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)

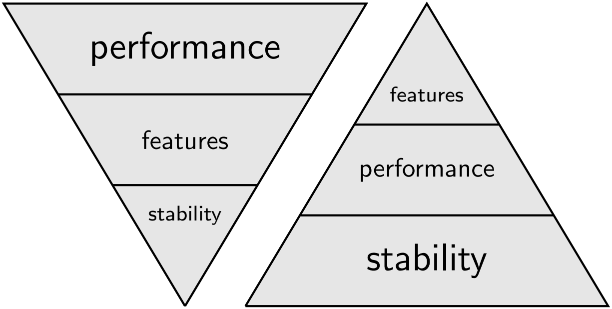
What is Lustre

- 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 **Xyratec**, **Whamcloud**, and more
- in 2012, **Intel** aquires Whamcloud
- in 2013, Xyratec 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]*

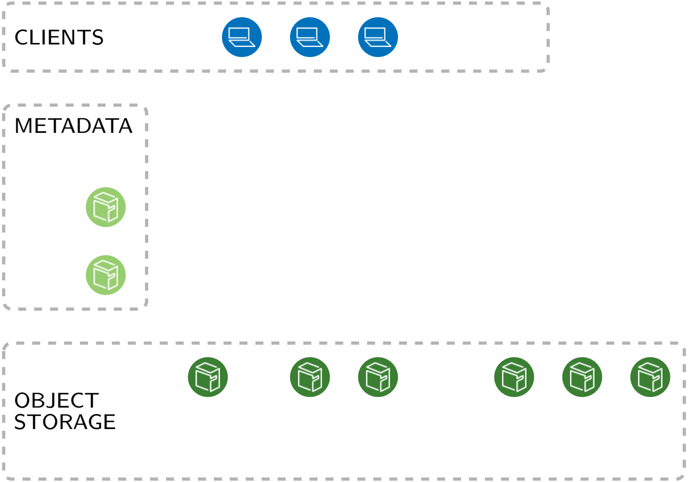
The biggest computer today (Titan by Cray, #1 on TOP500) uses Lustre.

Lustre Architecture

Network Structure

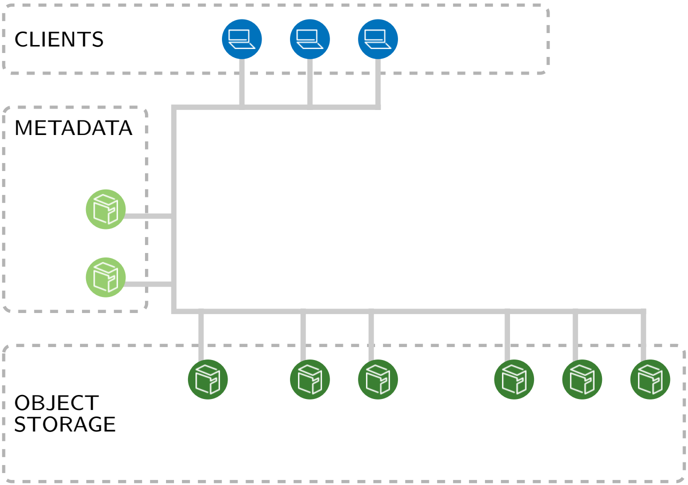
graph reproduced from [1]

Network Structure



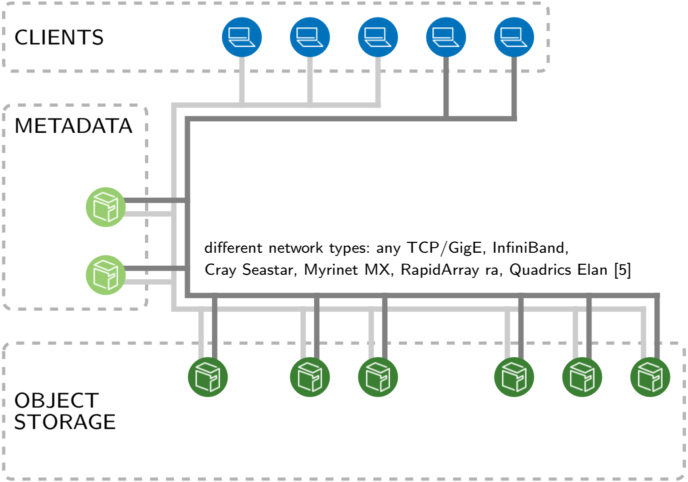
graph reproduced from [1]

Network Structure



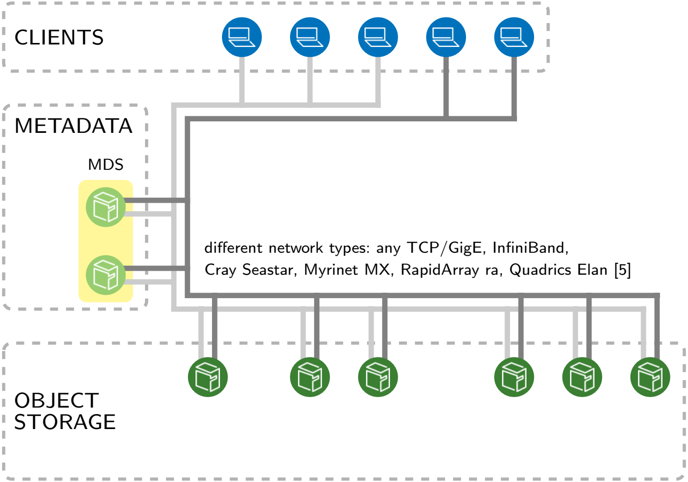
graph reproduced from [1]

Network Structure



graph reproduced from [1]

Network Structure

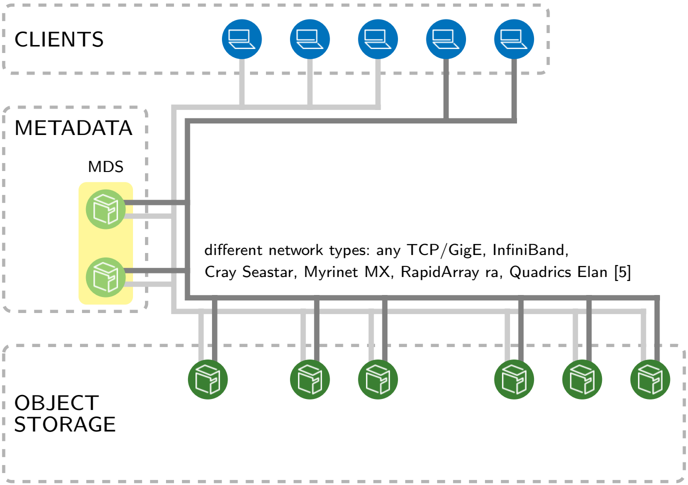


graph reproduced from [1]

Metadata Server (**MDS**)

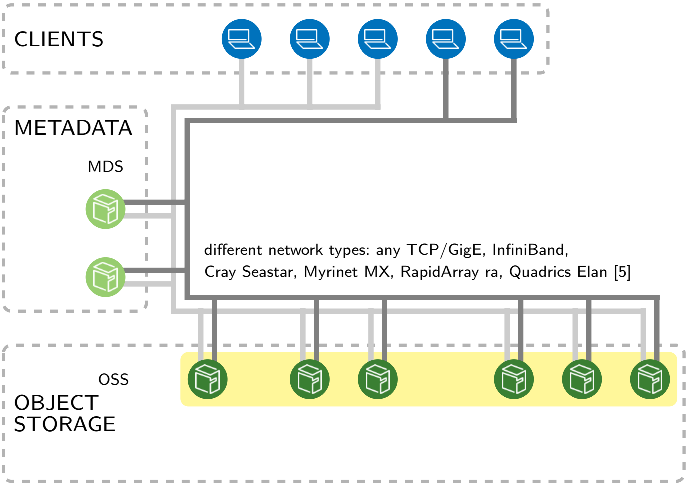
- store file information (metadata)
- accessed by clients to access files
- *manage* data storage
- at least one required
- up to ∼100 possible (failovers)

Network Structure



graph reproduced from [1]

Network Structure

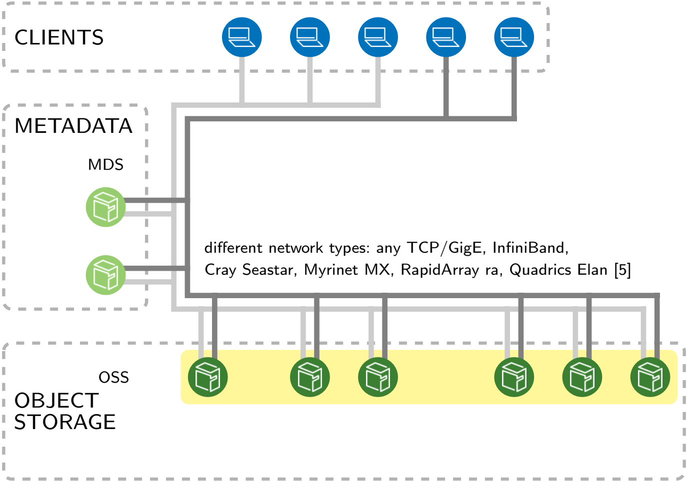


graph reproduced from [1]

Object Storage Server (**OSS**)

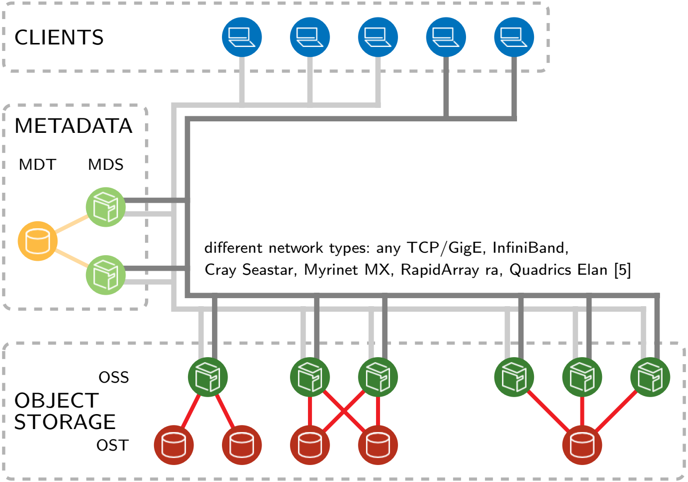
- store file content (objects)
- accessed by clients directly
- at least one required
- $>$ 10000 OSS are used in large scale computers
- multiple targets per server
- multiple servers per target

Network Structure



graph reproduced from [1]

Network Structure



graph reproduced from [1]

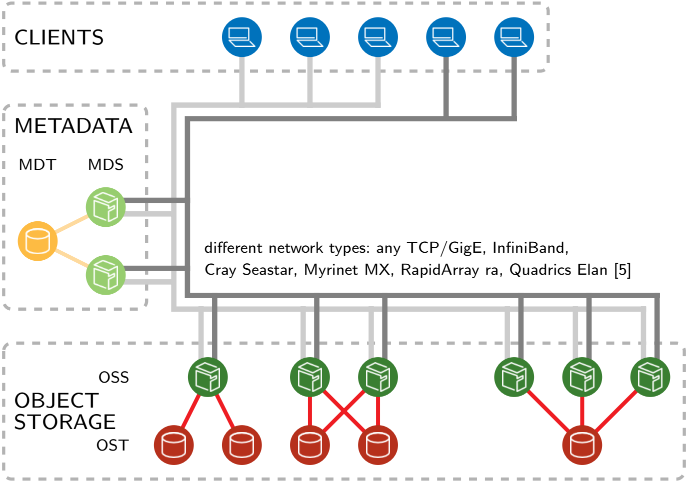
Targets

- 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 8 TiB / target (ext3 limit)

Failover

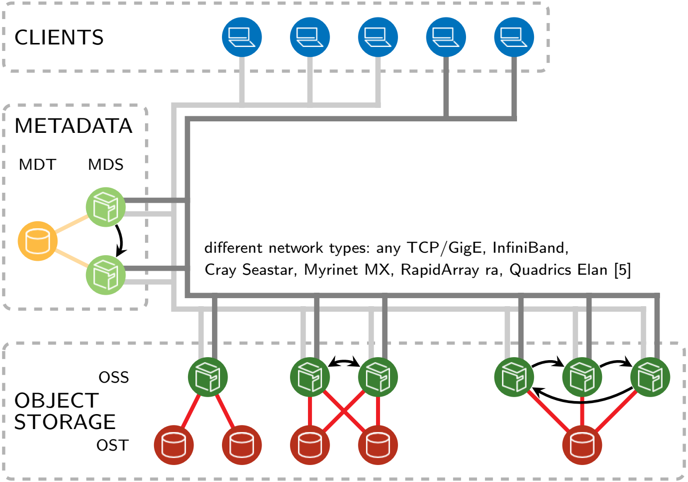
- if one server failes, another one takes over
- backup server needs access to targets
- enabled on-line software upgrades (one-by-one)

Network Structure



graph reproduced from [1]

Network Structure



graph reproduced from [1]

System characteristics

Subsystem	Typical number of systems	Performance	Required attached storage	Desirable hardware characteristics
Clients	1 - 100,000	1 GB/sec I/O, 1000 metadata ops	None	None
Object Storage	1 - 1000	500 - 2.5 GB/sec	total capacity / OSS count	Good bus bandwidth
Metadata Storage	1 + backup (up to 100 with Lustre 2.4+)	3000 - 15000 metadata ops	1 - 2% of file system capacity	Adequate CPU power, plenty of memory

table reproduced from [1]

Traditional INodes

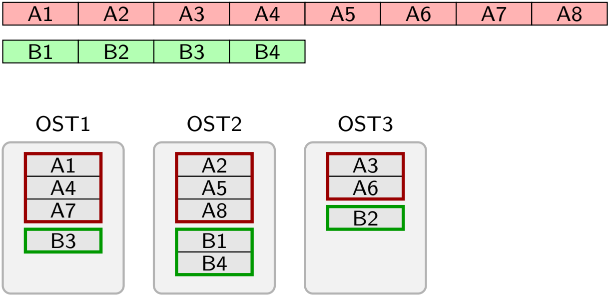
- used in many file system structures (e.g. ext3)
- each node has an index
- bijective mapping (file $\leftrightarrow$ inode)
- contains **metadata** and **data location** (pointer)

Metadata (Lustre INodes)

- lustre uses similar structure
- INodes are stored on MDT
- INodes point to objects on OSTs
- file is *striped* across multiple OSTs (limit was 160, now

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)



Data safety

- striping does **not** backup any data
- *but* for the targets, a software *or* hardware RAID can be used
- in target RAIDs, a drive may fail (depends on RAID type)
- with failovers, server availability is ensured
- for data consistency: lustre log (similar to journal)
- for 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 ext3)
- kernel requires patching (not available for Linux > 2.6)

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

- all 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 (old limits)

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 (old limits)
- $\rightarrow$ 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 (old limits)
- $\rightarrow$ 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 (old limits)
- $\rightarrow$ 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 (old limits)
- $\rightarrow$ 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 (old limits)
- $\rightarrow$ 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 (old limits)
- $\rightarrow$ 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

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 (old limits)
- → 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
- all OSS parallel, total speed 125 GiB/s

Metadata overhead

Common Task

- directory traversal and **stat** (`ls -l`)

Problem

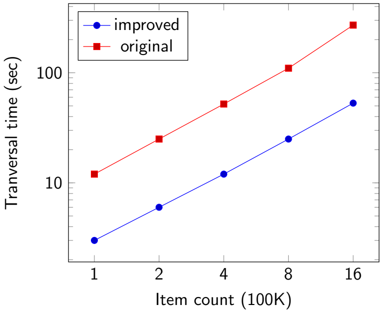
- one `stat` call for every file, each is a RPC (POSIX).
- each RPC generates overhead and I/O wait

Solution

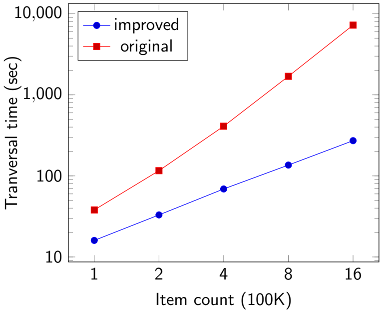
- kernel detects traversal+stat and requests all stats from OSS in advance (parallel)
- a combined RPC reply is sent (up to 1 MB)

Metadata overhead (cont'd)

`ls -l` with subdir items



`ls -l` with 4-striped files



graph data from [4]

Improvements

Recently implemented

- OSS/file limit extended (wide striping, $>$ 160 OSS possible)

Planned features

- ZFS instead of ldiskfs
- metadata striping / namespacing (multiple MDS)

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)
- current bottleneck: MDS

Conclusion

- still heavyly 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