Revisiting the Metadata Architecture of Parallel File Systems

Nawab Ali[#1], Ananth Devulapalli[*2], Dennis Dalessandro[*3], Pete Wyckoff[*4] and P. Sadayappan[#5]

[#]The Ohio State University
[*]Ohio Supercomputer Center

Presentation Outline

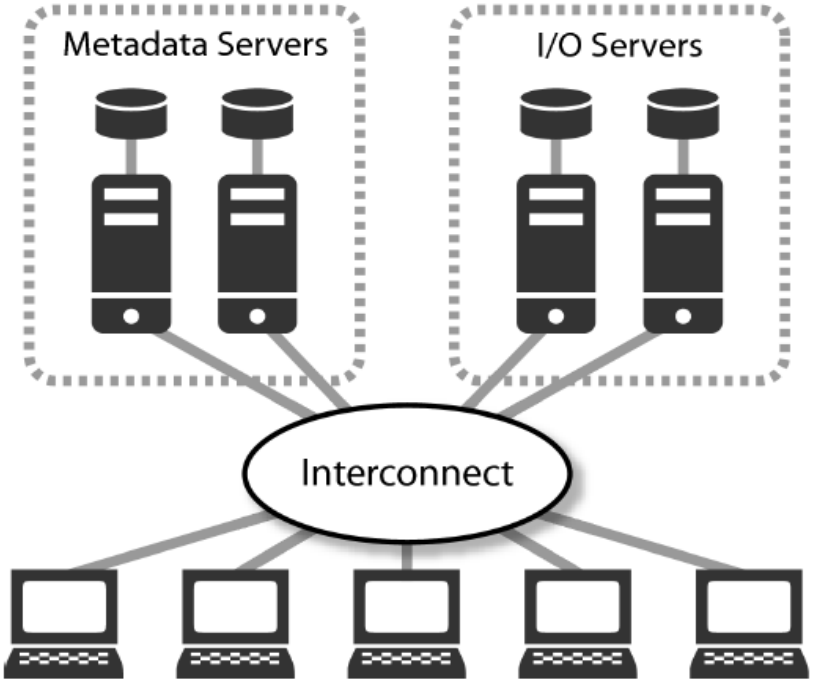
- Introduction
- Object-based Storage Devices
- Parallel File Systems Design Goals
- Offloading MDS Operations to OSDs
- OSD System Design
- Experiments
  - Microbenchmarks
  - Applications
- Conclusions & Future Work

Introduction

- HPC applications increasingly generate or process large data sets
  - Sloan Digital Sky Survey
  - Large Hadron Collider
  - Climate Research
- Existing parallel file systems unable to cope with the I/O throughput requirements
  - Server-oriented design inhibits high-performance

Parallel File System Design Limitations

- I/O bandwidth and latency limit file system performance
- Store-and-forward latency
- Dedicated I/O and metadata servers limit
  - Scalability
  - Manageability
  - Performance

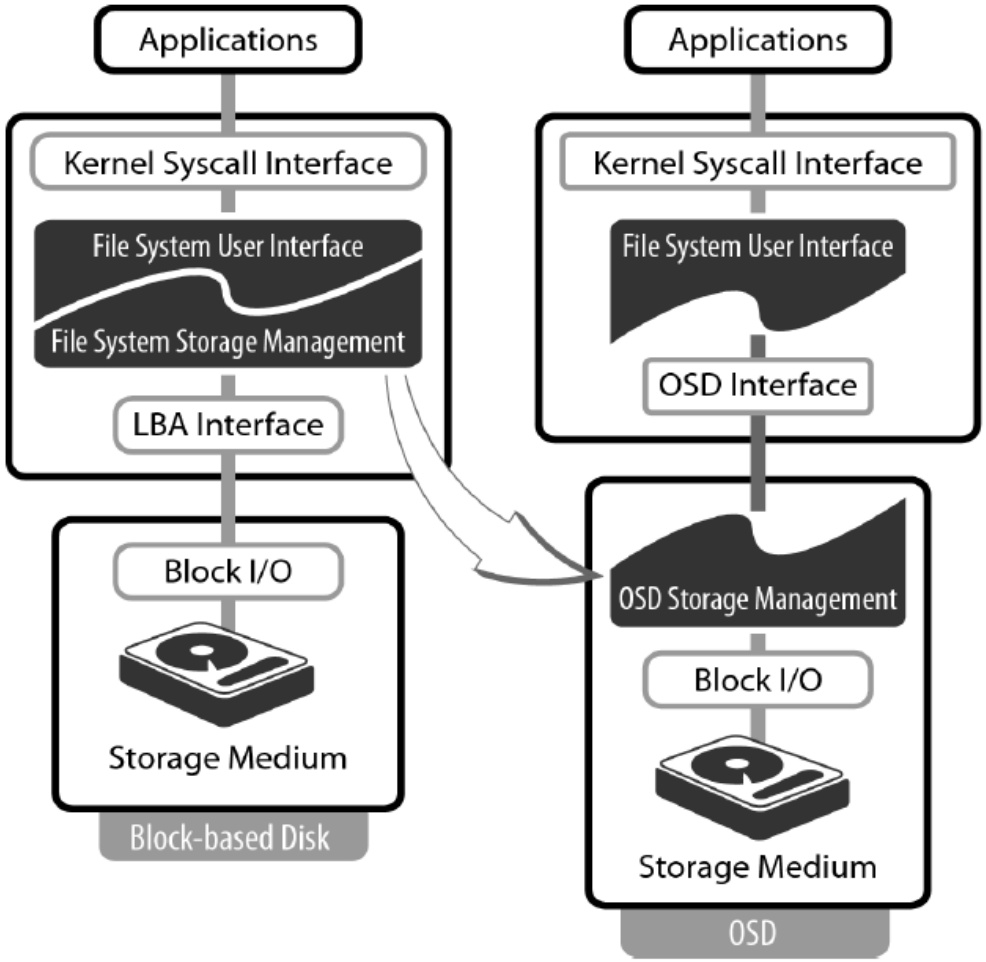


Typical parallel file system design

Object-based Storage Devices

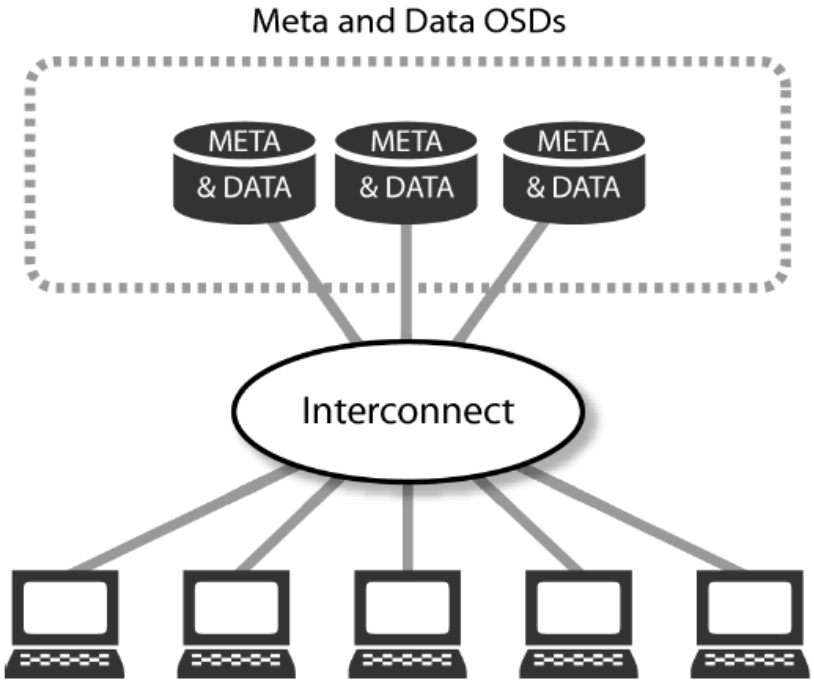
- New storage technology
  - SCSI extension
- Intelligent, higher-level object interface
  - Object encapsulation
  - Attributes
    - User assigned, but device managed
    - Large space for rich metadata
- Secure building block for direct-access file systems

OSD Architecture



Parallel File System Design Goals

- Use intelligent peripherals (OSDs) to improve performance, scalability and manageability
- Serverless, direct-access storage model

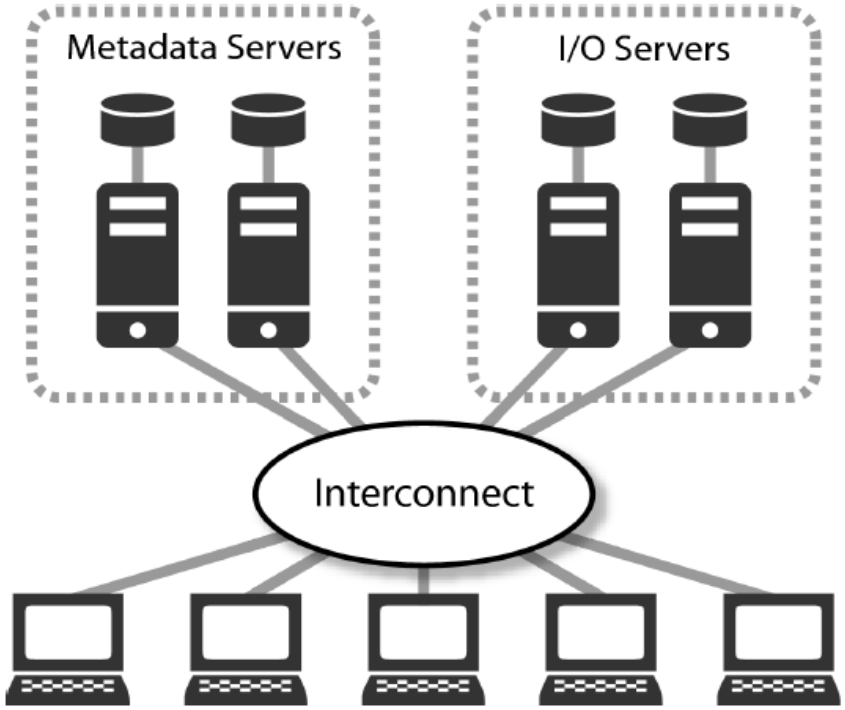


OSD-based parallel file system design

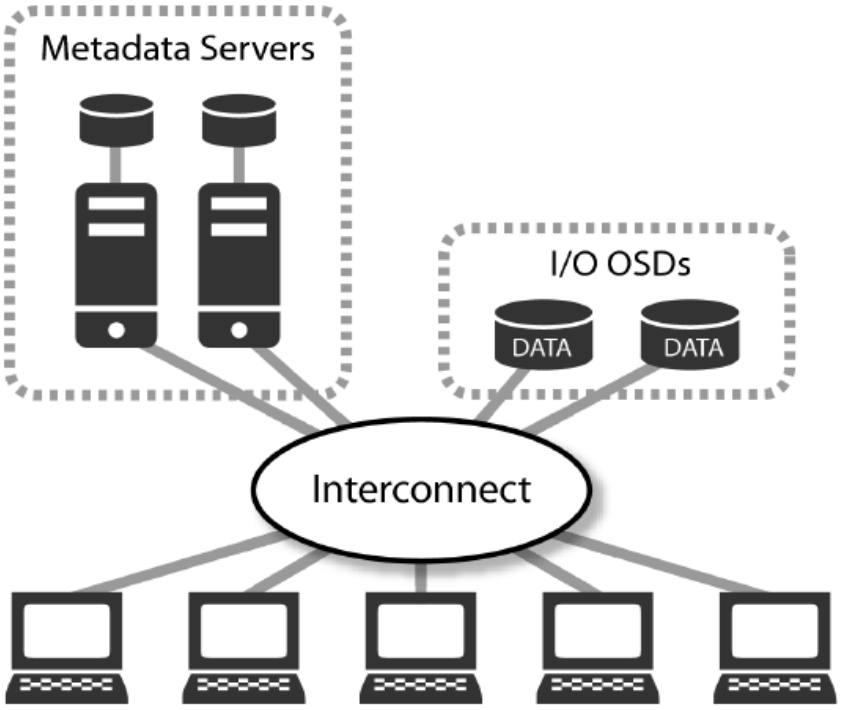
Metadata Design Goals

- Offload file metadata operations to OSDs
- Make the case for recoupling data and metadata
  - Simplify parallel file system design

Existing Metadata Architectures

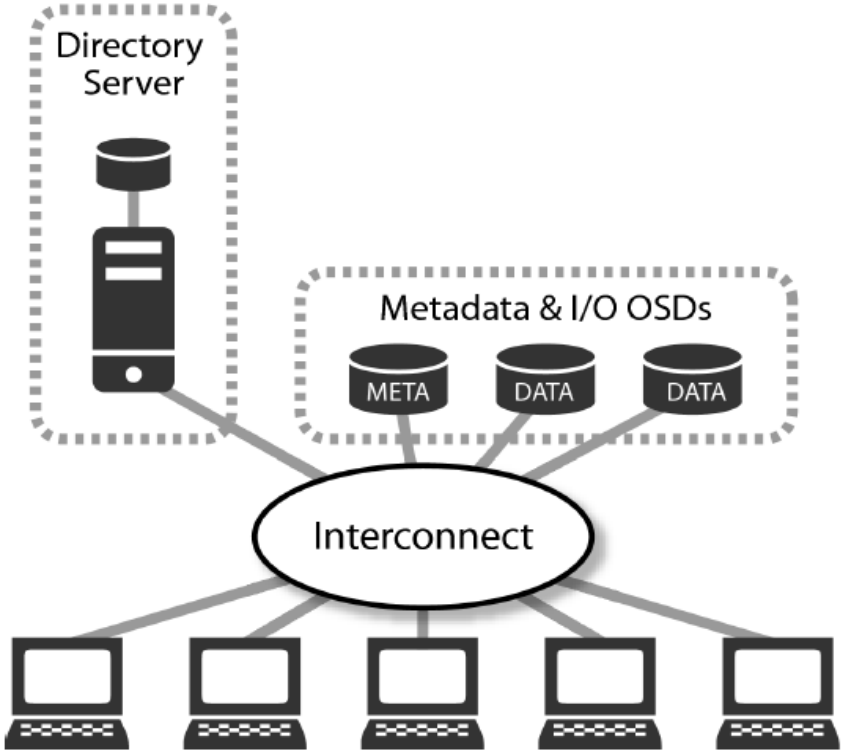


Stock PVFS

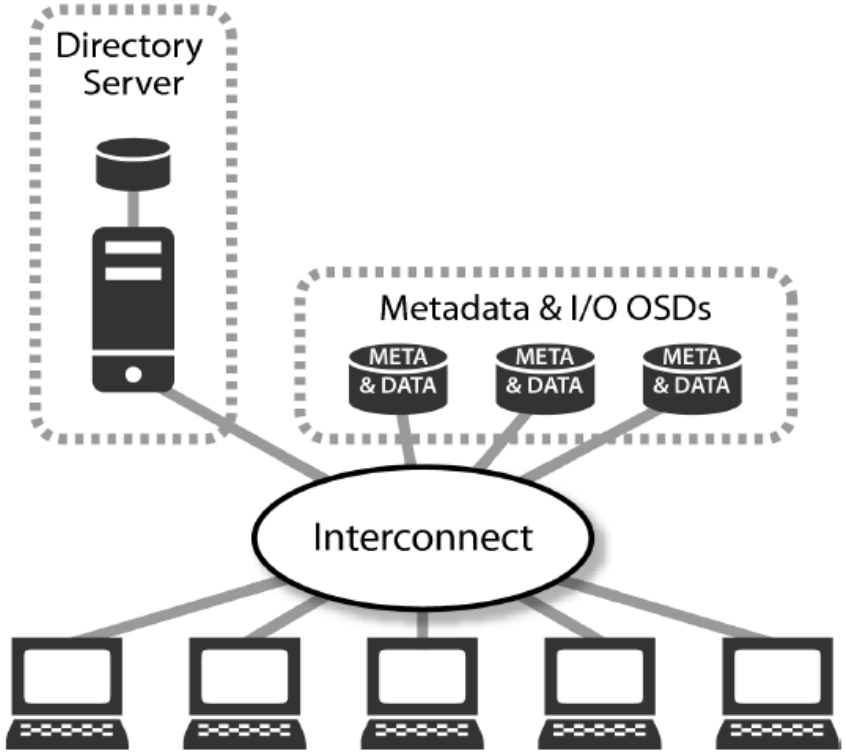


OSD IOS

Offloading MDS Operations to OSDs



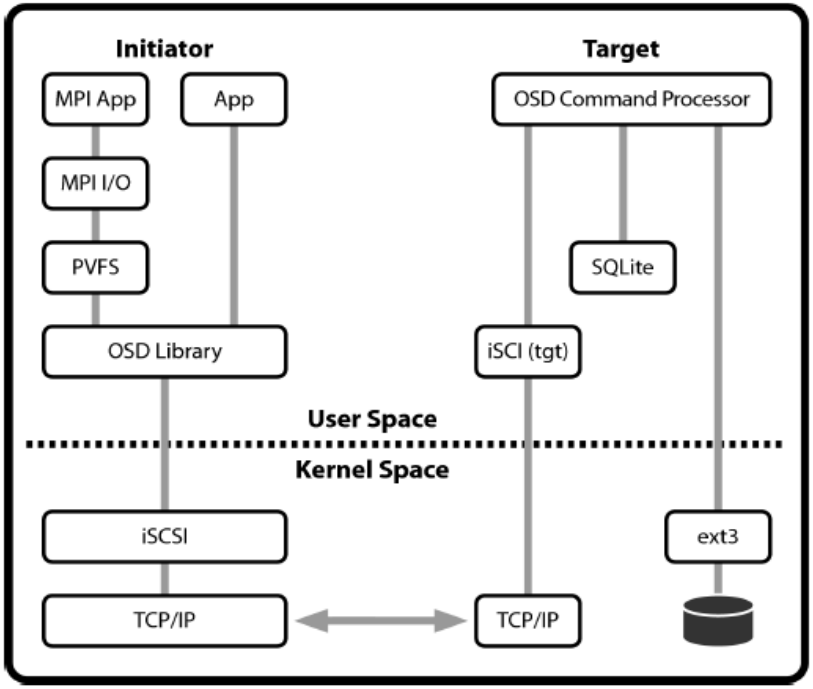
Dedicated OSD MDS



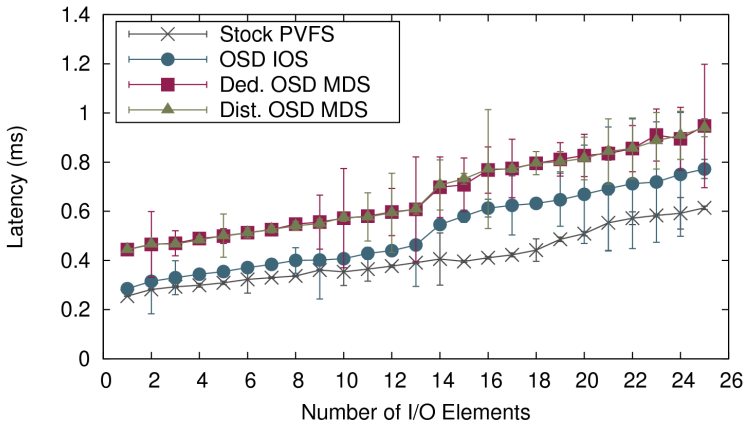
Distributed OSD MDS

OSD System Design

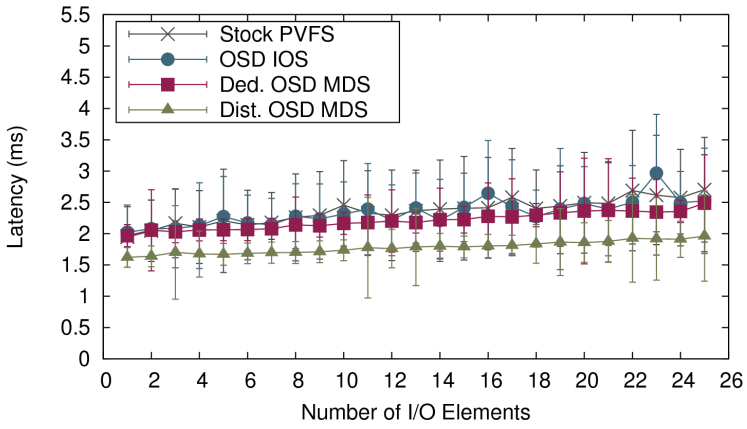
- OSD Initiator
  - Exports OSD interface to client applications
  - Generates SCSI commands
- OSD Target
  - Software Implementation of OSD
  - OSD Command processor
  - Data Management
  - Attribute Management



Latency Microbenchmarks



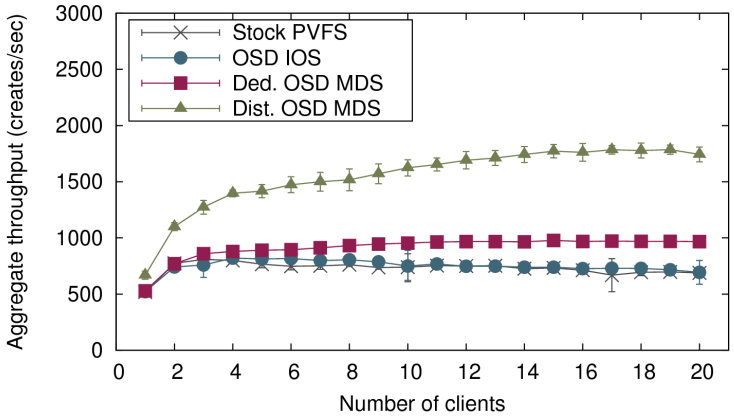
PVFS stat



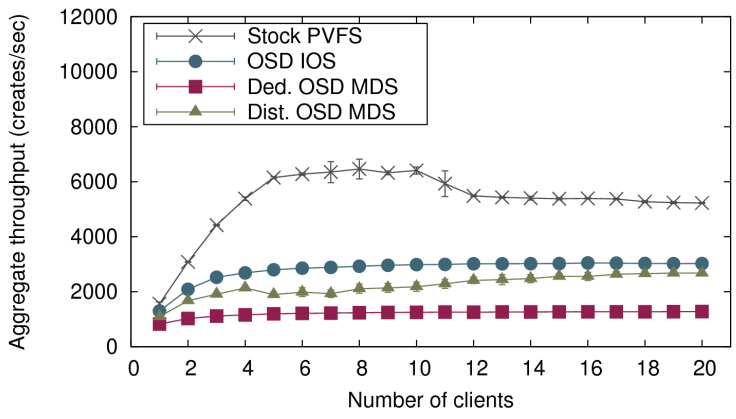
PVFS create

- 1 client
- Latency as a function of number of I/O elements
- PVFS stat: OSD-based MDS has higher latency than PVFS
- PVFS create: Dist. OSD MDS has lower latency because it does not create a metafile

Throughput Microbenchmarks



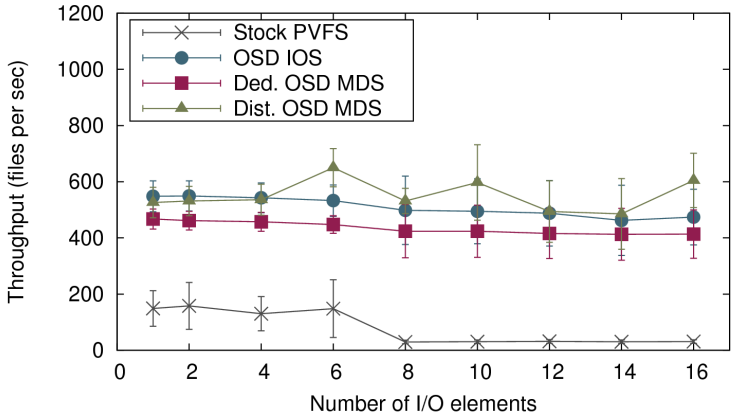
Disk-based storage



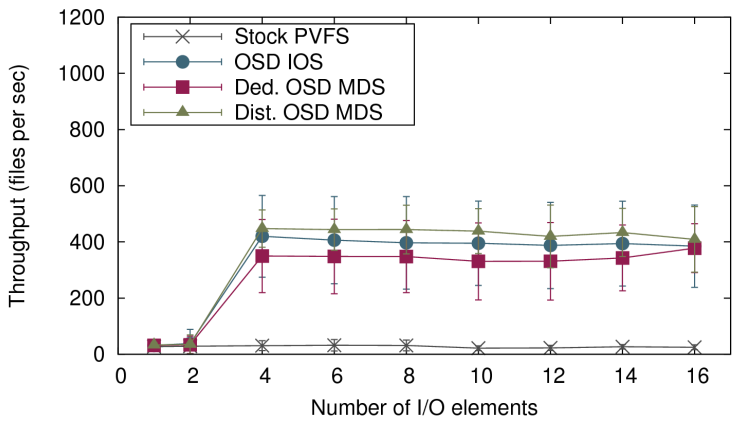
RAM-based storage

- Create throughput as a function of number of clients
- 4 PVFS servers or 4 OSDs respectively
- Disk-based storage: Dist. OSD MDS has a higher create throughput than PVFS
- RAM-based storage: PVFS outperforms other OSD variants

Checkpoint Performance Results



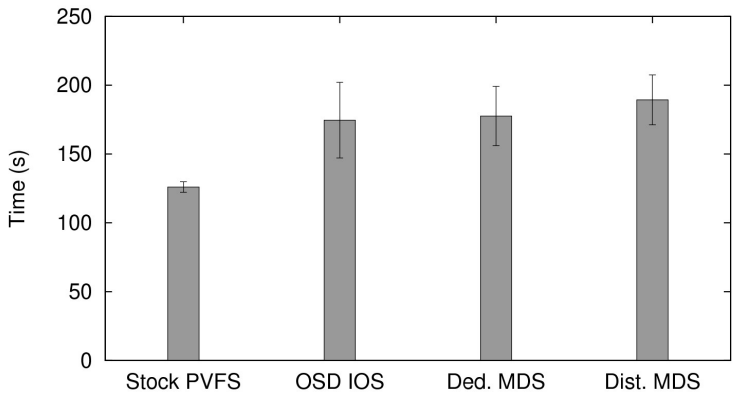
Checkpoint size = 32 kB



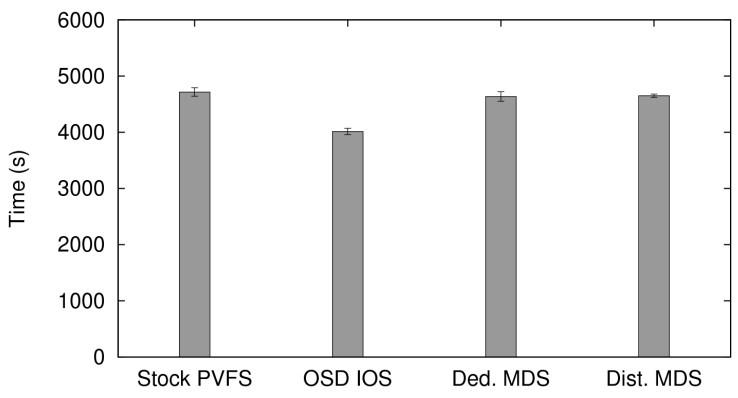
Checkpoint size = 256 kB

- 8 clients, varying I/O elements
- Processes write to individual checkpoint files
- Distributed OSD MDS performs better than other schemes
- Performance degrades with number of I/O elements because of datafiles

SSCA-3 Performance Results



SSCA-3 test1



SSCA-3 test7GB

- 1 client. 4 PVFS servers or 4 OSDs respectively
- test1: 5000 files. 1 GB data. Small metadata footprint
  - Execution time dominated by small I/O
- test7GB: 95000 files. 7 GB data. Metadata intensive

Conclusions

- Dedicated MDS limit the performance of parallel file systems
- Possibly recouple data and metadata using OSDs
- Presented two metadata offloading techniques to OSDs
- Performance of OSD-based file system is comparable to that of PVFS
  - Expect performance to improve significantly with the availability of hardware OSDs

Future Work

- Metadata query based on user-defined attributes
- File system scalability analysis
  - OSC Glenn cluster
  - iSER
- Replicated Metadata

Thank You