IEEE MSST
Asilomar Symposium
April 2012

Lustre Future Development

- Andreas Dilger

  Principal Lustre Engineer
  Whamcloud, Inc.
  adilger@whamcloud.com

What is Lustre?

- A scalable distributed parallel filesystem
- Hardware agnostic
  - Can use commodity servers, storage, and networks
  - Many vendors also integrate with their hardware/tools
- Open source software (GPL v2)
  - Ensures no single company controls Lustre
  - Protects users and their storage investments
  - Large, active, motivated development community
- POSIX compliant
  - What applications expect today…
    … though Lustre is flexible for future demands
- The most widely used filesystem in HPC
  - 7 or 8 of top 10 supercomputers for many years
  - ~70 of top 100 systems in most recent Top-500

Lustre development timeline

- 1999 – Lustre project startup
- 2001 – ASCI Pathforward

- 2003 v1.0 – CFS
- 2004 v1.4 – CFS

- 2007 v1.6 – CFS/Sun

- 2009 v1.8 – Sun

- 2010 v2.0 – Oracle

- 2011 v2.1 – Whamcloud
- 2012 v2.2 – Whamcloud

lustre

CFS Cluster File Systems, Inc.

Sun microsystems

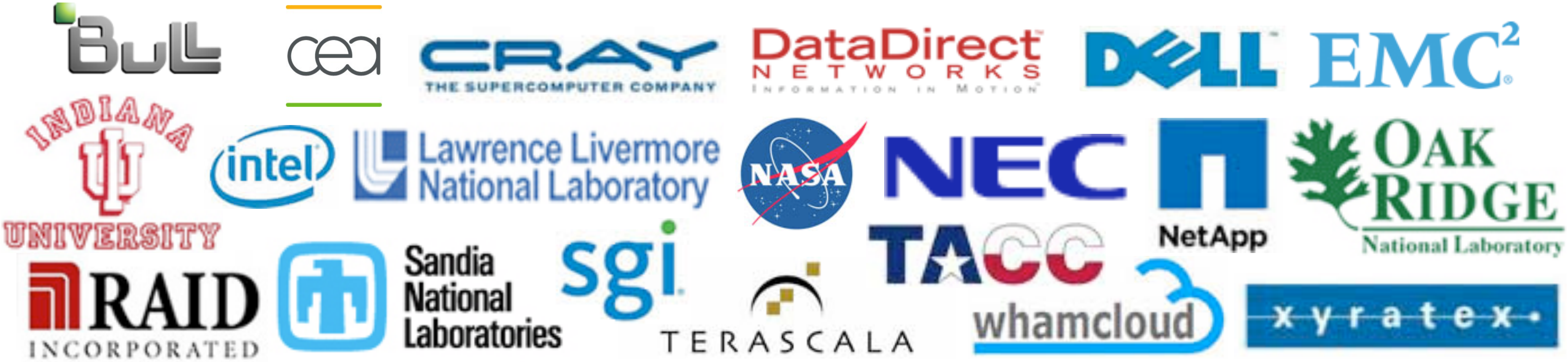
ORACLE

whamcloud

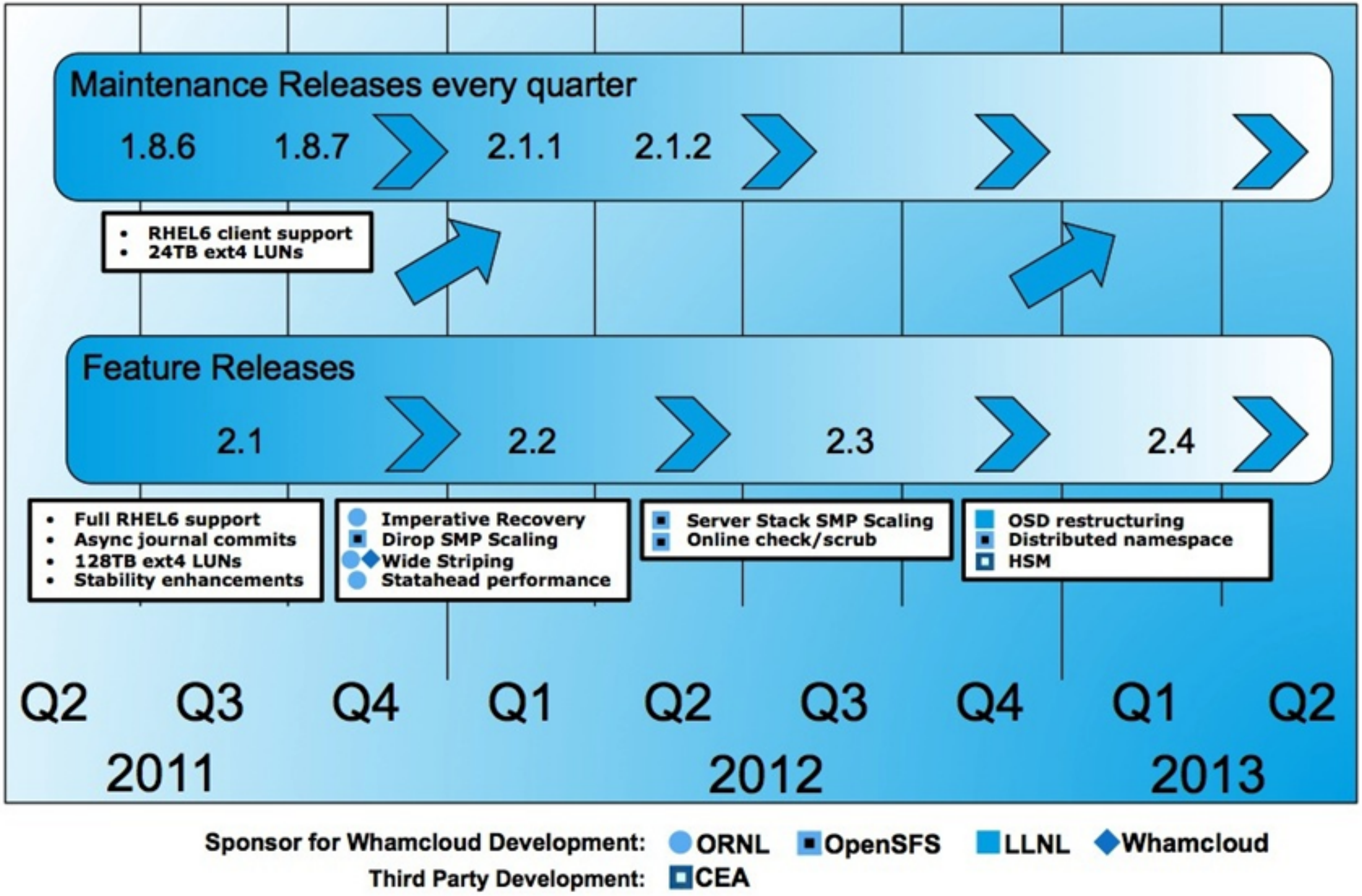
whamcloud

Lustre Community Organizations

http://www.opensfs.org http://www.eofs.eu



Community Lustre Roadmap



Maintenance Releases every quarter

1.8.6 1.8.7 2.1.1 2.1.2

- RHEL6 client support
- 24TB ext4 LUNs

Feature Releases

2.1 2.2 2.3 2.4

- Full RHEL6 support
- Async journal commits
- 128TB ext4 LUNs
- Stability enhancements

● Imperative Recovery
▪ Dirop SMP Scaling
●◆ Wide Striping
● Statahead performance

▪ Server Stack SMP Scaling
▪ Online check/scrub

■ OSD restructuring
▪ Distributed namespace
□ HSM

Sponsor for Whamcloud Development: ● ORNL ▪ OpenSFS ■ LLNL ◆ Whamcloud

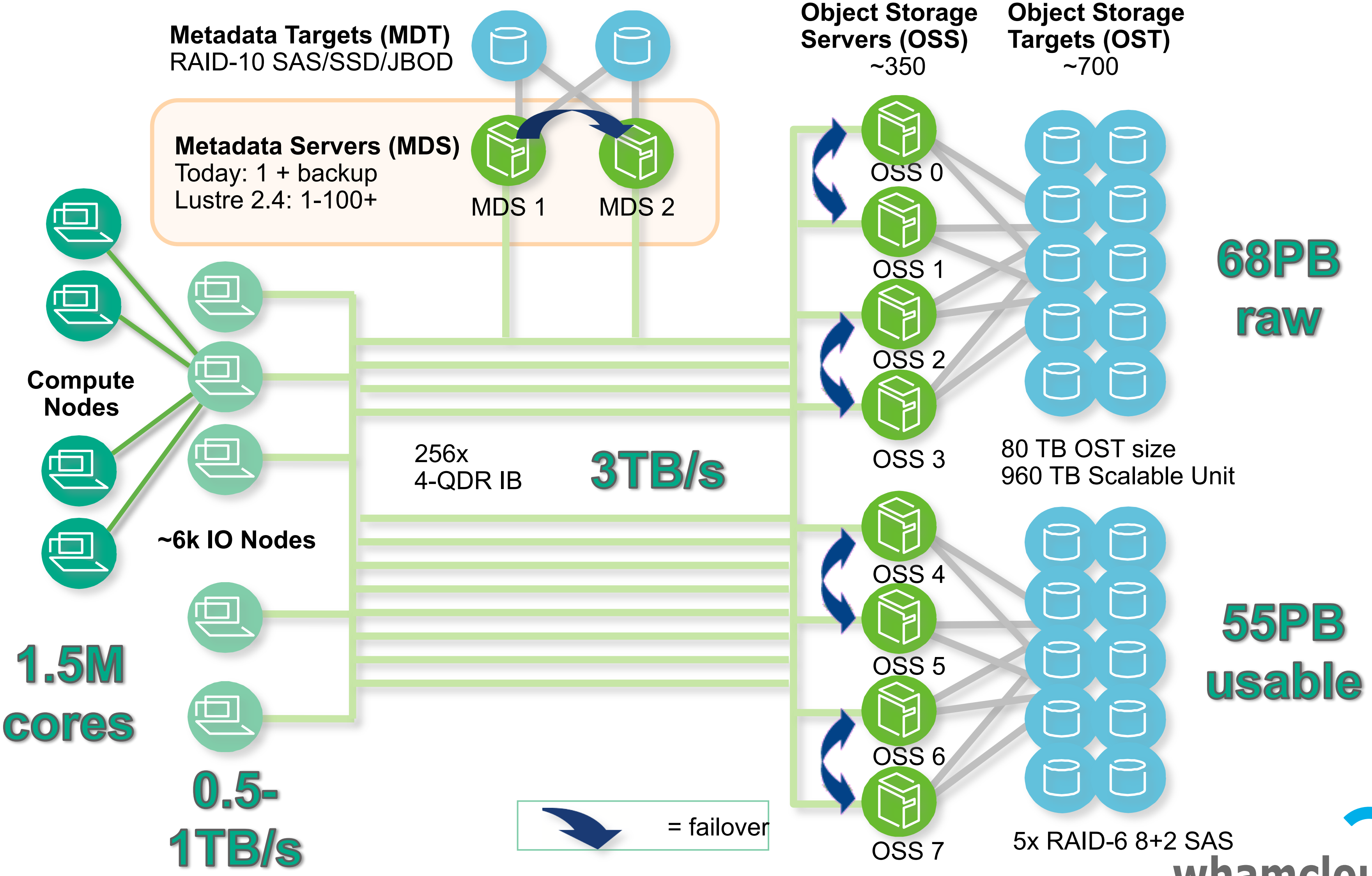
Third Party Development: □ CEA

http://wiki.whamcloud.com/display/PUB/Community+Lustre+Roadmap

Lustre Architecture Overview

- Client operations split into metadata/data
  - Each operation class goes to a dedicated server
- Metadata Server (MDS = node)
  - Stores dirs, filenames, mode, permissions, xattrs, times
  - Allocate data object(s) for file
  - MDS **NOT** needed for file IO/block allocation/file size
- Object Storage Servers (OSS = node)
  - Objects store file data, size, block count, timestamps
  - Files may be striped across N objects/storage targets
  - IO to OSTs is completely independent
- Client merges meta/data on read/stat
  - File size and timestamps remain distributed
  - POSIX is an attribute of the client, not server or protocol

LLNL Sequoia Lustre Architecture



Metadata Targets (MDT)
RAID-10 SAS/SSD/JBOD

Metadata Servers (MDS)
Today: 1 + backup
Lustre 2.4: 1-100+

Object Storage Servers (OSS)
~350

Object Storage Targets (OST)
~700

Compute Nodes

~6k IO Nodes

256x
4-QDR IB

3TB/s

68PB raw

55PB usable

1.5M cores

0.5-1TB/s

80 TB OST size
960 TB Scalable Unit

5x RAID-6 8+2 SAS

= failover

whamcloud

Lustre 2.3 and Beyond

- Lustre 2.3 (September 2012)
  - Server SMP metadata performance
  - LFSCK Online check/scrub - Internal OSD consistency
- Lustre 2.4 (March 2013)
  - OSD Restructuring (ZFS support)
  - LFSCK Online check/scrub - MDT-OST consistency
  - Distributed Namespace - Remote directories
  - HSM
- Many other projects underway
  - Not scheduled for releases until they are ready
- Lustre 2.5+ in the planning/funding stage
  - LFSCK Online check/scrub - DNE MDT-MDT consistency
  - Distributed Namespace - Shard/Stripe directories
  - Working with OpenSFS to prioritize other features
    - Object mirroring/migration
    - Storage tier management/quota/migration

Lustre+ZFS Benefits

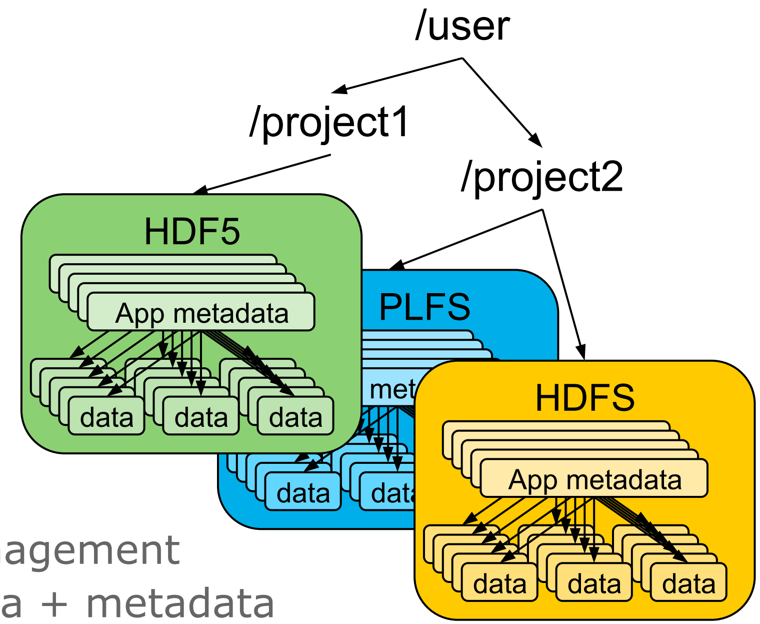
- Can leverage many features immediately
  - Robust code with 10+ years maturity
  - Data checksums on disk + Lustre checksums on network
  - Online filesystem check/scrub/repair - no more *e2fsck*!
  - Scales beyond current filesystem limits (object, filesystem)
  - Easier management of large pools of disks
  - Drive commodity JBOD storage without RAID hardware
  - Integrated with flash storage cache (L2ARC)
- More features usable by Lustre in the future
- Will be an option for Lustre 2.4 (2013)
  - http://zfsonlinux.org/lustre.html

whamcloud

Lustre HSM

- Originally developed by CEA France
- Simple archive back-end interface
- Initially supports HPSS and POSIX API
  - HPSS copytool only available to HPSS users
- Uses CEA Robin Hood for policy engine
  - Leverages Lustre ChangeLog to avoid scanning
- Infrastructure usable for other projects
  - Data migration between storage pools/tiers
  - Asynchronous data mirroring
- Planned integration into Lustre 2.4

Exascale Challenges



- APIs beyond POSIX
  - Need to be usable by applications
  - Cannot be vendor/filesystem specific
  - Leverage existing APIs/models
- Simplify data management
  - Use filesystem for user/project/job management
  - Separate namespace for application data + metadata
- Distributed Application Object Storage (DAOS)
  - Containers for application data, application metadata
  - Export object API to userspace (filesystem specific or agnostic?)
  - Integrate with higher-level data libraries (HDF5, HDFS, PLFS, etc)
- Preserve model integrity in the face of all failures
  - Very large atomic, durable transactions
  - Integrity APIs at all levels of the I/O stack
- Lustre well suited to provide this foundation

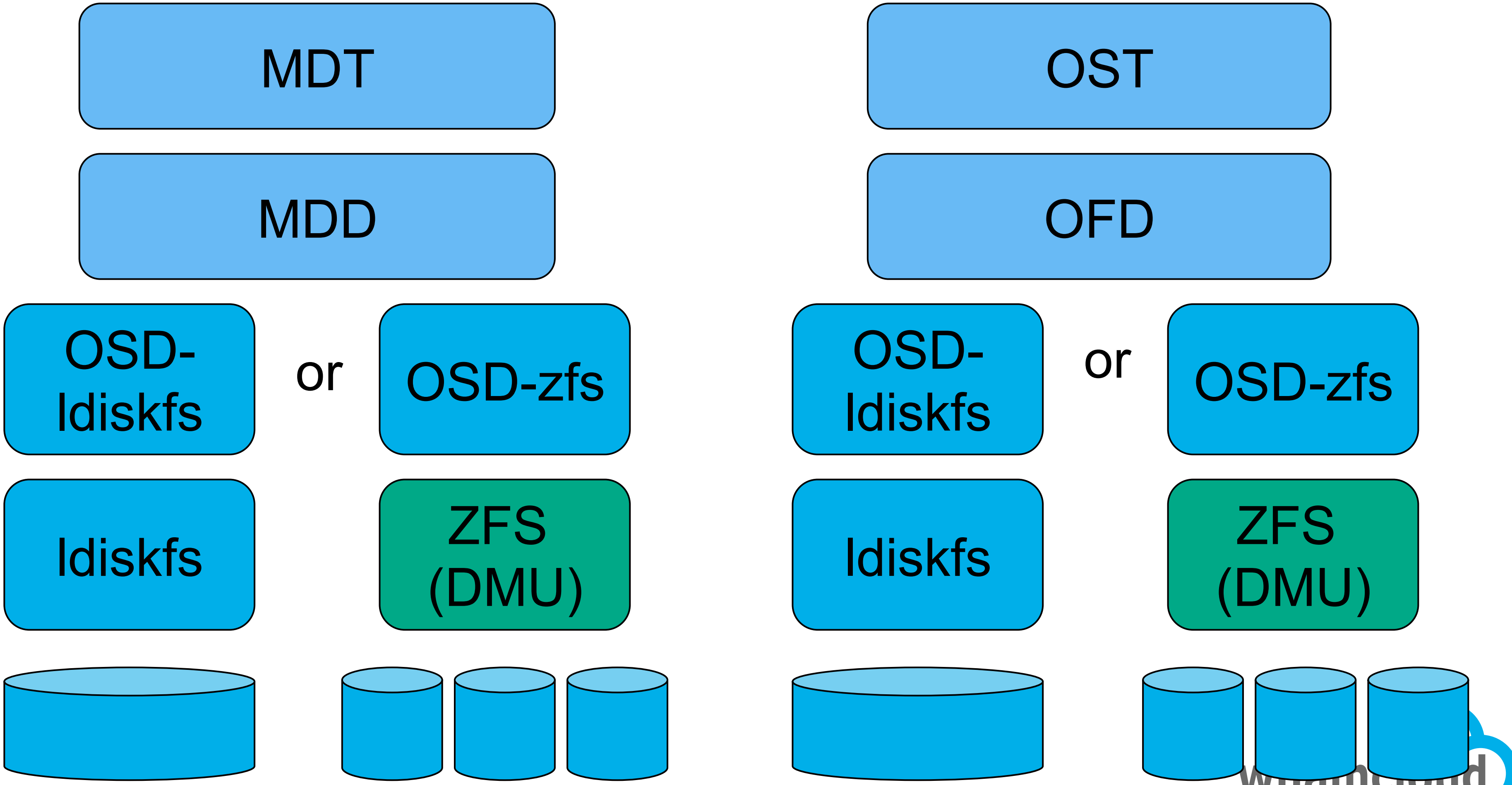
- Andreas Dilger

  Principal Lustre Engineer
  Whamcloud, Inc.
  adilger@whamcloud.com

Lustre + ZFS Implementation

- On-disk format is ZFS compatible
  - Can mount MDT/OST with Linux ZFS filesystem module
- Lustre protocol filesystem agnostic
- Integrates with Data Management Unit
  - ZFS OSD integrate with DMU engine directly (no FUSE/VFS)
  - Can manage ZFS transactions directly for Lustre recovery
- Fixed hard-coded assumptions on client
  - Assumed maximum object size was 2TB (ext3 limit)
  - Assumed OST blocksize <= PAGE_SIZE when reserving space

Lustre on ZFS - Server Layering



ZFS on Linux Licensing Concerns

- ZFS is NOT a derived work of Linux

  “It would be rather preposterous to call the Andrew FileSystem a 'derived work' of Linux, for example, so I think it's perfectly OK to have an AFS module, for example.” – Linus Torvalds

  “Our view is that just using structure definitions, typedefs, enumeration constants, macros with simple bodies, etc., is NOT enough to make a derivative work. It would take a substantial amount of code (coming from inline functions or macros with substantial bodies) to do that.” – Richard Stallman (The FSF's view)

- Companies use/support OpenSolaris ZFS
  - CDDL provides patent indemnification, unlike GPLv2