Summary

Open-Channel Solid State Drives

* **Need of Open-Channel SSD:**Traditional SSDs manage the data storage and IO task on their own, i.e they do not take *advice* of underlying Operating System as to how the data should be managed. For example, in case of NAND Flash Die, the data storage depends on the selection of plane, block, pages and then subsequent individual cells. But this trivial method of storing data has many limitations, provisioning of location within SSD. For balancing out the *wear-and-tear* of SSDs, the actual location of data storage needs to be balanced out and stored in such a way that removes the needs of fragmentation.   
    
  Open-Channel SSDs differ from these traditional SSDs as they offer the controlling unit to the hosts itself. The host can manage the internal parallelism and thus provision the workloads as efficiently as needed.
* **Brief of Open Channel SSDs (OC-SSD):**The main purpose of Open-Channel SSDs is to provide the controlling reign to the host machine to manage data storage. Since Read and Write are pretty light operations when compared to Erase (R/W works at page level, Erase works at Block level), it’s important to perform as few operations as possible and still maintain the data integrity. The three main purposes of OC-SSDs are:  
  + I/O Isolation:  
    OC-SSD can be used efficiently in multi tenant applications as it provides separate non-interfering channels to each user, thus providing ease and security both at the same time.
  + Predictable Latency:  
    The host has complete control over the submission of IO task and thus at latency of IO and delay can be predicted at software level (before it had to sense at SSD level).
  + Software Defined Non-Volatile Memory:  
    The host having the authority over FTL layer can now provide optimizations at software level, that helps in performing IO operations and management of data. This advancement can even be done at application layer now. liblightnvm (a set of OC-SSD controlling APIs) helps in provisioning the task.