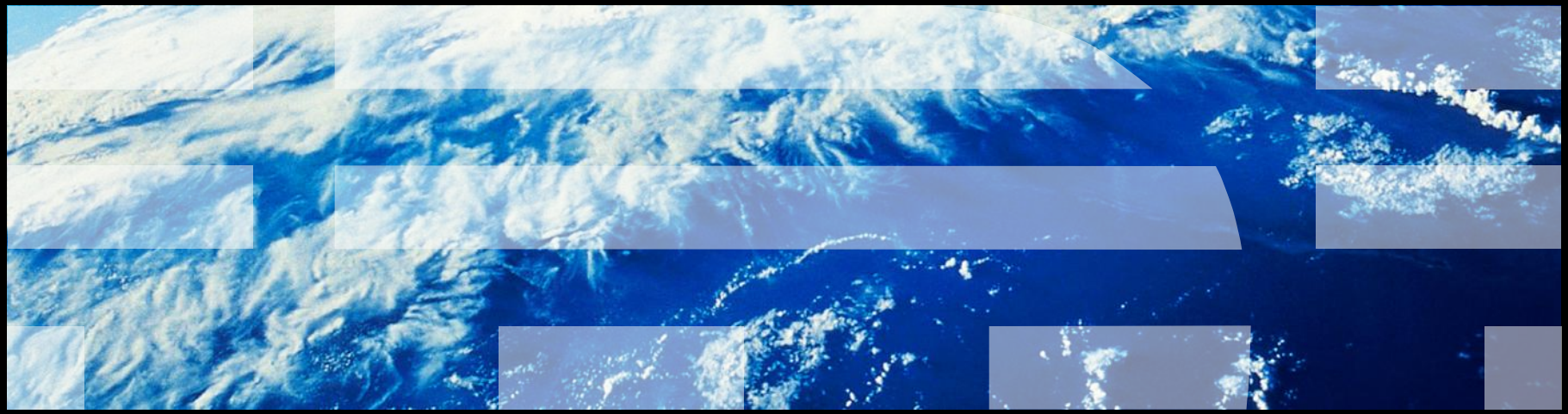
Dean Hildebrand
dhildeb@us.ibm.com

Fadvise Support for NFSv4.2

Dean Hildebrand – IBM Almaden
Trond Myklebust – Netapp
Sam Falkner – Oracle



Sending Application I/O Hints to Server

- POSIX requirements
  - POSIX supports operation for applications to inform file system of expected I/O workload
  - Per-file descriptor
  - Specified range need not exist
  - If length is 0, all data post offset is specified

```
int posix_fadvise(int fd, off_t offset, off_t len, int advice);

POSIX_FADV_NORMAL     Specifies that the application has no advice to give on its
                      behavior with respect to the specified data. It is the default
                      characteristic if no advice is given for an open file.

POSIX_FADV_SEQUENTIAL Specifies that the application expects to access the specified
                      data sequentially from lower offsets to higher offsets.

POSIX_FADV_RANDOM     Specifies that the application expects to access the specified
                      data in a random order.

POSIX_FADV_WILLNEED   Specifies that the application expects to access the specified
                      data in the near future.

POSIX_FADV_DONTNEED   Specifies that the application expects that it will not access the
                      specified data in the near future.

POSIX_FADV_NOREUSE    Specifies that the application expects to access the specified
                      data once and then not reuse it thereafter.
```

Sending Application I/O Hints to Server

- Other possible use cases and requirements
  - O_DIRECT at file open time
    - Use FADVISE_NOREUSE flag on entire file
  - Additional useful I/O hints for your use case??
    - Enterprise Apps…

Hint Usage and Obligations

- Server has no obligation to carry out hint
- Server should never return an error unless the request is malformed
- Would be nice to have server inform client of what it intends to do
    - Just for debugging purposes
    - Client MAY change behavior based upon result from server (but I wouldn't)
        - Need a good example here
- If allow multiple hints per request, need to define 'acceptable' combinations

OPERATION FADVISE: Details

- Might want to change name to IO_ADVISE
- Client and server can send/return multiple hints per operation (bitmap4)
- Per open file instance
  - Must be preceeded with current FH
  - Includes stateid
- Specified range need not exist in file
- If length is 0, all data post offset is specified
- Server return success if the operation is properly formed, error otherwise
  - May also return hint it plans to implement
    - Can be same or different
    - Client SHOULD NOT change or modify its file access behavior
  - MUST NOT return error if it does not recognize or does not support hint
- Server MAY ignore/implement the advice
- pNFS***
  - FADVISE only sent to metadata server (may be inefficient for some implementations)
  - MDS must communicate with its DS’s if it intends to implement hint

OPERATION: FADVISE – Read from file with extensible results

```
ARGUMENTS
      enum fadvise_type {
            FADVISE_NORMAL     = 0,
            FADVISE_SEQUENTIAL = 1,
            FADVISE_RANDOM     = 2,
            FADVISE_WILLNEED   = 3,
            FADVISE_DONTNEED   = 4,
            FADVISE_NOREUSE    = 5,
      };

      struct FADVISE4args {
            /* CURRENT_FH: file */
            stateid4        stateid;
            offset4         offset;
            length4         count;
            bitmap4         hints;
      };

RESULTS
      struct FADVISE4resok {
            bitmap4             hints_res;
      };

      union FADVISE4res switch (nfsstat4 _status) {
            case NFS4_OK:
                  FADVISE4resok  fadvise_resok4;
       default:
            void;
      };
```

NFSv4.2 FADVISE Hints

- FADVISE_NORMAL
  - No advice to give on its behavior with respect to the specified data
  - Server always assume this value unless it is explicitly set differeingly
  - Can be used to ‘reset’/’cancel’ hint for a range
- FADVISE_SEQUENTIAL
  - Specified data will be accessed sequentially from lower offsets to higher offsets
- FADVISE_RANDOM
  - Specified data will be accessed in a random order

- FADVISE_WILLNEED_TOREAD
- FADVISE_WILLNEED_TOWRITE
  - Specified data will be read/written in the near future
  - To implement posix, set one or both depending on file open flag
- FADVISE_DONTNEED
  - Specified data will not be accessed in the near future
- FADVISE_NOREUSE
  - Specified data will be accessed once and then not accessed afterwards

Enterprise Applications Draft I/O Hints

- Most of this draft covered by our fadvise draft
- Except…
  1. Per-Read and Per-Write Hints
     - No longer needed
     - Hints useful only if they are stable for a non-zero period of time
  2. IO_ADVISE4_PREFETCH_OPPORTUNISTIC: Stateid holder expects to access the data soon; prefetch if it can be done at a marginal cost
     - Mike to find a specific application example/use case
     - If the client knows that it will want to read the data soon, then when would it not want the server to prefetch the data at any cost?
     - What does it mean for an application to ‘maybe’ want data?

  1. IO_ADVISE4_RECENTLY_USED: The client has recently accessed the byte range in its own cache
     - Use Case?
     - Might relate more to exclusive caching, as some research has shown that more exclusive caches are useful, unaware of research showing the reverse

Crazy Brainstorming Use Cases

- Hints
  - FADVISE_MANY – It is expected that many clients will access specified range in the near future
    - Cache data on multiple nodes
    - Can return different pNFS layouts to indicate the same data is cached on multiple nodes