**混合存储中Flashcache使用的误区以及解决方案**

Flashcache是facebook释放出来的开源的混合存储方案,用ssd来做cache提升IO设备的性能.很多硬件厂商也有类似的方案，比如说LSI raid卡. 但是这个方案是免费的软件方案，而且经过产品的考验，具体参见：  
主页：<https://github.com/facebook/flashcache>  
开源混合存储方案(Flashcache)：<http://blog.yufeng.info/archives/1165>  
Flashcache新版重大变化：<http://blog.yufeng.info/archives/1429>

但是flashcache在使用中很多人会有个误区，导致性能很低。首先我们看下flashcache的设计背景和适用场景：

*Introduction :  
============  
Flashcache is a write back block cache Linux kernel module. This  
document describes the design, futures ideas, configuration, tuning of  
the flashcache and concludes with a note covering the testability  
hooks within flashcache and the testing that we did. Flashcache was  
built primarily as a block cache for InnoDB but is general purpose and  
can be used by other applications as well.*

它是为数据库这样的应用的离散读写优化。如果你用在了顺序读写，就有非常大的性能问题。  
那么为什么呢？我来分析下：

flashcache把内部的cache空间分成很多set, 是以set而不是整体为单位提供cache以及flush后备操作. 也就是说当一个set里面的dirty page达到一个预设的值的时候，就需要把这么dirty page 淘汰并且flush到后备设备去，以便腾出空间给更热的数据使用。  
那么每个set多大呢？

*To compute the target set for a given dbn  
target set = (dbn / block size / set size) mod (number of sets)  
Once we have the target set, linear probe within the set finds the  
block. Note that a sequential range of disk blocks will all map onto a  
given set.*

set默认是 512\*4k = 2M大小，也就是说如果你的这个set刚好是一个文件所在的块，而且每次这个文件都不停的顺序写，很快这个set都变成dirty, 那么flashcache就选择马上刷，这样加速效果就没有了。

幸好作者Mohan认识到了这个问题，提供了解决方案：

见文档： https://github.com/facebook/flashcache/blob/master/doc/flashcache-sa-guide.txt 中的章节Tuning Sequential IO Skipping for better flashcache performance

他引入了配置参数来解决这个问题：

*dev.flashcache.skip\_seq\_thresh\_kb:  
Skip (don’t cache) sequential IO larger than this number (in kb).  
0 (default) means cache all IO, both sequential and random.  
Sequential IO can only be determined ‘after the fact’, so  
this much of each sequential I/O will be cached before we skip  
the rest. Does not affect searching for IO in an existing cache.*

这样你可以把太大的顺序写操作给过滤掉了，大大提升cache的性能。