Data Reduction xattr in nfsv4

draft-faibish-data-reduction-xattrs-pvt-00

Sorin Faibish <faibish.sorin@dell.com>

David Black <david.black@dell.com>

Philip Shilane <philip.shilane@dell.com>

Motivation: making the case for DR xattr

- New storage arrays use expensive flash devices
  - Size of data sets is growing exponentially
  - But size of backend flash storage grows linearly at higher price per GB
  - Storage servers compute power increases with use of much larger number of cores
  - Memory of the servers also increases using NVMe devices based memory (Intel Apache Path)
  - New faster NVMe over fabric interconnect is available (Intel)

Motivation: making the case for DR xattr

- How we address this problem
  - New data reduction algorithms for deduplication and compression improve DR
  - Variable block deduplication improves 2-5x size of data on disk versus fixed block
  - New compression HW using MS zipline methods and/or Intel QAT chips are becoming the norm
  - New DR require larger memories and core numbers; new servers have both
  - What's missing: user information related to DR that arrays cannot know
  - The draft offers a way of transmitting DR attributes from NFS client to NFS server

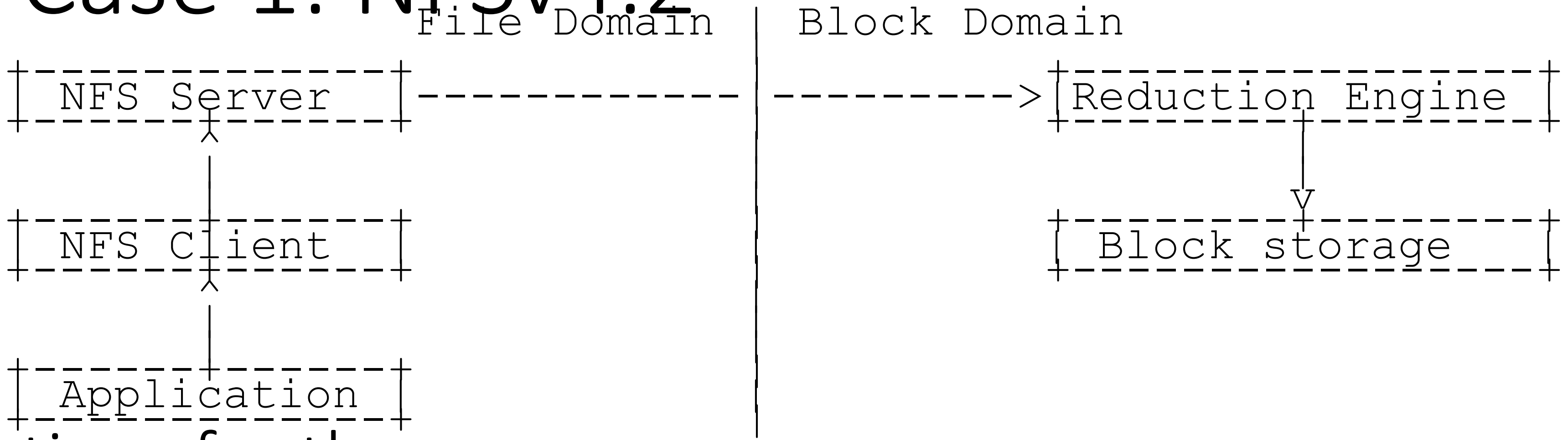
Solving the Problem

- Currently NFSv4 has no means to communicate DR attributes to NFSv4 server
- NFSv4 server DR engine operates at FS block size typically 8k
- There is analysts data regarding compression and dedupe of different types of files that can improve DR engines in the array efficiency
- There is no way to take advantage of this data as the application DR characteristics are not visible to the DR engine
- There is an unfinished draft of Christophe extending pNFS SCSI to NVMe protocol that can be used; we propose to finish that draft and add DR xattr
- We propose to expand the new xattr to apply to pNFS SCSI

What is needed from NFSv4

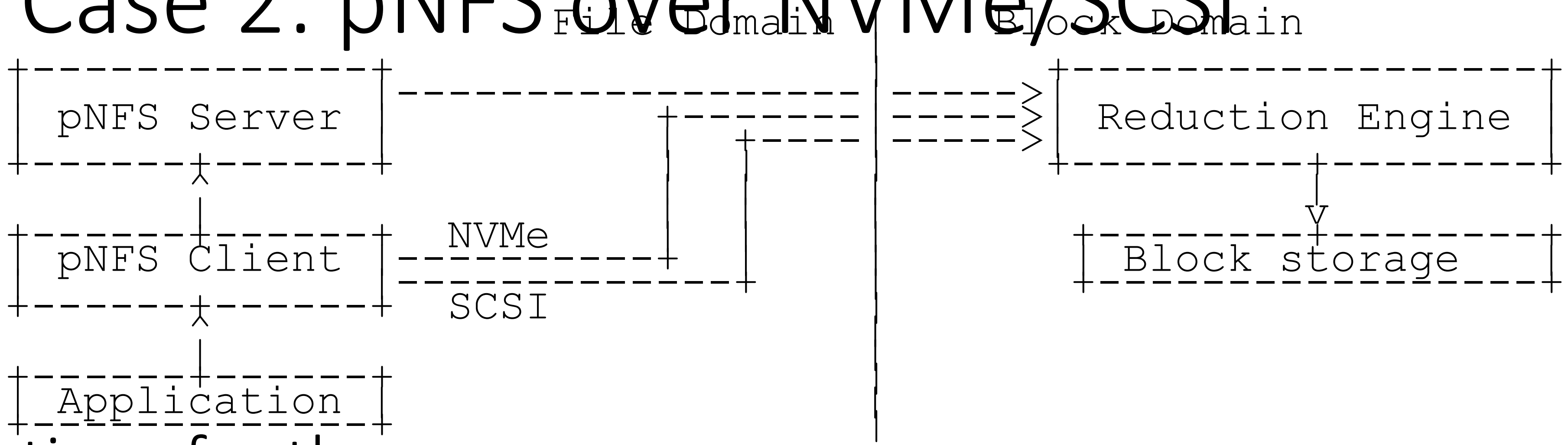
- We need a way to communicate DR characteristics from client to server
- We propose to add a new type of xattr to send compression and dedupe ratios from user/client to the NFSv4 server
- We do not propose any implementation of new algorithms; we leave this to the NFSv4 vendor how to do compression and deduplication
- We just allow better DR when the DR engine knows what to target
- We assume that:
  - File extensions can indicate file type=> DR characteristics => xattr
  - File header can indicate compression type (MPEG) => DR characteristics =>xattr
  - Prevent applying DR on uncompressible data using file attributes => can use a flag

Use Case 1: NFSv4.2



- Assumptions for the use case:
  - Backend DR engine can access the XATTR of the file
  - Backend identifies FS blocks of a “file” and can associate DR data to each block

Use Case 2: pNFS over NVMe/SCSI



- Assumptions for the use case:
  - DR xattr are communicated to Block storage DR engine by the pNFS server
  - Block storage is informed by NFSv4 layer of DR type for each block of a “file”

Typical DRR for different applications

- Different applications have known DR/CR:

EDA	DR/CR=50%/50%
SWBUILD	DR/CR=0/80%
VDI	DR/CR=55%/70%
DB	DR/CR=0/50%
VDA	DR/CR=0/0
IT infrastucture	DR/CR=30%/50%
Oracle DW	DR/CR=15%/70%
Oracle OLTP	DR/CR=0%/65%
Exchange 2010	DR/CR=15%/35%
Geoseismic	DR/CR=3%/40%

xattr versus file attributes

- The proposed data reduction xattr are opaque to the file system
- xattr are file system-agnostic
- xattr can associate opaque metadata with file system objects
- xattr are rich in space not only Boolean values: 0/1
- xattr can be retrieved from the local file systems on the client
- Many Linux and Windows hosts support extended attributes
- There are no clear indications on how xattrs can be mapped to existing file attributes

Protocol Enhancements

- We proposes extensions to the NFSv4 protocol operations to allow data reduction xattrs to be queried and modified by NFSv4/pNFS clients.
- Add new attribute bitmap4 data type to allow xattr support to be queried
- Add 4 new operations, namely:
  - GETDRATTR,
  - SETDRATTR,
  - LISTXATTR and
  - REMOVEXATTR

Asks from NFSv4 WG

- Should DR xattr be added to the NFSv4 protocol?
- Should this become a WG item?
- Should we first define the protocol details before adoption?
- Is the WG interested in this draft at all?
- Is the WG interested to revive the pNFS over NVMe draft and make it a WG item?
- We want to ask for WG review of the draft
- Next steps?

Future work

- We intend to write a new draft to allow hash keys exchange between NFSv4 server and client
- Main target to improve DR efficiency for cloud storage
- Is there any interest in the WG for this new protocol?