<goo.gl/ZaeMiy>

[Non-NDA] Host-Managed Fast-Fail Read Proposal

Proposer: Lawrence Ying (lyying@google.com)

Problem Statement

In a typical large-scale distributed file system, a piece of data is stored across multiple HDDs. Thus, if a given HDD cannot complete a read request quickly, it is often much better for that HDD to abandon the read request, and have the distributed file system read from another HDD instead.

High-Level Intent and Scope

The intent is to create two policies for reads. The default policy would be the current one, which is "really try hard to read". However, many reads would be issued with a "fast read" policy, which returns an error to the host if the read cannot be completed quickly (due to any reason, including read retry or uninterruptable firmware background task). In addition, after a "fast read" fails, the host still has the option to use the "really try hard to read" option if it desires.

Note that the followings are specifically out of scope under this proposal:

- Changes to command queueing and/or caching in order to support different read traffic classes or quota system, as well as changes to HDD queue management in general
- Generic host management of disk background commands, tasks, or services
- Generic logging or health monitoring

Proposed Schedule

Would like to standardize an OCP specification in ~3 months.

Technical Details

The technical details are intended to be discussed under the OCP Storage Project Umbrella. Note that Google only cares about SATA HDDs and not SAS HDDs, but is open to creating a standard for both SAS and SATA depending on preference of the OCP Storage Community.

- There needs to be a unique “bit” identified for both DMA and FPDMA Read commands.
    - Bit unset = Traditional (“really try hard”) read
    - Bit set = Fast-fail (“limited deadline”) read (ie, return Read Error if the data cannot be read within “XX” number of milliseconds, see below)
- There needs to be a way for the host to set the “XX” milliseconds timeout limit globally for all fast-fail reads. The definition of this timeout will need to be explicitly defined as well (ie, is this purely a command timeout, or an error recovery timeout). This timeout can be accomplished through one of the followings:
    - Set Feature command
    - Write to a custom (but standardized) Log Page

Note 1: Supported range of “XX” may be ~20ms to ~1sec? Alternatively, would X00ms be sufficient?

Note 2: There is already a way to set the timeout limit for all traditional reads. An example would be the SCT Error Recovery Control (ERC) command set.

Note 3: NCQ Isochronous Command Completion (ICC) is a fine FPDMA alternative, but the full command set may be too complex for HDD Vendors to implement.

Note 4: For testability purpose, may want a way to artificially inject read failures in either buckets.

Note 5: May be nice to have FW counters or logs that track the various status of this behavior.

Software and Firmware Changes

- When a fast-fail read timeout occurs, the HDD FW (and HBA FW) should return a proper read error to the Host, but internally treat it as if the read has never happened (ie, HDD FW’s data integrity related counters and processes should not trigger).
- The Host software should determine what type of read to issue on a per IO basis, and also properly handle the resulting Read Error accordingly if it arises.
- For workloads where the fast-fail read becomes the dominating read method, and that data durability is a concern, then:
    - Either the HDD FW or the Host SW may need to implement a secondary counter to track the rates of fast-fail reads happening, and/or
    - The HDD FW or the Host SW may need to perform background data scanning.