Lustre

Paul Bienkowski
2bienkow@informatik.uni-hamburg.de

Proseminar “Ein-/Ausgabe - Stand der Wissenschaft”

2013-06-10

Outline

What is Lustre

- parallel filesystem
- well-scaling (capacity *and* speed)
- based on Linux kernel
- optimized for clusters (many clients)

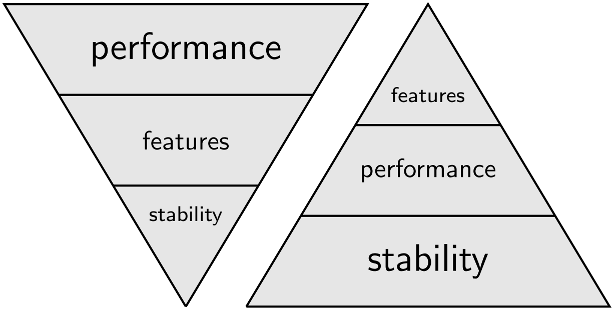
What is Lustre

- parallel filesystem
- well-scaling (capacity *and* speed)
- based on Linux kernel
- optimized for clusters (many clients)

Linux cl**uster**

The Project

Goals



until **2007**
"it's a science project"
(prototype)

2010
used in high-performance production environments

History

- Started as a research project in 1999 by Peter Braam
- Braam founds **Cluster File Systems**
- Lustre 1.0 released in 2003
- **Sun Microsystems** aquires Cluster File Systems in 2007
- **Oracle Corporation** aquires Sun Mircrosystems in 2010
- Oracle ceases Luster development, many new Organizations continue development, including **Xyratex**, **Whamcloud**, and more
- In 2012, **Intel** aquires Whamcloud
- In 2013, Xyratex purchases the original Lustre trademark from Oracle

Who is involved?

Oracle	*no development*, only pre-1.8 support
Intel	funding, preparing for *exascale computing*
Cray	funding, development (Titan Supercomputer)
Xyratex	hardware bundling
OpenSFS	(Open Scalable File Systems) “keeping Lustre open”
EOFS	(EUROPEAN Open File Systems) (community collaboration)
FOSS Community	many joined one of the above to help development (e.g. Braam works for Xyratex now)
DDN, Dell, NetApp, Terascala, Xyratex	storage hardware bundled with Lustre

Supercomputers

Lustre File System is managing data on more than 50 percent of the top 50 supercomputers and seven of the top 10 supercomputers.

— *hpcwire.com, 2008 [9]*

The biggest computer today (Titan by Cray, #1 on TOP500) uses Lustre.

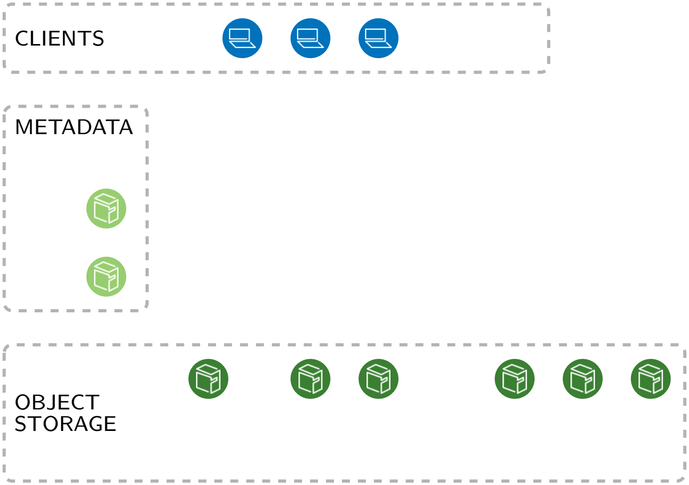
Lustre Architecture

Network Structure

CLIENTS
METADATA
OBJECT STORAGE

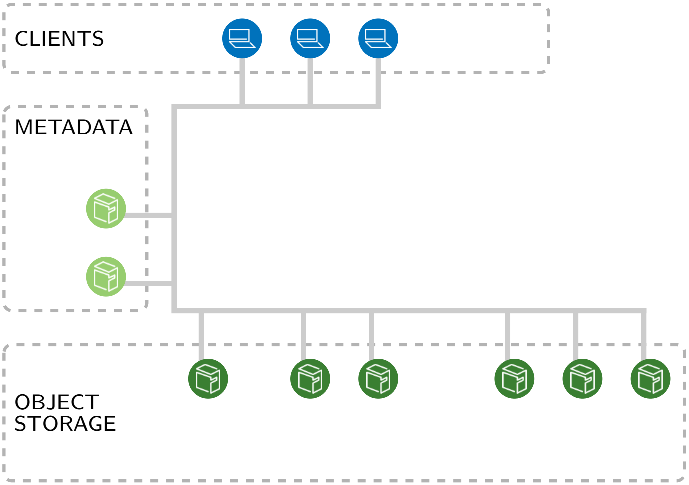
graph reproduced from [1]

Network Structure



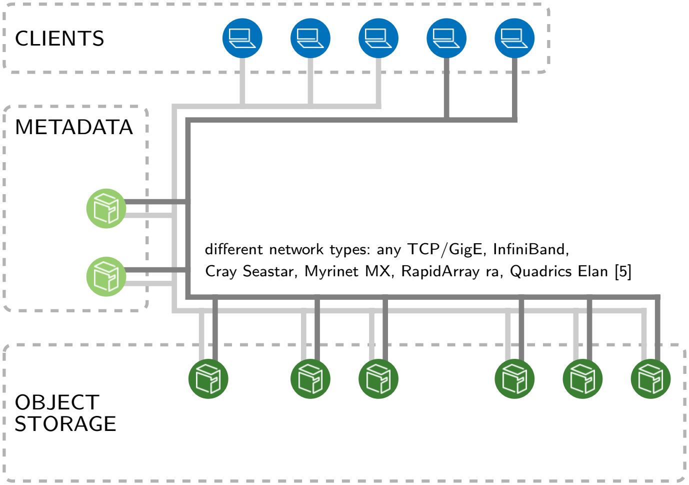
graph reproduced from [1]

Network Structure



graph reproduced from [1]

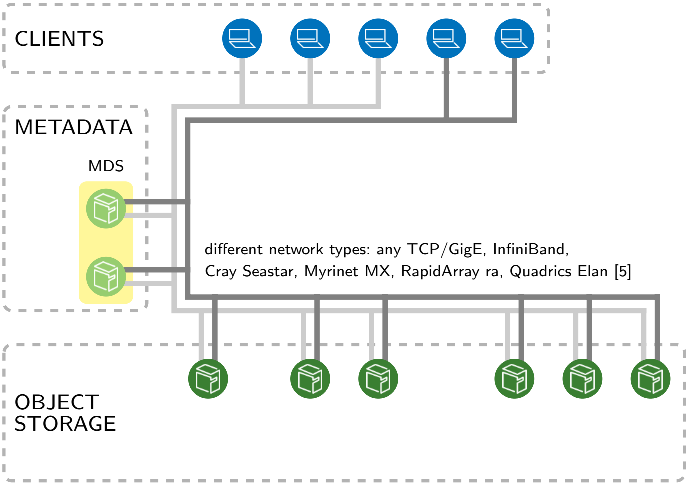
Network Structure



different network types: any TCP/GigE, InfiniBand, Cray Seastar, Myrinet MX, RapidArray ra, Quadrics Elan [5]

graph reproduced from [1]

Network Structure

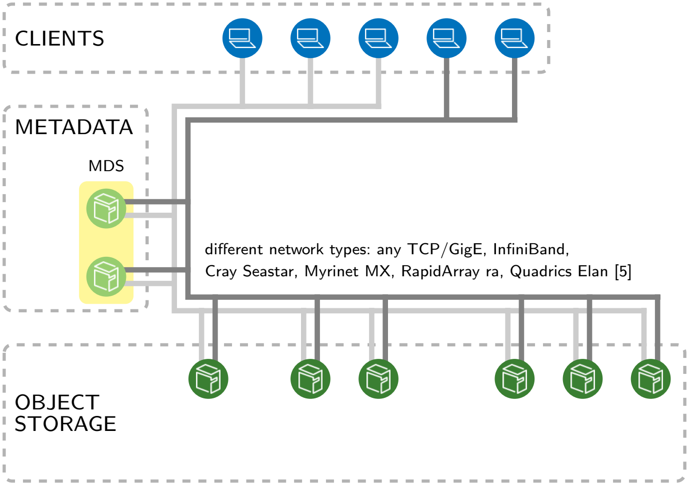

CLIENTS
METADATA
MDS
different network types: any TCP/GigE, InfiniBand,
Cray Seastar, Myrinet MX, RapidArray ra, Quadrics Elan [5]
OBJECT
STORAGE

graph reproduced from [1]

Metadata Server (**MDS**)

- store file information (metadata)
- accessed by clients to access files
- *manage* data storage
- at least one required
- multiple MDS possible (different techniques)
- recent focus for performance improvement

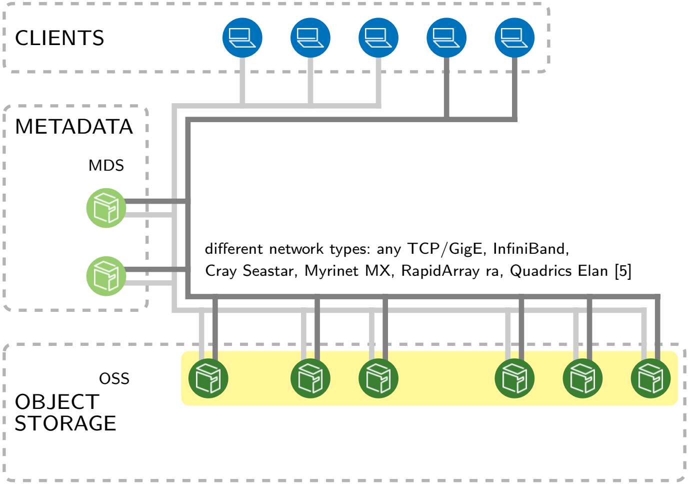
Network Structure



different network types: any TCP/GigE, InfiniBand, Cray Seastar, Myrinet MX, RapidArray ra, Quadrics Elan [5]

graph reproduced from [1]

Network Structure



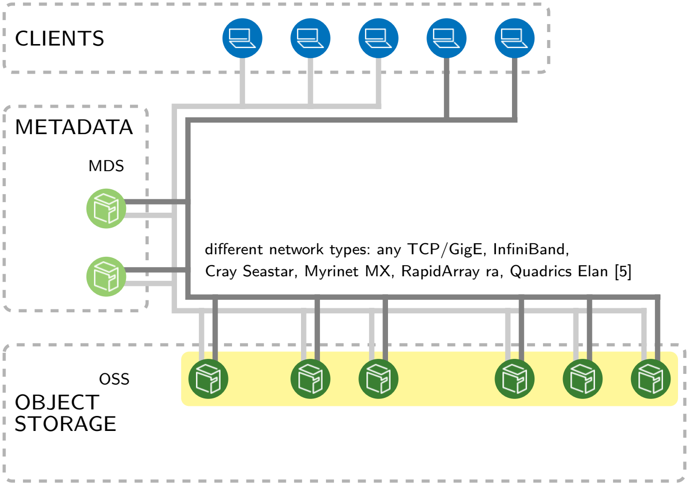
different network types: any TCP/GigE, InfiniBand, Cray Seastar, Myrinet MX, RapidArray ra, Quadrics Elan [5]

graph reproduced from [1]

Object Storage Server (**OSS**)

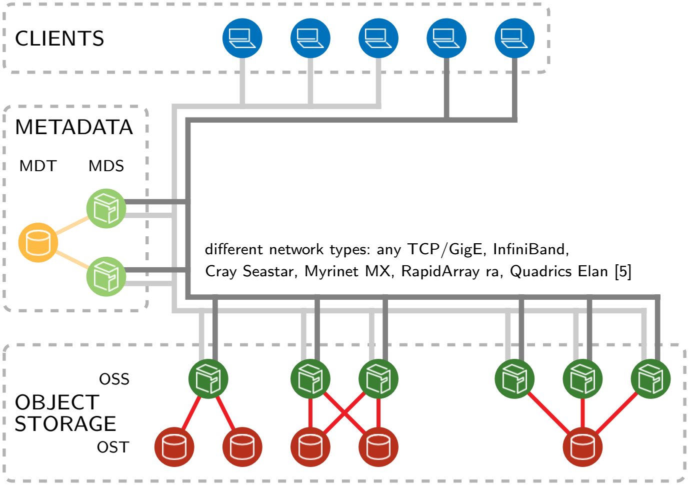
- store file content (objects)
- accessed by clients directly
- at least one required
- $> 10,000$ OSS are used in large scale computers
- multiple targets per server
- multiple servers per target

Network Structure


CLIENTS
METADATA
MDS
different network types: any TCP/GigE, InfiniBand,
Cray Seastar, Myrinet MX, RapidArray ra, Quadrics Elan [5]
OSS
OBJECT
STORAGE

graph reproduced from [1]

Network Structure



different network types: any TCP/GigE, InfiniBand, Cray Seastar, Myrinet MX, RapidArray ra, Quadrics Elan [5]

graph reproduced from [1]

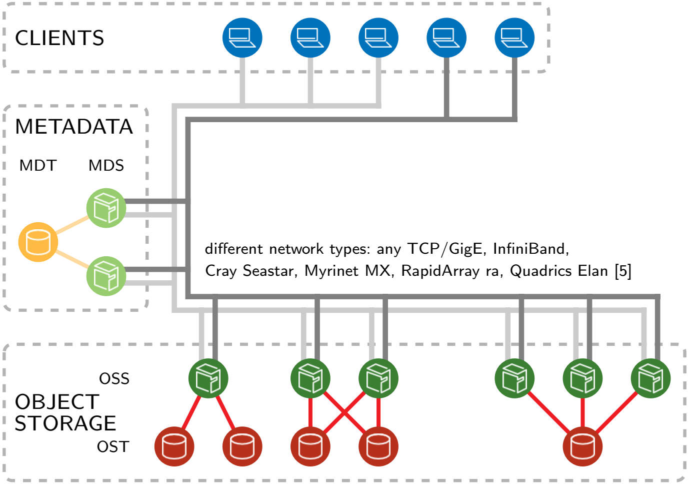
Targets

- two types
  - object storage target (OST)
  - metadata target (MDT)
- can be any block device
  - normal hard disk / flash drive / SSD
  - advanced storage arrays
- will be formatted for lustre
- up to 16 TiB / target (ext4 limit)

Failover

- if one server fails, another one takes over
- backup server needs access to targets
- enabled on-line software upgrades (one-by-one)

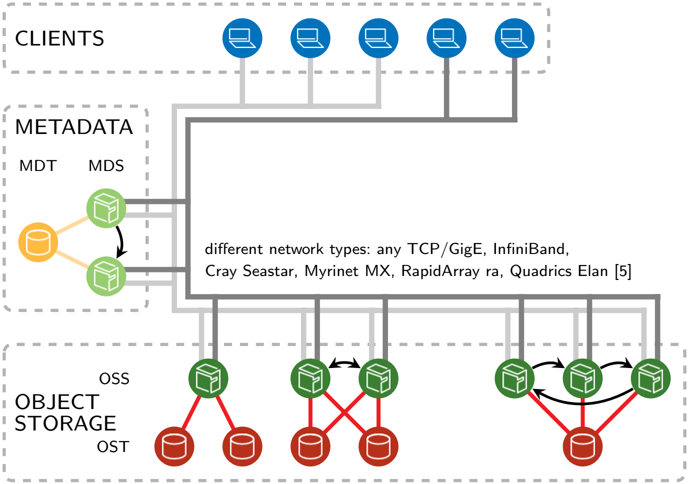
Network Structure



different network types: any TCP/GigE, InfiniBand, Cray Seastar, Myrinet MX, RapidArray ra, Quadrics Elan [5]

graph reproduced from [1]

Network Structure



different network types: any TCP/GigE, InfiniBand, Cray Seastar, Myrinet MX, RapidArray ra, Quadrics Elan [5]

graph reproduced from [1]

System characteristics

Subsystem	Typical number of systems	Performance	Required attached storage	Desirable hardware characteristics
Clients	1 - 100,000	1 GB/s I/O, 1000 metadata ops	–	–
Object Storage	1 - 1,000	500 MB/s - 2.5 GB/s	$\frac{\text{total capacity}}{\text{OSS count}}$	good bus bandwidth
Metadata Storage	1 + backup (up to 100 with Lustre $>$ 2.4)	3,000 - 15,000 metadata ops	1 - 2% of file system capacity	adequate CPU power, plenty of memory

table reproduced from [1]

Traditional Inodes

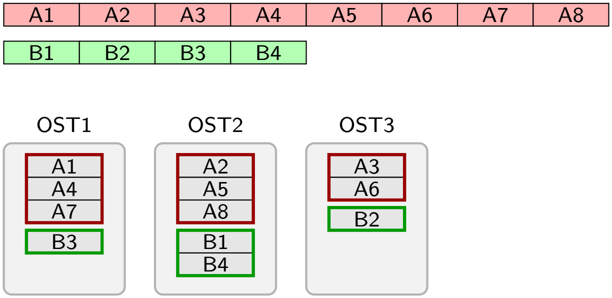
- used in many file system structures (e.g. ext3)
- each node has an index
- bijective mapping (file $\leftrightarrow$ inode)
- contains **metadata** and **data location** (pointer)

Metadata (Lustre Inodes)

- Lustre uses similar structure
- inodes are stored on MDT
- inodes point to objects on OSTs
- file is *striped* across multiple OSTs
- inode stores information to these OSTs

Striping

- RAID-0 type striping
- data is split into blocks
- block size adjustable per file/directory
- OSTs store every n-th block (with n being number of OSTs involved)
- speed advantage (multiple simultaneous OSS/OST connections)
- capacity advantage (file bigger than single OST)



Data Safety & Integrity

- data safety
  - striping does **not** backup any data
  - *but* for the targets, a RAID can be used
  - in target RAIDs, a drive may fail (depends on RAID type)
- availability
  - failovers ensure target reachability
  - multiple network types/connections
- consistency
  - lustre log (similar to journal)
  - simultaneous write protection: LDLM (Lustre Distributed Lock Manager), distributed across OSS

Software Architecture - Server

- MDS/OSS has mkfs.lustre-formatted space
- ldiskfs kernel module required (based on ext4)
- kernel requires patching (only available for some Enterprise Linux 2.6 kernels, e.g. Red Hat)

Limitations

- very platform dependent
- needs compatible kernel
- not a problem when using independent storage solution

Software Architecture - Client

- “patchfree” client: kernel module for Linux 2.6
- userspace library (liblustre)
- userspace filesystem (FUSE) drivers
- NFS access (legacy support)

Platform Support

- most Linux kernel versions $>$ 2.6 supported
- NFS for Windows
- NFS/FUSE MacOS

Interversion Compatibility

- Lustre usually supports **interoperability** [6].
- e.g. 1.8 clients $\leftrightarrow$ 2.0 servers and vice versa
- $\rightarrow$ on-line upgrade-ability using failover systems

Performance

Theoretical Limits

A well designed Lustre storage system can achieve 90% of underlining hardware bandwidth.

— Zhiqi Tao, Sr. System Engineer, Intel [3]

Theoretical Limits

A well designed Lustre storage system can achieve 90% of underlining hardware bandwidth.

— Zhiqi Tao, Sr. System Engineer, Intel [3]

Example

- 160 OSS, 16 OST each, 2 TiB each

Theoretical Limits

A well designed Lustre storage system can achieve 90% of underlining hardware bandwidth.

— Zhiqi Tao, Sr. System Engineer, Intel [3]

Example

- 160 OSS, 16 OST each, 2 TiB each
- $\rightarrow$ 2.5 PiB (Pebibyte) total storage

Theoretical Limits

A well designed Lustre storage system can achieve 90% of underlining hardware bandwidth.

— Zhiqi Tao, Sr. System Engineer, Intel [3]

Example

- 160 OSS, 16 OST each, 2 TiB each
- $\rightarrow$ 2.5 PiB (Pebibyte) total storage
- each OST delivers 50 MiB/s

Theoretical Limits

A well designed Lustre storage system can achieve 90% of underlining hardware bandwidth.

— Zhiqi Tao, Sr. System Engineer, Intel [3]

Example

- 160 OSS, 16 OST each, 2 TiB each
- $\rightarrow$ 2.5 PiB (Pebibyte) total storage
- each OST delivers 50 MiB/s
- $\rightarrow$ 800 MiB/s combined throughput per OSS

Theoretical Limits

A well designed Lustre storage system can achieve 90% of underlining hardware bandwidth.

— *Zhiqi Tao, Sr. System Engineer, Intel [3]*

Example

- 160 OSS, 16 OST each, 2 TiB each
- $\rightarrow$ 2.5 PiB (Pebibyte) total storage
- each OST delivers 50 MiB/s
- $\rightarrow$ 800 MiB/s combined throughput per OSS
- stripe size 16 MiB

Theoretical Limits

A well designed Lustre storage system can achieve 90% of underlining hardware bandwidth.

— Zhiqi Tao, Sr. System Engineer, Intel [3]

Example

- 160 OSS, 16 OST each, 2 TiB each
- $\rightarrow$ 2.5 PiB (Pebibyte) total storage
- each OST delivers 50 MiB/s
- $\rightarrow$ 800 MiB/s combined throughput per OSS
- stripe size 16 MiB
- write 200 GiB file (80 stripes per OSS, 5 stripes per OST)

Theoretical Limits

A well designed Lustre storage system can achieve 90% of underlining hardware bandwidth.

— *Zhiqi Tao, Sr. System Engineer, Intel [3]*

Example

- 160 OSS, 16 OST each, 2 TiB each
- → 2.5 PiB (Pebibyte) total storage
- each OST delivers 50 MiB/s
- → 800 MiB/s combined throughput per OSS
- stripe size 16 MiB
- write 200 GiB file (80 stripes per OSS, 5 stripes per OST)
- → 1.25 GiB per OSS, written in 1.6 seconds

Theoretical Limits

A well designed Lustre storage system can achieve 90% of underlining hardware bandwidth.

— Zhiqi Tao, Sr. System Engineer, Intel [3]

Example

- 160 OSS, 16 OST each, 2 TiB each
- $\rightarrow$ 2.5 PiB (Pebibyte) total storage
- each OST delivers 50 MiB/s
- $\rightarrow$ 800 MiB/s combined throughput per OSS
- stripe size 16 MiB
- write 200 GiB file (80 stripes per OSS, 5 stripes per OST)
- $\rightarrow$ 1.25 GiB per OSS, written in 1.6 seconds
- all OSS parallel, total speed 125 GiB/s

Recent Improvements

- “wide striping”
  - OST/file limit extended
  - $>$ 160 OST possible
  - inode xattrs
- ZFS support
  - instead of ldiskfs on targets
  - better kernel support
  - more widely used $\rightarrow$ better developed
  - all advantages of ZFS (checksums, up to 256 ZiB[1]/OST, compression, copy-on-write)
- directory traversal and stat’ing
- multiple MDS
  - metadata striping / namespacing
  - metadata performance as bottleneck

[1] kibi, mebi, gibi, tebi, pebi, exbi, **zebi**, yobi

Metadata overhead

Common Task

- **readdir** (directory traversal) and **stat** (file information)
- `ls -l`

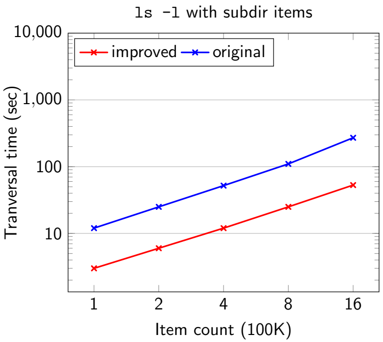
Problem

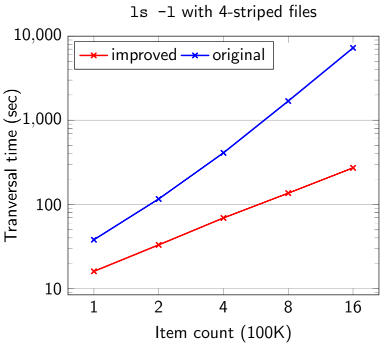
- one `stat` call for every file, each is a RPC (POSIX).
- each RPC generates overhead and I/O wait

Solution

- Lustre detects readdir+stat and requests all stats from OSS in advance (parallel)
- a combined RPC reply is sent (up to 1 MB)

Metadata overhead (cont'd)





graph data from [4]

Metadata overhead

Common Task

- **readdir** (directory traversal) and **stat** (file information)
- `ls -l`

Problem

- one `stat` call for every file, each is a RPC (POSIX).
- each RPC generates overhead and I/O wait

Solution

- Lustre detects readdir+stat and requests all stats from OSS in advance (parallel)
- a combined RPC reply is sent (up to 1 MB)

Alternative

- `readdirplus` from POSIX HPC I/O Extensions [11]

SSDs as MDT

- Metadata often bottleneck
- SSDs have higher throughput
- SSDs achieve way more IOPS (important for metadata)
- only small capacity required (expensiveness!)

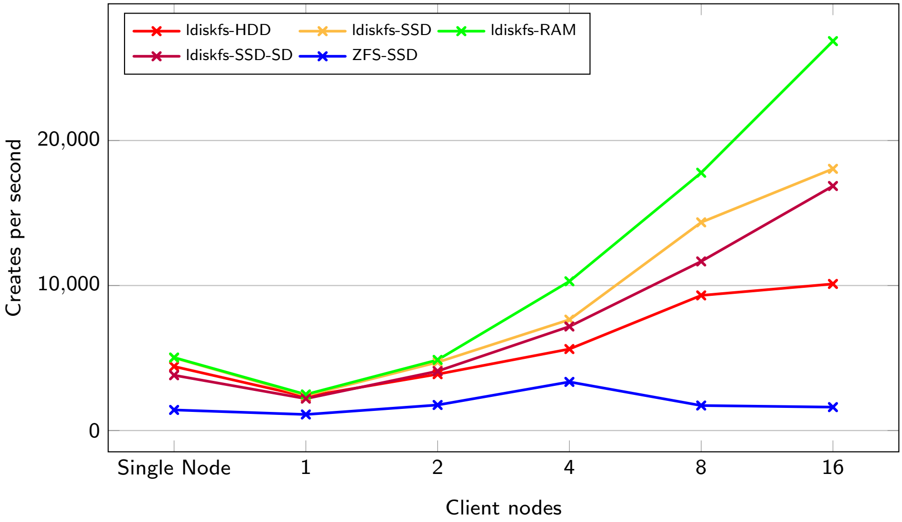
SSDs as MDT

- Metadata often bottleneck
- SSDs have higher throughput
- SSDs achieve way more IOPS (important for metadata)
- only small capacity required (expensiveness!)

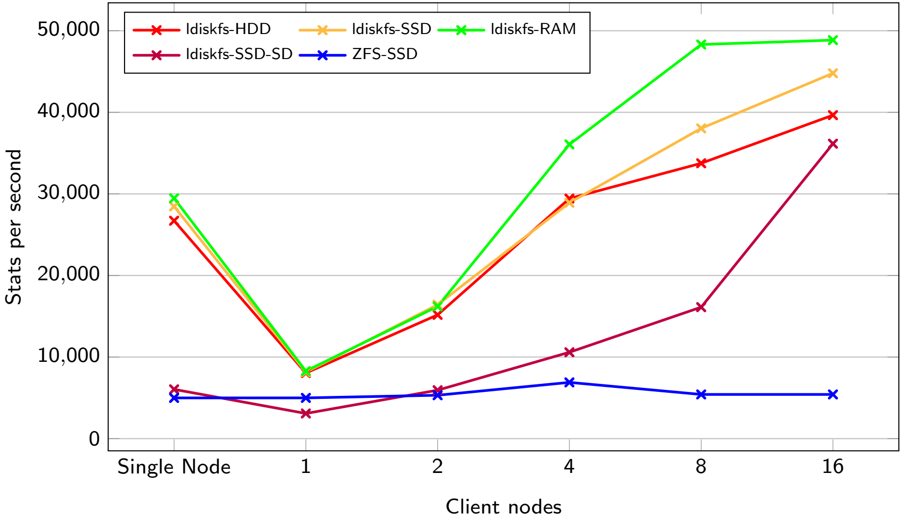
Following Graphs:

- plot metadata access (create, stat, unlink)
- 8 processes per client-node
- HDD/SSD/RAM
- shared / per-process directory
- ldiskfs / ZFS (Orion-Lustre branch)
- data from [10]

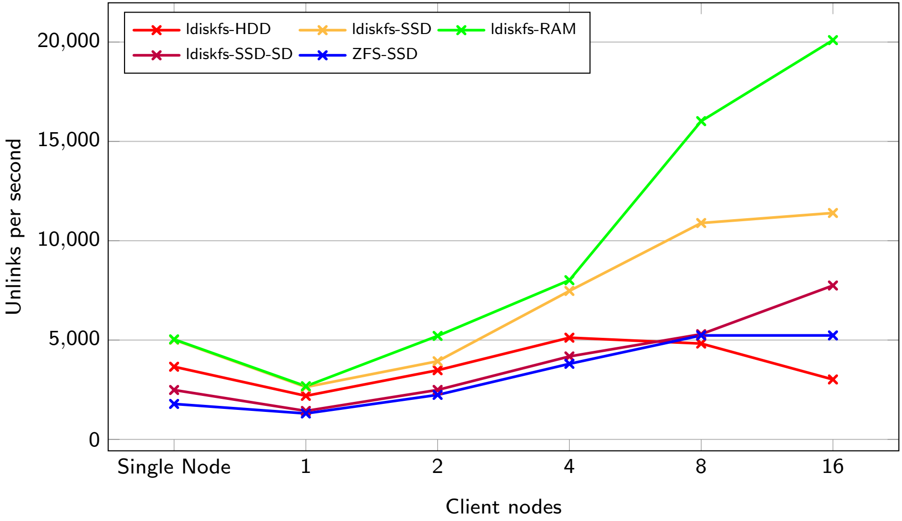
SSDs as MDT


ldiskfs-HDD
ldiskfs-SSD
ldiskfs-RAM
ldiskfs-SSD-SD
ZFS-SSD
Creates per second
20,000
10,000
0
Single Node
1
2
4
8
16
Client nodes

SSDs as MDT


ldiskfs-HDD
ldiskfs-SSD
ldiskfs-RAM
ldiskfs-SSD-SD
ZFS-SSD
50,000
40,000
30,000
20,000
10,000
0
Stats per second
Single Node
1
2
4
8
16
Client nodes

SSDs as MDT


ldiskfs-HDD
ldiskfs-SSD
ldiskfs-RAM
ldiskfs-SSD-SD
ZFS-SSD
Unlinks per second
20,000
15,000
10,000
5,000
0
Single Node
1
2
4
8
16
Client nodes

Scalability

- Lustre distributes bandwidth evenly over OSS (striping)
- different network types simultaneously (InfiniBand, TCP: GigE)
- more OSS can always be added (for more bandwidth and/or capacity)

Scalability

- Lustre distributes bandwidth evenly over OSS (striping)
- different network types simultaneously (InfiniBand, TCP: GigE)
- more OSS can always be added (for more bandwidth and/or capacity)

- still current bottleneck: MDS
- . . . not much data on 2.4 available yet
- ZFS improvements required

Conclusion

- still heavily developed
- many interested/involved companies + funding
- actively used in HPC clusters
- well scalable
- throughput depends on network
- still improvements for metadata performance required
- Linux 2.6 (Redhat Enterprise Linux, CentOS) only

References

[1] http://www.raidinc.com/assets/documents/lustrefilesystem_wp.pdf 2013-05-17

[2] http://www.opensfs.org/wp-content/uploads/2011/11/Rock-Hard1.pdf 2013-05-17

[3] http://www.hpcadvisorycouncil.com/events/2013/Switzerland-Workshop/Presentations/Day_3/10_Intel.pdf 2013-05-21

[4] http://storageconference.org/2012/Presentations/T01.Dilger.pdf 2013-05-21

[5] http://wiki.lustre.org/images/3/35/821-2076-10.pdf 2013-05-28

[6] http://wiki.lustre.org/index.php/Lustre_Interoperability_-_Upgrading_From_1.8_to_2.0 2013-05-25

[7] http://wiki.lustre.org/index.php/FAQ_-_Installation 2013-05-12

[8] https://wiki.hpdd.intel.com/display/PUB/Why+Use+Lustre 2013-05-21

[9] http://www.hpcwire.com/hpcwire/2008-11-18/suns_lustre_file_system_powers_top_supercomputers.html 2013-05-28

[10] http://www.isc-events.com/isc13_ap/presentationdetails.php?t=contribution&o=2119&a=select&ra=sessiondetails 2013-05-31

[11] http://www.pdl.cmu.edu/posix/docs/POSIX-IO-SC05-ASC.ppt 2013-05-31