Sun Storage Perspective & Lustre Architecture

Dr. Peter Braam
VP
Sun Microsystems

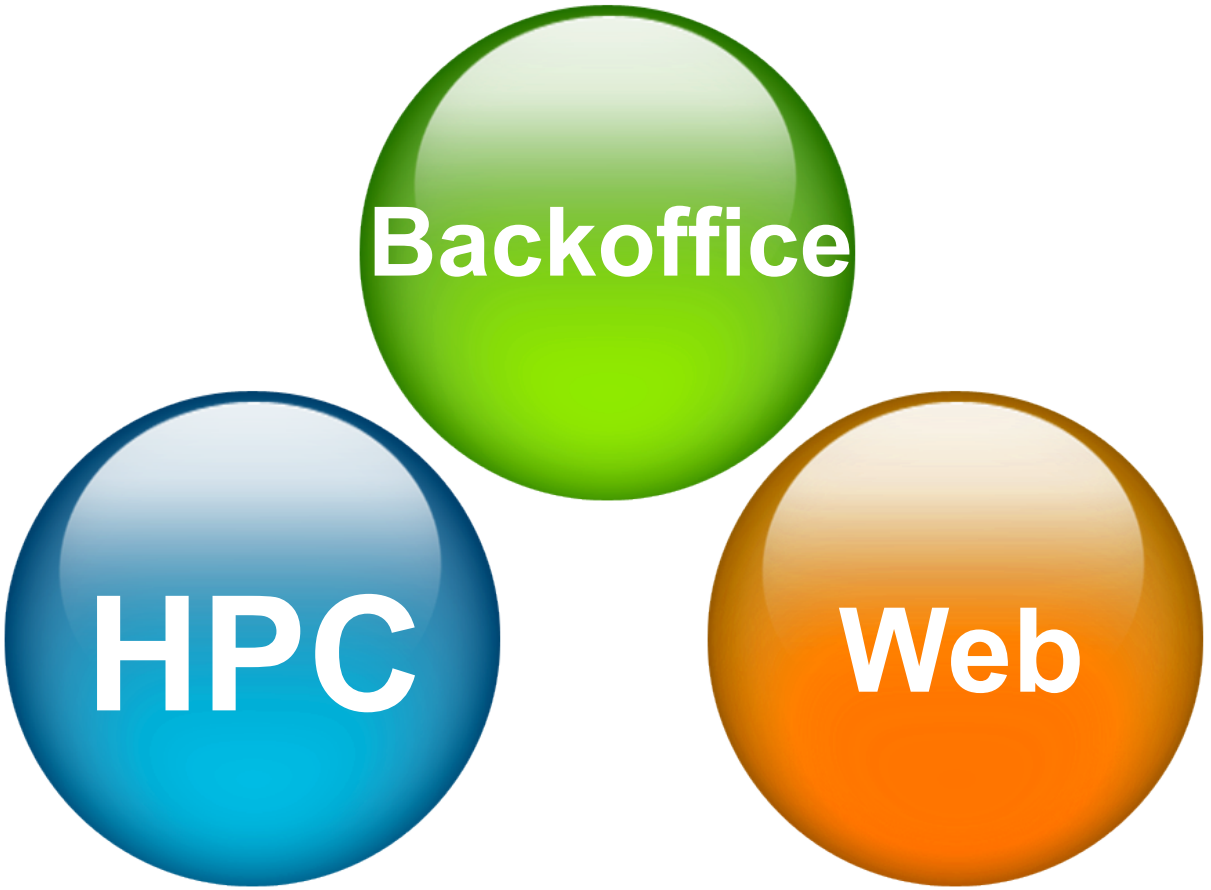
Agenda

- Future of Storage – Sun's vision
- Lustre - vendor neutral architecture roadmap

Sun's view on storage

introduction

The IT Infrastructure



Big Changes

- Everything is a cluster
- Open Source everywhere (Computer, Network, Storage)
- Fully virtualized processing, IO, and storage
- Integration, datacenter as a design center

COMPUTE:
Many cores, many threads, open platforms

STORAGE OPEN PLATFORMS:
$/performance
$/gigabyte

NETWORKING:
Huge bandwidth
Open platforms

What's Ahead

Open Servers

- Leveraging innovative product design and packaging
- Common components
- Open source software
- Wide interoperability to deliver breakthrough economics

Open Storage

A storage architecture that leverages:

- Open software
- An open architecture
- Common components
- Open interoperability to create innovative storage products
- Delivers breakthrough economics

Open Networks

- Unified datacenter network that utilizes common components
- Open source software
- Seamless integration with exitsting evironments
- Delivers breakthrough ecomonics

ZFS

the central component of Open Storage

What is ZFS?

A new way to manage data

End-to End Data Integrity

With check-summing and copy-on-write transactions

Immense Data Capacity

The world's first 128-bit file system

Easier Administration

A pooled storage model – no volume manager

Huge Performance Gains

Especially architected for speed

Trouble with Existing File Systems?

Good for the time they were designed, but...

No Defense Against Silent Data Corruption

Any defect in datapath can corrupt data... undetected

Difficult to Administer–Need a Volume Manager

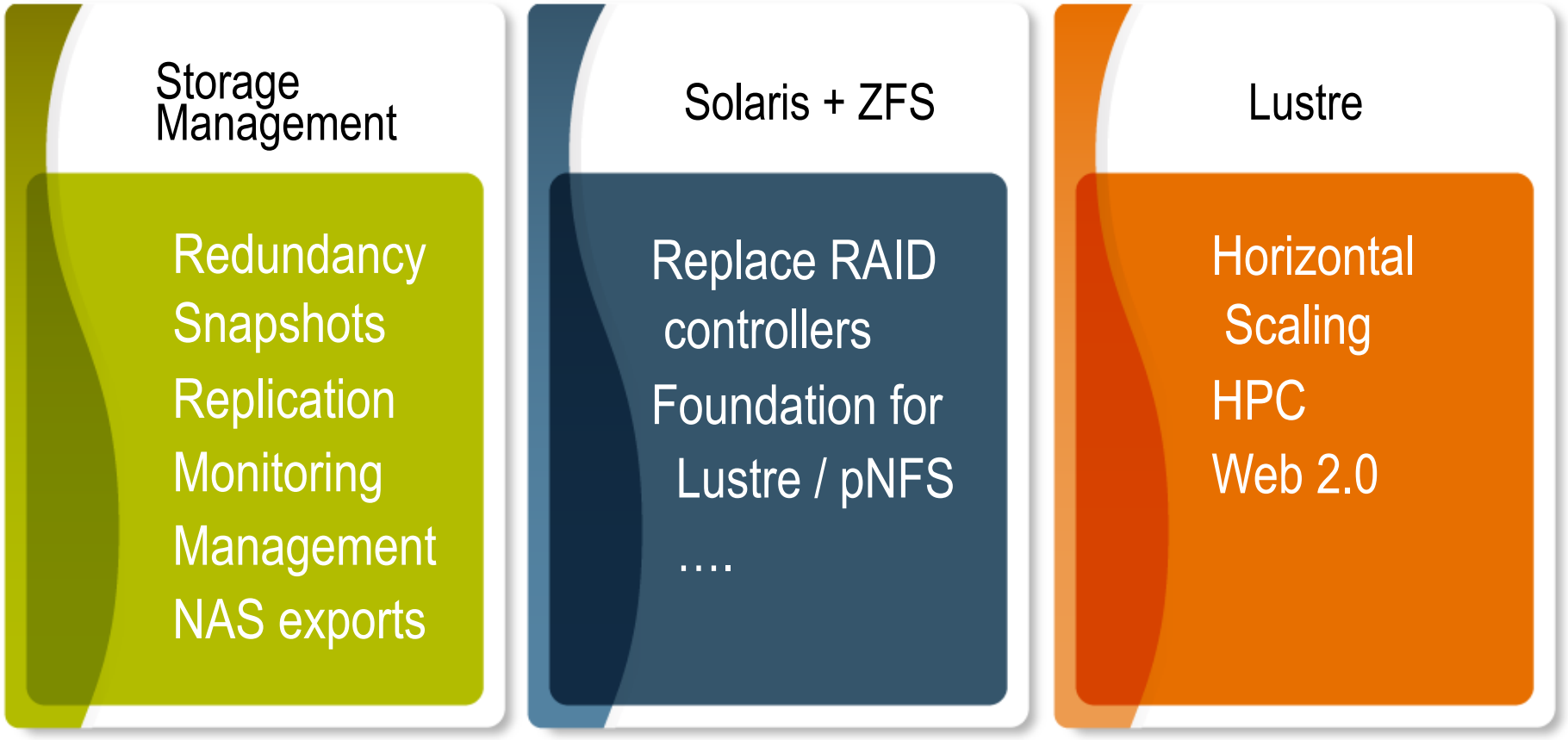
Volumes, labels, partitions, provisioning and lots of limits

Older/Slower Data Management Techniques

Fat locks, fixed block size, naive pre-fetch, dirty region logging

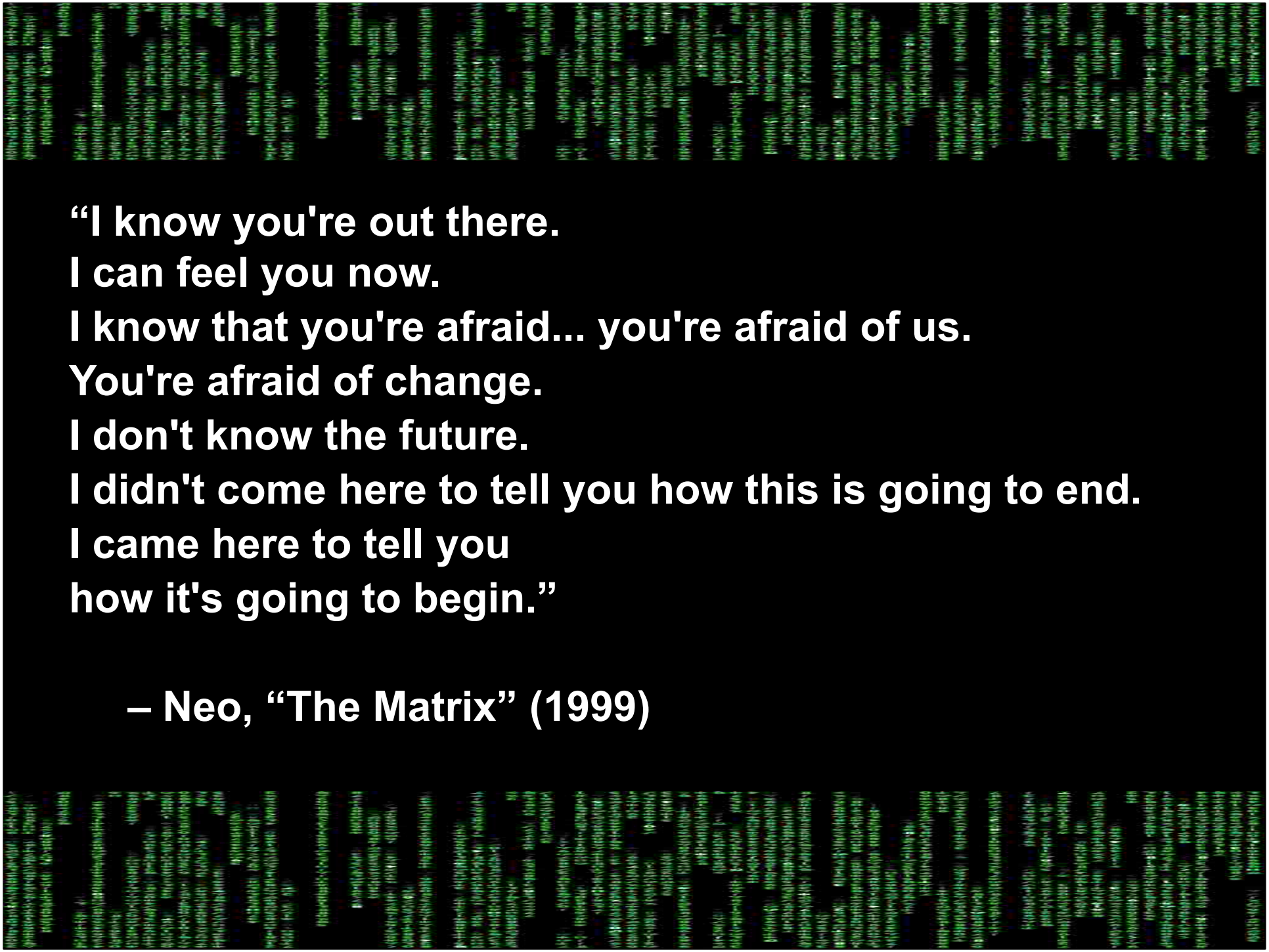
Storage software features

Getting out of the controller…



ZFS re-usability

- Storage controller – iSCSI or IB volume exports
  - With the enterprise goodies
- Local file system
- NAS server
- Storage layer for clustered storage
  - pNFS, Lustre, others


“I know you're out there.
I can feel you now.
I know that you're afraid... you're afraid of us.
You're afraid of change.
I don't know the future.
I didn't come here to tell you how this is going to end.
I came here to tell you
how it's going to begin.”
– Neo, “The Matrix” (1999)

Lustre
introduction

World's Fastest and Most Scalable Storage

- Lustre is the leading cluster file system
  - > 7 of Top 10 HPC systems
  - > Half of Top 30 HPC systems
- Demonstrated Scalability and Performance
  - > 100 GB/sec I/O
  - > 25,000 Clients
  - > Many systems with 1000s of nodes

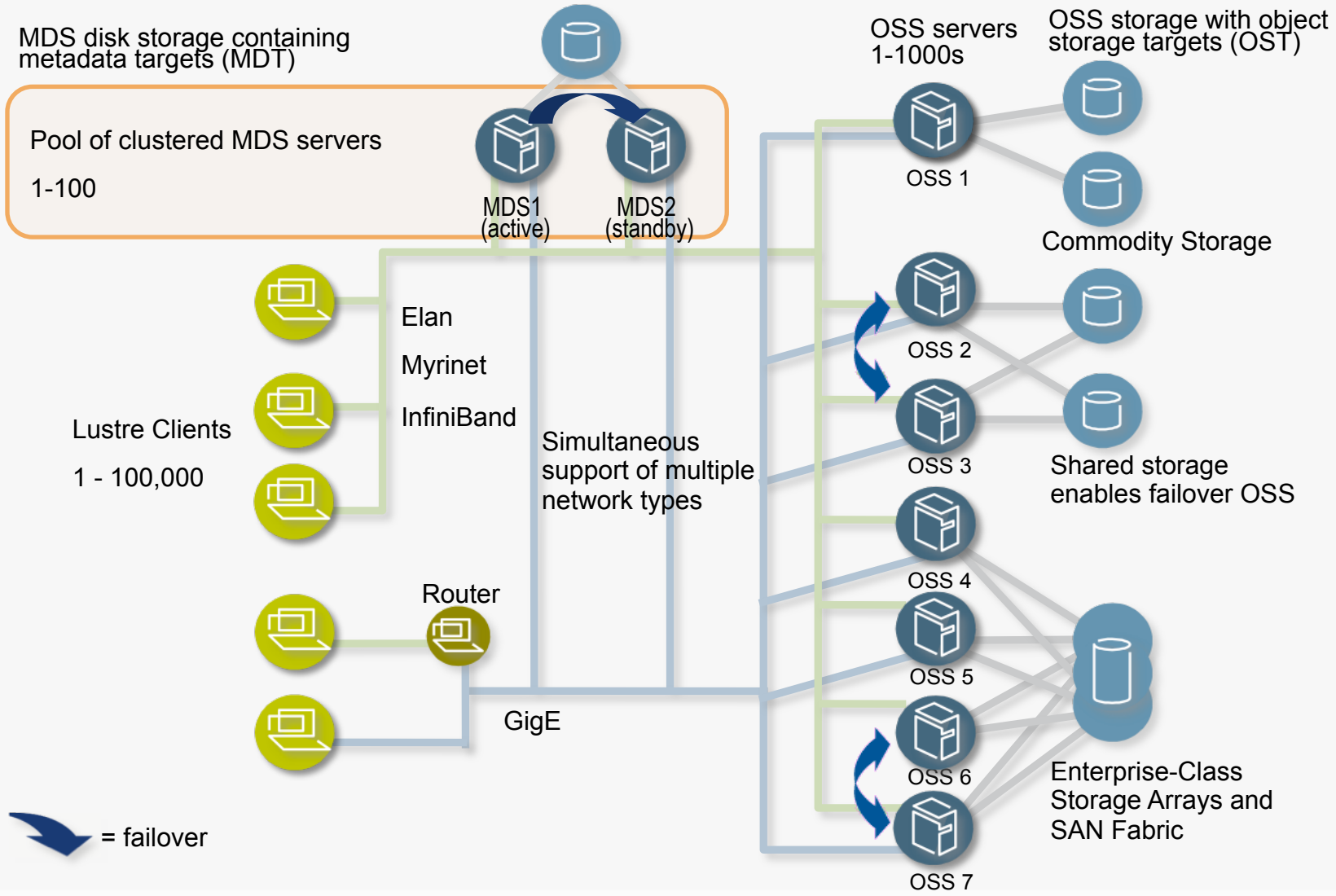
Lustre – scalable file system

- Lustre is a shared file system
  - Software only solution, no hardware ties
  - Developed as company – gvmt lab collaboration
  - Open source, modifiable, many partners
  - Extraordinary network support
  - Smoking performance and scalability
  - POSIX compliance and High Availability
- Lustre is for “extreme storage”
  - Horizontal scaling of IO over all servers
    - parallelizes I/O, block allocation and locking
  - Similar for metadata over MDS servers
  - add capacity by adding servers
  - Example: week1 of LLNL BG/L system: 75M files, 175TB

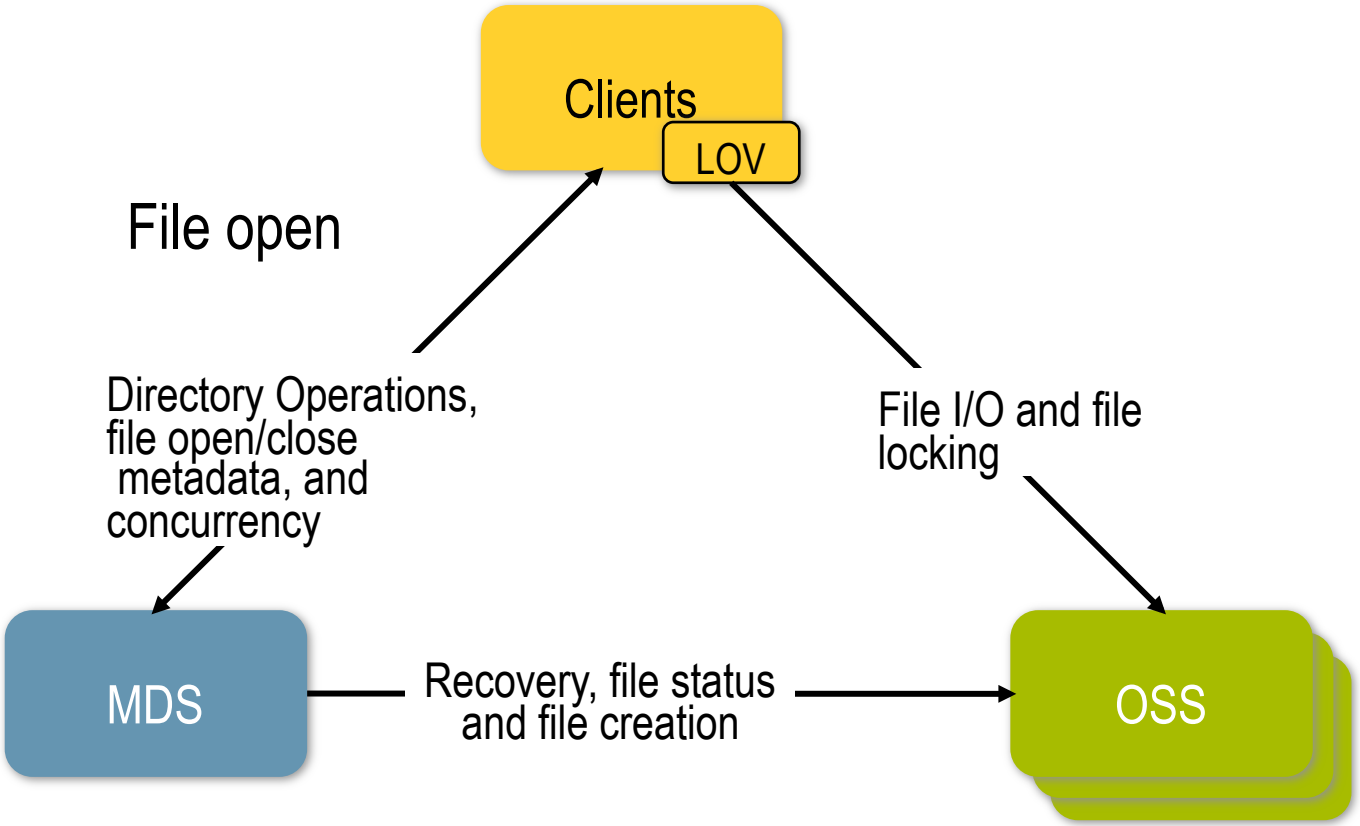
What kind of deployments?

- Extremely Large Clusters
  - > Deployment: extremely high node count, performance
  - > Where: government labs, DoD
  - > Strengths: modifiability, special networking, scalability
- Medium and Large Clusters
  - > Deployment: 32 – low thousands of nodes
  - > Where: everywhere
  - > Strengths: POSIX features, HA
- Very large scale data centers
  - > Deployments: combine many extremely large clusters
  - > Where: LLNL, ISP's, DoD
  - > Strengths: security, networking, modifiability, WAN features

A Lustre Cluster

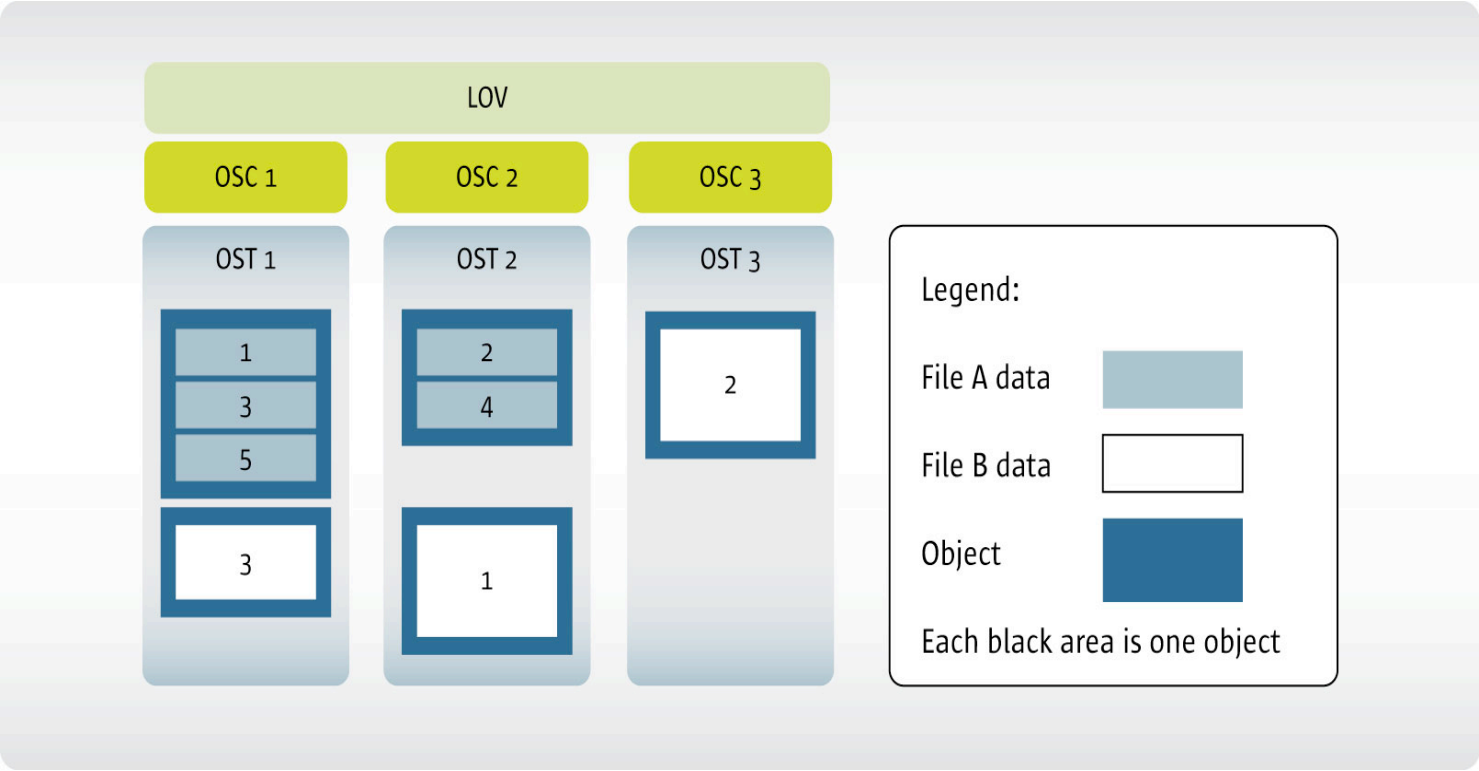


How does it work?



Lustre Stripes Files with Objects

- Currently objects are simply files on OSS resident file systems
- Enables parallel I/O to one file
  - > Lustre scales that to 100GByte/sec to one file



Vision

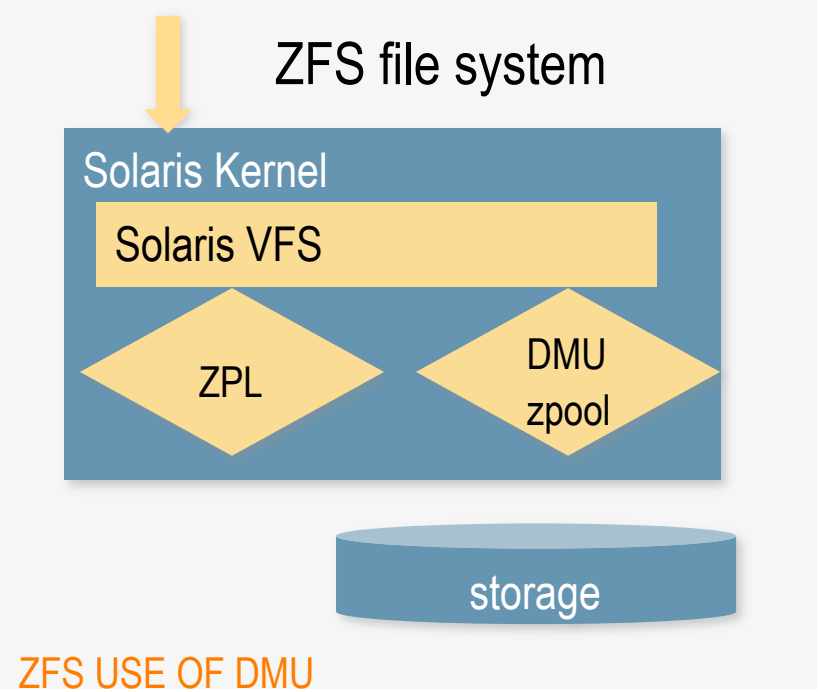
Facet	Activity	Difficulty	Priority	Timeframe
Product Quality	Major work is needed except on networking	High	High	2008
Performance fixes	Systematic benchmarking & tuning	Low	Medium	2009
More HPC Scalability	Clustered MDS, Flash cache, WB cache, *Request Scheduling*, Resource management, ZFS	Medium	Medium	2009-2012
Wide area features	*Secuirty,* WAN performance, proxies, replicas	Medium	Medium	2009-2012
Broad adoption	Combined nNFS /Lustre exports	High	Low	2009-2012

Note: These are visions, not commitments

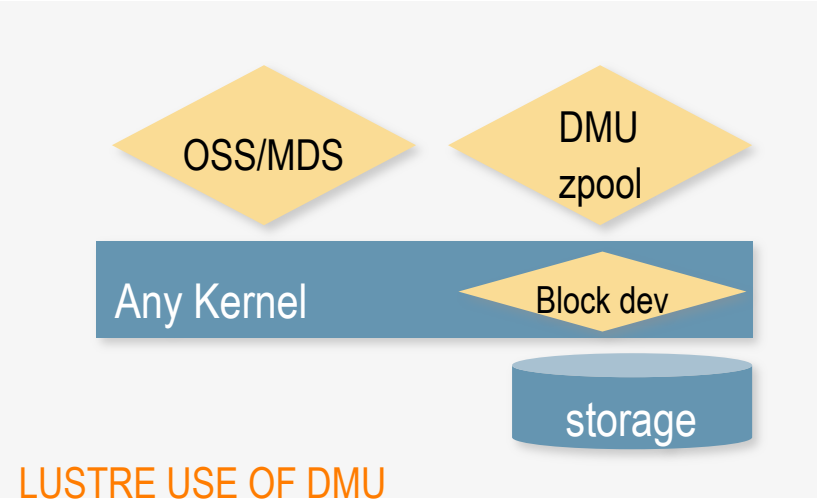
Lustre
ZFS-DMU

Lustre & ZFS

- User space!
  - > DMU talks to block devices
  - > OSS / MDS talks to DMU
    - > ztest and FUSE work similarly
  - > LNET: user space or kernel
- OSS / MDS
  - > Will write ZFS formats on disk
    - > Like we currently write ext3
  - > Use DMU API's for transactions
- DMU
  - > Already ported to Linux, OS X



ZFS USE OF DMU



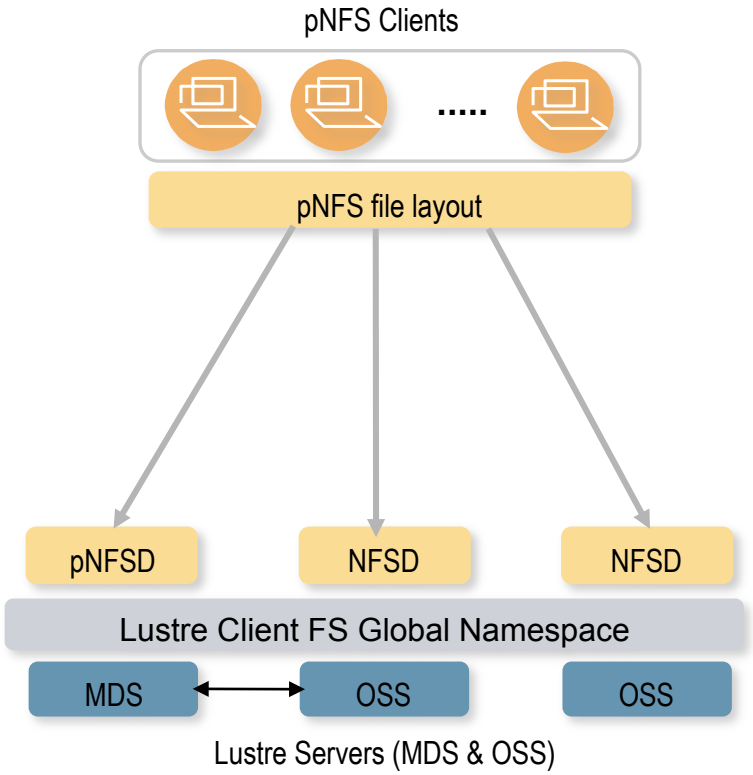
LUSTRE USE OF DMU

Lustre
pNFS

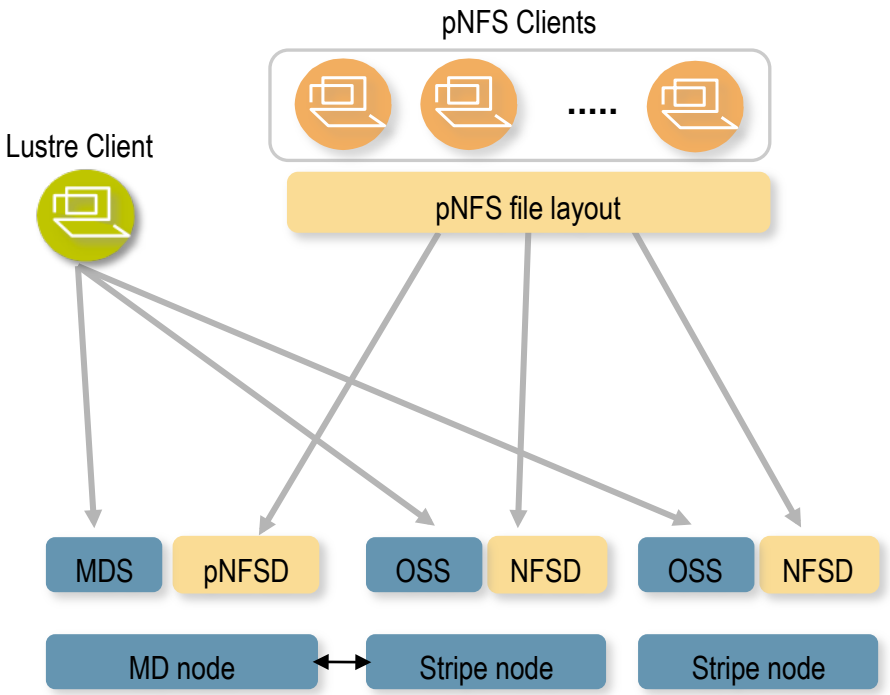
pNFS & Lustre

- pNFS integration
- Soon – pNFS exports from Lustre on Linux
  - First participation in a Bakeathon by Lustre!
- Longer term possibilities
  - Let Lustre servers offer pNFS & Lustre protocol
    - Requires an interesting Lustre storage layer
  - Make LNET an RDMA transport for NFS?
  - Offer proven Lustre features to NFS standards efforts

Layered & direct pNFS



pNFS layered on Lustre Clients



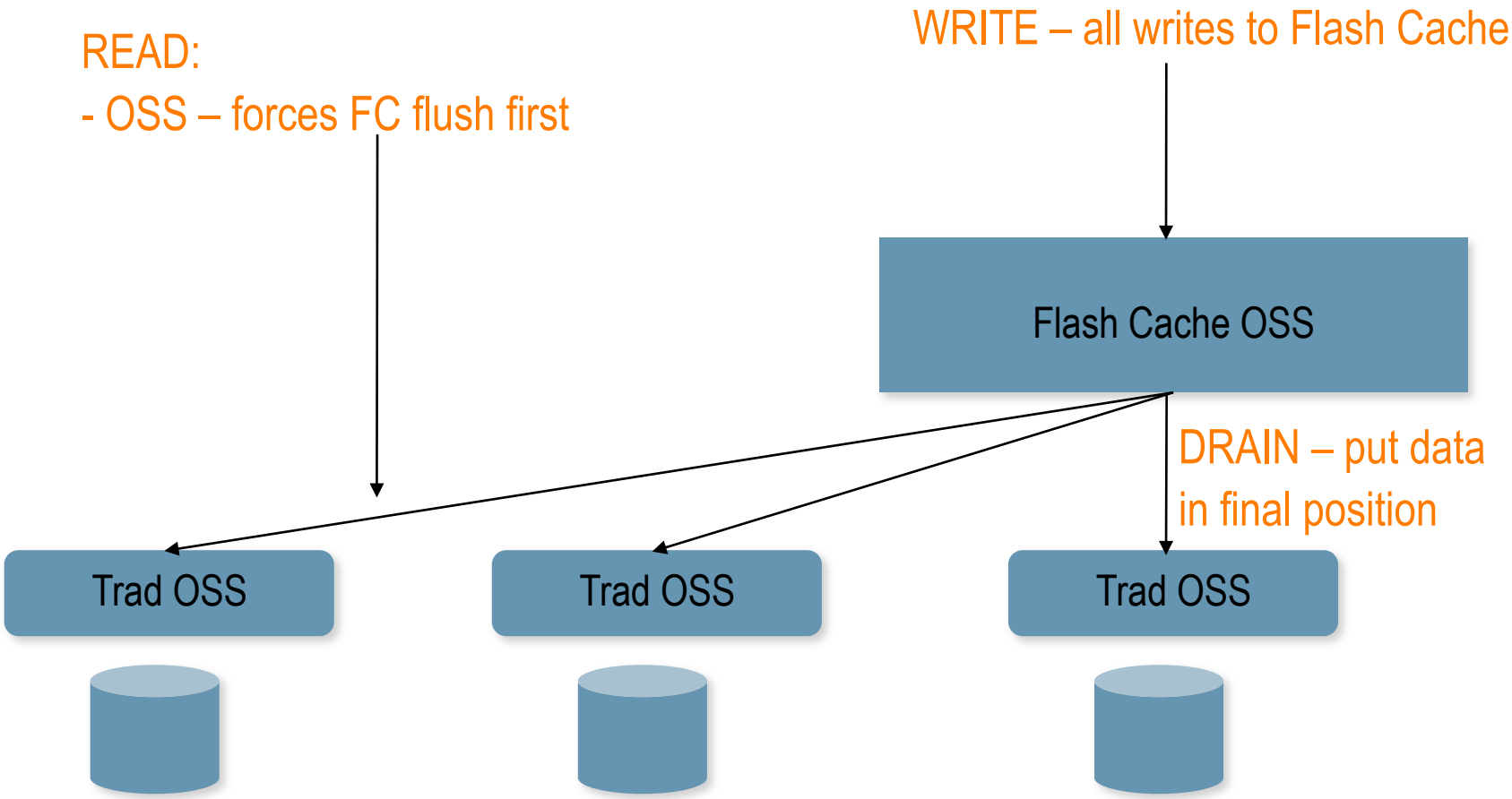
pNFS and Lustre servers on
Lustre / DMU storage system

Lustre
flash cache

Flash cache

- Exploit storage hardware revolution
  - Very high bandwidth available from flash
  - Add Flash Cache OSTs– capacity ~ RAM of cluster
  - Cost: small fraction of cost of RAM of cluster
- Fast I/O from compute node memory to flash
- Then drain flash to disk storage - ~ 5x slower
  - E.g. cluster finishes I/O in 10 mins, on disk in 50 mins
  - Need 5x fewer disks
- Lustre manages file system coherency

Flash Cache interactions



Lustre

client write back cache

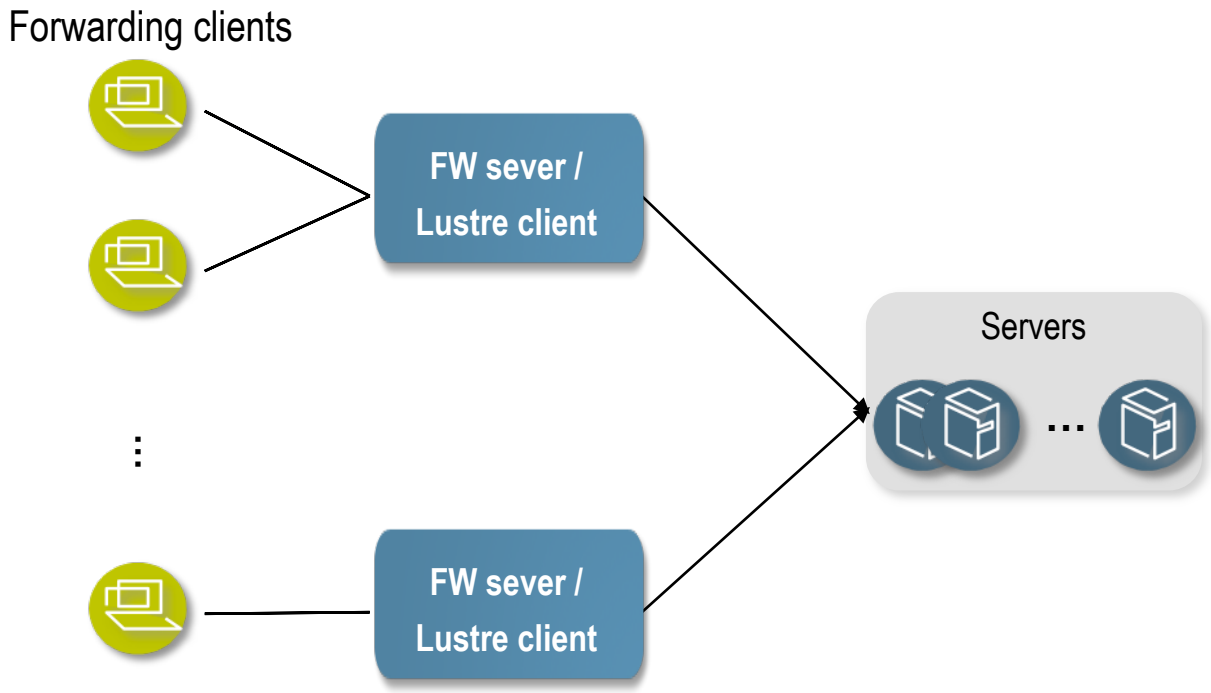
Metadata WBC & replication

- Goal & problem:
  - Disk file systems make updates in memory
  - Network FS's do not - metadata ops require RPCs
  - The Lustre WBC should only require synchronous RPCs for cache misses
- Key elements of the design
  - Clients can determine file identifiers for new files
  - A change log is maintained on the client
  - Parallel reintegration of log to clustered MD servers
  - Sub-tree locks – enlarge lock granularity

Uses of the WBC

- HPC
  - > I/O forwarding makes Lustre clients I/O call servers
  - > These servers can run on WBC clients
- Exa-scale clusters
  - > WBC enables last minute resource allocation
- WAN Lustre
  - > Eliminate latency from wide area use for updates
- HPCS
  - > Dramatically increase small file performance

Lustre with I/O forwarding



FW servers should be Lustre WBC enabled clients

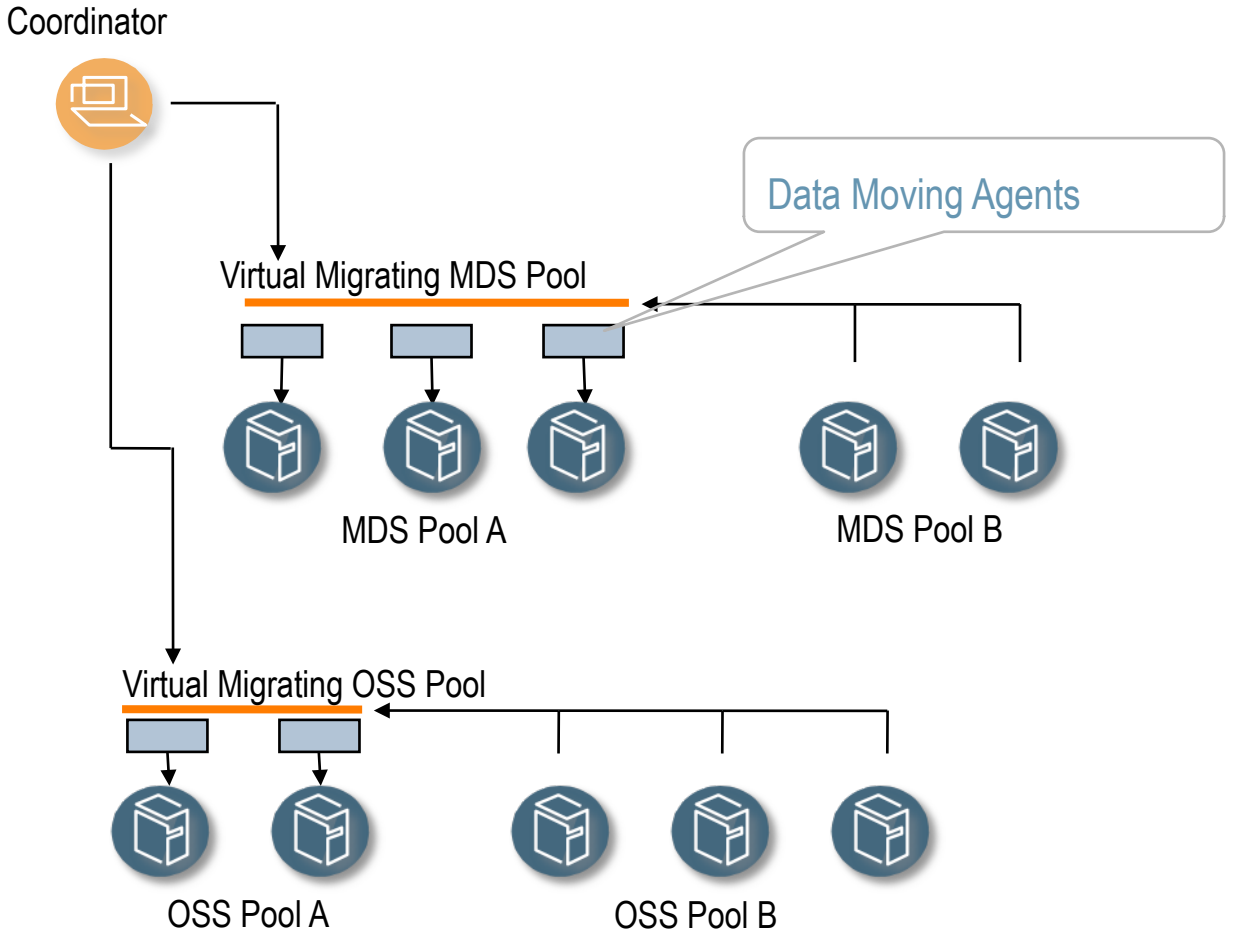
Lustre

data migration & file system replication

Migration – many uses

- Between ext3 / ZFS servers
- For space rebalancing
- To empty servers and replace them
- In conjunction with HSM
- To manage caches & replicas
- For basic server network striping

Migration


Coordinator
Data Moving Agents
Virtual Migrating MDS Pool
MDS Pool A
MDS Pool B
Virtual Migrating OSS Pool
OSS Pool A
OSS Pool B

General purpose replication

- Driven by major content distribution networks
  - DoD, ISPs
  - Keep multi petabyte file systems in sync
- Implementing scalable synchronization
  - Changelog based
  - Works on live file systems
  - No scanning, immediate resume, parallel
- Many other applications
  - Search, basic server network striping

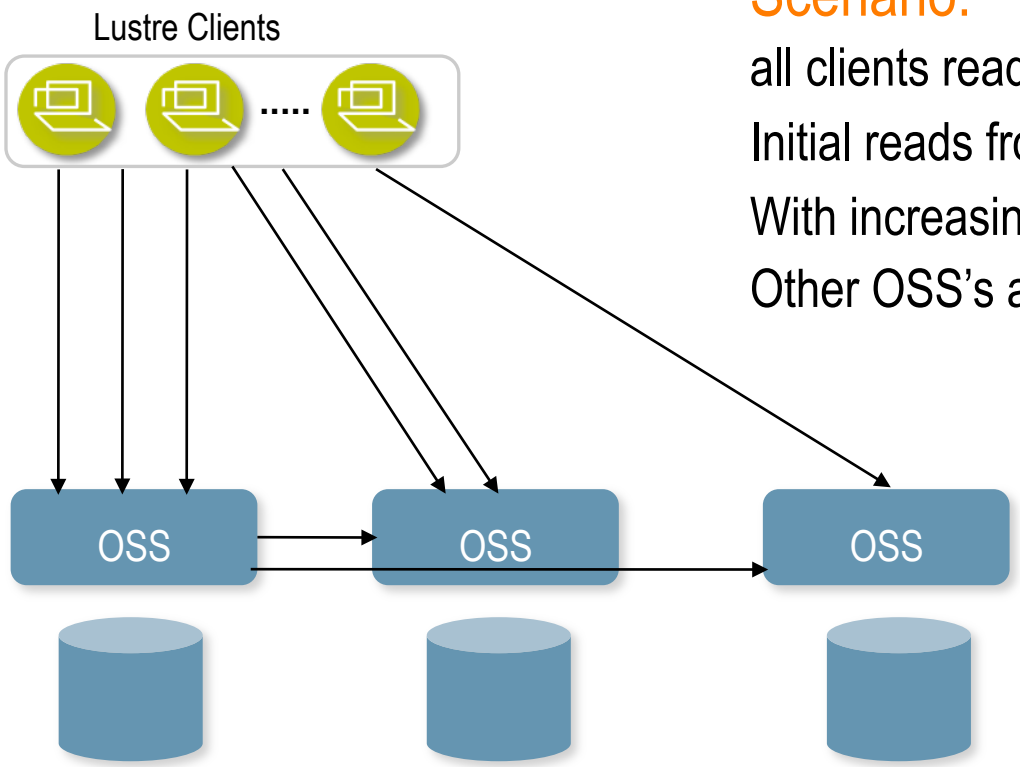
Lustre

server caches & proxies

Caches / proxies

- Many variants
  - HSM – Lustre cluster is proxy cache for 3rd tier storage
  - Collaborative read cache
    - Bit-torrent style reading or
    - When concurrency increases use other OSS's as proxies
  - Wide area cache – repeated reads come from cache
- Technical elements
  - Migrate data between storage pools
  - Re-validate cached data with versions
  - Hierarchical management of consistency

Collaborative cache



Scenario:

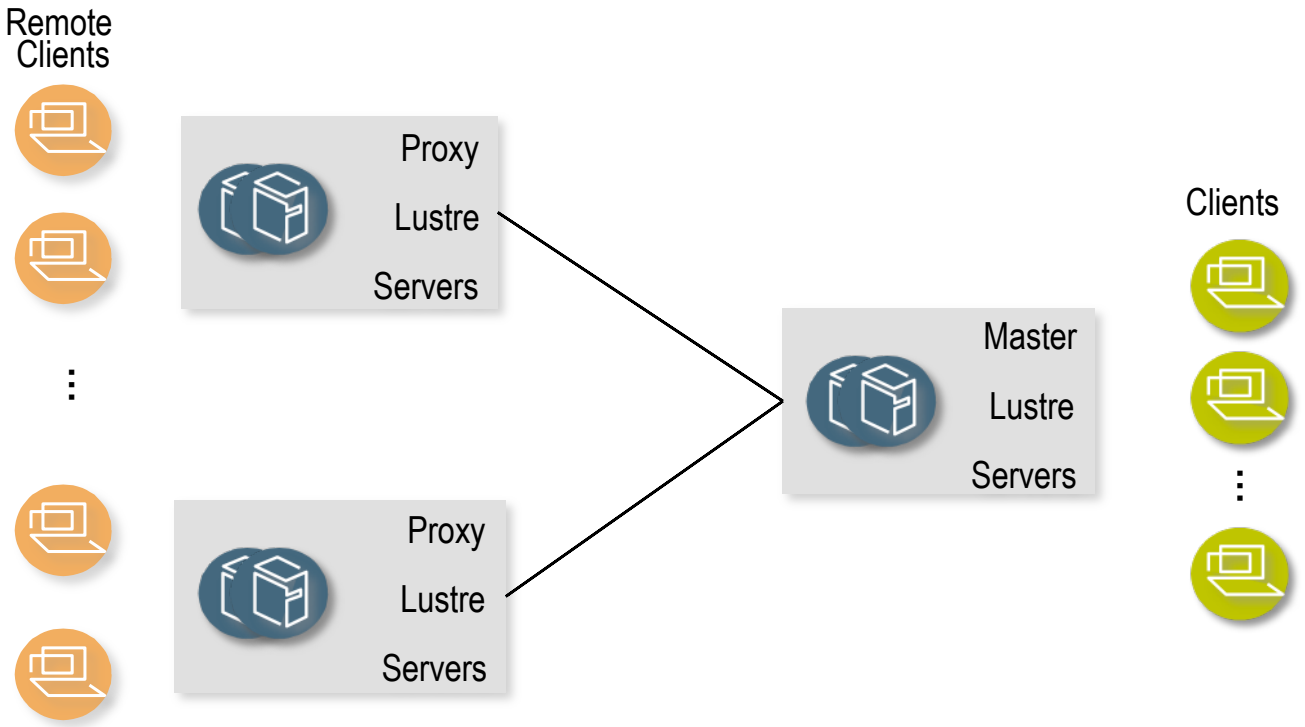
all clients read one file

Initial reads from primary OSS

With increasing load – redirect

Other OSS's act as caches

Proxy clusters



Local performance after the first read

peter.braam@sun.com