IBM FlashSystem Technical Whitepaper
October 2014

Understanding Sustainable Performance in Flash Technology

By

Justin Haggard
Matthew Key
Ron Herrmann

Also thank you to:

IBM FlashSystem WW Solutions Engineering Team
IBM FlashSystem Engineering Team
Erik Eyberg

Purpose

The purpose of this document is to provide education to IBM FlashSystem Sellers and Business Partners on the topics of “garbage collection,” “write-cliff,“ and other nuances of flash technology. Sections include an Executive Summary, overview of flash storage, technical explanations of garbage collection and the write-cliff, flash storage system design, advantages of IBM FlashSystem, and the typical impact garbage collection will have on the production environment.

This document was produced with input from IBM’s top flash storage engineers and is designed to be a sales aid and assist sellers and business partners in educating customers on the benefits of the IBM FlashSystem offerings, as well as countering some of the misinformation from various competitors.

Executive Summary

Flash technology has emerged into the enterprise from wide adoption in cellular and portable consumer products. We all know this as we use our smart phones, our cameras, and our USB drives. It's even in our cars, refrigerators, and traffic lights! The benefits of flash such as being fast, compact, low power, and with ever increasing density enable new capabilities for the individual and enterprises. Arguably, the greatest benefit of flash technology has been its speed. For generations, spinning media has been the core of data storage, but it has been limited by the rotational speed of the disk. Disk has increased from 7200 RPM to 15000 RPM over the past ten years, but it is still the bottleneck for always faster CPUs running applications.

However, applications do see visible impact from the speed of flash. It's not a single read or a write that make a difference - it's millions. Compound these millions of read and write requests (I/Os) and you reach beneficial results in the levels of seconds and hours. These are scales of time that applications do benefit from. When applications benefit, people benefit – with better response times and shorter batch jobs.

With this new speed and efficiency come greater scrutiny when this speed is impacted. Taking a process that originally clocked at six hours down to one hour is quite an improvement. When that process one day takes 1hour 20 minutes, people want to know why. It is not a factor of "1 hour 20min is still faster than 6 hours." Instead, it is "Why is the process running 25% slower?" When quicker time to results happens, it becomes the new baseline.

Hence this paper. Flash is fast yet has inherent characteristics that must be considered during the development of a data storage product. Most of these are handled through product engineering, while the others must be handled at the architecture or application level.

Garbage collection is probably the most misunderstood topic in the area of flash storage. Various storage vendors make erroneous claims or try to scare customers with misinformation on this topic. Some flash storage vendors will even dismiss the idea and claim their product isn't impacted by garbage collection at all. The purpose of this whitepaper is to bring all the facts to the table in a common sense manner with the goal of describing how IBM FlashSystem products have been physically and logically designed from the ground up to handle garbage collection properly and avoid the known limits of flash as much as is technically possible.

Most technologies have their equivalent of garbage collection and consequent impacts. A favorite analogy is the concept of rust with automobile manufacturers. How is it possible for two cars made in the same year, for example a mini-van and a luxury sedan, to sit in the same driveway in Cleveland and yet only one of them rusts? Why is the mini-van showing signs of rust after three years and yet the luxury car doesn't rust for eight years? Or why does the same mini-van rust after three years in Cleveland, yet is rust free in Tucson after ten years?

There are various reasons – 1) the quality of the materials used; 2 the design of the car (are there places where water enters seams?, etc); 3) the manufacturing processes (are parts treated with rust proofing?, etc), and finally 4) the climate that the car is operated in. In our example, the luxury car is made with a better quality of metals and those materials are also treated to be rust resistant, so it takes much longer to rust in equal climate conditions than the mini-van. Also, the climate in Cleveland, which gets bitter cold in the winter, very humid in the summer, and where lots of salt is sprinkled on the road to melt snow, causes rust on the mini-van much faster than it does one in Tucson, where it is dry and relatively warm year round. How does this relate to garbage collection? As you will learn in this whitepaper, the quality of the parts, the manufacturing processes, and the system design, as well as the host application use cases of the storage have an impact on flash behavior.

How is flash storage different from disk and tape?

There are two differences in how flash and disk or tape store data. The first is flash uses a current of electricity to program data, while tape and disk use a magnetic process. The second difference comes when data must be rewritten. For disks and tape, a storage block or sector is simply overwritten again with the updated data. On the other hand, flash storage blocks can only be written if they are empty. If they are not already empty, they must first be erased.

There is already some asymmetry in read and write performance of flash where reads are quickest and writes are relatively slower than reads. Erases are significantly slower than either. When blocks must be erased before being written, the overall write performance is degraded by the time of the erase and the time of the writes averaged together. This creates an even greater asymmetry.

The size and arrangement of blocks in flash further exacerbates the differences between flash and disk. Disk reads and writes operate on a small sector of storage (512 bytes or 4,096 bytes on newer disks). On flash, reads are performed at 4,096 bytes or smaller, writes are generally performed at 4,096 bytes or larger, and erases are performed at 1MB or larger. Because the storage must maintain backwards compatibility with the applications, this stresses flash controllers significantly for writes.

Overwriting existing data on flash is like having to bulldoze the entire city block and rebuild *just* to redecorate your bedroom.

Since we cannot force the application to write exclusively at the size of flash blocks, the most common method used by flash controller vendors to maintain performance is to virtualize the application data and write all data sequentially onto the flash and maintain a Flash Translation Layer (FTL) where host addresses are translated to a physical flash location. This means that all data that is to be read must go through a lookup process and find where the address truly resides on the flash chips.

This raises the issue, "what if the application updates data or writes the same address repeatedly?" At the flash layer, all writes are done to erased areas of flash. This results in the same host address repeatedly stored in flash with old/invalid data alongside the most recently written valid copy of the address. Because the capacity of flash is finite, it is inevitable that the space devoted to holding that old data must be recovered.

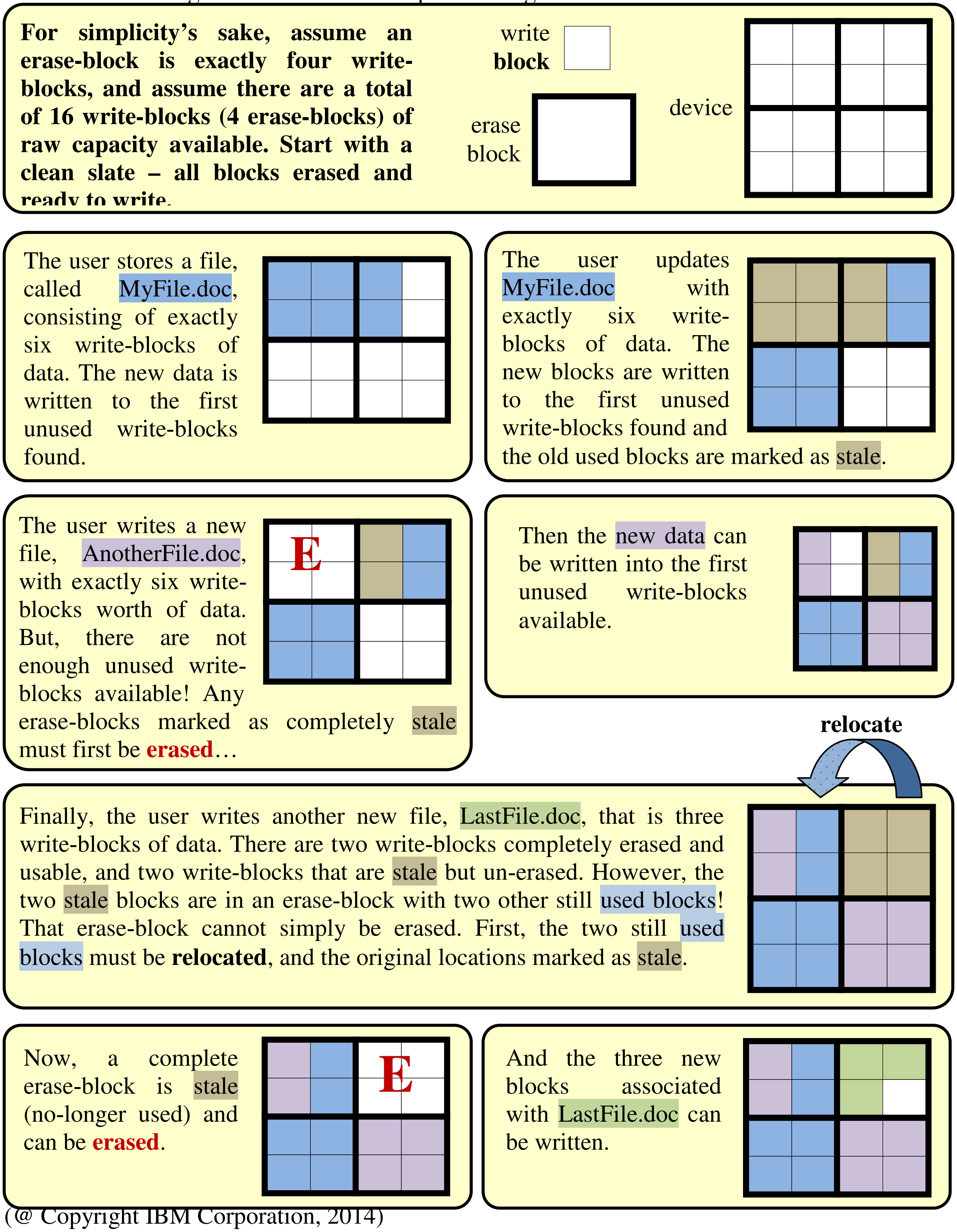
What is garbage collection?

As data is written, if that address has been written before, there is an old copy of that address sitting somewhere in the flash as the new data for that address goes to another location in flash. Unless it is referenced for some other reasons like a backup point or a snapshot, that old copy is garbage. The valuable flash capacity must be recovered.

If only it were as easy as erasing that small portion of flash occupied by that old address. Instead, that address is within a much larger block of flash shared with a large amount of other valid, most recent copies of addresses. To free the flash of the garbage, we first must move the good data from the block. After the good data has been safely copied to another location, the block gets erased. At this point, that block is now ready for new data. The good data did not get copied to some random location prior to the erase. See Figure 1. on the next page for a "comic strip" illustration of garbage collection.

Flash controllers often have access to flash capacity beyond what is presented to the attached application servers. Vendors call this the raw capacity, over-provisioning, or buffer capacity of flash. In the IBM FlashSystem 840, a 40TB usable system can present 40TB of storage (LUNs) to application server hosts; however, with data parity, a RAID spare, and flash over-provisioning, the total is nearly 67TB of raw capacity. Much of this 27TB of excess capacity is used by the flash controllers for garbage collection and erase scratch space.

Figure 1. The Comic Strip – "Garbage Collection Man"



For simplicity's sake, assume an erase-block is exactly four write-blocks, and assume there are a total of 16 write-blocks (4 erase-blocks) of raw capacity available. Start with a clean slate – all blocks erased and ready to write.

The user stores a file, called MyFile.doc, consisting of exactly six write-blocks of data. The new data is written to the first unused write-blocks found.

The user updates MyFile.doc with exactly six write-blocks of data. The new blocks are written to the first unused write-blocks found and the old used blocks are marked as stale.

The user writes a new file, AnotherFile.doc, with exactly six write-blocks worth of data. But, there are not enough unused write-blocks available! Any erase-blocks marked as completely stale must first be **erased**…

Then the new data can be written into the first unused write-blocks available.

Finally, the user writes another new file, LastFile.doc, that is three write-blocks of data. There are two write-blocks completely erased and usable, and two write-blocks that are stale but un-erased. However, the two stale blocks are in an erase-block with two other still used blocks! That erase-block cannot simply be erased. First, the two still used blocks must be **relocated**, and the original locations marked as stale.

Now, a complete erase-block is stale (no-longer used) and can be **erased**.

And the three new blocks associated with LastFile.doc can be written.

Is it possible to eliminate the need to do garbage collection?

Since garbage collection is the process of freeing the space of addresses that have been most recently written, there is only one condition where a flash storage array would not need to do garbage collection – if the flash storage array was filled with original data (an amount less than the capacity of the unit) and this data was never erased or altered, essentially becoming read-only. Unless this use case is why you are purchasing a flash-based storage array, then garbage collection will be needed!

All vendors' flash products, whether solid state drives (SSD), all-flash module arrays, or PCIe based flash storage, must have the ability to deal with garbage collection. It is a normal part of flash storage operations and good design and engineering will deal with it before it gets to be a problem.

The Write-Cliff

There are two direct impacts of garbage collection: the write-cliff and write amplification. The write-cliff is the phenomenon that can occur when the performance of a flash storage system drops considerably during garbage collection and impacts production. As a flash storage system moves data from a fragmented block to free up space for more writes, it also writes to a portion of a new block. These writes are generally done in-line with new data coming in from the application. This means that we now have two feeds of data sharing the write performance of the new block because they are sharing the same data-bus to the new block. As you will read in the coming sections, the IBM FlashSystem unit has been designed to eliminate the possibility of a write-cliff in also all production environments.

You may not immediately recognize it, but you have seen this behavior before. In cached arrays, there is a honeymoon period while cache is filling up and, once full, starts to flush or destage down to the disks, causing an extreme decline in performance, or write cliff, from DRAM response time to disk response time. In flash, the time before garbage collection can be seen as cache-like performance. However, because flash offers significantly more capacity than the DRAM cache of an array, more time and throughput is required to reveal the cliff.

Suppose you buy a stock for $5 and within a day it shoots up to $25. It stays at $25 for one day and then over the next three weeks it slips slowly down to $20. After 30 days, you still have a 400% gain – would you be upset? Most people would be very happy with this situation, but there might also be some that were disappointed that the stock wasn't still at $25. This is the situation that can occur with a well-designed storage array like the IBM FlashSystem 840 after the system fills with data for the first time and garbage collection begins. The $20 a share represents the performance that the IBM FlashSystem was engineered to deliver in your environment. The IBM FlashSystem 840 and V840 are designed to handle garbage collection in a well-engineered manner and IBM Sales Engineers conservatively integrate these products into your production environment – we understand the performance you need to meet your workloads and we will not sell something that misses this mark.

Let's tell another stock story. You buy a stock for $5 and within a day it shoots up to $10. It stays at $10 for a day, but then begins to quickly sink below $5 and ends up at $2, where it sits for several days. It then climbs back to $5, sinking again to $2 after several more days. It then fluctuates between $2 and $5. If you haven't put a stop-loss on this stock and sold it at $4.50 (which a professional trader would), this situation would drive most people nuts. This can be the experience you may have with some competing flash storage arrays – dramatic fluctuations from $10 to $2 (and from $5 to $2 over and over again) - because these systems handle garbage collection poorly and therefore their performance is very unpredictable. Sometimes the performance is good enough ($5) and sometimes it doesn't even meet production needs (when it is under $4.50).

Figure 2 below illustrates the effect garbage collection can have on flash storage performance. The systems shown below were first filled with random data (which most represents a production workload). Then, some of the data was deleted and replaced by the attached host operating systems (beginning at about minute 5), which again represents a real-world operation. A 100% random write pattern was then used to test the "worse case." The IBM FlashSystem 840 array under test was designed to deliver performance of about 3500MB/s. The competitive system was sold by another vendor having been presented the same business needs by the customer. As the chart shows, the IBM FlashSystem 840 handles garbage collection in an effective manner and production speeds are maintained. However, under the same test loads and data sets, the competitive flash storage system didn't do so well. This is an example of a write-cliff that would have a serious impact on production. The competitor's system, although able to reach speeds of 3500MB/s in the beginning, cannot keep up with garbage collection and is in a write-cliff spiral.

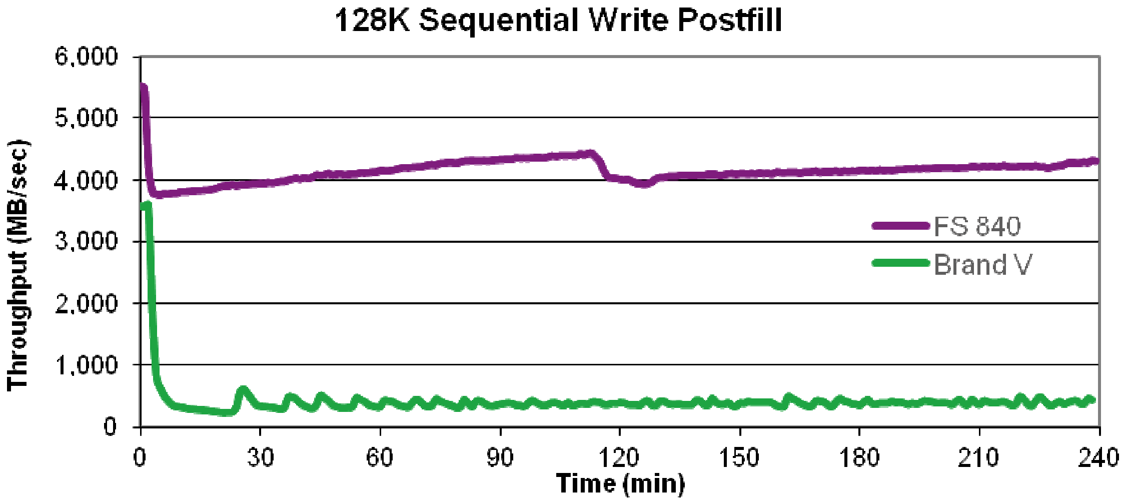


Figure 2 – FlashSystem 840 Performance during garbage collection

How often will a write cliff happen in production?

As shown in Figure 2 above, if deployed in an appropriate application environment, an IBM FlashSystem array will never suffer from a write-cliff that will impact production.

However, not all vendors' flash storage systems are designed and engineered properly and not all application environments are a good fit for the particular flash storage system deployed. The write-cliff can happen when there are not enough erased flash cells to handle incoming write needs. This is different from the flash storage array simply running out of space – because the application server's OS is involved in this logic as well. A write-cliff may develop when the attached host operating system believes there is space left on the array, yet the flash storage system has not yet erased these cells. This can happen to competitors' storage systems that are getting very close to maximum capacity while experiencing heavy delete and (re)write I/O patterns.

Write Amplification

The secondary impact of garbage collection is related to flash endurance. Flash is a consumable media and every write issued to a block counts against the total number of writes supported by the flash.

Writes from the host go to flash, but every portion of data that is moved from a used block to a new block is also considered a write to flash. The number of writes performed by the application versus the number of writes that the flash blocks actually endure is called write amplification.

Write amplification is a multiplier to the rate of flash consumption. If the host is writing 100MBps and the flash is seeing a write amplification of 5, the flash is consumed at an actual rate of 500MBps. The means used to lower write amplification are the same as those used to lessen the write cliff. Both issues of write cliff and write amplification are very important to the architecture of the array, as well as the architecture of the application stack.

IBM FlashSystem 840 Technical Approach to Garbage Collection

The IBM FlashSystem design is based on a high degree of distributed hardware-based parallelism. This parallelism utilizes an extremely intelligent hardware flash-controller that can natively issue as many concurrent commands as the flash-chips themselves can handle. We have fine-tuned the amount of over-provisioning required to maximize sustained performance with an optimal price-point (since over-provisioning is capacity that you pay for but don't get to use). Our garbage collection is also effectively "out-of-band" in that it is handled by an auxiliary hardware and processor, not the main data-path hardware. The flash controller continuously streams data to flash, while the auxiliary processor runs transparently in the background, cleaning up behind. It is a well tuned dance in which the ingest engine constantly produces new "fragmented" pages and the cleanup engine constantly tosses fresh, clean pages back. All of this is accomplished with full 2-dimensional (2-D) RAID Protection featuring patented Variable Stripe RAID^{TM}. This means an entire plane of a flash chip, even the entire flash chip itself, can fail and the data will be automatically, seamlessly migrated to new storage without a loss in performance or protection. No degraded state. No lengthy rebuild time.

The major advantages of IBM FlashSystem garbage collection and wear leveling center around a few key aspects:

IBM features a hardware-based flash controller (built using Field Programmable Gate Arrays) with dedicated out-of-data path IBM POWER processors. Each system features up to 48 controllers, and each controller handles 20 flash chips. This enables very low latency and significant parallelism through the primary hardware data path.
(@ Copyright IBM Corporation, 2014)

We can then leverage the inherent parallelism of flash. Flash chips can handle concurrent operations on each die of the flash chip without interfering with operations on other dies, and there are eight dies per flash chip. Our hardware-based flash controller issues concurrent operations on all of the flash dies while actively transferring data. This means that the system can be actively servicing 1,600 4K DMA data transfers, while also performing 5,600 other operations on the flash. These operations include erases and internal moves necessary for garbage collection. This translates into the ability to run active garbage collection and wear-leveling without significant impact on the performance or availability/reliability of the system.

IBM utilizes an over-provisioning of the listed usable capacity of flash that supports the performance of the system and longevity of the data. A 40TB FlashSystem 840, has a physical raw capacity of nearly 67TB. The extra capacity is used for RAID sparing, RAID parity, and active data and is factored in for wear-leveling, failure mitigation, and proactive garbage collection.

Our wear-leveling algorithm keeps all flash blocks on the same module within 10% of each other in Program/Erase cycles (the key measure for flash wear). It also reduces disturb errors - a little known caveat to working with flash. Disturb errors are errors induced in a cell simply by reading, writing, or erasing adjacent cells.

Our systems also pro-actively monitor the wear of flash modules. They have the ability to issue wear-out warnings when a module has lost enough capacity that performance will be impacted so that maintenance can be scheduled for flash module replacement. And they have the ability to initiate failover to the online-spare module when a module has lost enough capacity to impact user-data. All of these are tied into the SNMP trap, email alert, and call-home features so the user and IBM support can be notified of the situation both proactively and retroactively.

This level of successful engineering comes from over seven years experience working with flash and over 30 years experience designing massively parallel data acquisition and digital signal processing systems.

> *All NAND flash chips have an imbalance between read and write performance. Any system built with NAND flash chips has to mitigate that imbalance, whether those chips are packaged in commodity SSDs (like many flash arrays on the market) or differentiated flash modules (like IBM FlashSystem). Worst case workloads can be constructed to show a decrease in write performance under certain conditions for ALL flash/SSD based storage systems, including ours. But our IBM FlashSystem technology is extremely effective at reducing the impact of that imbalance so the vast majority of realistic enterprise workloads will not experience a "write cliff." We encourage competitive bakeoffs and POCs to prove that our technology is superior for your workloads.*
>
> *Erik Eyberg – IBM FlashSystem Technical Executive*

<u>Flash Storage Array capacity provisioning</u>

Some flash storage array vendors will tell customers that they recommend keeping some of the usable capacity in reserve and therefore to not provision 100% of the usable capacity. Vendors may recommend only presenting 70%, 80%, or 90% of the capacity to avoid performance problems. Flash storage is expensive and so fast that most customers want to put as many applications on it as possible. In normal production environments, IBM does not have a recommendation to reserve some of the valuable space as "overhead." If a customer purchases a 40TB FlashSystem 840, they can use all 40TB. We do not have a "flash capacity tax" like many other vendors.

The importance of pre-sales design in implementing flash storage

One of most important aspects of implementing flash in Cloud or enterprise data center environments is to make sure the applications and workloads are going to match the capacity and performance capabilities of the flash storage system. No matter how well designed a storage system is, if it is used in the wrong environment or poorly over-provisioned, it is going to fail to meet expectations. One of IBM's strengths is the experience that our FlashSystem sellers, Client Technical Specialists (CTS), and Business Partners have in designing and implementing flash storage solutions. IBM's flash sellers and Business Partners go through very comprehensive design and implementation training to ensure that the equipment they sell will work well in your environment. IBM does not aggressively sell storage equipment into an environment simply to meet a sales goal. IBM and our Business Partners spend time with our customers to understand applications and usage patterns before we create a FlashSystem solution. ***This is critical to the success of any storage implementation.***

How to properly test a flash storage array for garbage collection and the write-cliff

Just as it is important to test drive a car to make sure that it meets your family's needs, many customers test flash storage arrays to make sure they can store their data with the performance required. This is a valid exercise if it consists of tests that represent the production environment (including using actual data). Figure 2 represents a good test to run any flash storage array through, because it makes sure the product being considered meets production requirements (or doesn't) while the standard flash housekeeping of garbage collection is occurring.

Many car dealers are willing to allow you to take a car home for the weekend for an extended test drive. They don't even mind if you get the car a little dirty by taking your family to the beach or park for an outing with the kids to see how the vehicle is going to work for you in everyday life. However, if you were to drive the car off a cliff to see how well it held up after a 200 foot fall, or you were to drive the car into the lake to see how waterproof it is, the car dealer would not be happy! The point of is, design useful and realistic torture tests in the process of testing a flash storage array.

Also, many competitors like to show testing results with data that is really "fluff" – by either using Dev0 or repeatable patterns that can be cached in DRAM prior to flash cells, and/or data that is very deduplication or compression friendly, thus eliminating flash many writes (and erases). While these testing methods may show dramatic results, customers can end up being disappointed when these systems go into production and performance is not acceptable.

Sometimes it isn’t easy to come up with 40TB or more of production data for a lab test; however, this is really the best way to test a 40TB flash storage array. It is also possible to simulate production data with random data generators and other tools. At all costs, avoid tests using data comprised of all 0s, repeatable patterns, or other data that can be cached.

The most important part of testing a flash storage system is to understand the I/O patterns of the applications that will be using it. Will they be 70% read and 30% write? What I/O block size – large or small? If you plan on filling your flash storage array close to 100%, then it is very important to test typical I/O patterns of your data when the system is getting close to full.

There are some vendors who suggest that their storage systems should be benchmarked at 100% of usable capacity. This is a flawed exercise for several reasons. Most importantly, if you're running a storage system at exactly 100% of capacity, then you have zero headroom for capacity growth, demand spikes, and other unanticipated events. If you have a flexible layer on top of the storage system that can leverage other storage devices as needed - e.g. IBM Elastic Storage or SAN Volume Controller - then that could be OK. But for all other environments, running at 100% is a very dangerous approach (Imagine if you ran your car to the redline 100% of the time!).

With flash storage, "usable capacity" is a very complex balancing act among performance, economics, system lifespan, and other factors. There is no single "right" answer but we believe that the default usable capacity points in FlashSystem products deliver the optimal balance between those factors for the vast majority of enterprise environments. We've also provided an easy way to adjust this balance; simply choose to provision less capacity than our advertised usable capacity and FlashSystem will automatically use the unprovisioned space to provide a "turbo boost" (While you might think this is common sense, many flash storage systems on the market apparently do not have this capability).

System performance is a curve, not a point. IBM has chosen to provide official FlashSystem performance figures at 90% capacity utilization, because we believe that is (a) <u>much</u> higher than typical disk storage utilization and (b) reasonable for nearly all customers. <u>All</u> flash storage architectures experience varying performance with utilization levels, unless the system's capabilities are artificially limited (which in our opinion is a poor design choice). IBM is happy to provide appropriate performance estimates at any utilization level, based on customer demand.

Erik Eyberg –IBM FlashSystem Technical Executive

Garbage collection and deduplication.

Deduplication is a technology that looks for repeated data patterns and eliminates the need to store these patterns more than once by using a hash or index to record a reference to duplicates (to the original or first time data). Deduplication can be beneficial to workloads such as VDI (virtual desktops) or system backup and recovery, where many duplicate data patterns (files) are stored. Deduplication is not as beneficial to workloads such as databases, but many flash storage vendors deduplicate all data coming into the system anyway. Deduplication also has its own requirement to do a form of garbage collection. Deduplication's garbage collection happens when the host operating system deletes data from the system and the deduplication process no longer needs to reference this data. Any flash storage system that does deduplication will need to do "double garbage collection" at times – deleting deduplication reference data as well as erasing the flash cells that held that data. A challenging test for any flash storage system that does deduplication is to fill it to 90% full and then commence deleting data and replacing it with entirely new (unique) data sets. This will trigger deduplication and flash garbage collection and usually impact performance significantly.

Summary

Just as all automobile manufacturers must consider rust protection while designing their cars, all flash vendors must deal with garbage collection. While it is possible to create torture tests that can create a write-cliff, it rarely happens in a production environment as many competitors claim, hoping to make their product seem superior. This white paper has shown how a well-designed implementation of a well-engineered flash storage product can all but eliminate the effects of garbage collection on processing and storing production data and provide consistent performance for applications. The IBM FlashSystem product line has been engineered from the “ground up” to deal with garbage collection by properly using the most technologically advanced design and components. IBM’s FlashSystem sellers and Client Technical Specialists will ensure that the products deployed into your environment are the right fit and provide the best possible performance. All flash vendors must deal with garbage collection and we at IBM feel that the FlashSystem 840 and V840 products are the best at avoiding the write-cliff in production.

Important Reminders

IBM does not have a recommendation to limit the maximum utilization (capacity) of a FlashSystem 840 or V840. In most cases, it is possible to use all 40TB of a 40TB IBM FlashSystem 840.

In extreme cases with write patterns close to 100%, there may be trade-offs to capacity vs. performance, but this is very much the exception. Typical enterprise applications do not present any concern for maximum capacity.

It is very important to know the application I/O workload (including read/write mix and I/O block sizes). This information is critical to ensure that the IBM Sales Engineer or Business Partner can design the proper flash storage configuration.

Test the flash storage array with production workloads and actual data and I/O patterns, if possible.

IBM and our Business Partners have many years of flash storage expertise and the FlashSystem products are designed to handle garbage collection in the industry’s best manner.