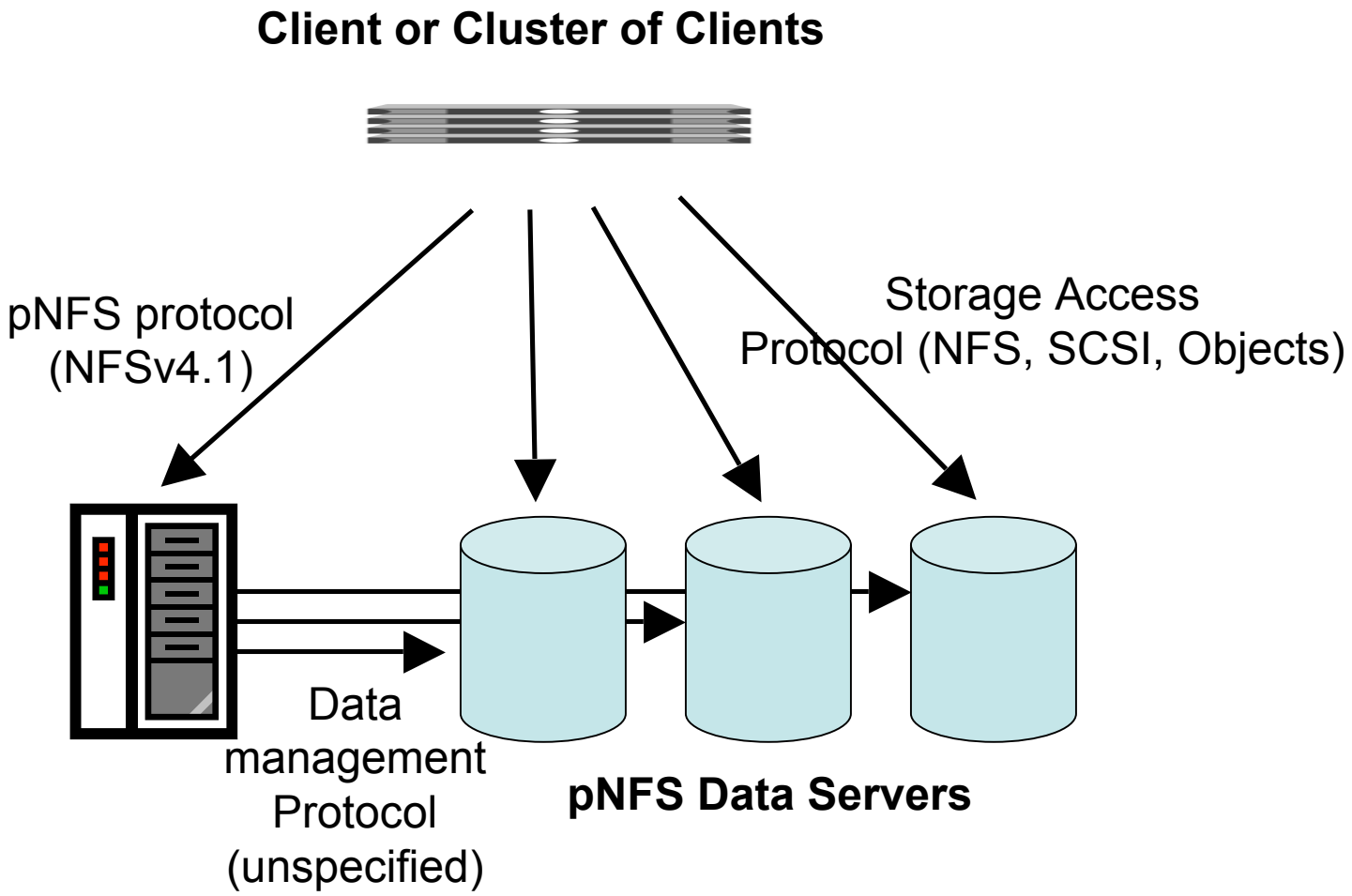
Block-Based pNFS

Andy Adamson CITI, University of Michigan
Sorin Faibish EMC

pNFS Architecture



pNFS Block Requirements

- Identify storage volumes by content
- Represent arbitrarily complex volume topologies per fsid
  - Break down and reset volume topology
  - multiple volume topologies (COW)
- Failover to NFS server
- Server file system block size vs.. page size

Discovery of pNFS exported Disks

- Identify storage by content
- Enumerate SCSI disks
- Match disk signatures returned from pNFS block server visible disk devices on the client.
- Interface into drivers/scsi

Volume Topologies

- Create a dm meta device to represent volume topology
  - Currently use the ioctl interface
- Prototype flattens the topology into a set of dm linear targets
  - Even stripe volume within a stripe volume
- Need to improve use of dm meta device
  - recursive meta device tree (sub-meta devices)
  - Build our own logical tree

Failover to NFS

- Generic pNFS client interface to pNFS I/O is post nfs page cache setup
- If pNFS I/O fails, we are set-up to try NFS
  - We do this for read
- For write, the block pNFS client has new entry point into nfs_write_begin
  - Reads data around block to be written
  - Writes server block size and mark nfs_page as PNFS_IO
  - If Block I/O fails, unmark page: next flush sends over NFS

Block I/O

- Want to use nobh_writepages and friends
  - Assumes no buffer heads
  - Upon failure, allocates buffer heads on page->private
- NFS uses page->private
  - nobh_writepages use of page->private removes ability to failover to NFS
  - Perhaps a new inode operation get_private()
- We also need our own callback
- We currently use our own bios routines

Other Issues

- Server filesystem block size vs. page size
  - Server block size typically 8192
- How to tell when a disk fails
  - What timeout value before trying NFS

pNFS Block Server

- CITI has developed a Python NFSv4.1 server for client testing
- Expanded it into a pNFS block server
  - Export a RAM based SCSI disk(s) over iSCSI
  - ‘File system’ consists of 4 files
  - Does all pNFS operations along with block payloads
  - Performs minimal I/O
- EMC Celerra and Celerra simulator
- Investigating options for native block server for Linux