Achieving cost/performance balance ratio using tiered storage caching techniques: A case study with CephFS

M D Poat¹, J Lauret¹
¹ Brookhaven National Laboratory, P.O Box 5000, Upton, New York 11973-5000, USA

Abstract. As demand for widely accessible storage capacity increases and usage is on the rise, steady IO performance is desired but tends to suffer within multi-user environments. Typical deployments use standard hard drives as the cost per/GB is quite low. On the other hand, HDD based solutions for storage is not known to scale well with process concurrency and soon enough, high rate of IOPs create a “random access” pattern killing performance. Though not all SSDs are alike, SSDs are an established technology often used to address this exact “random access” problem. In this contribution, we will first discuss the IO performance of many different SSD drives (tested in a comparable and standalone manner). We will then be discussing the performance and integrity of at least three low-level disk caching techniques (Flashcache, dm-cache, and bcache) including individual policies, procedures, and IO performance. Furthermore, the STAR online computing infrastructure currently hosts a POSIX-compliant Ceph distributed storage cluster - while caching is not a native feature of CephFS (only exists in the Ceph Object store), we will show how one can implement a caching mechanism profiling from an implementation at a lower level. As our illustration, we will present our CephFS setup, IO performance tests, and overall experience from such configuration. We hope this work will service the community’s interest for using disk-caching mechanisms with applicable uses such as distributed storage systems and seeking an overall IO performance gain.

1. Introduction

The Solenoidal Tracker at RHIC (STAR) experiment at the Relativistic Heavy Ion Collider (RHIC) is a worldwide known experiment that has been taking increasing amounts of Physics data year over year. Within STAR, our online compute farm contains 240 TB of raw storage which composes our CephFS distributed storage system offering users 80 TB of redundant safe storage (replication 3). Our initial published work was presented at the CHEP 2015 conference [1], where we studied and compared OpenStack Swift Object Storage and Ceph Object Storage based on architecture and IO performance followed by the deployment of CephFS (POSIX layer on top of Ceph Object Storage) which was our retained solution. Our choice was motivated by our findings that, within the hardware available to us, CephFS performed the best in terms of IO performance and met our needs by having a mountable POSIX FS. The work at CHEP 2015 was then preceded with a contribution presented at ACAT 2016 [2] where we implemented a portion of SSD drives into the Ceph cluster. This was a
natural progression of our studies, whilst the cost per/GB of SSDs has decreased since inception; SSD costs are still significantly more than standard HDDs. We initially thought that a possible approach could be the use of a mixture of both HDDs and SSDs coupled with a caching mechanism between the two types of drives. The POSIX/CephFS implementation however did not provide a “natural” caching mechanism so instead, we tried using the many built-in data placement techniques within Ceph such as Primary Affinity, Journals on SSD, and Cache Tiering. We found that each technique had its drawbacks and did not lead to the performance impact we sought. We found that Ceph would only perform as fast as the slowest of its components.

With our previous efforts in mind, we decided to take a different approach where we would implement the SSDs at a lower level where Ceph would be unaware of the underlying hardware configuration. With such approach, the most performant drive technology can be exposed to the application while the lower performing drives (in IOPs performance metric) would be used for storage capacity. Furthermore, least used files could be transparently migrated to the least performing storage in the background. With this agile concept, both low cost and performance may very well be achieved.

In this study we will show how one can use disk caching techniques such as Flachecache, bcache, and dm-cache to leverage SSDs within a Ceph context. These caching techniques allow you to put your fast SSD drive as the facing cache disk then use 1 or more HDDs as the backend storage. Multiple configurations have been tested such as a 1SSD over 1HDD cache, a 1SSD over 3HDD cache, a 1SSD over RAID0 cache, and a bare RAID0 configuration. We will discuss how we implemented each technique, what features the caching techniques offer along with the IO performance of the individual cache setup. Furthermore each cache configuration was implemented in Ceph where IO performance tests were run to show the performance impact vs. a HDD only Ceph cluster.

2. STAR’s Ceph Distributed Storage System

The STAR Ceph Distributed Storage System is run on our Online Linux Pool with a total composition of 120 x 2 TB drives. With replication 3, we are able to offer 80 TB of redundant storage to our users. CephFS is the POSIX compliant (NFS like) mountable storage that creates a seamless integration of our distributed storage system. Our use cases for a CephFS storage system range from processing such as QA workflows online from recovered DAQ files to a simple user scratch space and even a backup store. While CephFS is not considered a production ready product in all versions, it has proven to be very stable since inception at STAR.

2.1 Hardware and Configurations

The nodes we are running for our Ceph cluster are 30 Dell PowerEdge 2950 models each with 6 SAS hard drives (2 for the OS and 4 for Ceph). The SAS hard drives we are using are 2 TB Seagate ST2000NM001 drives which run at 3.0 Gbps link speed, the Dell PowerEdge 2950 SAS backplane ports only support up to 3.0 Gbps link speed. The SSD drives we used for this work were 1 TB Mushkin MKNSSDRE1TB SSD SATA drives which support up to SATA III 6.0 Gbps link speed. SAS backplane ports are compatible with SATA drives; however, it was found that after connecting our 1 TB Mushkin SSD drives the link speed obtained was only 1.5 Gbps. When testing 1 TB Crucial M550 SSD drives we had from past work, the link speed obtained with those drives was 3.0 Gbps. Evidently, not all SATA drives will obtain a 3.0 Gbps link speed with the Dell PowerEdge 2950 SAS backplane ports unless we used a Dell Interposer Card. The Dell interposer card is an intermediary device between the SATA drive and the Dell SAS backplane that ensures proper link speed. After obtaining enough of the interposer cards for all of our machines we were able to obtain a consistent 3.0 Gbps link speed for all SATA SSD drives.
Figure 1. The top diagram is describing one node in a “Stock” Ceph cluster where there are four disks per machine. The arrows are pointing to how we transitioned to each of the new configurations such as 1 SSD in front of 3 HDDs, 1 SSD in front of a RAID0 consisting of 3 HDD, followed by a RAID0 configured in Ceph.

The Ceph configurations shown in Figure 1 illustrate each of the caching layouts we defined to the machines. In our initial ‘stock’ Ceph cluster we implemented 4 HDD per node to each run as individual Object Storage Daemons (OSD) within Ceph (4 HDD = 4 OSD). We then implemented three different drive layouts to implement into Ceph, the first being 1 SSD as a cache to 3 HDD (3 OSD total), second was 1 SSD as a cache to a 3 HDD RAID0 (1 OSD total), and lastly was a plain 3 HDD RAID0 for comparison (1 OSD total). With the configuration shown above, this design allows the underlying configuration to be hidden from Ceph. By defining an underlying cache mechanism, the virtual device can decide when/what to put in and out of the cache and reduce the burden on Ceph.

3. Bare Disk

Figure 2. The plot above is showing IO performance results of the 1 TB Mushkin SSD, the 2 TB Seagate HDD and a 3x 2 TB Seagate RAID0. IO performance test is of 4096 KB data chunks per thread shown in MB/s as a function of the number of threads.
As shown in Figure 2, the 2 TB Seagate SAS HDD performance plateaus around 10 threads and reaches near the manufacturer bare disk performance (140 MB/s). With the HDD formatted with XFS (XFS format is consistent in all our performance tests and within Ceph) the drive reaches ~120 MB/s. The 1 TB Mushkin SSDs reach an IO performance in range of 200 to 250 MB/s depending on the number of threads. The SSD outperforms the HDD by a near factor of 2. The manufacturer of the 1TB Mushkin SSD states maximum write speeds of ~560 MB/s. Although due to the slower than maximum link speed of 3.0 Gbps, the SSD will not obtain maximum speed and could be recognized as a bottleneck. Lastly, a 3x HDD RAID0 was configured and tested. At a high concurrency (number of threads), its performance asymptotically reaches the IO of the SSD’s but otherwise remains below its performance, a promising outlook for SSD use.

4. bcache

bcache is a Linux kernel block layer cache. It allows one or more fast disk drives such as flash-based solid state drives (SSDs) to act as a cache for one or more slower hard disk drives [3]. bcache supports three cache modes – write-through, write-back, and write-around. Write-through caching directs the IO onto the caching drive (SSD) and through to the underlying storage (HDD). This ensures the data is on both the SSD and HDD. The benefit from write-through is when the data is read, it will be read from cache. Write-back caching is where the IO is only directed to the cache device and is immediately confirmed as a completed write to the client. Write-back caching is good for read and write intensive applications, however, there is a higher exposure to risk in terms of data loss due to an event the cache drive fails. Lastly, write-around cache only directs the IO to the underlying storage (HDD) by bypassing the cache. This configuration will not offer any performance benefit but could be set in the event there is a problem with the caching drive or for testing purposes.

![Figure 3. The plot above is showing IO performance results for bcache configurations composed of one SSD acting as a cache for one HDD (bcache 1SSD:1HDD – light green curve) and one SSD acting as a cache for three HDDs (bcache 1SSD:3HDD dark green curve). The two green curves are showing the IO performance of the bcache configurations while being the dotted line curves are the bare disk tests shown for comparison (look back at Figure 2). The IO performance tests are of 4096 KB data chunks per thread shown in MB/s as a function of the number of threads.](image)

The bare bcache performance tests were expected to be in par with the performance of the bare SSD. As shown in Figure 3, the performance of bcache 1SSD:1HDD (light green curve) is consistent with the bare SSD (blue curve) at a low number of threads but then begins to drop in performance around 15 threads. The bcache device was set to ‘write-back’ which in turn should write all IO to the caching device and return once completed. However, even in write-back mode bcache doesn’t cache
everything. What happens is when bcache recognizes sequential IO, the virtual device will try to skip the sequential IO and write it directly to the backing HDD. To circumvent this, bcache recommends disabling the ‘sequential_cutoff’ [4] which was disabled for our test. Lastly, bcache will track latency and will gradually throttle traffic if the latency exceeds a certain threshold (20 milliseconds for writes) -- latency tracking and IO throttling were also disabled for our test as well [5]. While taking the performance measurements shown in Figure 3, the IO was watched carefully using iostat [6] (monitors input/output statistics) to ensure all IO was being written to the SSD, which was confirmed. The 1SSD:3HDD bcache measurement exhibited a performance degradation right out of the gate as the number of threads increased; all IO was tracked and confirmed to the SSD as well. All bcache performance tests that we took have results that are reproducible. With that said, the drop in performance compared to the bare SSD may be caused by a discrepancy within bcache itself.

5. dm-cache

dm-cache is a device mapper target for generic block-level disk caching for storage networking. It aims to improve performance of a block device (e.g., a spindle) by dynamically migrating some of its data to a faster, smaller device (e.g., an SSD) [7]. dm-cache allows one or more SSDs to be mapped to one or more HDDs and supports the three main caching techniques, write-through, write-back, and write-around caching. dm-cache requires three logical volumes per logical device, a Metadata volume and Cache volume on the SSD, and an Origin volume on the HDD.

![Figure 4](image)

**Figure 4.** The plot above is showing IO performance results for dm-cache configurations composed of one SSD acting as a cache for one HDD (dm-cache 1SSD:1HDD light red curve) and one SSD acting as a cache for three HDD (dm-cache 1SSD:3HDD dark red curve). The two red curves are showing the IO performance of the dm-cache configurations while the dotted line curves are the bare disk tests shown for comparison (refer back to Figure 2 if needed). The IO performance tests are of 4096 KB data chunk per thread shown in MB/s as a function of the number of threads.

The dm-cache configurations of 1SSD:1HDD and 1SSD:3HDD were setup using all default parameters except the device was set to use write-back mode. Write-back ensures all IO will go to the caching device or SSD. In Figure 4, the IO performance of the dm-cache devices (red solid curves) is not drastically impacted by the number of threads and the performance is fairly consistent but is slightly slower than the bare SSD. The performance of the dm-cache devices compared to the bare SSD may contribute to the overhead of dm-cache. Nonetheless the dm-cache device does outperform the bare HDD and appears to be a potential good candidate for our Ceph distributed storage system.

6. CephFS Performance Results
After the individual drive and virtual device configurations testing, we moved to implement all of the storage devices into a working CephFS cluster. Multiple CephFS clusters were setup independent of one another. Each cluster was made up of 20 OSDs which will appear to Ceph as 20 individual block devices whether a single OSD be a single HDD, SDD, a 3x HDD RAID0 device, 1SSD:1HDD cache device or 1SSD:3HDD cache device. The software we used for performance testing was primarily IOZone [8]. The tests were composed of 10 clients that each ran tests simultaneously, each client test started at 1 thread and increased to 10 threads, each thread composed of 4096 KB data chunks. The total test(s) into Ceph was essentially IOZone tests starting at 10 threads (1 per client) increasing to 100 threads (10 per client) all writing into a single CephFS store.

![Figure 5](image)

**Figure 5.** The plots above are showing 10 clients writing into multiple CephFS clusters using IOZone. Each client is writing at 4096 KB data chunks with the number of threads increasing from 1 – 10. The curve(s) on each of the plots is the aggregate performance of each client. The curves represent 1 thread per client at the starting point X=1 (10 threads in total) increasing to 10 threads per client at X=10 (100 threads in total).

When comparing the performance of the CephFS clusters to the bare disk/device performance, the results are not consistent with our expectations. In Figure 5, the plot on the left were tests for the following CephFS cluster configurations – 20 HDD, 20 SSD, 20 1SSD:1HDD dm-cache devices, 20 3x HDD RAID0 configurations, and a 60 HDD cluster (60 OSD) cluster for comparison. The first two curves to mention in the left plot is the dark blue curve: HDD – 20OSD and the green curve: SSD – 20OSD. It can be seen that the IO performance of the HDD – 20OSD cluster outperforms at a factor of ~2 at a low number of threads over the SSD – 20OSD cluster. As the number of threads increase, the HDD – 20OSD cluster continues to outperform until the end of the test. If you refer back to Figure 2, it can be seen that the results appear to be a reversal of our expectations. Previously, the bare SSD performance tests surpassed those of the bare HDD at a factor of 2-2.5. Additionally in the left plot of Figure 5, the red curve which represents dm-cache 1SSD:1HDD – 20OSD outperforms both the HDD – 20OSD and SSD – 20OSD clusters while the bare performance tests of dm-cache 1SSD:1HDD shows the write performance falling between the bare SSD and bare HDD. With further investigation, the dm-cache cluster response to the Ceph OSD journal flush may be the reason for such inflated performance. Additionally, could the SSD – 20OSD Ceph cluster OSD journals be the bottleneck for performing below the performance of the HDD – 20OSD cluster?

The Ceph documentation [9] states that solid state drives can be used for OSD journals to improve performance. The SSD must be carefully chosen depending on the cluster, number of journals per SSD, IO expectations, etc. In our contribution at ACAT 2016 we tested this configuration with multiple consumer grade SSDs but did not see a performance impact. While digesting the results from
the left plot in Figure 5, we moved toward an unconventional configuration where instead of mounting the Ceph OSD journals onto “faster” SSD drives, we did the reverse and placed them on HDDs (one OSD journal per HDD). Each OSD would have the data and filesystem on one hard drive and the journals would be on a second hard drive. The right plot in Figure 5 is composed of four different CephFS cluster configurations, a 20 HDD OSD cluster with journals on 20 separate HDDs, a 20 SSD OSD cluster with journals on 20 separate HDDs, a 20 dm-cache 1SSD:1HDD cluster with journals on 20 separate HDDs, and lastly a 20 3x HDD RAID0 configuration with the journals kept on the RAID0 (used for comparison). With much surprise, the new SSD Ceph cluster with externally mounted journals on HDD performs at ~3x faster over the SSD only cluster. The HDD cluster with external HDD journals performs at ~2x faster over the HDD only cluster. And the dm-cache cluster gained a ~0.5x performance with external journals. How to explain this behaviour?

Ceph journal writes are synchronous and on Linux systems, synchronous writes to block devices will result in an ATA_CMD_FLUSH being issued to the device [10]. Some storage devices will guarantee that all writes will eventually succeed even in the event of a power failure if a battery exists or if the device has PLP (Power Loss Protection). Other storage devices, those that may not have PLP will wait until the data is flushed to persistent storage before moving on the next write, this can cause latency. Lastly, on some consumer-grade SSDs, the device may return a flush request immediately even if the device does not have any PLP, this would give the perception of a fast device but would not guarantee safe data. Power Loss Protection is typically built into Enterprise grade storage devices in both HDDs and SSDs. It has been found that the 2 TB Seagate SAS HDDs that we have used for our tests are “Enterprise” grade drives and do have PLP built in. As for the 1 TB Mushkin SSD drives we used, while documentation is sparse, it has no mention of a built in battery nor any form of PLP. With this, it would be safe to say that the reason for the slow performance of the bare SSD cluster is due to the handling of the ATA_CMD_FLUSH call causing latency. Our stock HDD – 20 OSD Ceph cluster and all the Ceph clusters we setup with journals mounted on the HDDs that have PLP, the increased write performance is understood in that the HDD device can skip the flush causing faster writes. Lastly, since the dm-cache devices are set to write-back (which will send all IO to the SSD), and the write performance of the dm-cache Ceph cluster outperforms the 20 SSD & 20 HDD Ceph clusters with and without external journals; it is difficult to conclude how this virtual device is handling the ATA_CMD_FLUSH call. Since the IO is going to the SSD and we know the SSDs do not support PLP, the dm-cache device may be skipping the flush call which may cause un-safe data.

7. Cost Analysis

Our initial intent was to implement a small number of SSDs into our Ceph cluster by leveraging the few drives using disk caching techniques. While this goal still may be attainable with the use of Enterprise SSDs or PLP supported SSDs, it may not be possible with the 1 TB Mushkin drives we had in stock (this should come as a lesson learned and an important consideration to anyone attempting to speed up IO using SSD drives).

In its original state without any added SSD drives, the stock STAR Ceph cluster composed of 120 x 2 TB HDD cost $14,400 for the HDDs alone ($120 each). If we were to simply replace our Ceph cluster with 120 2TB Enterprise grade SSDs, we would incur a huge cost of $126,000 ($1,050 per SSD), a ~9x cost increase. By manufacture specifications, the Seagate HDD drives can obtain a speed of 140 MB/s, as for the average 2TB Enterprise grade SSD manufacturers report obtainable speeds of 560 MB/s (4x performance increase). For such a setup, it would result in a near 2:1 Cost/Performance ratio in terms of MB/s.

8. Conclusion
When testing outside of Ceph, it is clear that the write performance of the SSD drives outperform the HDD with single and concurrent thread tests (results are based on 4096 KB chunk IOZone write tests). Additionally, while our bcache configurations performed well at a low number of threads but dropped at a higher number of threads, the dm-cache 1SSD:1HDD & 1SSD:3HDD configurations were consistent and performed well at all thread counts (dm-cache performance was slightly below bare SSD test which may account for dm-cache overhead, overall good performance).

Within Ceph, it has been found that the OSD journals play a critical role and can make or break your cluster in terms of write performance. Additionally, it has become clear that IO performance of bare drive tests and Ceph cluster tests are not to be compared on the same level. When observing a factor of ~2-2.5 performance increase of a bare 1 TB Mushkin SSD to a bare 2 TB Seagate HDD, it was natural to expect similar results when comparing a 20 OSD SSD Ceph cluster to a 20 OSD HDD Ceph cluster. The performance of the two Ceph clusters was not only dissimilar to the bare drive tests, but the results were actually flipped in that the SSD cluster performed worse than the HDD cluster. Considering the impact of the Ceph journals and the importance of PLP supported drives, the results are now understood. Individual drive support of PLP (Power Loss Protection) is a critical component for Ceph OSD journals when seeking IO performance. The handling of the ATA_CMD_FLUSH call can either make or break your cluster in IO performance. Finding that our SSD drives lack the needed PLP support helped us determine that Mushkin SSD drives were actually detrimental to Ceph over standard Enterprise grade HDDs. In order to obtain the IO performance we seek, enterprise grade or PLP supported SSDs would be required. Furthermore, if you look back at the left plot in Figure 5 and notice the light blue curve representing a 60 HDD (60 OSD) Ceph cluster, you can see that this configuration performs the best. Historically, Ceph has been known to perform best when the number of OSDs is increased. Since we cannot add to our initial 120 “Enterprise” HDD OSD Ceph cluster we plan to bring our cluster back to full size and continue use in this configuration.

Acknowledgements
This work was supported by the Office of Nuclear Physics within the U.S. Department of Energy’s Office of Science.

References
[1] Poat M D, Lauret J and Betts W 2015 POSIX and Object Distributed Storage systems – Performance Comparison Studies With Real-Life Scenarios in an Experimental Data Taking Context Leveraging OpenStack Swift & Ceph J. Phys.: Conf. Ser. 664 042031
[2] Poat M D and Lauret J 2016 Performance and Advanced Data Placement Techniques with Ceph’s Distributed Storage System 1. Phys.: Conf. Ser. 762 012025
[3] bcache https://bcache.evilpiepirate.org/
[4] https://evilpiepirate.org/git/linux-bcache.git/tree/Documentation/bcache.txt#n143
[5] https://evilpiepirate.org/git/linux-bcache.git/tree/Documentation/bcache.txt#n157
[6] https://linux.die.net/man/1/iostat
[7] dm-cache https://www.kernel.org/doc/Documentation/device-mapper/cache.txt
[8] Iozone http://www.iozone.org/
[9] Ceph Hardware Recommendations: Solid State Drives http://docs.ceph.com/docs/master/start/hardware-recommendations/?highlight=solid%20state#solid-state-drives
[10] Nelson M 2015 Ceph PCIE SSD Performance Part 1: OSD Journals