Lustre on ZFS

Andreas Dilger
Software Architect
High Performance Data Division
September, 24 2012

Introduction

Lustre on ZFS Benefits

Lustre on ZFS Implementation

Lustre Architectural Changes

Development Status

Landing Status

Preliminary Performance Results

Future Lustre/ZFS Development

Lustre on ZFS Benefits

Can leverage many ZFS features in Lustre 2.4

- 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 count/size, filesystem size)
- Easier management of many disks, commodity JBODs without RAID hardware
- Integrated with flash storage cache (L2ARC read cache)
- Optional data compression on disk can improve real-world IO performance

Other features will need extra effort to work with Lustre

http://en.wikipedia.org/wiki/Zfs

http://zfsonlinux.org/lustre.html

Lustre on ZFS Implementation

On-disk format is ZFS compatible

- Can mount MDT/OST with Linux ZFS filesystem module
- Simplifies debugging/maintenance/upgrades

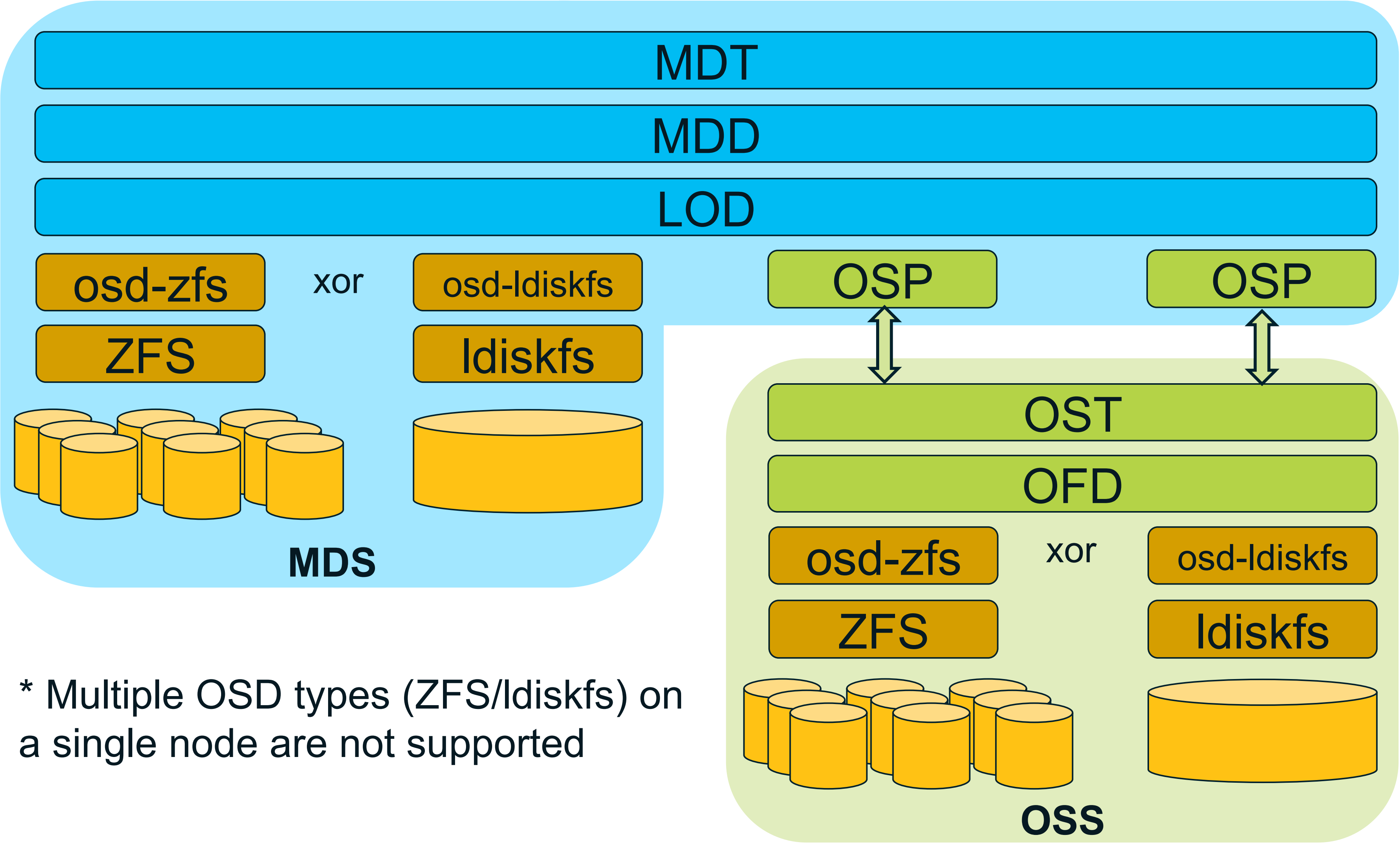
Network protocol independent of backing filesystem

- Fixed some hard-coded assumptions on client
  - Assumed maximum object size was 2TB (ext3 limit, fixed in 2.3)
  - Assumed OST blocksize <= PAGE_SIZE when reserving space

New Object Storage Device (OSD) module

- Integrates with ZFS Data Management Unit (no FUSE/VFS)
- Access ZFS transactions/features directly from Lustre

Updated MDS/OSS Module Layering



* Multiple OSD types (ZFS/ldiskfs) on a single node are not supported

Lustre Architectural Changes

Remove usage of VFS APIs

- Only access storage via OSD API

Clean up layering of MDS and OSS stacks

- Abstract MDS-to-OSS operations via OSD API
- Simplifies DNE design and implementation

Fix some longstanding MDS/OSS recovery issues

- MDS drives object destroy, avoids client failure issues

Allows ZFS support to co-exist with ldiskfs

- Potential for Btrfs OSD in the future, when it is faster/stable

Development Status

Feature development finished on Orion branch

- OSD API implemented for ZFS and finished for ldiskfs
- OST, MDT, MGT use only OSD API to access storage devices
- LOD/OSP are OSD API replacements for LOV/OSC
  - OSP proxies operations, transactions for remote OST storage
- Quota accounting and enforcement restructured for ZFS

Code has been under testing at LLNL for past year

- Development/testing clusters
- Scale testing up to ½ of Sequoia system (384 OSTs)
- Recent early deployment on ½ of Sequoia system

Landing Status

ZFS OST functionality landed for 2.3 (for testing only)

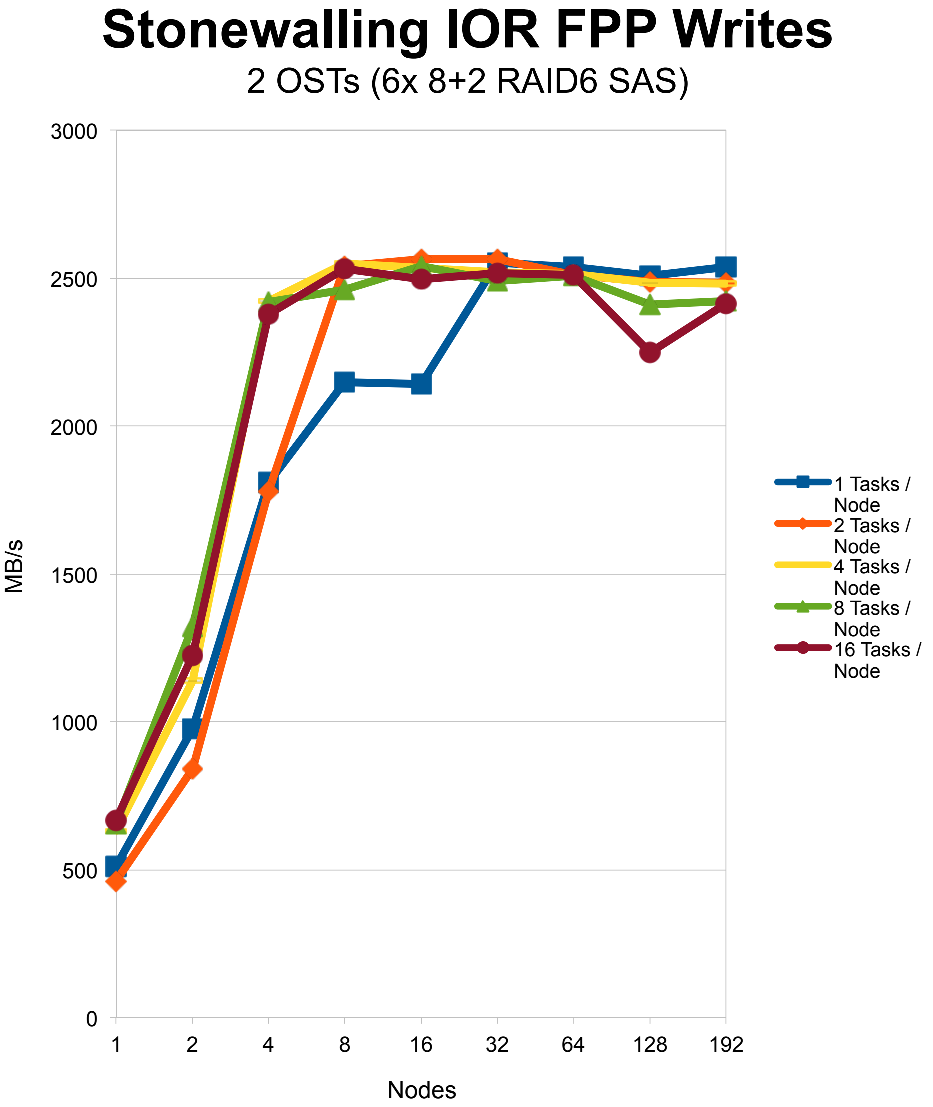
- Basic utilities support to format, mount ZFS OST filesystem

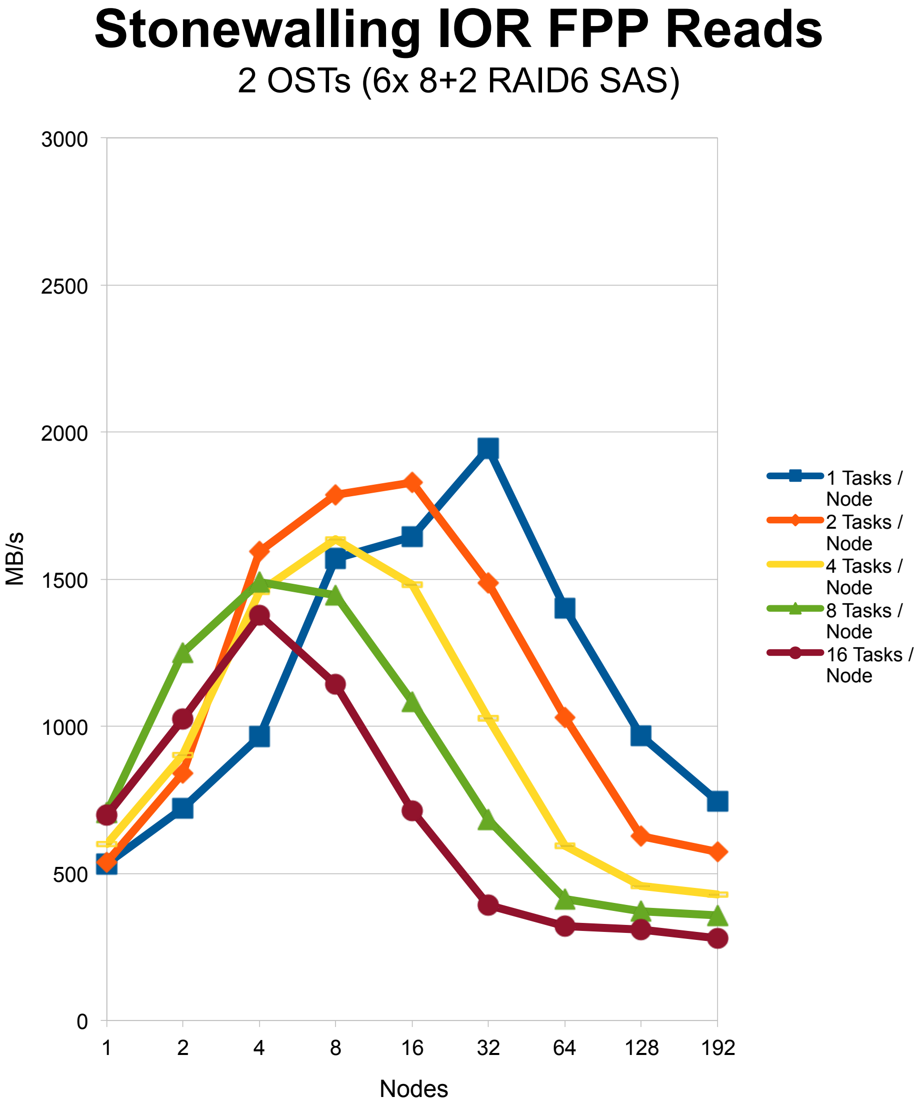
Work underway to land remaining Orion changes to master

- Utilities cleaned up for consistency between ZFS & ldiskfs
- MGS can run on ZFS
- llog functionality now landed
- MDD, LOD, OSP, quota landings underway

Will test commits to master branch with ZFS and ldiskfs

Preliminary Performance Results (1/384 scale)

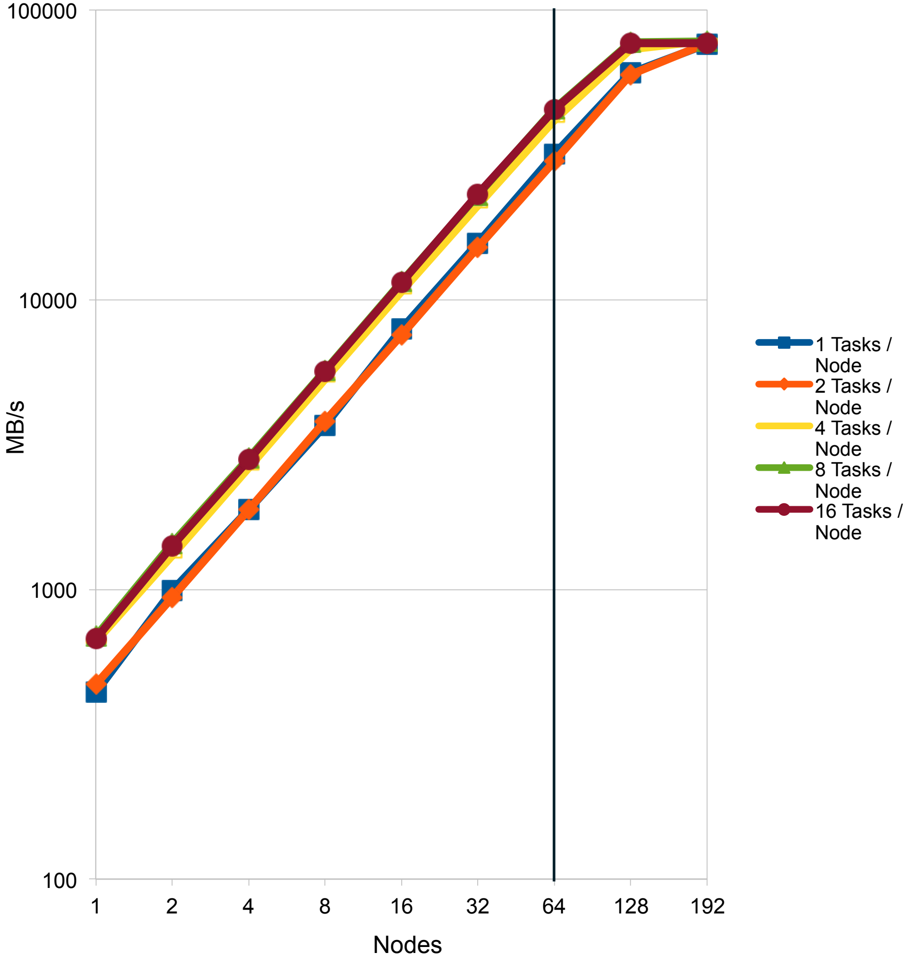




Preliminary Performance Results (1/12 scale)

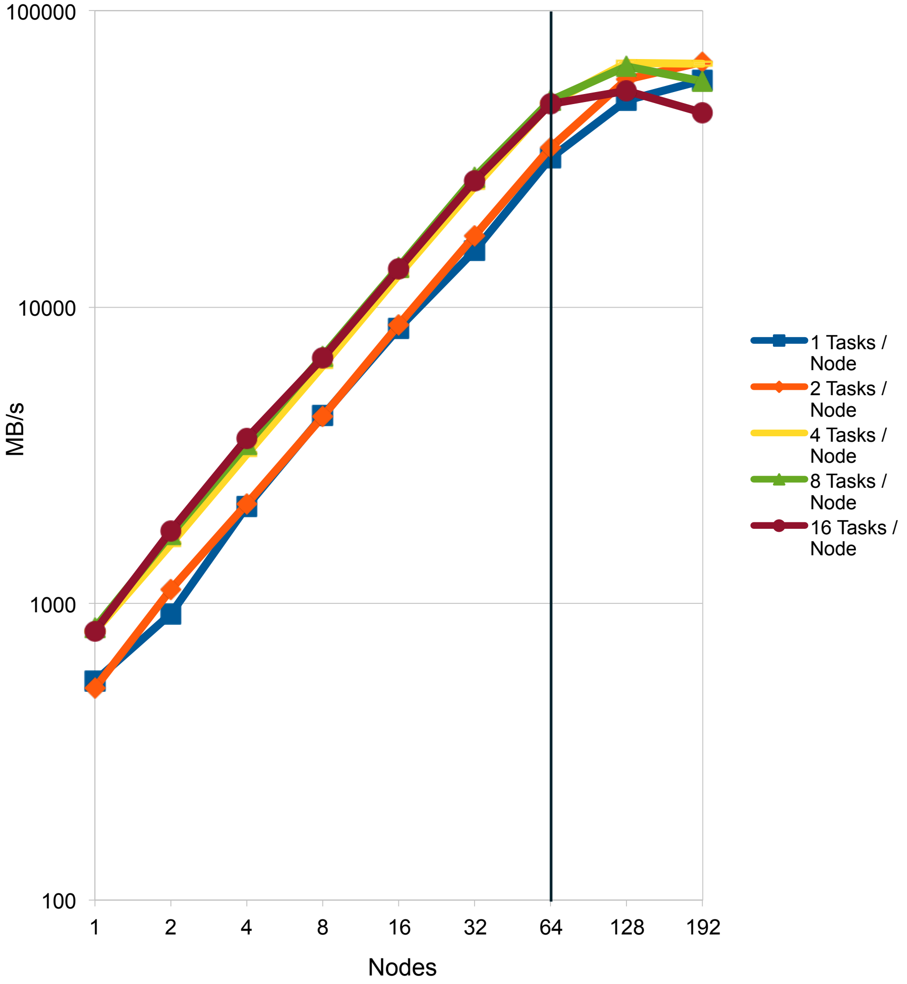
Stonewalling IOR FPP Writes

64 OSTs (192x 8+2 RAID6 SAS)



Stonewalling IOR FPP Reads

64 OSTs (192x 8+2 RAID6 SAS)



Future Lustre/ZFS Development

ZFS object iterator for online LFSCK

Performance investigation and tuning

- Allow up to 1MB ZFS block size to match Lustre RPC size
- osd-zfs ZFS Intent Log (ZIL) flash write cache integration
  - Allow fast synchronous IO operations
  - Avoid need to wait for full ZFS transaction commit

ZFS fault management and automatic hot-sparing

Longer term - better Lustre integration

- Snapshot support (synchronization, namespace visibility)
- Common network/disk checksum

Operational Changes

Relatively few operational changes for ZFS

- Can create ZFS pool/dataset manually, or via mkfs.lustre
- Recommend one target per pool, MGS always in separate dataset

```
mkfs.lustre ... --backfstype=zfs test-mgs/mgs mirror /dev/sda /dev/sdb
mkfs.lustre ... --backfstype=zfs test-mdt0/mdt0 mirror /dev/sdc /dev/sdd
mkfs.lustre ... --backfstype=zfs test-ost0/ost0 raidz2 /dev/sd[a-j] raidz2 /dev/sd[k-t]
:
mount -t lustre test-ost0/ost0 /mnt/ost/ost0
```

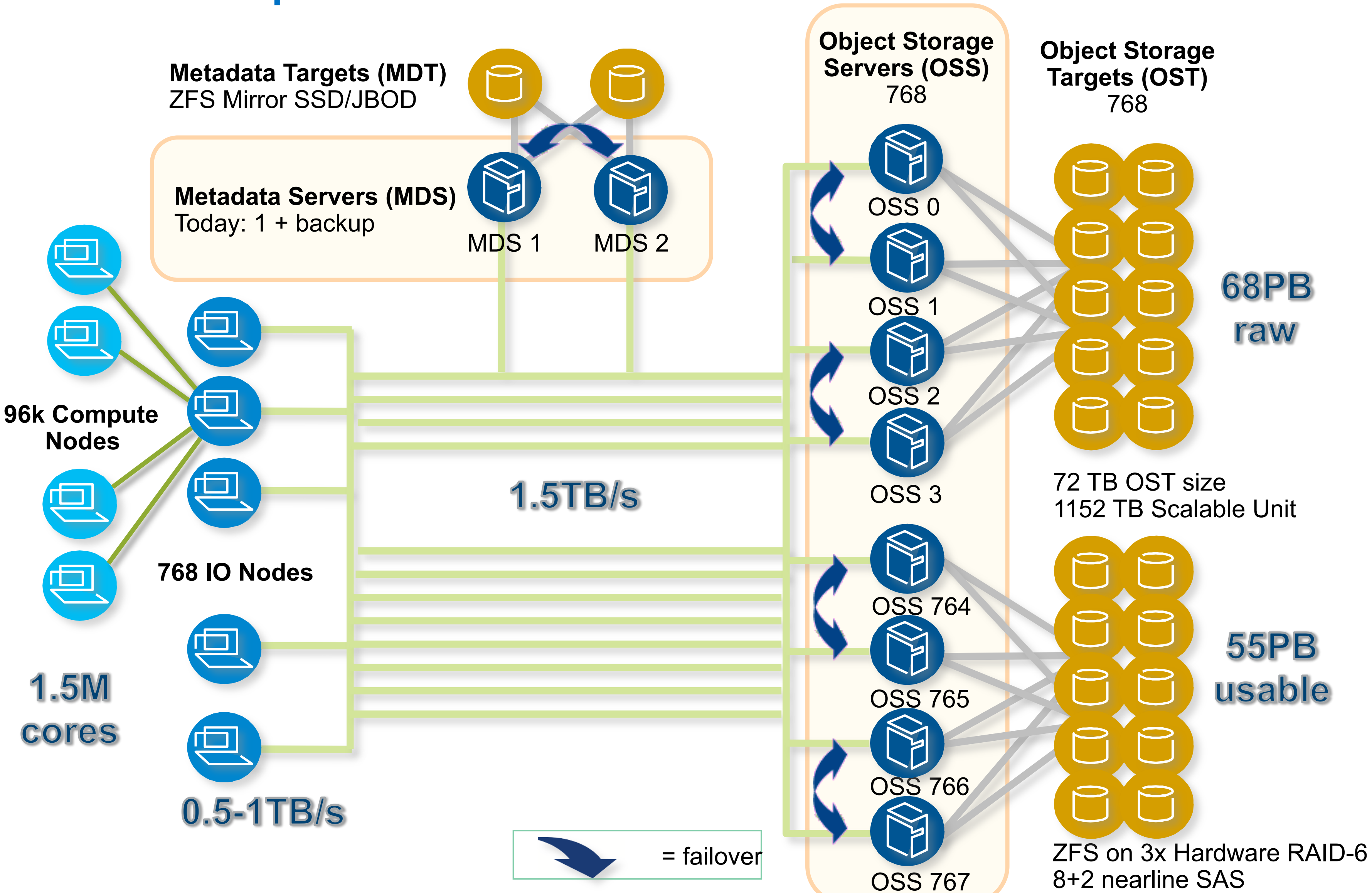
statfs/df blocks/inodes, quota data is not totally accurate

- Copy-on-write semantics make this impossible

No lfsck support for ZFS filesystems yet

Thank You

LLNL Sequoia Lustre Architecture



intel®