Experiments with IO Proxies over Lustre

LUSTRE ADMINS & DEVELOPERS WORKSHOP

Reims France 2014

Bull an atos company

2014/09/23

Grégoire PICHON

Parallel File Systems

Extreme Computing R&D

Agenda

- IO Forwarding
- Experiments
  - Goal
  - Architecture
  - Results
- IO Proxy sharing
  - Results
- Conclusion

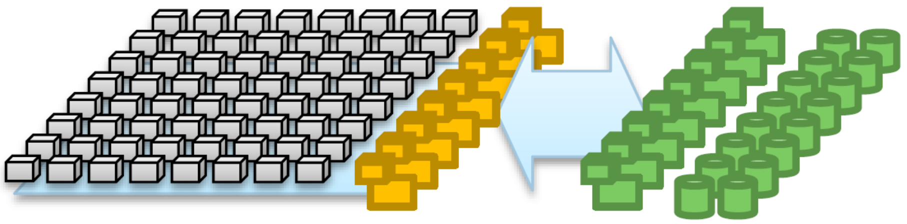
IO Forwarding

What ?

- ship IO calls from compute nodes to dedicated IO nodes
- IO proxies perform operations on behalf of compute nodes

Why ?

1. limit impact of IOs on compute nodes
2. reduce file system traffic and recovery time
3. allow : aggregation, rescheduling, caching of IO requests



Experiments' Goal

Project scope

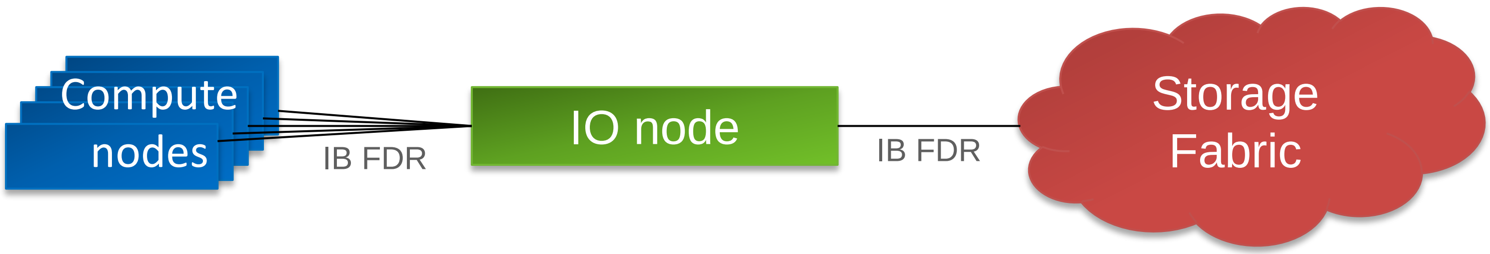
short term solution for 2015-2017 clusters

Goals

- setup software components
- evaluate performance
- compare with legacy architecture
- identify Lustre strengths and weaknesses

project in cooperation with CEA

Experiment's test bed



Compute nodes

- 8 nodes
- 16 cores per node total of 128 cores
- 32 GB memory per node
- 1 Infiniband adapter per node

IO node

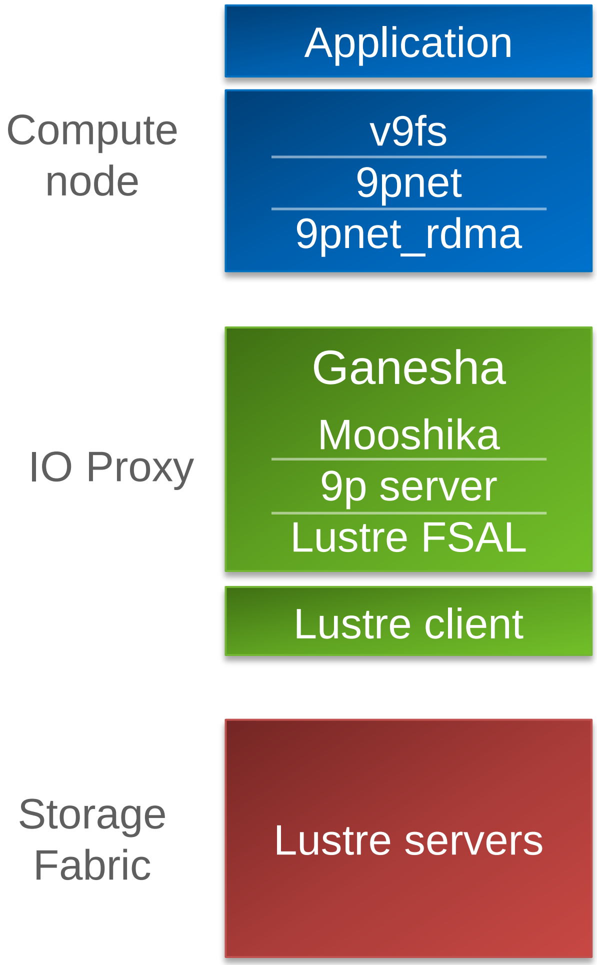
- 16 cores
- 64 GB memory
- 2 Infiniband adapters

Storage Fabric

- 2 OSSs, 16 OSTs
- 1 MDS, 1 MDT
- File system

  data bandwidth: ~6 GB/s

Experiment's software stack



v9fs

- v9fs Linux file system
- 9P protocol: Bell's lab Plan 9
- synchronous operations
- no-cache
- RDMA transport

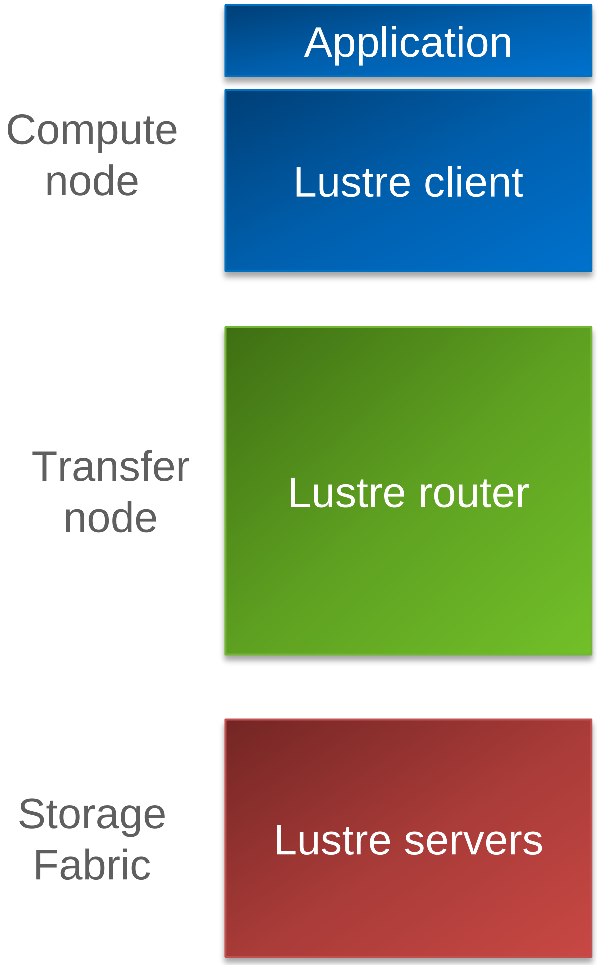
Ganesha

- open source project: NFS-Ganesha
- user mode
- 9p server
- Lustre FSAL (File System Abstraction Layer)
- RDMA library: Mooshika

Lustre

- version 2.4

Legacy Software Stack



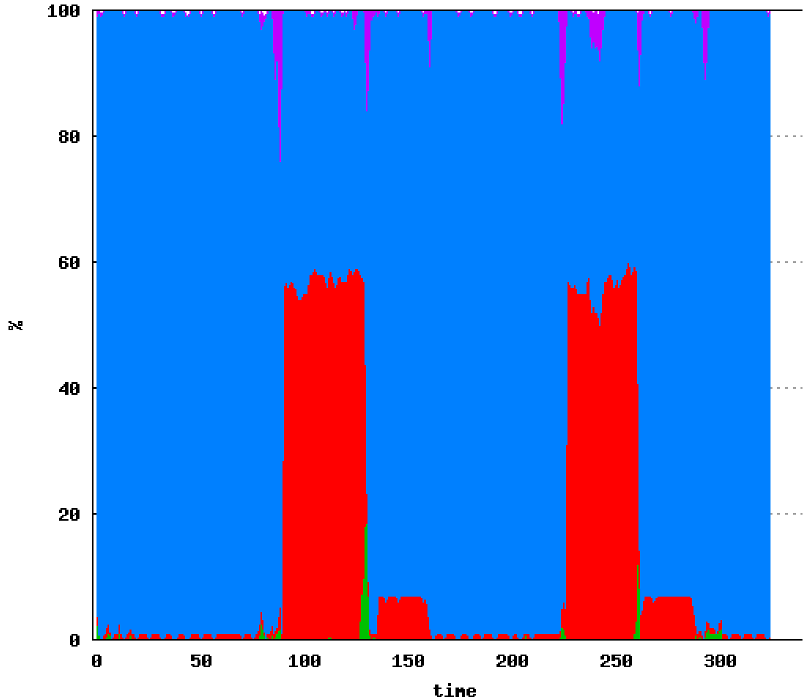
Legacy architecture

- compute cluster accesses storage cluster through routers

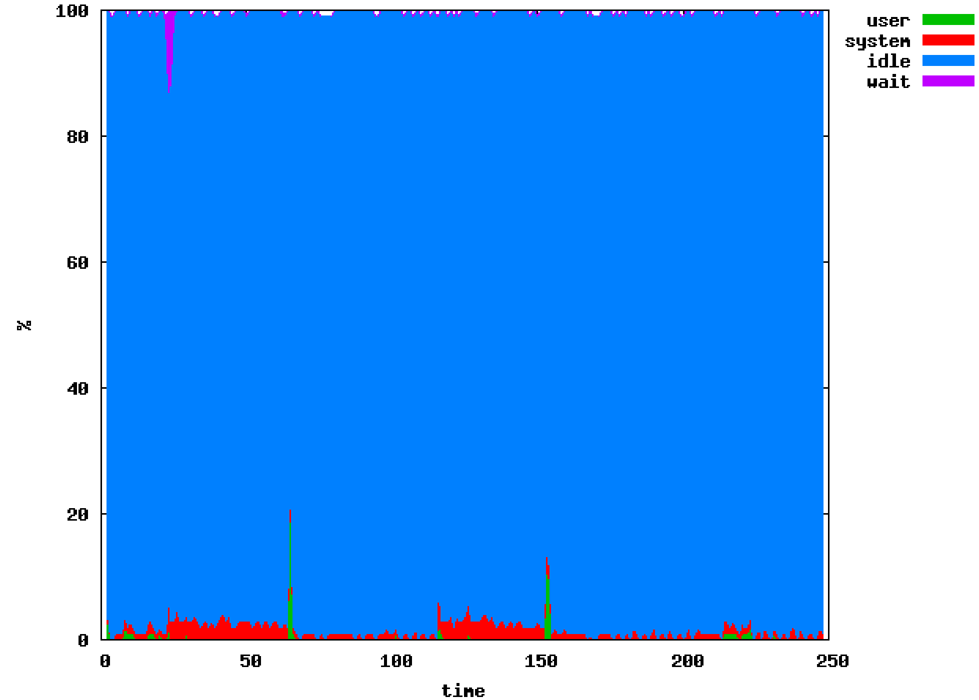
Compute node monitoring

CPU impact of IOs on compute node is minimal

Compute node CPU usage
during large sequential IOs (1.5GiB/s)



Legacy architecture

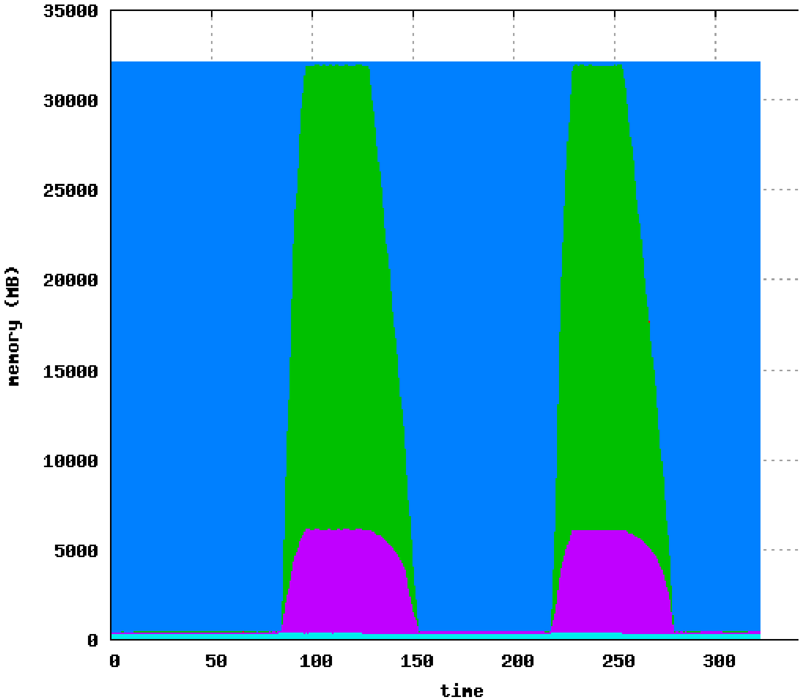


IO Proxy architecture

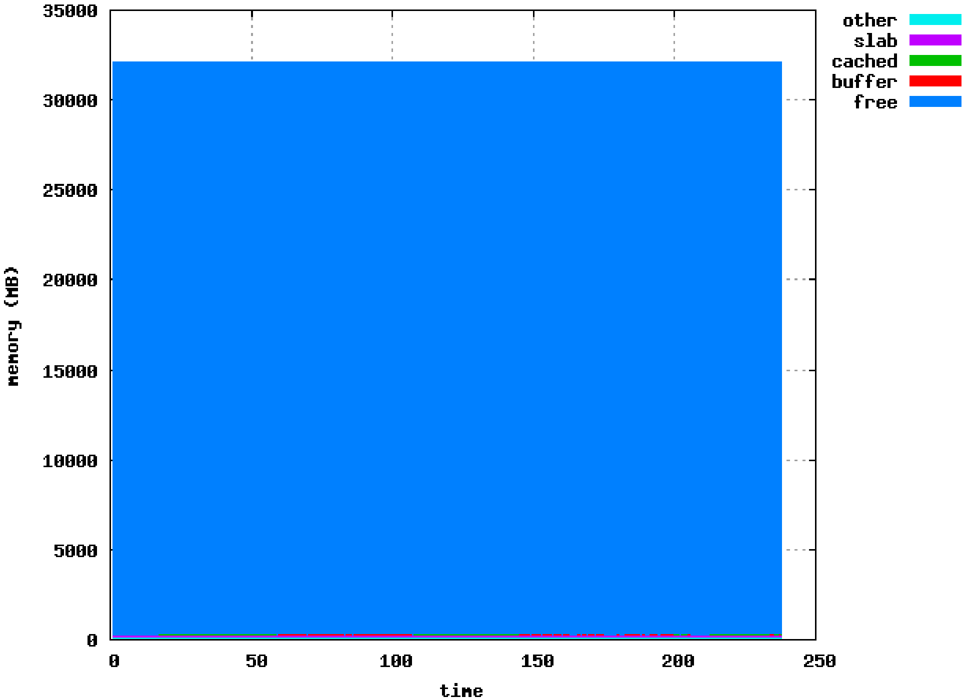
Compute node monitoring (2)

Memory impact of IOs on compute node is minimal

Compute node memory usage
during large sequential IOs (1.5GiB/s)

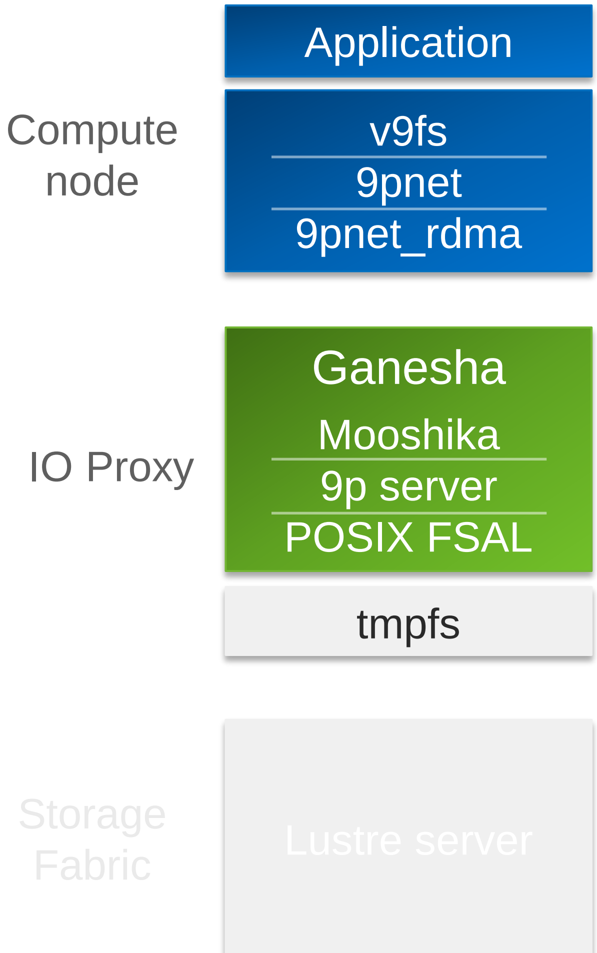


Legacy architecture



IO Proxy architecture

v9fs and Ganesha



v9fs and Ganesha performance

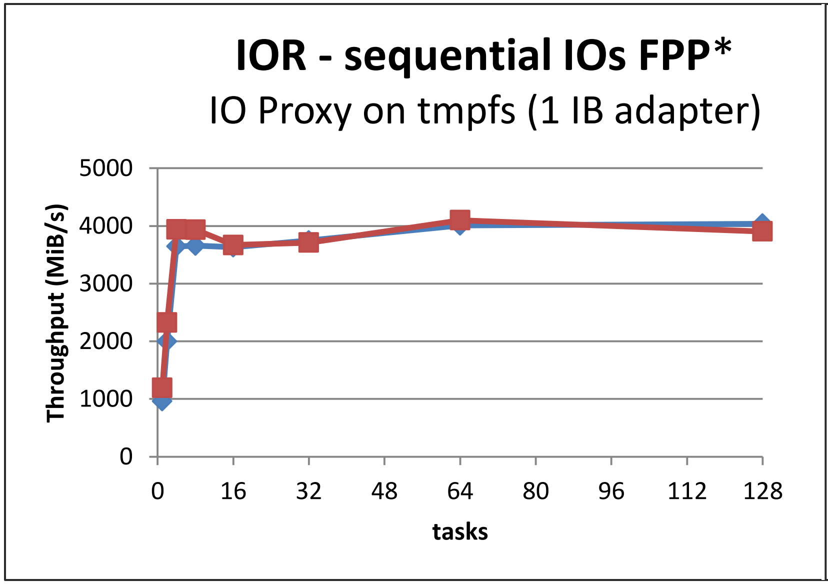
- no storage fabric
- in-memory filesystem on IO Proxy node

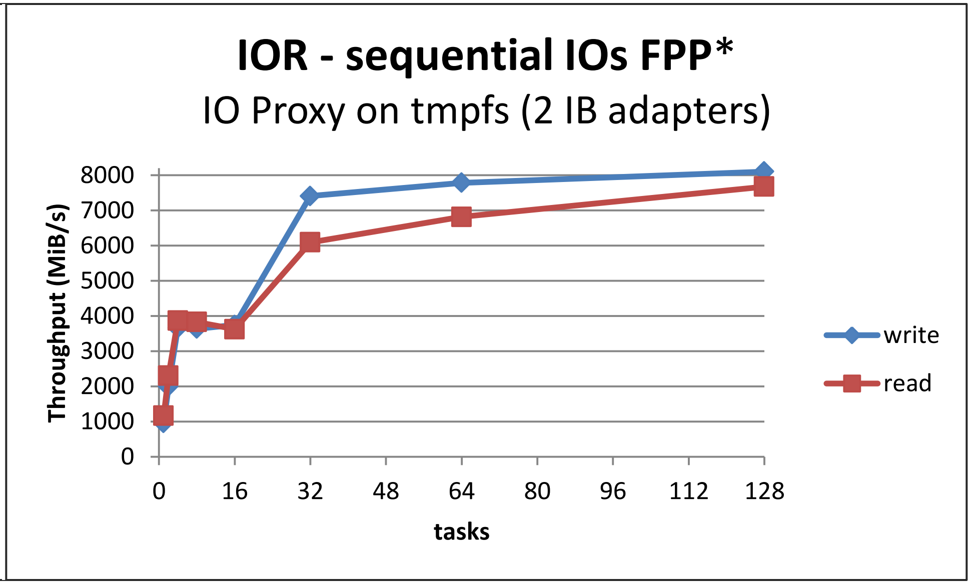
v9fs and Ganesha

v9fs and Ganesha performance

- reaches 2/3 of network bandwidth

-> might be possible to improve but enough for other tests



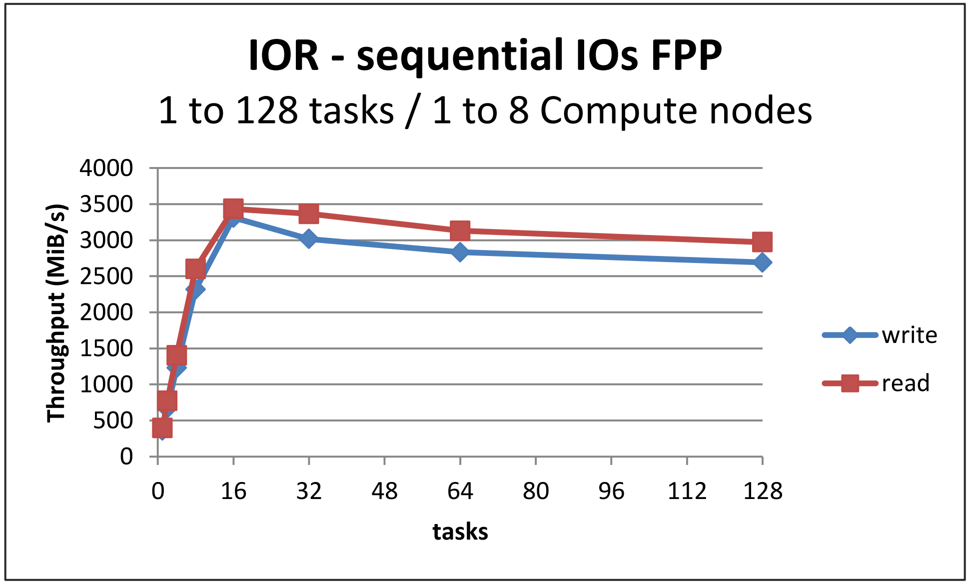


* FPP: file per process

Large sequential IOs FPP

large sequential IOs with File-Per-Process access

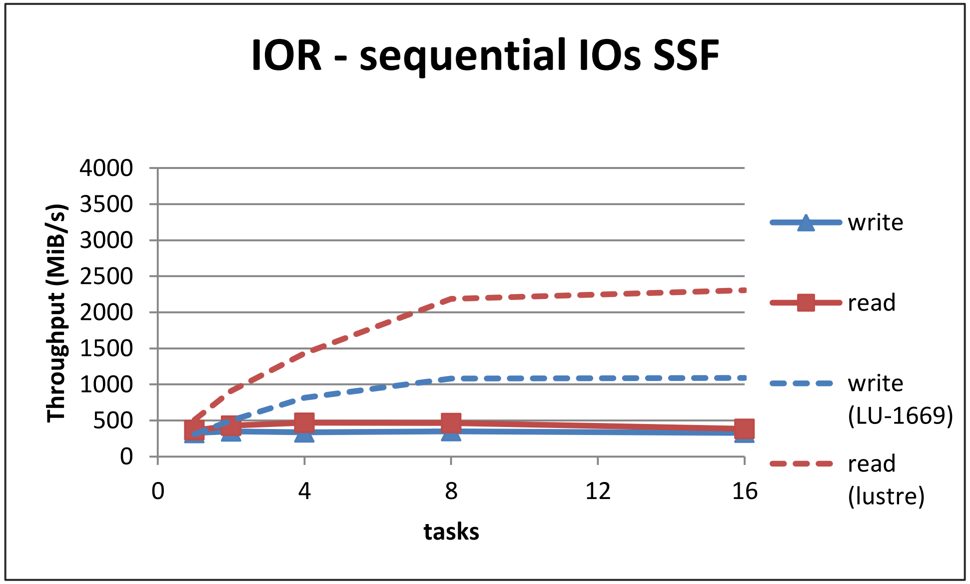
- reaches Lustre single client performance
- need to test with latest Lustre version, should be able to increase



Large sequential IOs SSF

large sequential IOs with Single Shared File access

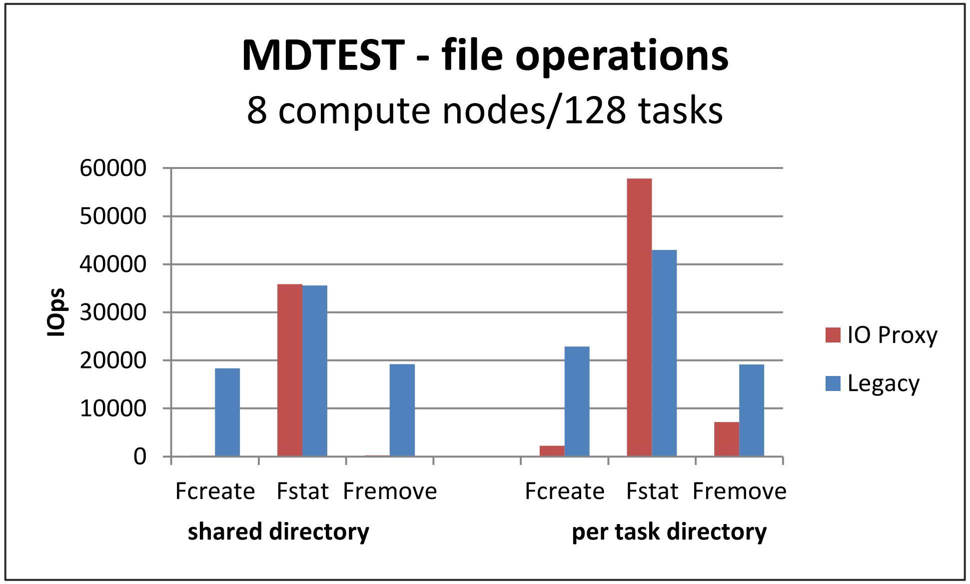
- limited in write by Lustre single client performance (LU-1669)
- limited in read by Ganesha



Metadata operations

metadata operations on files

- file creation and removal is limited by
  - MDT single slot for reply reconstruction (LU-5319)
  - Ganesha request handling overhead
- file stat benefits from Ganesha metadata cache

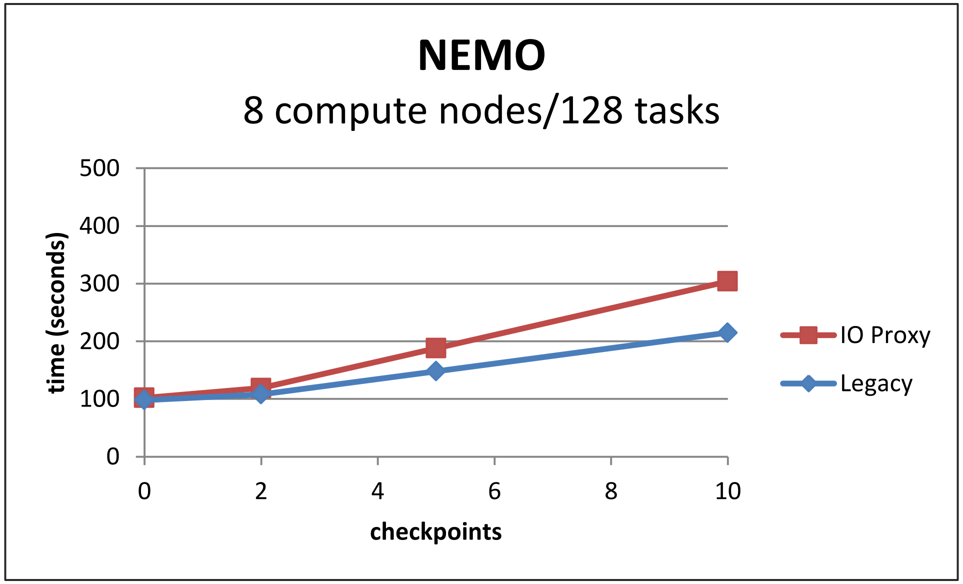


NEMO

NEMO : Nucleus for European Modelling of the Oceans

- with 10 checkpoints, 65% of runtime is spent doing IOs
- IO Proxy page cache is saturated and CPU usage is high

-> execution time is 50% longer than with Legacy architecture

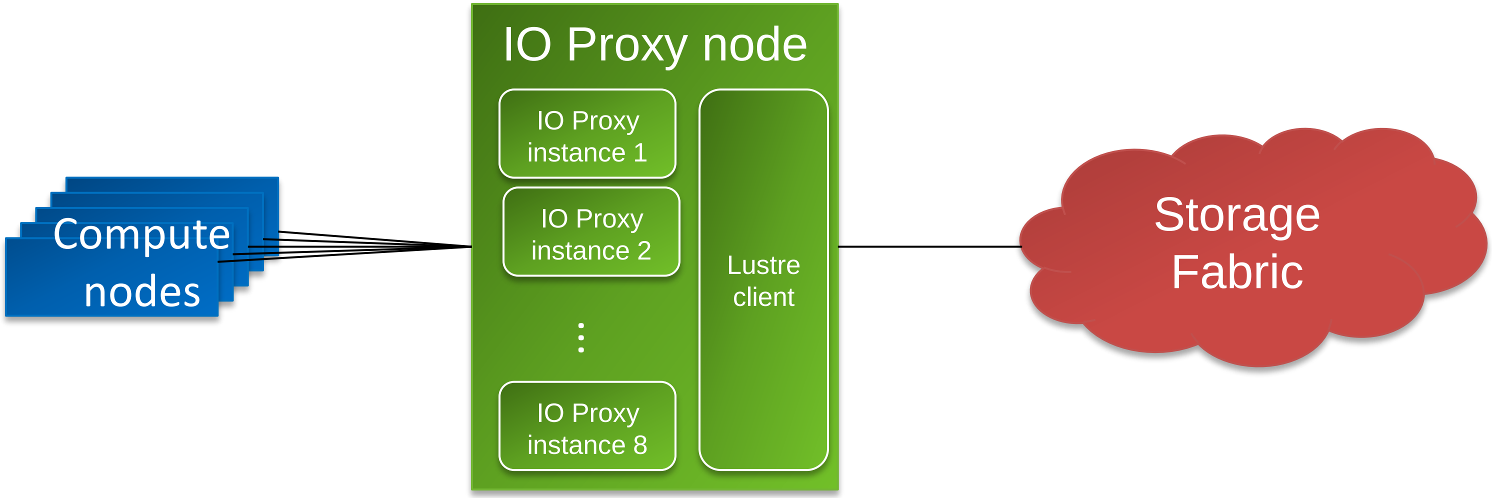


Agenda

- IO Forwarding
- Experiments
  - Goal
  - Architecture
  - Results
- **IO Proxy sharing**
  - Results
- Conclusion

IO Proxy sharing

- IO Proxy node is shared for several small jobs
  - IO Proxy instance (Ganesha) is dynamically allocated when job is scheduled
  - IO Proxy node instantiates several IO Proxy instances
- How to ensure fair partitioning of filesystem resources ?

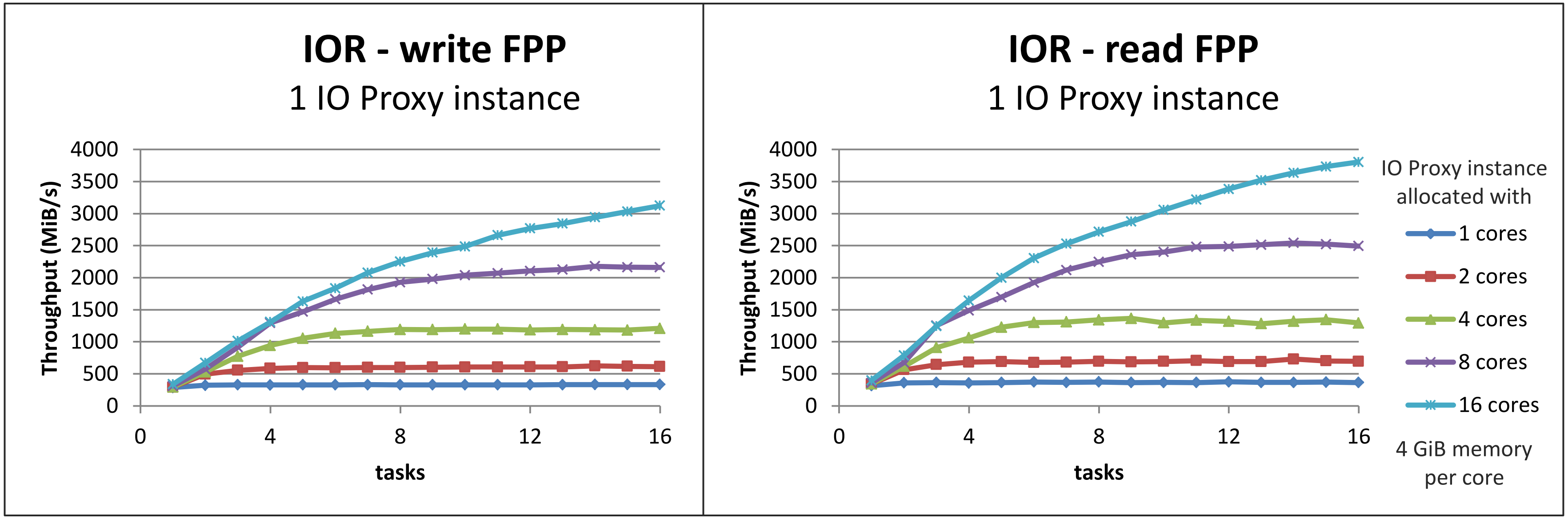


IO Proxy sharing (2)

large sequential IOs with File-Per-Process access

- 1 IOR job launched on a compute node
- 1 IO Proxy instance allocated with variable node resources

-> performance is almost proportionally bound to resources allocated to the IO Proxy instance

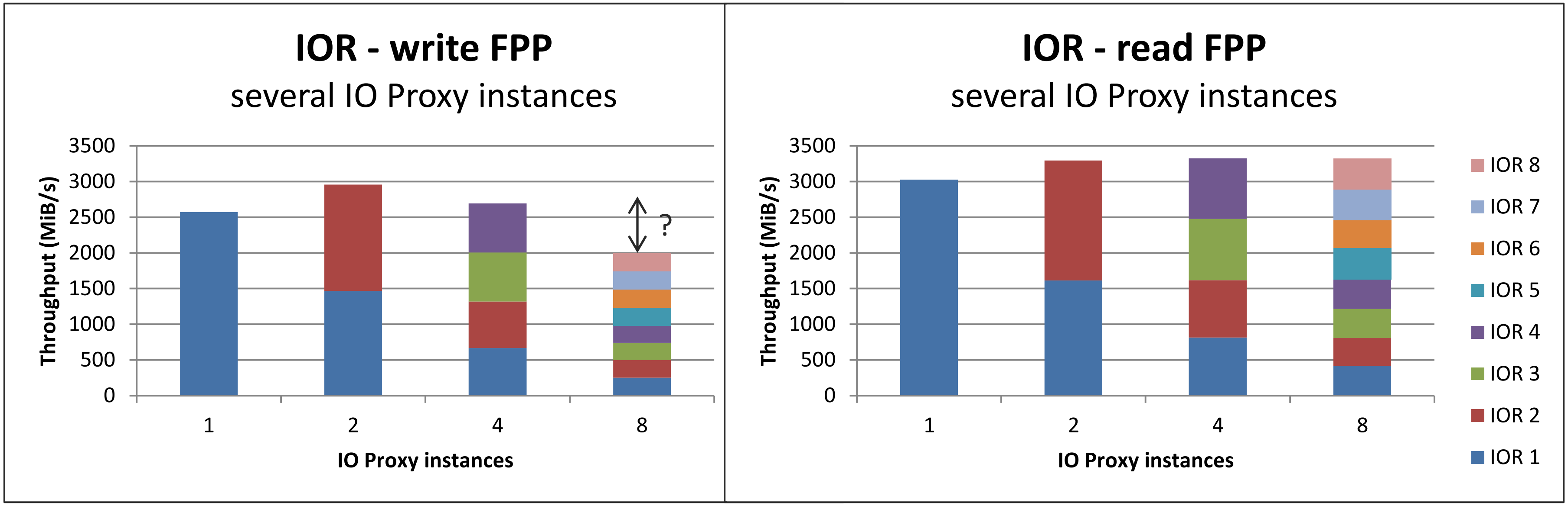


IO Proxy sharing (3)

large sequential IOs with File-Per-Process access

- several IOR jobs launched in parallel, one per compute node
- one IO Proxy instance allocated per job

-> almost all IO Proxy node filesystem bandwidth is exploited



Agenda

- IO Forwarding
- Experiments
  - Goal
  - Architecture
  - Results
- IO Proxy sharing
  - Results
- **Conclusion**

Conclusion

In the IO Forwarding architecture

Lustre single client performance is crucial

IOs of several compute nodes are concentrated on a single Lustre client

Strengths

- large sequential IOs with file-per-process access
- filesystem bandwidth can be easily shared by controlling cpu/memory resources

Weaknesses

- large sequential IOs with single-shared-file access
- filesystem modify metadata operations

an atos company

Test configuration: hardware

- Compute nodes
  - 8 Bullx R423E3 servers
  - 2 sockets/16 cores, Intel Xeon SandyBridge-EP E5-2660 2.20 GHz, 32 GB memory, 1 IB FDR adapter
- IO Proxy
  - 1 Bullx R425E3 server
  - 2 sockets/16 cores, Intel Xeon SandyBridge-EP E5-2650 2.00 GHz, 64 GB memory, 2 IB FDR adapter
- Storage Fabric
  - 2 OSS, 16 OSTs, raw performance: 8.8 GB/s write, 12.3 GB/s read
  - 1 MDS, 1 ram device MDT

Test configuration: software

- Bullx Super Computer Suite 4 Advanced Edition Release 4
  - based on Redhat 6.4
  - Linux kernel 2.6.32-358/431
  - OFED 1.5.4.1
  - Lustre 2.4
- v9fs
  - git://github.com/sderr/9pmod
  - version 1.0
- Ganesha
  - git://github.com/nfs-ganesha/nfs-ganesha
  - version 2.0 plus few patches
- Mooshika
  - git://github.com/martinetd/mooshika
  - version 0.4

Results : Linux kernel build

Linux kernel build

- single compute node build requires multithreading to avoid 9p latency impact
- kernel build in parallel by several compute nodes reveals overload of IO Proxy