Dean Hildebrand
dhildeb@us.ibm.com

IBM

Sparse and Direct I/O Support

Dean Hildebrand – IBM Almaden

Sparse File Support

- Sparse file are common way to represent huge files
  - Database files
  - HPC applications
  - Virtual machine images
- Problem
  - Application are not aware of file organization
  - Read and Prefetch holes
  - Simple change to NFS protocol
- IETF77 - “Why haven't we done this already?”

Source: If applicable, describe source origin

Sparse File Support Protocol Addition

```
Operation 25: READ - Read from File
RESULT
   struct READ4resok {
           bool            eof;
           opaque          data<>;
   };
+  struct READ4reshole {
+          offset4         data_offset;
+          count4          data_count;
+  };

   union READ4res switch (nfsstat4 status) {
    case NFS4_OK:
            READ4resok     resok4;
+   case NFS4ERR_HOLE:
+           READ4reshole   reshole4;
    default:
            void;
   };
```

data_offset
- offset of next region of allocated data in file

data_length
- length of the non-zero data segment at data_offset
- If data_length is greater than 0, then the data in the file from data_offset until data_length is allocated and does not contain a hole
- If data_length is set to zero, then either the server has no further information regarding holes in the remainder of the file or it can be assumed that all remaining bytes in the file are allocated and contain no holes. Either way, the client can ignore the information in READ4reshole.

Direct I/O Support

- Direct I/O common way of avoiding overhead of client data caching
- Many applications benefit from not caching data
  - HPC, DB, and virtual machines5
  - Random READ workloads
  - Write once, Read never
  - Sub-block data access
    - And block sizes could be growing with steaming pNFS data access!!
- With NFS
  - O_DIRECT flag affects only client
  - Server file system continues to cache all data
  - Can reduce I/O performance and/or pollute server cache
- “If client caching is ineffective, so is server caching”

Direct I/O Support

- No method to disable all file system data caching
- POSIX limits flags that can be passed at file open/read/write

- Possible Solution:
  - At Open(), pass existing POSIX O_DIRECT flag to NFS server
    - NFS server passes flag to underlying file system at file open
    - Server file system uses flag as a 'hint' to optimize data caching
      - May want to still use some form of data caching, e.g., on sequential read pattern, use temporary readahead cache

NFSv4.1 RFC Change

```
 struct OPEN4args {
           seqid4          seqid;
           uint32_t        share_access;
           uint32_t        share_deny;
           open_owner4     owner;
           openflag4       openhow;
           open_claim4     claim;
+          bool            direct;
 };
```