pNFS: High Performance Parallel IO for the NFS Standard

Brent Welch, Panasas, Inc.
Presented at SC06, Tampa Florida
welch@panasas.com

www.panasas.com

Abstract and Outline

- pNFS
  - An extension to the NFSv4 file system protocol standard that allows direct, parallel I/O between clients and storage devices
  - Eliminates the scaling bottleneck found in today's NAS systems
  - Supports multiple types of back-end storage systems, including traditional block storage, other file servers, and object storage systems
- This talk
  - Motivates pNFS by explaining problems with NAS and SAN
  - Describes the details of the pNFS protocol
  - Describes current and future activity around pNFS by leading storage companies and universities

Why a Standard?

- NFS is the only network file system standard
  - Proprietary file systems have unique advantages, but can cause lock-in
- NFS widens the playing field
  - Panasas, IBM, and EMC want to bring their experience in large scale, high-performance file systems into the NFS community. Sun and NetApp want a standard solution for the HPC market.
  - Broader market benefits vendors
  - More competition benefits customers
- What about open source?
  - NFSv4 Linux client is very important for NFSv4 adoption, and therefore pNFS
  - Still need vendors that are willing to do the heavy lifting required in quality assurance for mission critical storage

NFSv4 and pNFS

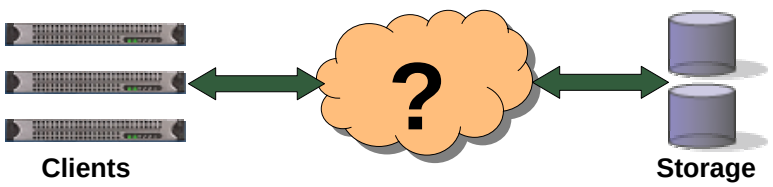
- NFS created in '80s to share data among engineering workstations
- NFSv3 widely deployed
- NFSv4 seven years in the making, lots of new stuff
  - Integrated Kerberos (or PKI) user authentication
  - Integrated File Locking and Open Delegations (stateful server!)
  - ACLs (hybrid of Windows and POSIX models)
  - Official path to add (optional) extensions
- NFSv4.1 adds even more
  - Details learned from early NFSv4.0 experience
  - pNFS for parallel IO
  - Directory Delegations for efficiency
  - RPC Sessions for robustness, better RDMA support

Whence pNFS

- Gary Grider (LANL) and Lee Ward (Sandia)
  - Spoke with Garth Gibson about the idea of parallel I/O for NFS in 2003
- Garth Gibson (Panasas/CMU) and Peter Honeyman (UMich/CITI)
  - Hosted pNFS workshop at Ann Arbor in December 2003
- Garth Gibson, Peter Corbett (NetApp), Brent Welch
  - Wrote initial pNFS IETF drafts, presented to IETF in July and November 2004
- Andy Adamson (CITI), David Black (EMC), Garth Goodson (NetApp), Tom Pisek (Sun), Benny Halevy (Panasas), Dave Noveck (NetApp), Spenser Shepler (Sun), Brian Pawlowski (NetApp), Marc Eshel (IBM), …
  - Dean Hildebrand (CITI) developed pNFS prototype based on PVFS2
  - NFSv4 working group commented on drafts in 2005, folded pNFS into the 4.1 minorversion draft in 2006
- *Many others*

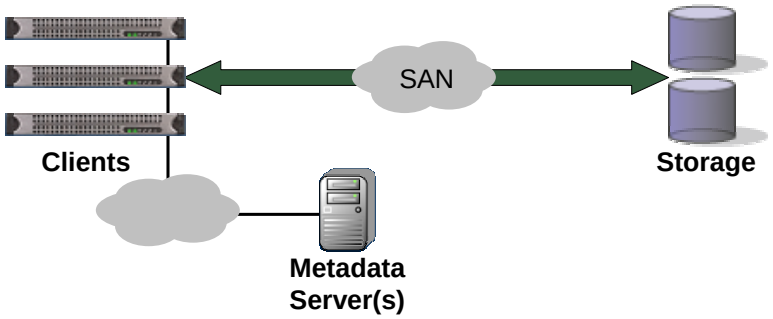
Cluster Storage Problem Statement

- Compute clusters are growing larger in size (8, 128, 1024, 4096…)
  - Scientific codes, seismic data, digital animation, biotech, EDA …
- Each host in the cluster needs uniform access to any stored data
- Demand for storage capacity and bandwidth is growing (GBs/sec)
- Apply clustering techniques to the storage system itself
- Maintain simplicity of storage management even at large scale



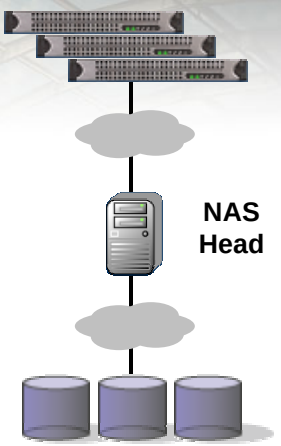
SAN File Systems

- Scalability dependent on scalable block management by the metadata server
  - Require proprietary non-standard file systems
  - NAS access may be limited



Network Attached Storage

- NFS is a widely deployed standard
- The Network File Server is a bottleneck
- Works well in small scale
- Today's typical scale-out solution is using a bunch of NAS islands

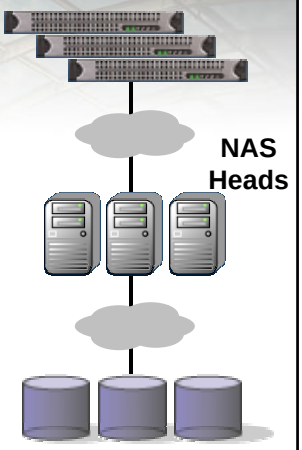


Problems with that are:

- Bandwidth and capacity for each file are still limited by the server hosting it
- Data needs to be manually moved and/or replicated to accommodate application needs
- This creates a painful storage management issue

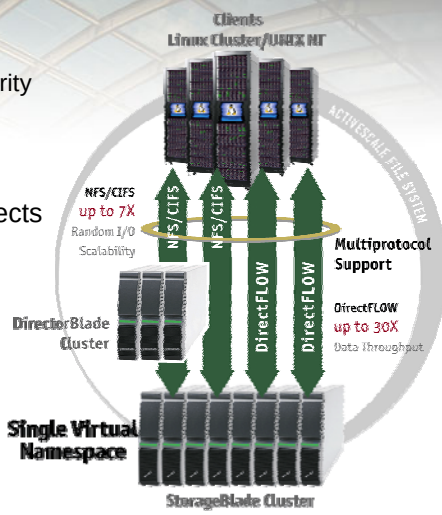
Clustered NAS

- More scalable than single-headed NAS
  - Multiple NAS heads share back-end storage
- Head-to-head synchronization limits scalability
  - Forward requests to “owner Head”
- “In-band” NAS head still limits performance and drives up cost
  - “In-band” means metadata requests use the same channel as read and write data requests
- NFS does not provide a good mechanism for dynamic load balancing
  - Clients permanently mount a particular Head



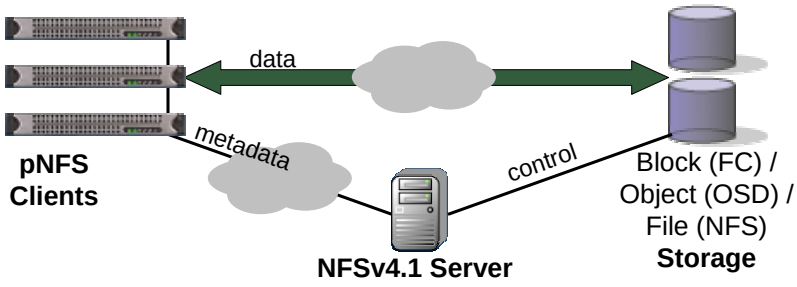
Object-based Storage Clusters

- Object Storage Devices
  - High-level interface that includes security
  - Block management inside the device
  - OSD standard (T10)
- Panasas file system layered over objects
  - National labs (also Lustre)
  - Seismic data processing
  - Digital animation
- High performance through clustering
  - Scalable to thousands of clients
  - 10 GB/sec demonstrated



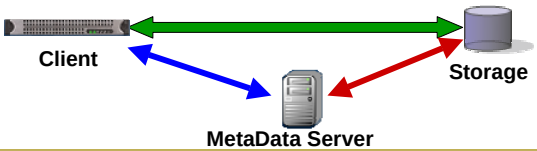
pNFS: Standard Storage Clusters

- pNFS is an extension to the Network File System v4 protocol standard
- Allows for parallel and direct access
  - From Parallel Network File System clients
  - To Storage Devices over multiple storage protocols
  - Moves the Network File System server out of the data path



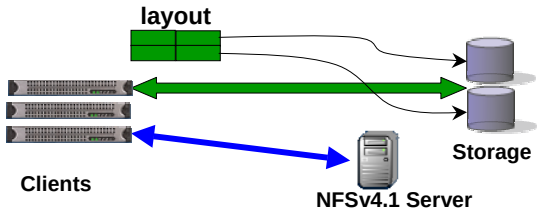
The pNFS Standard

- The **pNFS** standard defines the NFSv4.1 protocol extensions between the **server** and **client**
- The **I/O** protocol between the **client** and **storage** is specified elsewhere, for example:
  - SCSI **Block** Commands (**SBC**) over Fibre Channel (**FC**)
  - SCSI **Object**-based Storage Device (**OSD**) over iSCSI
  - Network **File** System (**NFS**)
- The **control** protocol between the **server** and **storage** devices is also specified elsewhere, for example:
  - SCSI **Object**-based Storage Device (**OSD**) over iSCSI



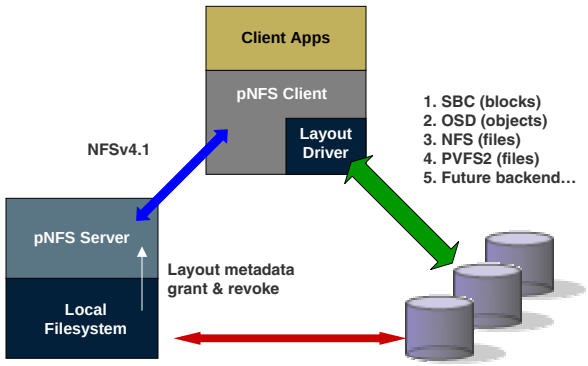
pNFS Layouts

- Client gets a *layout* from the NFS Server
- The layout maps the file onto storage devices and addresses
- The client uses the layout to perform direct I/O to storage
- At any time the server can recall the layout
- Client commits changes and returns the layout when it's done
- pNFS is optional, the client can always use regular NFSv4 I/O



pNFS Client

- Common client for different storage back ends
- Wider availability across operating systems
- Fewer support issues for storage vendors



pNFS is not…

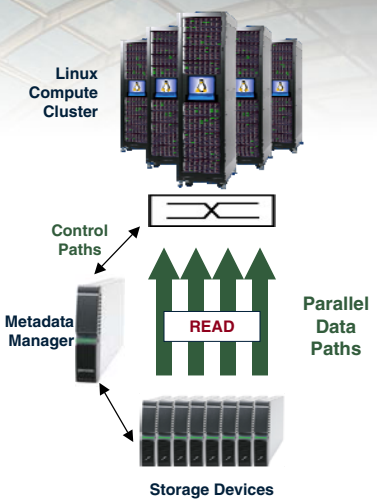
- Improved cache consistency
  - NFS has open-to-close consistency enforced by client polling of attributes
  - NFSv4.1 directory delegations can reduce polling overhead
- Perfect POSIX semantics in a distributed file system
  - NFS semantics are good enough (or, all we'll give you)
  - But note also the POSIX High End Computing Extensions Working Group
    - http://www.opengroup.org/platform/hecewg/
- Clustered metadata
  - Not a server-to-server protocol for scaling metadata
  - But, it doesn't preclude such a mechanism

pNFS Protocol Operations

- LAYOUTGET
  - (filehandle, type, byte range) -> type-specific layout
- LAYOUTRETURN
  - (filehandle, range) -> server can release state about the client
- LAYOUTCOMMIT
  - (filehandle, byte range, updated attributes, layout-specific info) -> server ensures that data is visible to other clients
  - Timestamps and end-of-file attributes are updated
- CB_LAYOUTRECALL
  - Server tells the client to stop using a layout
- CB_RECALLABLE_OBJ_AVAIL
  - Delegation available for a file that was not previously available
- GETDEVICEINFO, GETDEVICELIST
  - Map deviceID in layout to type-specific addressing information

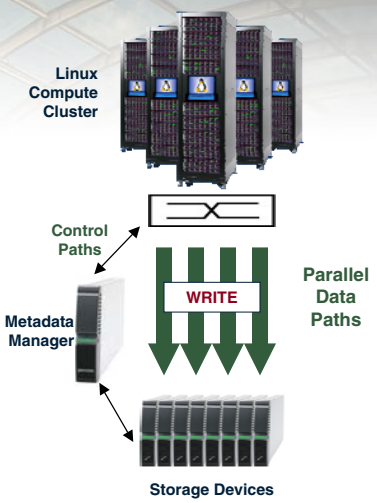
pNFS READ

- LOOKUP+OPEN client to NFS server, returns file handle and state ids
- LAYOUTGET client to NFS server, returns layout
- READ client to storage devices, many reads in parallel
- LAYOUTRETURN client to NFS server
- Clients can cache layouts for use with multiple READ and multiple LOOKUP+OPEN instances
- Server uses CB_LAYOUTRECALL when the layout is no longer valid



pNFS WRITE

- LOOKUP+OPEN client to NFS server, returns file handle and state ids
- LAYOUTGET client to NFS server, returns layout
- WRITE client to storage devices
- LAYOUTCOMMIT client to NFS server "publishes" write
- LAYOUTRETURN client to NFS server
- Server may restrict byte range of write layout to reduce allocation overheads, avoid quota limits, etc.



Example: pNFS over Blocks

- Layout describes an array of block or extents
- NFS server is responsible for block allocation
- Client uses SCSI/SBC commands to read and write data blocks
  - iSCSI or FC SAN access

Example: pNFS over Files

- Layout describes the set of file servers that store (parts of) a file
  - Layout parameters describe how data is striped over the component files
  - Simple striping is supported
- NFS server is responsible for creating and deleting component files, and establishing security and access control state on data servers
- Client uses NFS commands to read and write data (bytes)
  - Data File Servers are responsible for block management
  - Metadata File Server is responsible for attributes and access control

Example: pNFS over Objects

- Layout describes the set of component objects that store a file
  - Layout parameters describe how data is striped over these objects
  - RAID-0, RAID-1 (Mirroring), RAID-5, RAID-6 are all possibilities
  - Security credentials grant access to the client for individual objects
- NFS server is responsible for creating and deleting objects, and granting access credentials
- Client uses iSCSI/OSD commands to read and write data (bytes)
  - Object Storage Device (OSD) is responsible for block management

Is pNFS Enough?

- Standard for out-of-band metadata
  - Great start to avoid classic server bottle neck
  - NFS has already relaxed some semantics to favor performance
  - But there are certainly some workloads that will still hurt
- Standard framework for clients of different storage backends
  - Files
  - Objects
  - Blocks
  - PVFS2
  - Your project… (e.g., dcache.org)

Current Status

- pNFS is part of the IETF NFSv4 minor version 1 standard draft
  - Bi weekly conference calls to drive 4.1 draft to closure
- Reference open source client done in CITI
  - CITI owns NFSv4 Linux client and server
  - Weekly pNFS implementers' conference call
- Participants:
  - CITI (Files over PVFS2, Files over NFSv4)
  - Netapp (Files over NFSv4)
  - IBM (Files, based GPFS)
  - EMC (Blocks, based on HighRoad)
  - Sun (Files over NFSv4, Objects based on OSDv1)
  - Panasas (Objects, based on Panasas ActiveScale Storage Cluster OSDs)
  - Carnegie Mellon University, performance and correctness testing

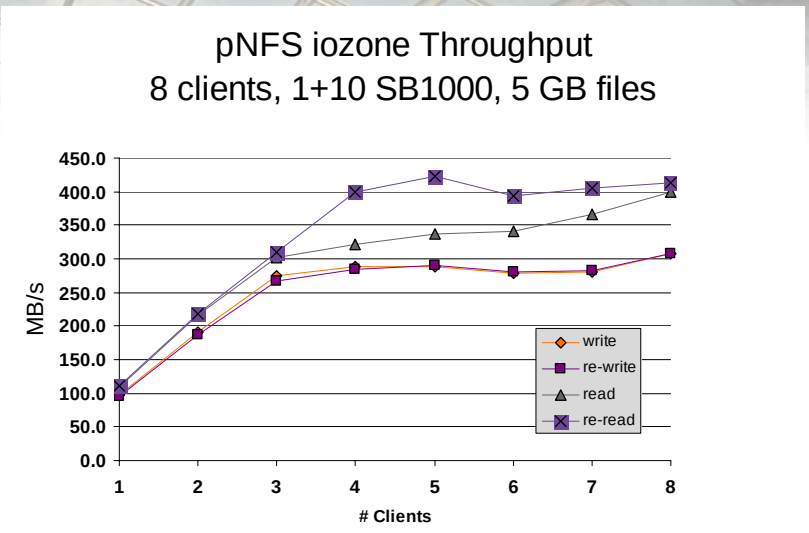
More Status

- Working out code structure for internal Linux APIs
  - Client: Merged PVFS2 prototype and NetApp NFSv4 files client
  - Client: APIs for generic layout cache
  - Server: standard API to export file system layouts
  - No recalls, yet (i.e., callbacks)
- Prototype interoperability began in 2006
  - San Jose Connect-a-thon March '06
  - Ann Arbor NFS Bake-a-thon September '06
- Availability
  - TBD – gated behind NFSv4 and, uh, working implementations of pNFS

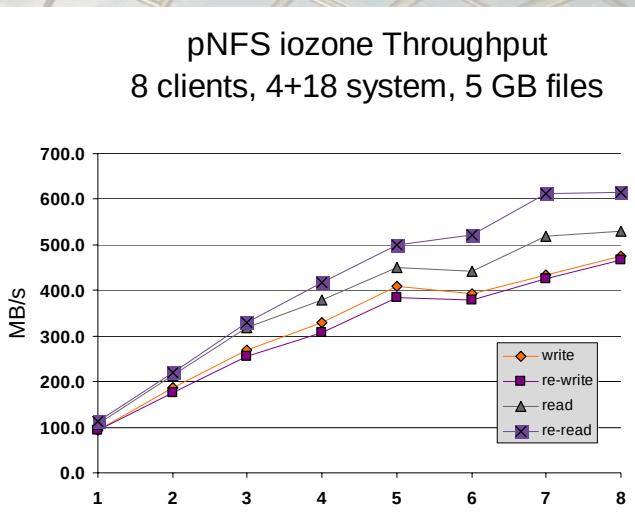
Notes from NFS Bake-a-thon

- NFS interoperability testing at UMich / CITI Sept 11-15
  - Test NFSv4.1 features, including pNFS
  - pNFS testing
    - CITI (linux client/server, files over PVFS2)
    - NetApp server (files)
    - IBM (linux server of files over GPFS)
    - Sun (solaris client/server files)
    - DESY (Java server exporting dcache)
    - Panasas (linux client/server of objects over Panasas)
  - Everyone is at “early prototype” state
    - It was possible to copy a small file between servers using a common client

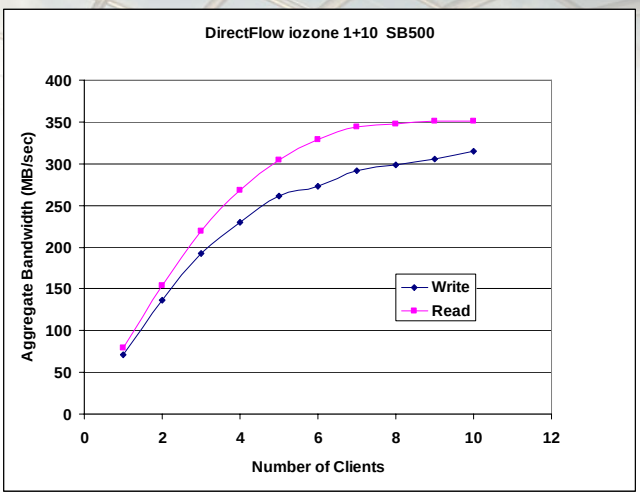
Prototype pNFS Performance



Prototype pNFS Performance



Panasas DirectFlow Protocol Performance



References

- "NFS Version 4 Minor Version 1"
  draft-ietf-nfsv4-minorversion1-08.txt
  draft-ietf-nfsv4-pnfs-obj-02.txt
- "pNFS Problem Statement"
  Garth Gibson (Panasas), Peter Corbett (Netapp), Internet-draft, July 2004,
  http://www.pdl.cmu.edu/pNFS/archive/gibson-pnfs-problem-statement.html
- "NFSv4 pNFS Extensions"
  G. Goodson (Netapp), B. Welch, B. Halevy (Panasas), D. Black (EMC), A. Adamson (CITI), Internet-draft, October 2005,
  http://www.ietf.org/internet-drafts/draft-ietf-nfsv4-pnfs-00.txt
- "Linux pNFS Kernel Development"
  CITI,
  http://www.citi.umich.edu/projects/asci/pnfs/linux/

Thank You

For more information,
call Panasas at:

1-888-PANASAS
(US & Canada)

00 (800) PANASAS2
(UK & France)

00 (800) 787-702
(Italy)

+001 (510) 608-7790
(All Other Countries)

www.panasas.com