

NFS Version 4.1 and Beyond

Mike Eisler, NetApp

Spencer Shepler, Storspeed

Dave Noveck, NetApp

- ❑ Comparison of NFSv3 and NFSv4.0
- ❑ NFSv4.1 Fixes and Improvements
 - ❑ ACLs
 - ❑ Delegation Management
 - ❑ Opens
 - ❑ Asynchronous Blocking Locks
 - ❑ Callbacks
- ❑ NFSv4.1 New Features
 - ❑ Sessions
 - ❑ Trunking
 - ❑ pNFS
 - ❑ Directory Delegations and Notifications
 - ❑ Non-regular file delegations
 - ❑ Global Namespace
- ❑ Status
- ❑ After NFSv4.1
- ❑ Links

Comparison of NFSv3 and NFSv4

NFSv3

- ❑ A collection of protocols (file access, mount, lock, status)
- ❑ Stateless
- ❑ UNIX-centric, but seen in Windows too
- ❑ Deployed with weak authentication
- ❑ 32 bit numeric uids/gids
- ❑ Ad-hoc caching
- ❑ UNIX permissions
- ❑ Works over UDP, TCP
- ❑ Needs a-priori agreement on character sets

NFSv4

- ❑ One protocol to a single port (2049)
- ❑ Lease-based state
- ❑ Supports UNIX and Windows file semantics
- ❑ Mandates strong authentication
- ❑ String-based
- ❑ Formal handshake
- ❑ Windows-like access
- ❑ Bans UDP
- ❑ Uses universal character set for file names

Access Control Lists (ACLs)

- ❑ NFSv4.0 ACLs are derived from Windows 2000
- ❑ NFSv4.1 ACLs add better (read: Windows-like) support for ACL inheritance
- ❑ Like Windows, NFSv4.1 ACLs now explicitly separate “discretionary” ACL (dACL) on a file from its “security” ACL (sACL)
 - ❑ dACL: ALLOW/DENY Access Control Entries (ACEs)
 - ❑ sACL: AUDIT/ALARM ACEs

Delegation Management

- ❑ Delegation Re-Acquisition
 - ❑ NFSv4.0 tied delegations to OPEN
 - ❑ Only by polling via another OPEN could a client re-acquire delegation
 - ❑ NFSv4.1 adds a separate WANT_DELEGATION operation to allow a delegation to be “pushed” to a client via a callback
- ❑ Delegation recall
 - ❑ Often a server wants to recall delegations due to general resource constraints
 - ❑ Server has no idea which delegations to recall
 - ❑ NFSv4.1 add a callback that asks the client to retain up to N delegations and return the rest

- ❑ Parallel Opens
 - ❑ NFSv4.0 requires a given “open owner” to serialize all OPENS on the same file
 - ❑ NFSv4.1 allows parallel OPEN, and includes a generation number to indicate which OPEN was executed last
- ❑ Open By Filehandle
 - ❑ NFSv4.0: Except for recovery, OPENS done by name only
 - ❑ NFSv4.1: Steady State OPENS can use filehandles
 - ❑ Enables proxies (e.g. cache appliances), thinner clients

Asynchronous Blocking Locks

- ❑ NFSv4.0 requires client to poll for a blocking byte range lock
- ❑ NFSv4.1 adds a notification callback to indicate when a lock is available

Callbacks

- ❑ NFSv4.0 requires callback to recall delegations
 - ❑ A separate connection path was used
 - ❑ Not firewall friendly
- ❑ NFSv4.1 requires clients to create connections and hand them to servers to be used for callbacks

- ❑ NFSv[2,3,4.0] do not support exactly once semantics
 - ❑ Approximated with the reply cache that has indefinite size
- ❑ Approximation drawbacks:
 - ❑ Unbounded size makes persistence across failover and reboot impractical
 - ❑ Prone to cache misses
 - ❑ Prone to false positives
 - ❑ Bad clients
 - ❑ Reply caches that are too big

Sessions (continued)

- ❑ NFSv4.1 achieves exactly once semantics via a “slot table”
- ❑ Slot table created when session created
- ❑ Specific number of slots
- ❑ NFSv4.0 requests consist of a list of operations in the COMPOUND request
 - ❑ NFSv4.1 mandates each COMPOUND start with SEQUENCE operation that carries a slot number and a sequence number (per slot)
- ❑ Each slot corresponds to a request in progress or a completed request
- ❑ Each request is either a retry of the last executed request, or a new request with a sequence number exactly one higher than previously executed
 - ❑ This series OK:
 - ❑ SEQUENCE slot 1, seq #2, SEQUENCE slot 1, seq #2, SEQUENCE slot 1, seq #3
 - ❑ This series NOT OK:
 - ❑ SEQUENCE slot 1, seq #2, SEQUENCE slot 1, seq #4, SEQUENCE slot 1, seq #5

Sessions – Slot Table Example

slot number	0	1	2	...	29	30	31
sequence number	100	200	150	...	25	125	175
cached reply	SEQ., PUTFH, WRITE	SEQ., PUTFH, RENAME	SEQ., PUTFH, OPEN, GETFH, GETATTR	...	SEQ., PUTFH, REMOVE	SEQ., PUTFH, READ	SEQ., PUTFH, LOOKUP

- ❑ CREATE_SESSION
 - ❑ Negotiates the size of the slot table
 - ❑ Negotiates persistence of the slot table, enabling true exactly once semantics (EOS) through server reboot
- ❑ DESTROY_SESSION
 - ❑ Client can indicate explicitly when it non longer needs a session (e.g. when unmounting)
- ❑ Thus reply cache size and lifetime are bounded

- ❑ BIND_CONN_TO_SESSION
 - ❑ Allows client to bind additional connections to the session
 - ❑ New connections may be to alternate network interfaces of a multi-homed server
- ❑ Trunking over multiple sessions also permitted
 - ❑ Enables the use of storage clusters where multiple server nodes have paths or replicas to the same data

Parallel NFS (pNFS)

- ❑ pNFS allows servers to stripe data of regular files across multiple storage devices
- ❑ A pNFS server consists of:
 - ❑ A metadata server (MDS) that implements the full NFSv4.1 protocol
 - ❑ One or more storage devices
- ❑ A pNFS client is an NFSv4.1 client that is prepared to directly access storage devices
- ❑ The pNFS client finds out about storage devices from the MDS via a new LAYOUTGET operation

- ❑ LAYOUTGET returns a layout that describes the striping pattern for a given file
- ❑ Layouts are recallable which allows pNFS servers to re-stripe a file if desired or necessary
- ❑ Striping patterns can indicate if some or all of a pattern has mirrors
 - ❑ Clients are not required to construct mirrors
 - ❑ The pNFS offers RAID 0 and RAID 1+0

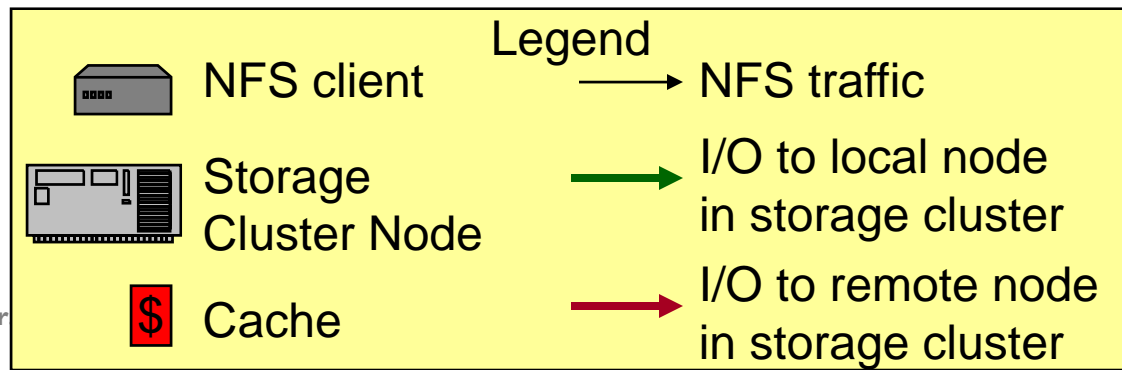
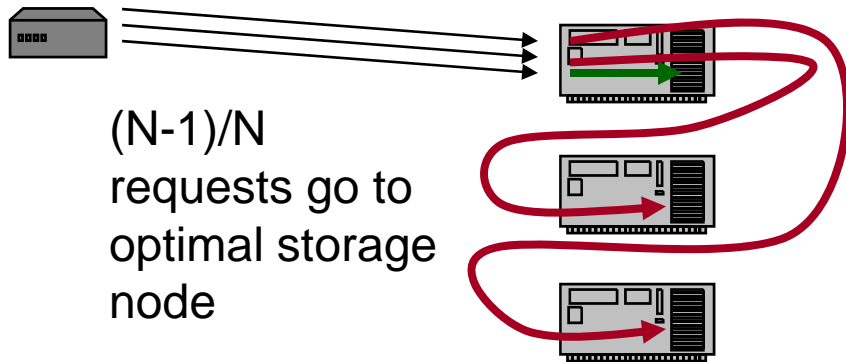
pNFS: Types of Storage Devices

- ❑ pNFS supports multiple Storage Device types (aka layout types)
- ❑ A layout can stripe a file over just one type of device
- ❑ The NFSv4 working group currently specifies three types:
 - ❑ Files
 - ❑ The storage “device” is an NFSv4.1 server
 - ❑ Blocks
 - ❑ The storage device is an iSCSI or FC target
 - ❑ Objects
 - ❑ The storage device is an Object Storage Device (OSD) target
- ❑ Additional types require a standards-track specification
- ❑ If a client does not support a given device type it can issue I/O directly to the MDS

- The file layout can work with or without a backing clustered file system
 - pNFS over a clustered file system is more optimal than conventional NFS access to a clustered file system because the client is directed to the optimal node
- The file layout uses the same security model (authentication, authorization, access control) as NFSv4.1

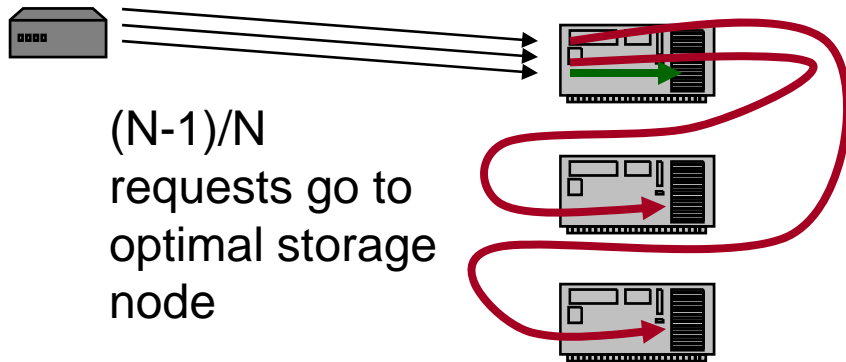
pNFS Optimizes Storage Clusters

“Share Nothing” Architecture w/out pNFS

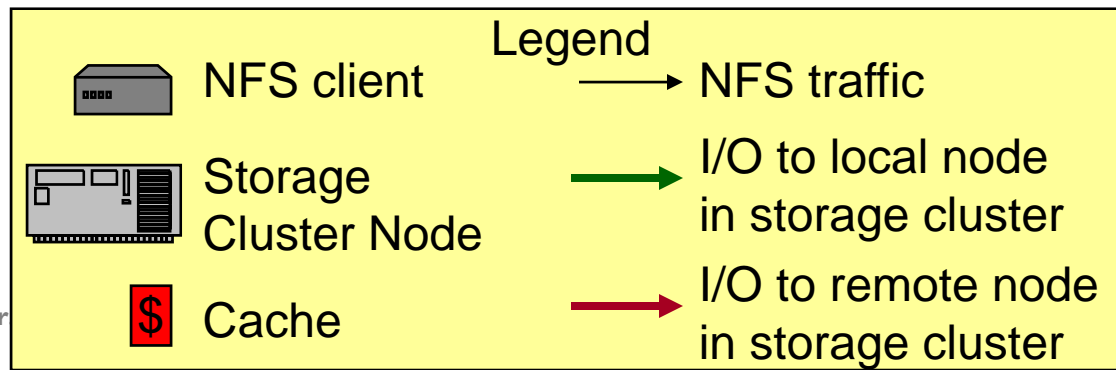
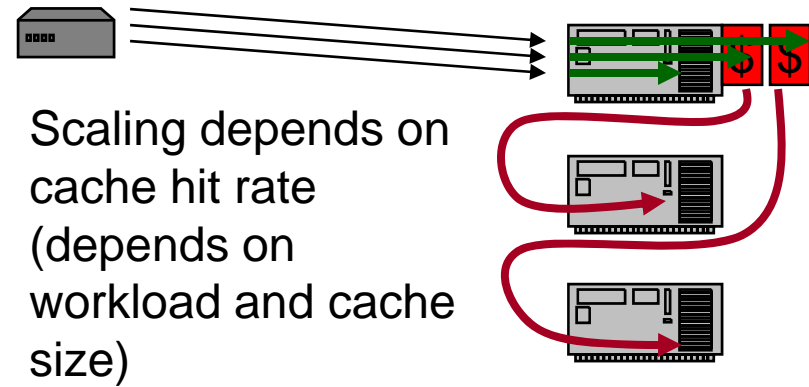


pNFS Optimizes Storage Clusters

“Share Nothing” Architecture w/out pNFS

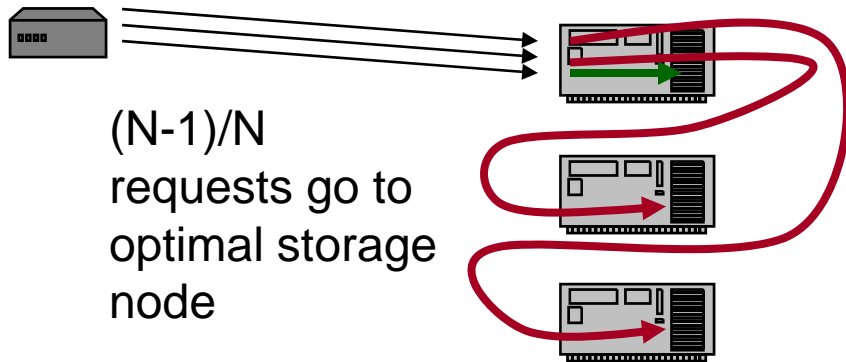


Cache Coherent Architecture w/out pNFS

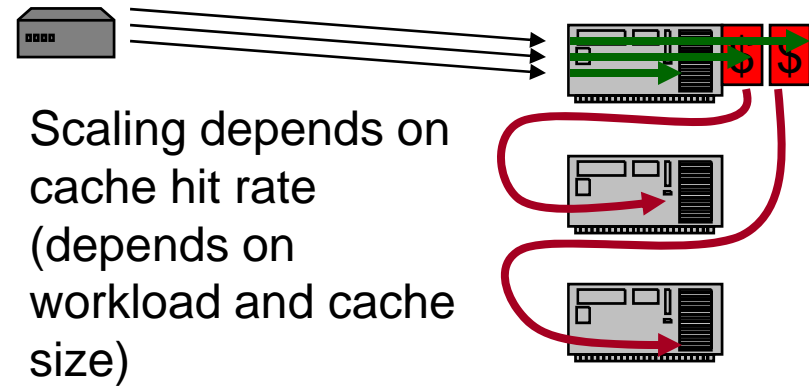


pNFS Optimizes Storage Clusters

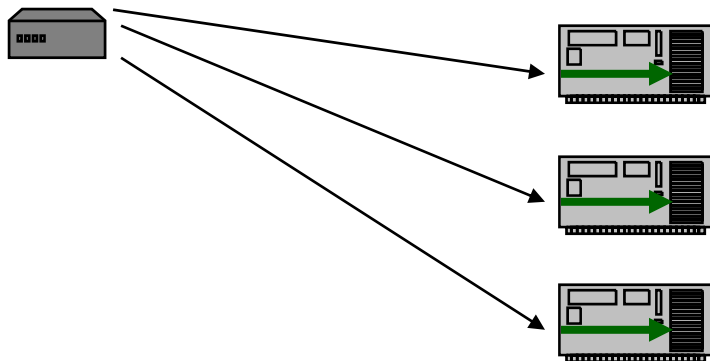
“Share Nothing” Architecture w/out pNFS



Cache Coherent Architecture w/out pNFS



With pNFS



In both architectures, I/O requests go to the optimal storage node.

- ❑ Intended to be compatible with SNIA XAM API
- ❑ Three new attributes:
 - ❑ *Retention*: if set, forces a file to be retained for a specified period of time
 - ❑ *Retention_event*: like retention for event-based retention
 - ❑ *Retention_hold*: bit mask of up to 64 administrative holds
 - ❑ If any bit set, file is retained regardless of state of *retention*, *retention_event*

Directory Delegations and Notifications

- ❑ Enabled by new GET_DIR_DELEGATION operation
- ❑ Directory Delegations are read-only in NFSv4.1
 - ❑ Optimal for workloads where directories are rarely updated
 - ❑ Write delegations for directories are difficult because creating entries requires a directory entry offset and a file inode number which only the server can produce
- ❑ Directory notifications allow client to be notified of updates to directories
 - ❑ Similar to CIFS change/notify
 - ❑ Optimal for workloads where directories are updated more frequently
 - ❑ Allows server to push changes to name caches versus existing poll model
 - ❑ Notifications are asynchronous
 - ❑ Designers did not believe synchronous notification would scale
 - ❑ Asynchronous model no worse than existing NFS directory caching implementations

Non-regular file delegations

- ❑ The new WANT_DELEGATION operation works on all types of files except directories
- ❑ Allows one to cache contents of symbolic links
 - ❑ Symbolic links are read-only content so delegations are very apropos

Global Namespace

- ❑ NFSv4.0 has a “referral” feature that allows a server to re-direct a client to another NFSv4.0 server
- ❑ NFSv4.1 builds on referrals to produce a more complete definition of multi-server global namespace
 - ❑ Defines lock and session state transitions
 - ❑ Indicates whether key attributes like inode number survive migration events

Standards Status

- ❑ NFSv4.I Internet Drafts have been ratified for publication as RFCs
- ❑ RFC publication was blocked up by indirect dependency (NFS/RDMA → Connection Latching)
- ❑ Using “Extra Time” to fix typos and clarify text
- ❑ Expect RFC number by 2009, RFC by 2010

Industry Status

- ❑ NetApp, EMC, IBM, LSI, Microsoft, NetApp, RedHat, Sun have announced plans to provide products
- ❑ SNIA Special Interest Group Formed

- ❑ NFS SIG drives adoption and understanding of pNFS across vendors to constituents
 - ❑ Marketing, industry adoption, Open Source updates
- ❑ NetApp, EMC, Panasas and Sun founders
 - ❑ NetApp and Panasas act as co-chairs
- ❑ Deliver Panels/Sessions on NFSv4.1 when possible
 - ❑ E.g. SNW Europe in October
- ❑ Developing (CY09) pNFS I0I document
 - ❑ Scale-out paradigm Enterprise and HPC
- ❑ www.snia.org/forums/esf/programs/about/nfs_sig

- NFSv4 WG has been re-chartered
- Confirmed Items on New Charter
 - Federated File System
 - NFSv4.0 BIS
- Proposed Items include:
 - Metadata Striping
 - Mandatory Access Control and Labeling
 - Client Aware De-duplication
 - End-to-End Data Integrity
 - Sub-file caching
 - Mode bit mask for file create
 - Sequential I/O hints
 - RPC over IPv6
 - Peer-to-peer pNFS
 - Clarification of Multi-Domain Access

- ❑ NFSv4 working group:
 - ❑ www.ietf.org/html.charters/nfsv4-charter.html
- ❑ SuperComputing 2008 pNFS BOF presentation:
 - ❑ http://pnfs.com/docs/sc08_pnfs_bof_slides.pdf
- ❑ Current Internet Draft
 - ❑ tools.ietf.org/wg/nfsv4/draft-ietf-nfsv4-minorversion1
 - ❑ blogs.netapp.com/eislers_nfs_blog
 - ❑ shepler.blogspot.com