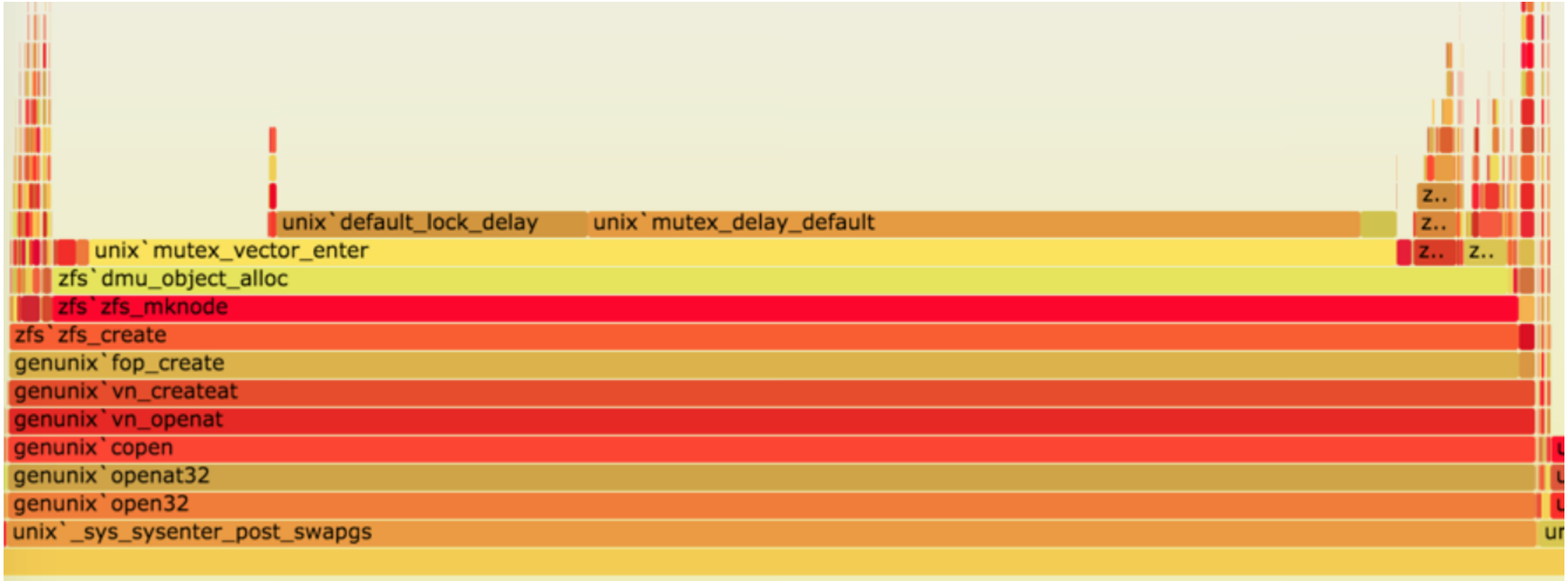
ZFS: more optimizations coming

- Active collaboration with upstream ZFS community
- Special thanks to Matt Ahrens @ Delphix who identified multiple bottlenecks
- Few experiments made to see potential improvement
  - Graph in a few slides

DMU object allocation

- dmu_object_alloc() serializes all object allocations with a single mutex
- Rescan every few thousand allocations
  - to reuse potentially freed dnodes
  - so-called *revisit problem*
- Every allocation looks up dbuf in the global hash table
- Then allocate and fill in-core structures from dbuf
- Summary:
  - very expensive
  - doesn't scale

ZFS: profiling with FlameGraph



ZFS: object allocation

- Possible improvement:
  - Smarter revisit algorithm to skip recently allocated dnodes
  - Concurrent allocation with metadnode broken into chunks
  - Cache last used dbuf to improve single-thread performance

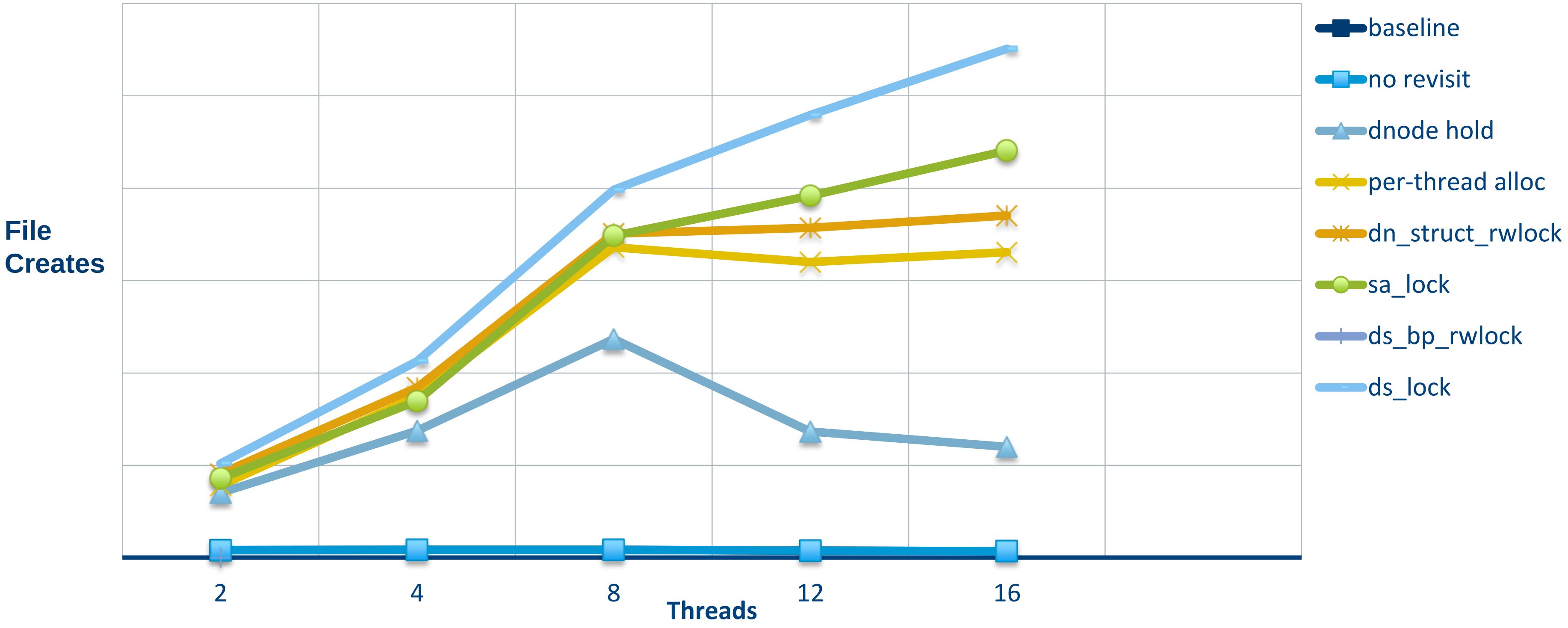
Look at the source

- DMU API uses dnode numbers instead of dnode_t *
  - Even internally
- Results in many dbuf lookups
- Doesn’t scale well with cores
- Lustre file create needs 14 lookups at least (even with all the fixes to Lustre)
- Options:
  - global dnode cache – additional locking
  - local per-cpu/transaction – additional lookups

ZFS: possible improvements

- dn_struct_rwlock
  - Used to access dnode fields, taken mostly shared
- sa_lock
  - Used to map specific set of EAs into a number encoding this set
  - Doesn’t need to be exclusive
- ds_bp_rwlock
  - Used for debugging, can be disabled
- ds_lock in dnode_setdirty()
  - Use multilist

ZFS: performance estimations



Summary

- Lustre 2.9 improves metadata performance with ZFS
  - 2x in some cases using same ZFS 0.6.5.7
  - ZFS 0.7 and Lustre 2.10 should bring ZFS in line (or even ahead) of ldiskfs
- Significantly better numbers are expected in the future
  - As ZFS gets fixes for the problems discussed above

Test configuration

- 2 x Intel(R) Xeon(R) CPU E5-2660 v2 @ 2.20GHz – 20 cores
- 64GB RAM
- 3 x 500GB local SATA HDD 7200 RPM
- CentOS 7.2.1511
- 3.10.0-327.28.2.el7 kernel (RHEL7)
- No remote clients, just local MDS testing (mds-survey script)

intel®