PowerVault MD3 SSD Cache Overview

White Paper

Table of contents

1 Overview

With the introduction of Dell PowerVault 3.0 firmware, Dell PowerVault MD3 array series takes the next step in providing affordable performance through a new feature called “SSD cache. “

The traditional mainstay in storage technology has been the hard disk drive (HDD). However, while the capacity of HDDs has increased, their random input/output (I/O) performance has not increased at the same rate. This means that for some of today’s enterprise, web, cloud, and virtualized applications that require both high capacity and performance, HDDs may not deliver a cost-effective storage solution, even with their significant drop in cost per GB. SSDs offer exceptionally high performance but have much less capacity per drive. They are also relatively expensive when compared to HDDs, and have a write endurance limit.

Given the properties of HDDs and SSDs, IT departments now have a choice, but also a challenge, in determining the best way to cost-effectively fulfill the performance and capacity requirements of their enterprise applications. To meet this challenge and determine how they should integrate HDDs and SSDs into their storage fabric, IT departments must first quantify the performance, capacity, and cost value of SSDs vs. HDDs for different applications.

One of the best ways to approach this is to combine the benefits of both SSDs and HDDs and that is exactly what SSD Cache enables you to do. SSD cache enables the PowerVault MD3 array to use Solid State Disks (SSDs) as extended read only cache, thus increasing the performance of random read intensive applications such as file servers, web servers, and databases, etc.

In general SSD Cache is useful for applications with –

1. Read Performance bottlenecked due to HDD IOPS.
2. High Percentage of Reads compared to Writes
3. The Size of data that is repeatedly accessed (Working Set) is smaller than the size of SSD cache capacity.

SSD Cache is available as part of High Performance Tier Premium Feature Key on PowerVault MD3 Array Series.

2 Architecture

SSD cache is secondary cache used with primary cache to enable better performance. When SSD Cache is created, the cache is split into two internal RAID 0 virtual disks (one per controller). This cache volume is not available for regular data storage.

In duplex mode, each controller uses a separate virtual disk and manager or accesses only half of the SSD capacity, even if a controller fails or is in maintenance mode. In Simplex mode, one controller managers or accesses the entire SSD capacity and uses both virtual disks.

The maximum SSD cache size allowed by PowerVault MD3 arrays is 2TB.

3 SSD Cache Algorithm

Whenever host read or write, the SSD cache feature copies the data from HDD virtual disk to SSD virtual disk. Any subsequent host read of the same logical block addresses can be read directly from SSD with a much lower response time thus increasing the overall performance.

Since the data in SSD is a copy of the original data residing on HDD, a SSD virtual disk failure does not cause data loss.

SSD cache is divided into groups of sectors of equal sizes. Each group is called a cache block; each block is divided into sub-blocks. The metadata that resides in DRAM tracks the contents of each block and sub-block. The amount of DRAM used to manage the SSD cache for PowerVault MD3 arrays is approximately 200MB per controller.

3.1 Populating the Cache

Populating the cache is a background operation that typically immediately follows a host read operation or a host write operation. It is a simple read from the user HDD volume and a write to the SSD cache volume. By using volume reads and writes, the SSD cache leverages all of the applicable volume features, such as (primary) caching, data assurance, and full disk encryption, that are available to user volumes.

One of two parameters is used to determine when to start a cache-populate operation: populate-on-read threshold or populate-on-write threshold. Each cache block has associated read and write counts.

The read count is incremented each time a host read attempts to access a cache block to determine whether user data is present. If a cache miss occurs (for example, if at least one sector of data is missing in the SSD cache), the populate-on-read count is greater than zero, and the read count equals or exceeds the populate-on-read threshold, then a populate operation is scheduled concurrently with the host read of the base volume. If a cache hit occurs, then a populate operation is not performed.

The write count is incremented each time a host write attempts to access a cache block. If the populate-on-write count is greater than zero, and the write count equals or exceeds the populate-on-write threshold, then a populate operation is scheduled following the successful write to the base volume.

For workloads where a write is an indicator of a subsequent read, a nonzero populate-on-write threshold should be used to populate the cache.

These parameters are not directly selectable, but the user can select an I/O type that controls the populate-on-read and populate-on-write threshold when creating an SSD cache.

3.2 I/O Type

The I/O type is a user-selectable SSD cache configuration parameter. The user can specify an I/O type when creating an SSD cache or the user can change the I/O type on an existing SSD cache at any time, even while I/O is active. Changing the I/O type on an existing SSD cache causes all user data to be purged and caching to restart with an empty cache.

The SSD cache uses the I/O type selection to control certain internal configuration settings. The I/O type should be used as a guideline and does not imply an exact match with the actual usage of the base volumes. It is possible that an I/O type of database might obtain better performance for HDD volumes that contain file systems than an I/O type of file system. It might also be that the base volumes are of mixed usage. It is recommended to experiment with different settings to obtain optimal performance.

The user-selectable I/O type controls the SSD cache internal settings for cache block size, sub-block size, populate-on-read threshold, and populate-on-write threshold.

I/O Type	**Block Size (Sectors)**	**Sub-block Size (Sectors)**	**Populate-on-Read Threshold**	**Populate-on-Write Threshold**
Database	2,048	16	2	0
File system	4,096	32	2	2
Web server	8,192	128	2	0

Table 1: Pre-Defined Cache Settings per I/O Type

The block size generally affects the cache use and the warming time. Warming cache is the process of filling the cache for the first time. The cache use shows how much of the allocated cache actually holds user data.

The highest cache use is obtained when data that is frequently reread is located very close to other data that is frequently reread. Using an I/O type with a larger cache block size can be more beneficial to performance than a smaller cache block size. Cache fills at approximately the same rate for any cache block size with this type of data locality.

Conversely, when data that is frequently reread is located far from other data that is frequently reread, the lowest cache use is obtained. In this scenario, the I/O type with the lowest cache block size allows the most user data to be cached. Also, with this type of locality, cache fills at a faster rate with a larger cache block size than with a smaller cache block size. Even with low cache use, performance gains can still be achieved over HDD performance because the SSD cache size can be significantly larger than the primary DRAM cache size.

The sub-block size generally affects the cache warming time. A larger sub-block size causes cache to fill more quickly than a smaller sub-block size, but it can also affect host I/O response time when a controller is nearing one or more of its saturation points, such as CPU utilization, memory bandwidth, or channel utilization. Whether this is beneficial depends on the locality between blocks that are frequently reread. A very high locality of reference can benefit more from a larger sub-block size than from a smaller sub-block size, especially if those blocks that are reread frequently reside in the same sub-block. This occurs when one I/O causes the sub-block to be populated and another I/O in the same sub-block gets a cache hit.

Cache population is based on the starting LBA of the host read or the host write and the block count aligned to sub-block boundaries and sub-block lengths. A small I/O request can cause much more data to be populated than the actual size of the I/O, especially when an I/O spans cache block boundaries and sub-block boundaries. This can be beneficial if the additional blocks being populated are reread. However, if the additional blocks are never reread, it can be a waste of time and controller bandwidth resources.

3.3 I/O Handling

3.3.1 Host Read with Cache Miss

Figure 1 shows how the controller handles a host read request when some of the data is not in the SSD cache.

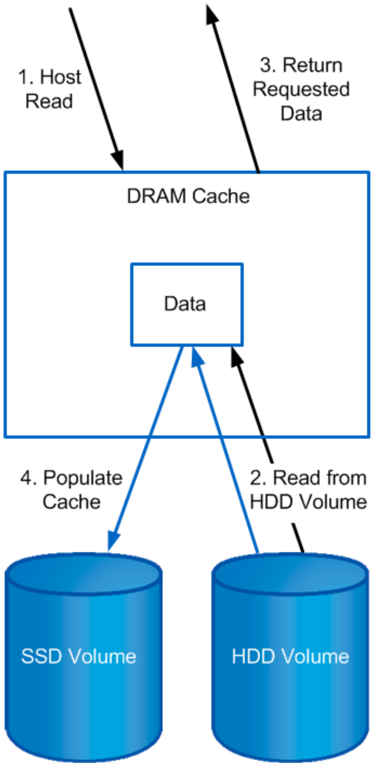


Figure 1: Host Read with Cache Miss

The following steps provide details about a host read with a cache miss:

1. Receive the host read.
    a. The SSD cache metadata is searched
    b. Any LBAs missing in the metadata indicate an SSD cache miss.
    c. If the read data is not to be cached, the host read is passed to the HDD volume (step d).
    d. If the read data is to be cached, the host read is passed to the HDD volume (step d) and a background cache populate operation is scheduled (step 3).
2. Read from the HDD volume.
    a. If the read is successful, the data and good status are returned to the host (step b).
    b. If there is a read error, the error is returned to the host.
3. Return requested data to the host.
4. Populate the cache.
    a. Data is read from the HDD volume and then is written to the SSD volume.

b. If successful, the sub-block bitmap in the DRAM metadata is updated to indicate that the LBAs are now present.
c. If an error occurs, the sub-block bitmap in the DRAM metadata is updated to indicate that the LBAs are missing.
d. In either case, no additional action is required.

3.3.2 Host Read with Cache Hit

Figure 2 shows how the controller handles a host read request when all of the data is in the SSD cache.

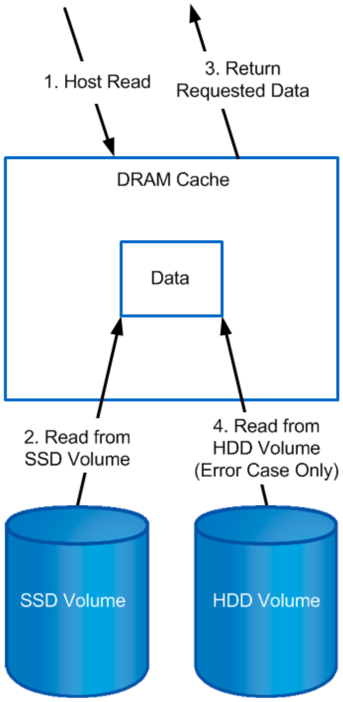


Figure 2: Host Read with Cache Hit

The following steps provide details about a host read with a cache hit:

1. Receive the host read.
   a. The SSD cache metadata is searched.
   b. All LBAs present in the metadata indicate an SSD cache hit.
   c. The translated host read is passed to the SSD volume (step c).
2. Read from the SSD volume.
   a. If the read is successful, the requested data is returned to the host (step b).
   b. If there is a read error, the sub-block bitmap in the DRAM metadata is updated to indicate that the LBAs are now missing and the host read is passed to the HDD volume (step 3).
3. Return the requested data and good status to the host.
4. Read from the HDD volume (error-handling case only)
   a. If the read is successful, the requested data and good status are returned to the host (step 2.b).
   b. If a read error occurs, the error is returned to the host.

3.3.3 Host Write

Figure 3 shows how the controller handles a host write request.

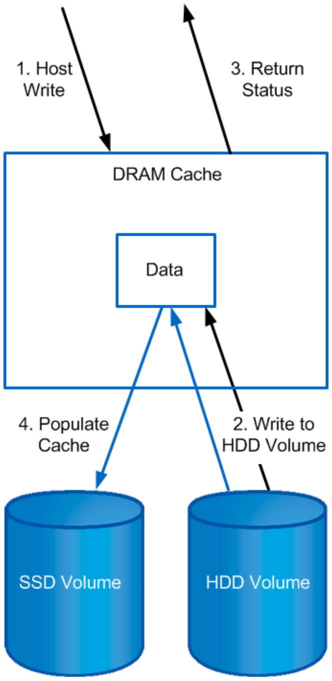


Figure 3: Host Write

The following steps provide details about a host write:

1. Receive the host write.
   a. The sub-block bitmap in the DRAM metadata is updated (prior to write) to indicate that the LBAs are now missing.
   b. Determine whether the write data must be cached.
2. Write to the HDD volume.
   a. If the write is successful, a good status is returned to the host (step b), and if the write data must be cached, a background cache populate operation is scheduled (step 3).
   b. If there is a write error, the error is returned to the host (step b). There is no update of the metadata and no change to the SSD cache state.
3. Return the status to the host.
4. Populate the cache.
   a. Data is read from the HDD volume and then is written to the SSD volume.
   b. These operations are performed only if write data is to be cached and the host write to the HDD volume is successful.
   c. If successful, the sub-block bitmap in the DRAM metadata is updated to indicate that the LBAs are now present.
   d. If an error occurs, the sub-block bitmap does not need to be updated.
   e. In either case, no additional action is required.

3.3.4 Host Verify

Figure 4 shows how the controller handles a host verify request.

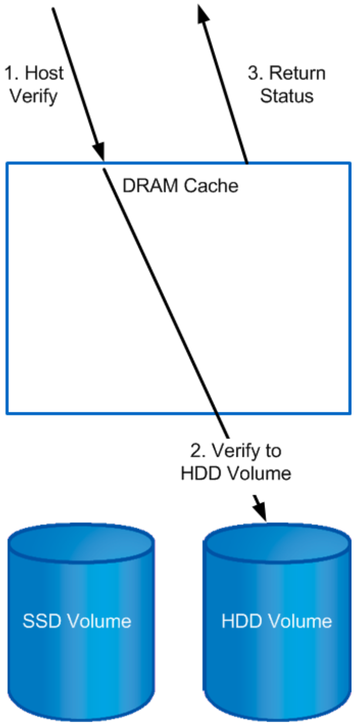


Figure 4: Host Verify

The following steps provide details about a host verify request:

1. Receive the host verify.
   a. The bitmap in the DRAM metadata is updated to indicate that the LBAs are now missing.
2. Send verify to the HDD volume.
   a. If successful, a good status is returned to the host (step b).
   b. If there is an error, the error is returned to the host. No additional changes are made to the bitmaps and the SSD cache state does not change.
3. Return the status to the host.

3.3.5 SCSI Disable Page Out Handling

Disable Page Out (DPO) is an option on the SCSI Read, SCSI Write, and SCSI Write and Verify commands. With the DPO option set, any of these commands causes the cache blocks to be removed from the SSD cache after the command completes.

3.3.6 SCSI Force Unit Access Handling

Force Unit Access (FUA) is an option on the SCSI Read and SCSI Write commands. With the FUA option set, any of these commands causes the cache to be invalidated. In that case, the sub-block bitmap in the

DRAM metadata is updated to indicate that the LBAs are now missing and the cache is not repopulated. For host read requests, this effectively forces a cache miss and the host read is passed to the HDD volume.

3.3.7 Sequential Streams

Data from sequential reads or writes is not written to the SSD cache.

3.3.8 Data Assurance (T10/PI) Support

SSD cache is automatically data assurance (DA) enabled if all of its SSDs are DA capable and the DA feature is enabled. If DA is enabled on an SSD cache, it cannot be disabled. Drives that are not DA capable cannot be added to a DA-enabled SSD cache.

A DA-enabled base volume cannot be enabled for SSD caching when the SSD cache is not DA enabled, because there is no place for DA fields in the SSD.

If a base volume is enabled for DA and the SSD cache volume is enabled for DA, only the guard field of the protection information (PI) is checked when reading from or writing to the SSDs. If a volume is not enabled for PI, none of the PI fields are checked when reading from or writing to the SSDs.

Error! Reference source not found. shows which PI fields are checked for volumes with and without DA enabled.

	SSD Cache with DA Enabled	SSD Cache without DA Enabled
Volume enabled for DA	Guard field only	Not supported
Volume not enabled for DA	No PI fields	No PI fields

Table 2: Protection Information matrix with and without Data Assurance enabled

3.3.9 Full Disk Encryption Drive Support

SSD cache with full disk encryption (FDE) capability is not supported in PowerVault Firmware 3.0 because FDE-capable SSDs are not available.

4 Performance Modeling Tool

The SSD cache feature contains a built-in performance modeling tool. When run with any one of a user's multiple workloads, it can help the user determine the relative performance improvements for various cache sizes up to the maximum supported SSD cache capacity for that workload. The tool provides an estimate of performance with two metrics:

1. SSD cache hit percentage
2. Average latency

The tool shows the actual performance for an installed cache, and by using software modeling techniques, it estimates the performance for a variety of cache sizes. The estimated results are close to the actual results that could be achieved with an SSD cache of that size.

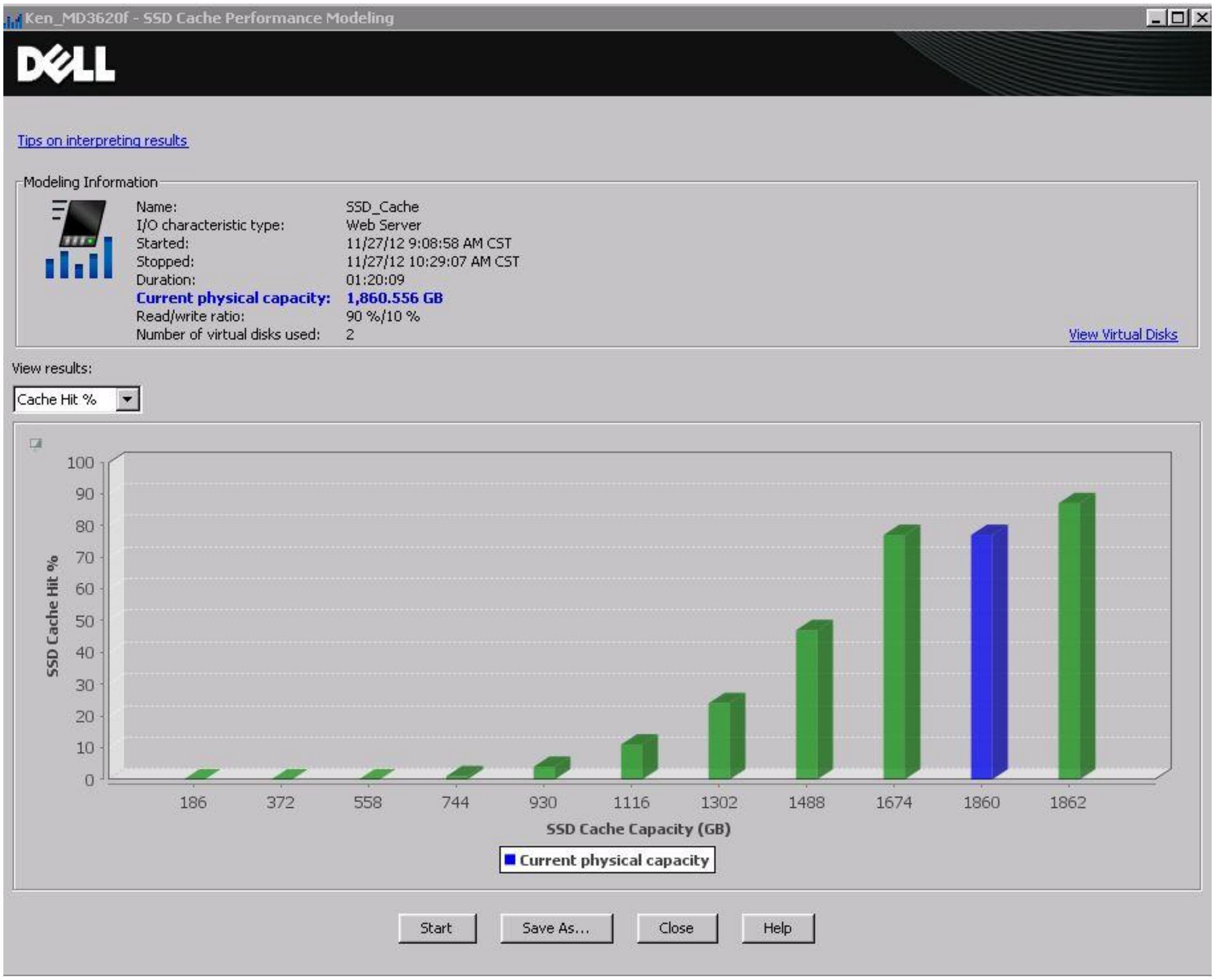
NOTE: When operating closer to the maximum performance of the controllers, the predictions might be less accurate.

The latency chart uses calculated latencies from the operation of the SSD cache to estimate the time it takes to execute external and internal I/O operations. The tool uses these latency measurements and the measurements from the I/O operations performed during the run of the user's workload to estimate the average latency for external I/O operations.

NOTE: The latency chart values are estimates and should be used to understand the relative performance of various cache sizes. They should not be used as absolute measurements for quality of service or other purposes.

Depending on the cache capacity and workload, it might take several hours to fully populate the cache. Valid information is available even after a run of a few minutes, but the modeling tool should be allowed to run for several hours to obtain the most accurate predictions.

Figure 5 shows the actual cache hit percentage (represented by the blue bar) and the predicted cache hit percentages for different cache sizes (represented by the green bars).



Ken_MD3620f - SSD Cache Performance Modeling

Tips on interpreting results

Modeling Information

Name: SSD_Cache
I/O characteristic type: Web Server
Started: 11/27/12 9:08:58 AM CST
Stopped: 11/27/12 10:29:07 AM CST
Duration: 01:20:09
Current physical capacity: 1,860.556 GB
Read/write ratio: 90 %/10 %
Number of virtual disks used: 2

View Virtual Disks

View results: Cache Hit %

Figure 5: Performance modeling tool (cache hits)

The point at which the bars in the cache hit percentage view flatten out — that is, when they reach a maximum value and are the same for all subsequent bars — is the point at which the working set size (WSS) of the data fits in the capacity of the SSD cache. If this occurs, it indicates that there is no added gain in using a cache capacity that is larger than the first bar where the bars flatten out. This applies only to the workload, the SSD cache configuration selections, and the duration for which the SSD cache performance modeling tool was allowed to run. Changing any of these variables might change the results. Configuration selections consist of the following items:

1. I/O type (file system, database, or Web server)
2. Set of volumes enabled for SSD caching
3. Number of SSDs - More SSDs might perform better than fewer SSDs.

If the bars in the cache hit percentage view rise but never flatten out, that might indicate that the working set size of the data is greater than the size of the SSD cache. It might also indicate that the SSD cache is thrashing and that the workload is not favorable for use with the SSD cache.

Thrashing occurs when data is constantly copied from the base volume into the SSD cache but is not accessed again until it is cleared from the SSD cache. This happens because all SSD cache blocks have been allocated and another cache block must be allocated for the same (or a different) volume and/or LBA.

Numerous statistics are available that can be used to monitor SSD cache operation. The recycle actions statistic can be used to determine whether cache thrashing is occurring. This and other statistics and their interpretation are described in section 4.1, "Cache Statistics."

If only one read occurs into a cache block, the SSD cache process does not populate the SSD cache. Therefore a purely random read workload does not cause thrashing or unnecessary overhead with any working set size.

Thrashing can be reduced or eliminated by decreasing the working set size. If numerous applications are running concurrently during an SSD cache performance modeling tool run, it might be beneficial to run the applications serially, if possible. Running the applications serially might be faster than running them concurrently, because the WSS of the individual applications might fit in the SSD cache, whereas the WSS of the combined applications might not fit in the SSD cache.

The SSD cache performance modeling tool does not currently take into consideration controller limitations, such as maximum IOPS or bandwidth. Therefore increasing the working set size by running more applications concurrently might not offer any additional performance benefits.

4.1 Cache Statistics

Table 3 describes the statistics that are available from the SSD cache.

Statistic	Description
Timestamp	Date and time statistics sample taken.
Reads	Total number of host reads to SSD cache-enabled volumes.
Read blocks	Number of blocks in reads.
Writes	Total number of host writes to SSD cache-enabled volumes.
Write blocks	Number of blocks in writes.
Full cache hits	Total number of host reads to SSD cache-enabled volumes that were satisfied from the SSD cache.
Full cache hit blocks	Number of blocks in full cache hits.
Partial cache hits	Number of host reads in which at least one block, but not all blocks, were

	in the SSD cache. This is an SSD cache miss where the reads were satisfied from the user volume.
Partial cache hit blocks	Number of blocks in partial cache hits.
Complete cache misses	Number of host reads in which none of the blocks were in the SSD cache. This is an SSD cache miss in which the reads were satisfied from the user volume. It is expected that there will be a larger number of partial hits and misses as compared to cache hits while the SSD cache is warming.
Complete cache miss blocks	Number of blocks in complete cache misses.
Populate on reads	Number of host reads in which data was copied from the user volume to the SSD cache.
Populate on read blocks	Number of blocks in populate on reads.
Populate on writes	Number of host writes where data was copied from the user volume to the SSD cache.
Populate on write blocks	Number of blocks in populate on writes.
Invalidate actions	Number of times that data was invalidated and removed from the SSD cache. A cache invalidate operation is performed for every host write request, every host read request with FUA, and every verify request.
Recycle actions	Number of times that an SSD cache block has been reused for another user volume and/or a different LBA range.
Available bytes	Number of bytes available in the SSD cache for use by this controller.
Allocated bytes	Number of bytes allocated from the SSD cache by this controller. Bytes allocated from the SSD cache can be empty; or they might contain data from user volumes.
Populated clean bytes	Number of allocated bytes in the SSD cache that contain data from user volumes.

Table 3: SSD Cache Statistics

Table 4 describes the information that can be derived from the statistics.

Information	Calculation	Description
Cache hit percentage	Full cache hits/reads	Percentage of host reads to SSD cache-enabled volumes that were satisfied from the SSD cache.

Cache allocate percentage	Allocated bytes/available bytes	Amount of SSD cache storage that is allocated, expressed as a percentage of the SSD cache storage that is available to this controller.
Cache utilization percentage	Populated clean bytes/allocated bytes	Amount of SSD cache storage that contains data from enabled volumes, expressed as a percentage of SSD cache storage that is allocated. This value represents the utilization or density of the SSD cache.

Table 4: Information that can be derived from statistics

4.2 Interpreting Statistics

The SSD cache warming can take many hours, depending on the workload and the SSD cache size. The interpretation advice in this section is for a warmed cache. The interpretations might be different for a cache in the warming process. An indicator of a warm cache is cache allocation equal to 100% or cache allocation that remains at a stable value less than 100% during several readings that are spaced approximately 10 minutes apart.

Compare the reads relative to the writes. The reads must be greater than the writes for effective SSD cache operation. The greater the ratio of reads to writes, the better the cache operates.

The cache hit percentage should be greater than 50% for effective SSD cache operation. A small percentage could indicate many things, including:

1. Ratio of reads to writes is too small
2. Reads are not repeated
3. Cache capacity is too small

Cache allocation percentage is normally 100%. If the number is less than 100%, it means that either the cache has not been warmed or the SSD cache is larger than all of the data being accessed. In the latter case, a smaller SSD cache could provide the same level of performance.

This does not indicate that cached data has been placed into the SSD cache.

Cache utilization percentage is normally lower than 100%, perhaps much lower. This number shows the percent of SSD cache space that is filled with cache data. This number is typically lower than 100% because each allocation unit of the SSD cache (the SSD cache block) is divided into smaller units, the sub-blocks that are filled somewhat independently. A higher number is generally better, but performance gains can be significant even with a smaller cache utilization percentage.

The SSD cache is beneficial to performance only for those operations that are full cache hits. Partial cache hits are the result of an operation that has only a portion of its data in the SSD cache. In this case, the operation must get the data from the HDD volume. The SSD cache offers no performance benefit for this type of hit. If the partial cache hit blocks count is higher than the full cache hit blocks, then a different cache configuration setting might improve the performance by changing the manner in which data is loaded into the cache.

The populate-on-write threshold might be zero for the cache configuration settings that do not fill the cache as a result of a write I/O operation.

For effective cache operation, it is important that the number of recycle actions is small compared to the combined number of read and write operations. If the number of recycle actions is close to the combined number of read and write operations, then the SSD cache will be thrashing. Either the cache size must be increased or the workload is not cacheable.

5 Performance

As part of our test setup we tried to show the performance improvements you get using SSD cache. We used two set ups -

	Setup 1	Setup 2
Virtual Disk 1 (Size)	800GB	
Virtual Disk 2 (Size)	800GB	
Hard Drives	18 * 15 K SAS	18 * 7.2K NL-SAS
SSDs	5 * 400GB SLC 6Gb SAS SSD	

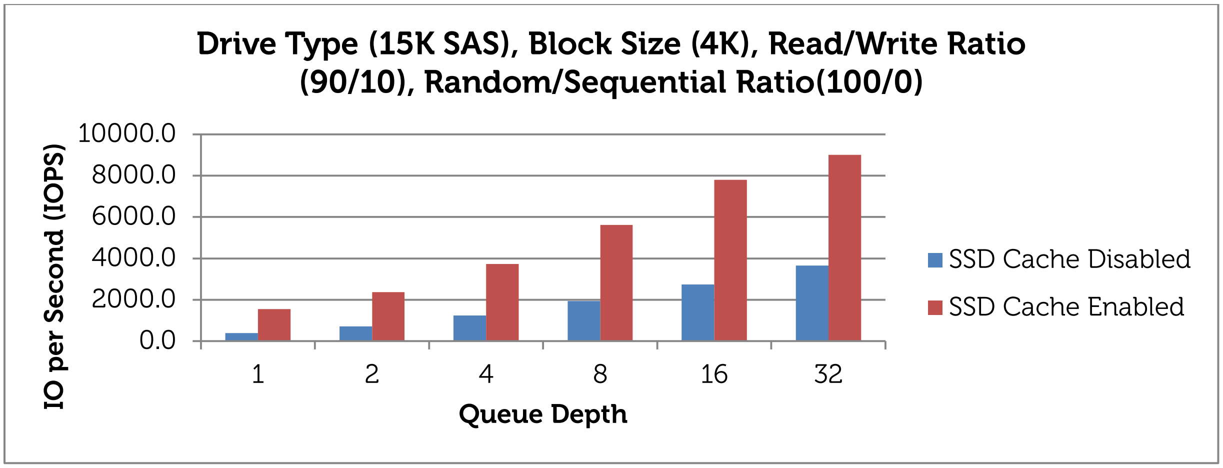


Figure 5: Impact of SSD Cache on a system using 15K SAS Drives.

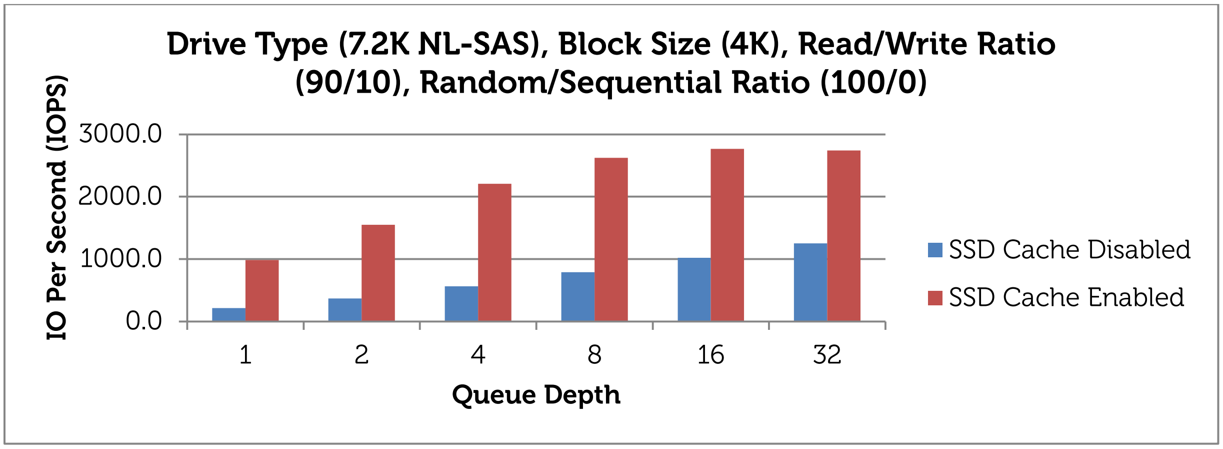


Figure 6: Impact of SSD Cache on a system using 7.2K NL-SAS Drives.

As you can see from the charts above, with SSD cache enabled, the IOPS have improved by as much as 363% in some instances.

6 Recommendations

From our testing we found that workloads with the following characteristics benefit from the SSD cache feature:

1. Read performance limited by HDD IOPS
2. High percentage of reads relative to writes
3. Working size set is smaller than the SSD cache capacity

It is recommended to use SSD Cache if a balanced, cost-effective approach that uses a mix of HDDs and an SSD cache is desired and the cost of dedicated SSDs for RAID volumes is prohibitive.

SSD cache is not an effective option if:

1. Workloads are write intensive
2. Workloads are sequential read intensive
3. Working set size is larger than the SSD cache capacity
4. Read workloads have a high percentage of read-once data (no repeat reads)