pNFS Storage Preference

IETF 78 NFSv4 WG Meeting

July 28, 2010

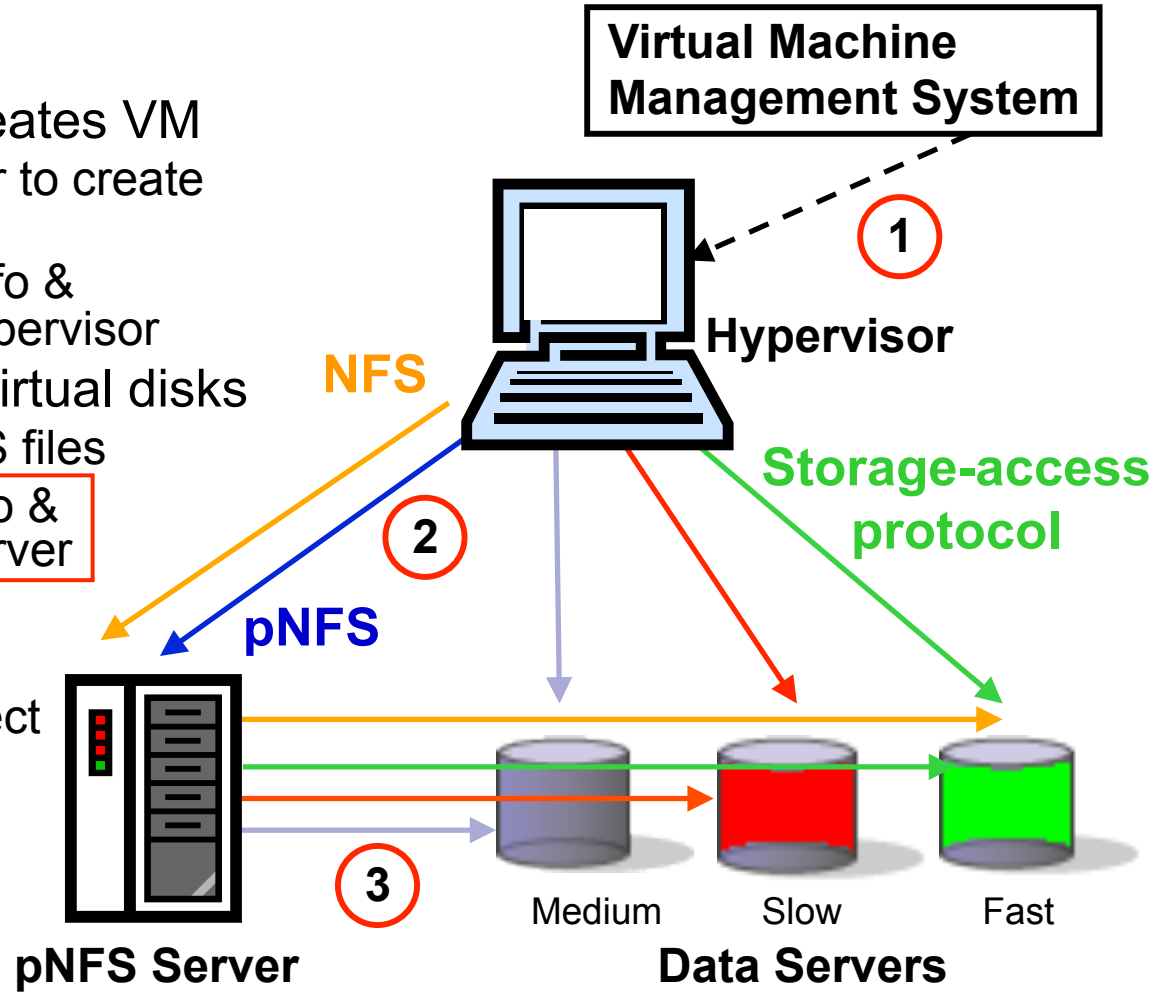
David Black – EMC (david.black@emc.com)
Peng Dai – VMware
Mike Eisler – NetApp (mike@eisler.com)
Sorin Faibish – EMC
Christos Karamanolis – VMware

pNFS and file placement

- Before: choose directory and hope for the best
  - File pathname implies storage placement & details
- After: pNFS layout separates file name & location
  - pNFS data server(s) do not depend on file pathname
- Opportunity: client input to help place new files
  - Server uses info to improve storage placement for file
  - Hypervisor virtual disk files (for new VMs) are a good example
- Goal: Describe file usage and requirements
  - Avoid specifying actual placement (e.g., no data server names!)
- Functionality can be used with pNFS **and** without pNFS

Example: Virtual Disk Creation

1. VM Management creates VM
   - Request hypervisor to create virtual disks
   - Pass virtual disk info & requirements to hypervisor
2. Hypervisor creates virtual disks
   - ⇛ Request new pNFS files
   - Send file usage info & requirements to server
3. Server creates files
   - Uses usage info & reqts. to better select data server[s]

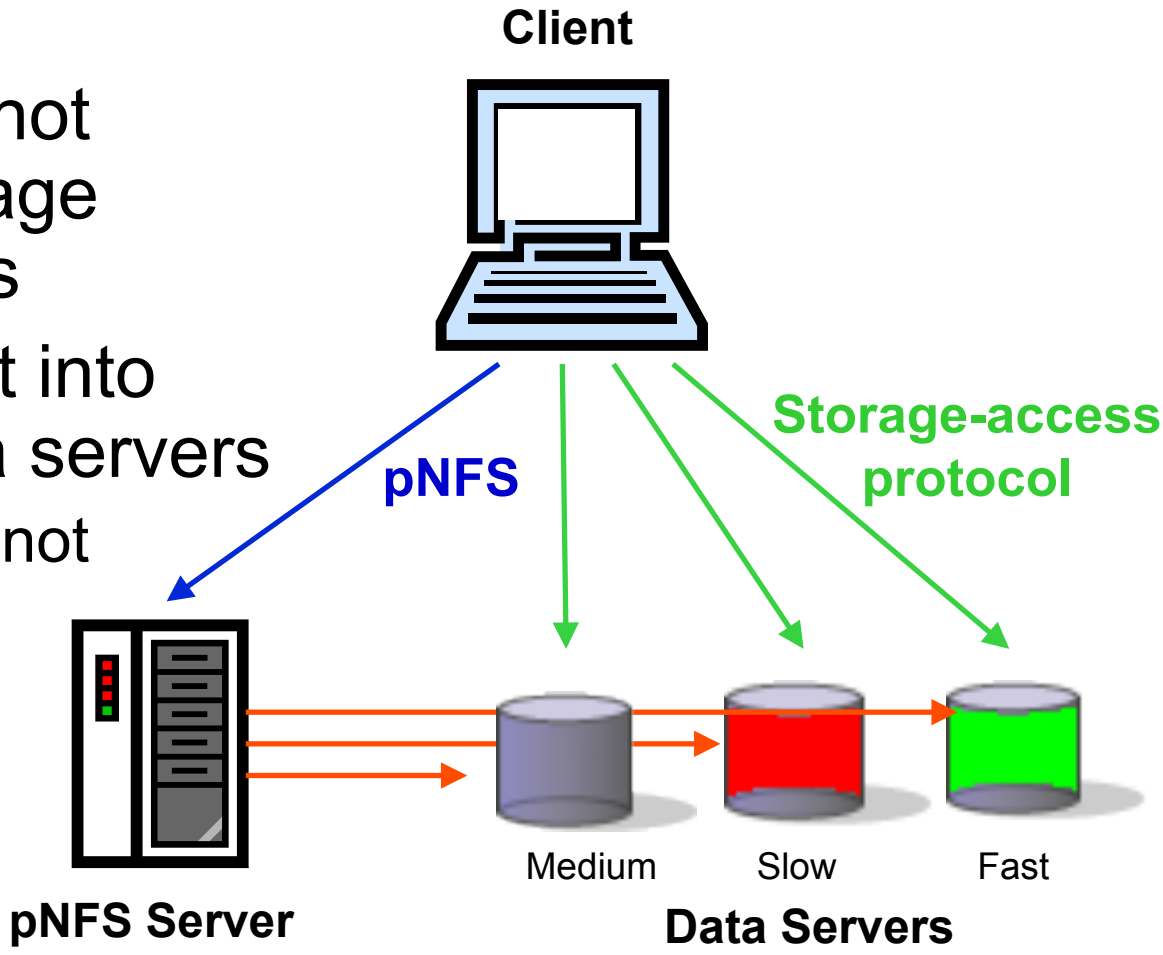


Storage Preference Examples

- Data Availability (RAID/mirrored: yes)
- Hardware Redundancy (single point of failure: yes/no)
- Storage tier
  - String-encoded or enumerated values (e.g., “orange”, “green”)
  - Relationships among values are crucial
- Expected typical case
  - A few <property, value> pairs for new file
- CAVEAT: These are POSSIBLE examples
  - We reserve the right to change our opinion of them ☺ ☺ ☺

Current Situation

- pNFS clients not aware of storage characteristics
- No client input into choice of data servers
  - layout hint is not sufficient
  - FedFS is not per-file



Storage Preference Goals

- Storage placement independent of filename and pathname
- Granularity: Individual (large) files or groups of files
- Single preference property namespace, with versioned properties
  - Basic set of r/w standard properties
    - Read: check current values, Write: set values
  - Extensible to add new properties and versions
    - Allow experimental & private use properties
- Property types (per property instance, not pre-defined)
  - Mandatory: Don't create file if <this> can't be done
    - Not allowed for private use properties (IETF policy)
  - Advisory: Try to do <this>
  - Informational: Remember <this> information
- Client can determine which properties are supported by server
  - Don't make client guess

Storage Preference Non-goals

- Advertise/expose storage capabilities of any type
  - Out-of-band interface preferable for capabilities
  - Setting new file properties out-of-band can be problematic
- Explicit storage selection (e.g., put this file <there>)
- Server implementation details
  - Storage preference properties should specify “what”, not “how”

Storage Preference Approach

- New NFS operation
  - Provide usage info, requirements
  - Same COMPOUND as file or directory creation
    - Avoid changing/adding creation operation
  - Set a new attribute of some form (or attributes?)
- Start with directory as grouping scope
  - Files default to attribute on directory (if available)
  - No directory to directory inheritance

Solution Details

- Extensible information approach
  - XDR or XML or ???
  - Expect site-specific usage (e.g., “My Purple Storage”)
- Lots of details to work through
  - Interaction w/other features at client and server
  - Error handling
  - Group files by means other than directory?
- Need interoperable baseline
  - Examples: Remember slide 4?
  - Here it is again ...

Storage Preference Examples

- Storage tier
  - String-encoded or enumerated values (e.g., “orange”, “green”)
  - Relationships among values are crucial
- Data Availability (RAID/mirrored: yes)
- Hardware Redundancy (single point of failure: yes/no)

- Expected typical case
  - A few <property, value> pairs for new file

- CAVEAT: These are POSSIBLE examples
  - We reserve the right to change our opinion of them ☺ ☺ ☺

Conclusion

- Client input into file placement on storage
  - Info on file flows: Client ➡ Server
- Requests:
  - Now: Discussion on what to do and how
  - Soon: Review and comment on -00 draft

Thank You

pNFS layout hint: Not suitable for this purpose

- hint is pNFS layout-type-specific
  - Typically used for striping
- “hint” not enough to impose requirements
- layout hint is pNFS-only
  - Want to supply client info for plain NFS
- Prefer layout-type-independent structure