

# Data Structure Consistency Using Atomic Operations in Storage Devices

*Ananth Devulapalli*   Dennis Dalessandro   Pete Wyckoff

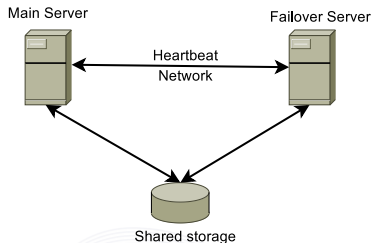
Ohio Supercomputer Center

SNAPI 2008

- ▶ Why atomics operations on disks?
- ▶ OSD background
- ▶ Present locking mechanisms
- ▶ Design space for implementing locks
- ▶ A case study
- ▶ Experimental evaluation
- ▶ Integration of atomics in future OSDs

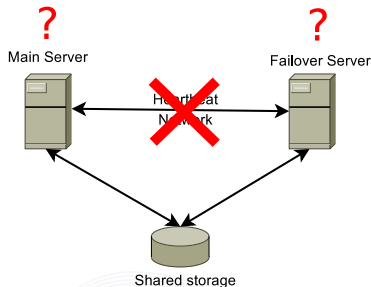
# Motivating scenarios

- ▶ Failover mechanism
  - Networks for heartbeat fails
  - Split brain problem
- ▶ Sometimes disks are the only way of communication
- ▶ Lack of atomics  $\Rightarrow$  serialized ops rely on third party
- ▶ SCSI RESERVE too coarse grained



# Motivating scenarios

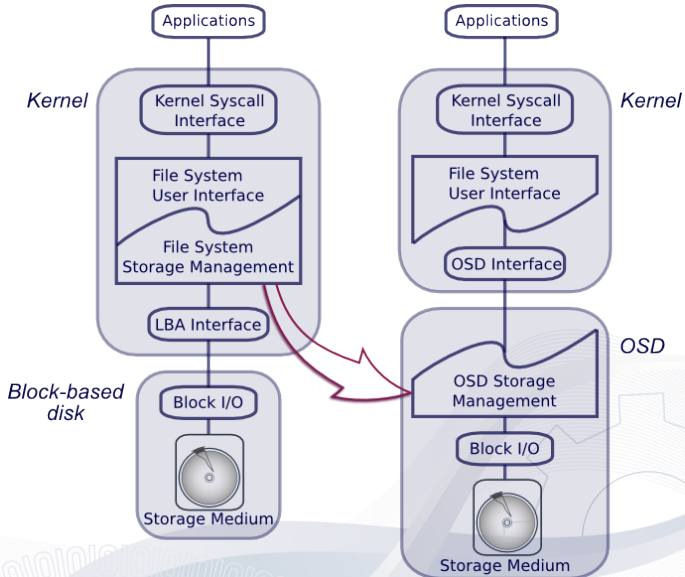
- ▶ Failover mechanism
  - Networks for heartbeat fails
  - Split brain problem
- ▶ Sometimes disks are the only way of communication
- ▶ Lack of atomics  $\Rightarrow$  serialized ops rely on third party
- ▶ SCSI RESERVE too coarse grained



# Intelligent hard drives

- ▶ Active Disks, Active Storage, iDisks for decision support systems
- ▶ NASD precursor to OSD
- ▶ Object-based storage Devices, ver. 2 rev. 4 out this August

# Object-based Storage Device (OSD) Architecture



# Object-based Storage Device (OSD) Architecture (contd)

- ▶ OSD Technology
  - SCSI Extension
- ▶ Object
  - Root, Partition, Collection, User Object
- ▶ Object encapsulation
- ▶ Attributes
  - User assigned, but device managed
  - Large space for rich metadata
  - Shared attributes
- ▶ Transport
  - iSCSI, iSCSI Extensions for RDMA (iSER), Fiber Channel

## Object

### Attributes

Attribute Page	Attribute Number	Value
1	9	"foo"
2	1	1MB
3	1	01:00:00
3	2	02:00:00

### Data

```
1010101111100010101
101010101010101010
0000000111111100101
.....
```

# Locking in OSD

- ▶ OSD exports higher level abstraction
- ▶ Locking makes sense in context of objects
- ▶ Exploit OSD's on-board accounting and management infrastructure

# Current approaches for locking

- ▶ Simulate locks by sequence of reads/writes
- ▶ SCSI RESERVE and SCSI RELEASE
  - Coarse operation
  - Vulnerable to bus reset or power failure
  - *Persistent* reservations in SCSI primary command, ver. 3

- ▶ At present no atomic operations supported by OSDs
- ▶ LOCK command
  - Claims object for a client
  - Requires separate state
  - Every command needs a state check to enforce locking semantics
- ▶ Leverage OSD attributes
  - Atomic primitives to modify attributes
  - No separate resources required
  - Leverages existing attribute infrastructure
  - Minimize side effects
- ▶ Several atomics primitives
  - Compare-and-Swap (CAS) and Fetch-and-Add (FA)

# Design space for implementing locking

- ▶ Number of atomic attributes
  - One is natural, but induces false sharing
- ▶ Location of locking attribute
  - Forces apps. to use fixed location
- ▶ Argument length
- ▶ Undefined attributes
- ▶ Advisory or mandatory
  - Can `SET ATTRIBUTES` modify lock attributes?
- ▶ Isolation level
  - Operation, object, sub-object levels?
- ▶ CAS effect on other attributes

# Implementation

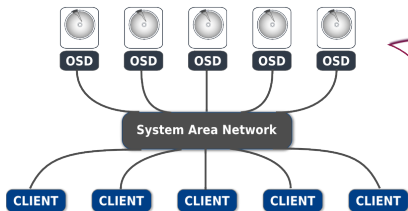
- ▶ Defined two new bidirectional commands: CAS and FA
- ▶ Modeled after `SET ATTRIBUTES` command

Design dimension	CAS	FA
Number of atomic attr.	$\geq 1$	$\geq 1$
Location of locking attr.	Not fixed	Not fixed
Argument length	64 kB	8 B
Undefined attributes	“cmp” succeeds	Treated as zero
Advisory or Mandatory	Advisory	Advisory
Isolation level	At attribute level	At attribute level

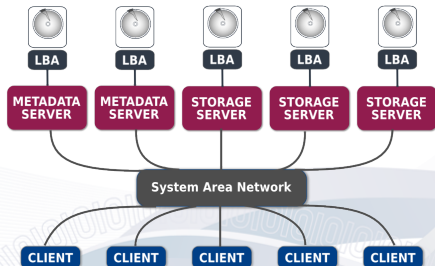
- ▶ CAS effect on other attributes
  - Deferred due to lack of features and use case.

# Case study: Serverless PVFS

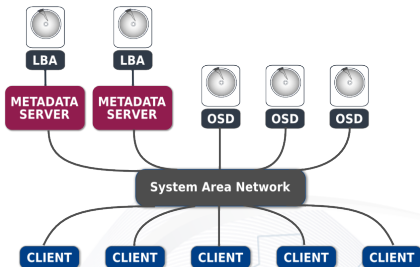
## DIRECT-ACCESS MODEL



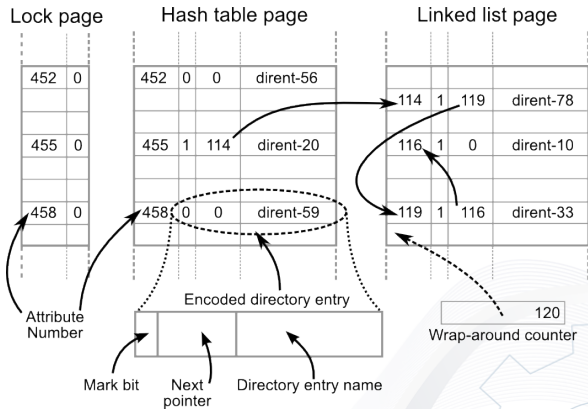
## SERVER FRONTED MODEL



## OSD AS STORAGE SERVER



# Case study: Serverless PVFS (contd.)

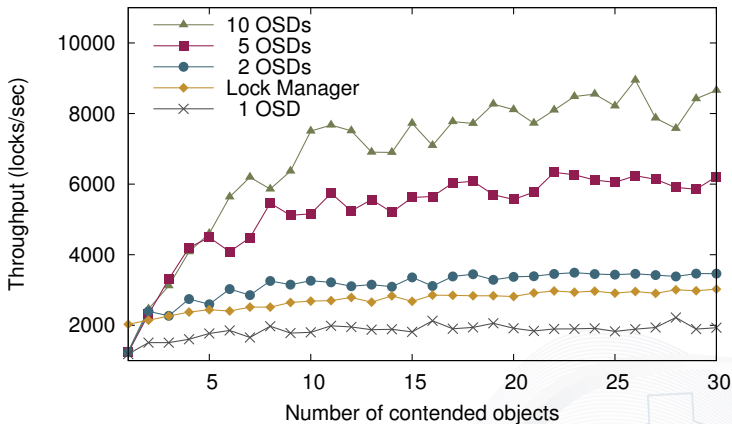


- ▶ Traditional implementation using DLM
- ▶ Some caveats:
  - Measure relative performance of on-disk locks with DLM
  - Not designed to demonstrate performance of real OSD
  - Demonstrate within and cross device parallelism
  - Lock manager implemented as a single server
  - Light weight DLM protocol: only exclusive locks
  - DLM commits its state regularly
- ▶ OSDs: no active entity to serialize their requests
  - We use a variation of truncated exponential backoff

# Experimental setup

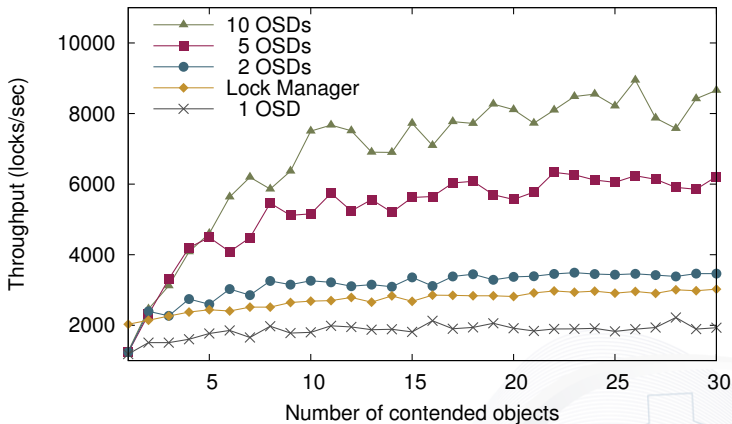
- ▶ 30 nodes cluster
- ▶ Dual AMD Opteron 250
- ▶ 2 GB RAM, 80 GB SATA disk
- ▶ Linux 2.6.25-rc1
- ▶ Tigeon 3 Gigabit Ethernet NIC
- ▶ Single SMC 8648T 48-port switch

# Lock throughput



- ▶ Throughput inversely related to contention
- ▶ Single DLM better, but OSDs leverage economies of scale

# Lock throughput



- ▶ Throughput inversely related to contention
- ▶ Single DLM better, but OSDs leverage economies of scale

**Leverage large number of disks against servers**

- ▶ OSD-2 rev 3 incorporates “Isolation level” ✓
  - Atomicity guarantees at data and attribute level
  - No facilities for CAS or FA
- ▶ Closely working with OSD TWG to incorporate and generalize changes

- ▶ Acknowledgment: NSF
- ▶ OSD source code: *git://git.open-osd.org/*
- ▶ *http://www.osc.edu/research/network\_file/projects/object/index.shtml*
- ▶ Contact: `ananth@osc.edu`