Technical Report

NetApp SANtricity SSD Cache for E-Series

Don Humlicek, NetApp
September 2012 | TR-4099

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 Overview

The NetApp® E-Series SANtricity® solid state drive (SSD) cache feature, available in NetApp SANtricity 10.84 software, uses SSD storage to hold frequently accessed data from user volumes. It is intended to improve the performance of workloads that are performance limited by hard disk drive (HDD) input/output operations per second (IOPS). Workloads with the following characteristics can benefit from using the SANtricity SSD cache feature:

- Read performance is limited by HDD IOPS.
- There is a high percentage of read operations relative to write operations.
- A large number of reads are repeat reads to the same or adjacent areas of disk.
- The size of the data that is repeatedly accessed is smaller than the SSD cache capacity.

2 Architecture

The SANtricity SSD cache feature uses a set of SSDs that are logically grouped together in the storage array. SSD cache is a secondary cache that is used with the primary cache in the controller's dynamic random-access memory (DRAM). In controller cache, the data is stored in DRAM after a host read. In SSD cache, the data is copied from user-specified base volumes and stored on two internal RAID 0 volumes (one per controller) that are automatically created when an SSD cache is created. These volumes are used for internal cache processing and are not accessible or displayed in the user interface.

In duplex mode, each controller uses a separate volume. Each controller manages or accesses only half of the SSD capacity, even if a controller fails or is in maintenance mode. In simplex mode, one controller manages or accesses the entire SSD capacity and uses both volumes.

Figure 1) SSD cache internal volumes.

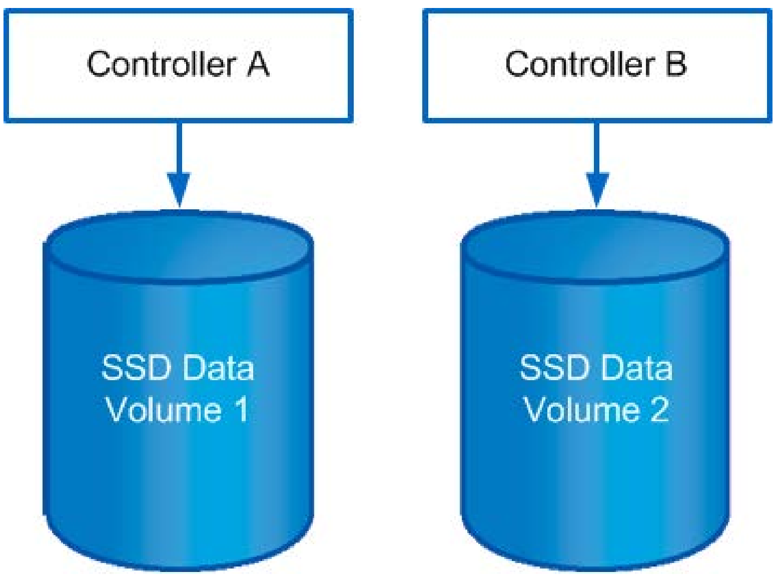


Table 1 lists the maximum supported SSD cache capacity per array. It is based on the size of the DRAM installed on a controller.

Table 1) Maximum supported SSD capacities.

Installed Memory Size (Per Controller)	Maximum Supported SSD Capacity (Per Array)
1GB (binary)	1TB (decimal)
2GB	2TB
4GB	4TB
Greater than 4GB	5TB

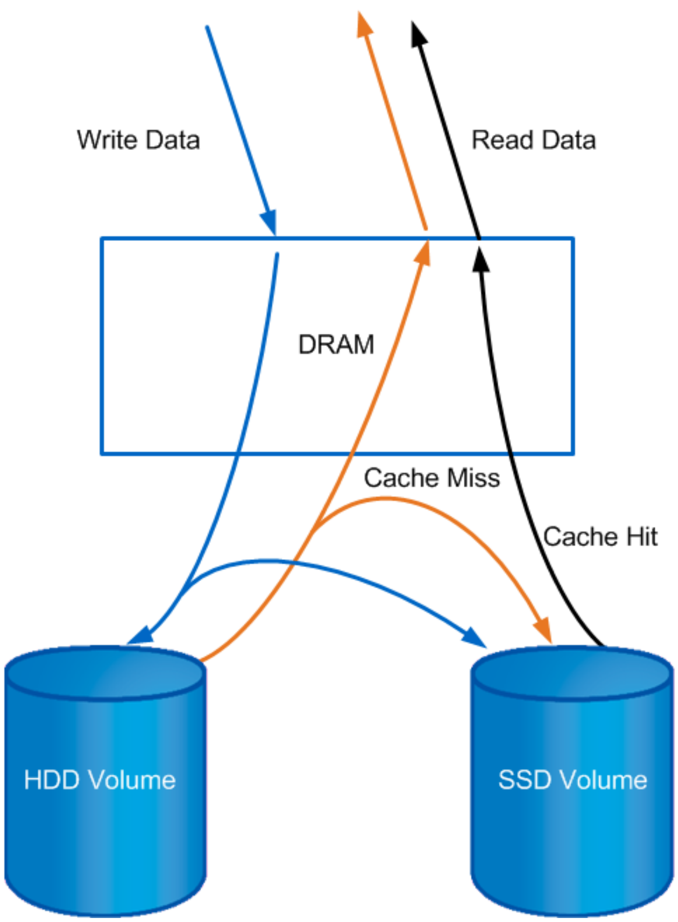
In duplex mode, the maximum supported capacity is split between each controller. In simplex mode, one controller uses the entire maximum supported capacity. Any installed SSD capacity that is greater than the maximum supported capacity is not used.

2.1 Theory of Operation

Following a host read or write, the SSD cache feature copies data from an HDD volume to its SSD volume. A subsequent host read of the same logical block addresses (LBAs) can be read directly from the SSD volume with a much lower response time than if the data were reread from the HDD volume.

Because the data in the SSD cache is only a copy of the data from the HDD volume, any failure of the SSD volume, or any failure while reading from or writing to the SSD volume, does not cause data loss.

Figure 2) General operation.

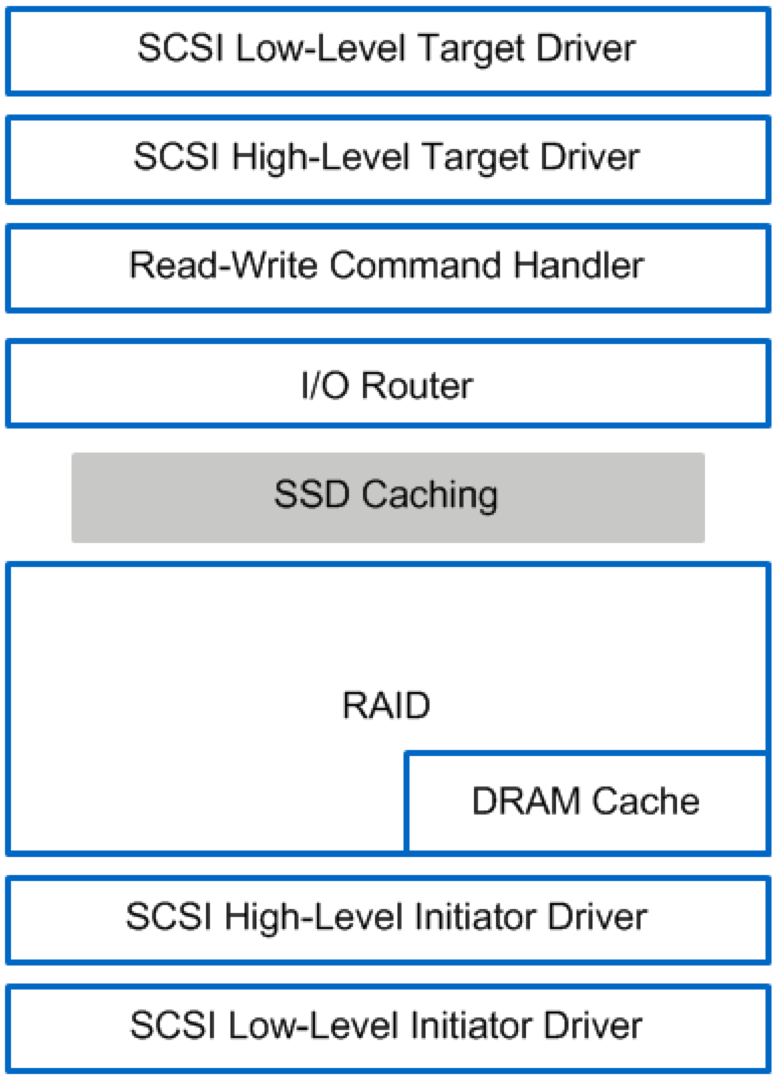


2.2 I/O Stack

The SSD caching resides at a level above the RAID volumes so that it can leverage RAID data protection methods and use a common data movement interface to move data in and out of the SSD cache.

Figure 3 illustrates the I/O stack in the controller.

Figure 3) I/O stack.



2.3 Cache Block and Subblock Sizes

The SSD capacity is divided into groups of sectors of equal sizes. Each group is called a *cache block*. This is similar to the way the primary DRAM cache is divided into cache blocks.

Each cache block is divided into subblocks. A subblock is the smallest amount of storage managed by the SSD cache. Metadata that is maintained in DRAM tracks the contents of each cache block and subblock.

To keep the DRAM metadata memory at a reasonable amount, the cache block sizes and subblock sizes of the SSD cache are much larger than the size of the primary cache.

Table 2 compares the cache block sizes and subblock sizes of the primary DRAM cache and the SSD cache.

Table 2) Cache block and subblock sizes.

	Primary DRAM Cache	SSD cache
Cache block size (sectors)	8, 16, 32, and 64	2,048, 4,096, and 8,192
Subblock size (sectors)	1	16, 32, 64, and 128

2.4 DRAM Usage

The SSD cache feature uses DRAM to manage the SSD cache. The amount of DRAM used to manage the SSD cache varies with the maximum supported SSD cache capacity. Table 3 lists these values.

Table 3) DRAM usage.

Maximum Supported Capacity (Per Array)	Approximate DRAM Usage (Per Controller)
1TB (decimal)	100MB (binary)
2TB	200MB
4TB	400MB
5TB	500MB

This DRAM is always allocated on controllers with firmware that supports the SSD cache feature, whether or not the feature is actually used. DRAM used for the SSD cache feature reduces the primary cache size by an equivalent amount.

2.5 Populating the Cache

Populating the cache is the term used to describe the operation of copying data from an HDD volume to the SSD 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.

2.6 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. NetApp recommends experimenting with different settings to obtain optimal performance.

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

Table 4) I/O types.

I/O Type	Block Size (Sectors)	Subblock 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

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 subblock size generally affects the cache warming time. A larger subblock size causes cache to fill more quickly than a smaller subblock 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 subblock size than from a smaller subblock size, especially if those blocks that are reread frequently reside in the same subblock. This occurs when one I/O causes the subblock to be populated and another I/O in the same subblock 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 subblock boundaries and subblock 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 subblock 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.

Figure 4 shows the effect of different subblock sizes as the cache is filling. The workload that produced this example is one in which the working set size fits in the SSD cache, and every block is reread repeatedly. With this workload, the cache block size does not matter.

Note: This workload is highly unlikely in an end-user environment.

Figure 4) IOPS during cache warming.

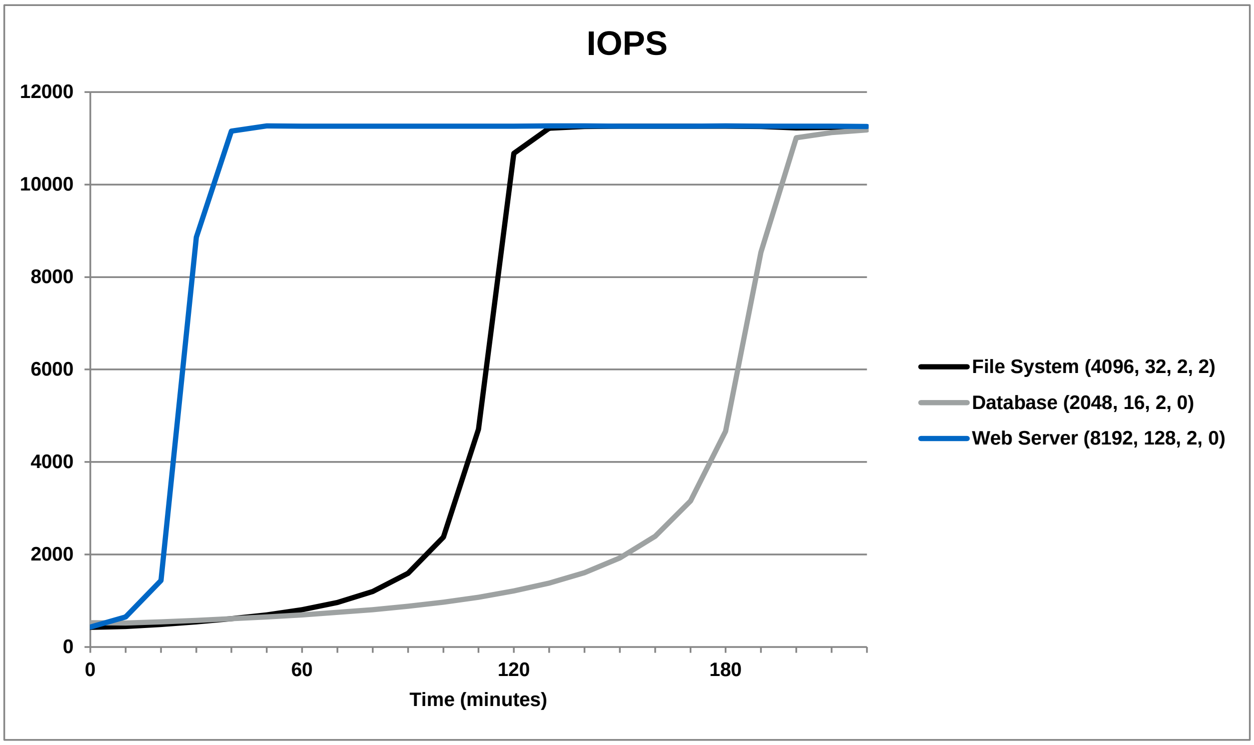
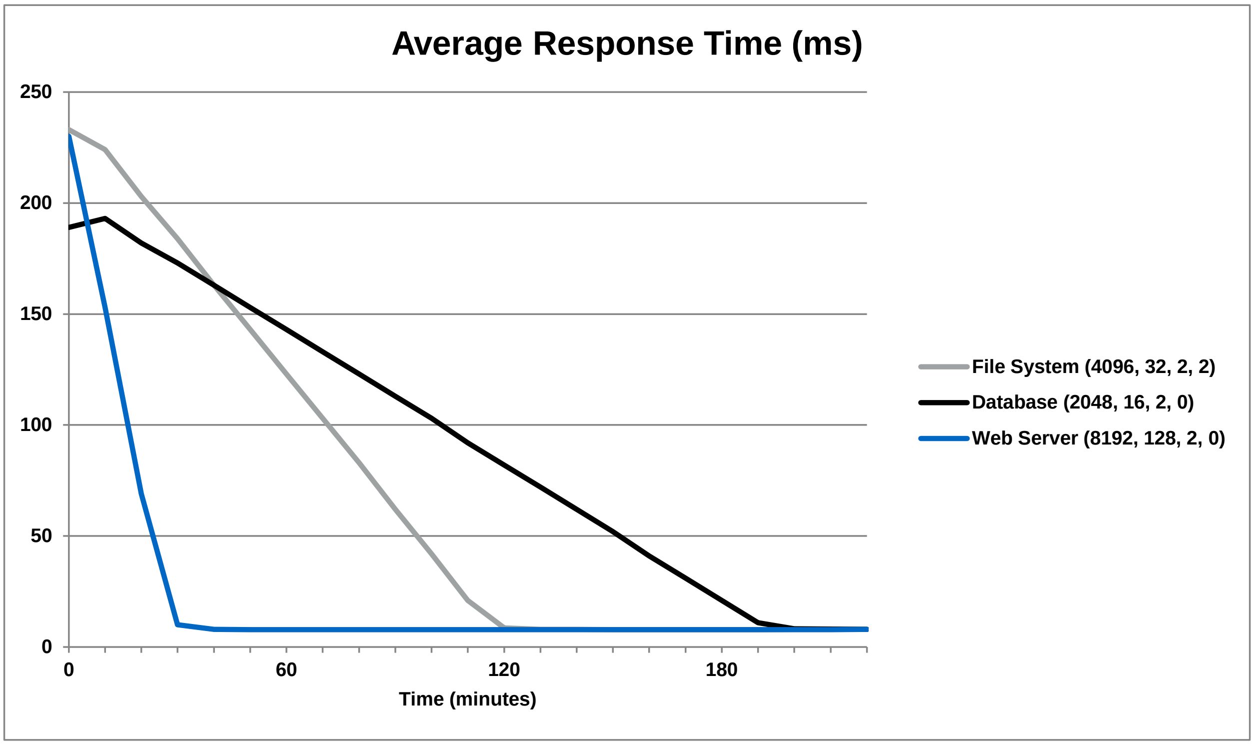


Figure 5 shows the host read response time for the same workload as cache is warming. With this workload, the faster the cache fills, the sooner the host read requests obtain cache hits and the lower the response times.

Figure 5) Average response time during cache warming.

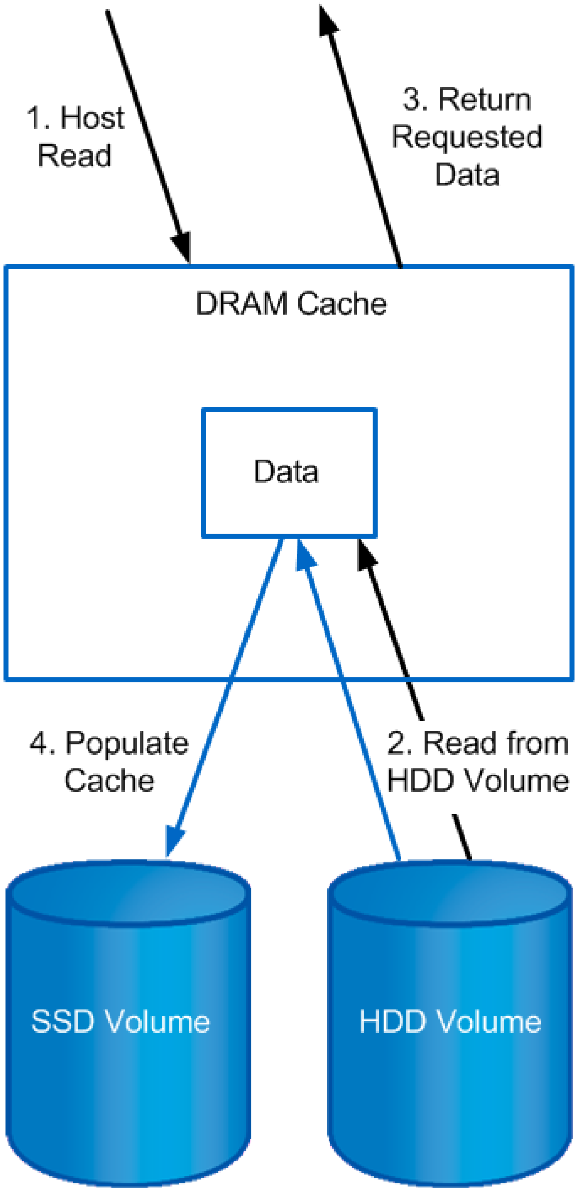


2.7 I/O Handling

Host Read with Cache Miss

Figure 6 shows how the controller handles a host read request when some of the data is not in the SSD cache.

Figure 6) Host read with cache miss.



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

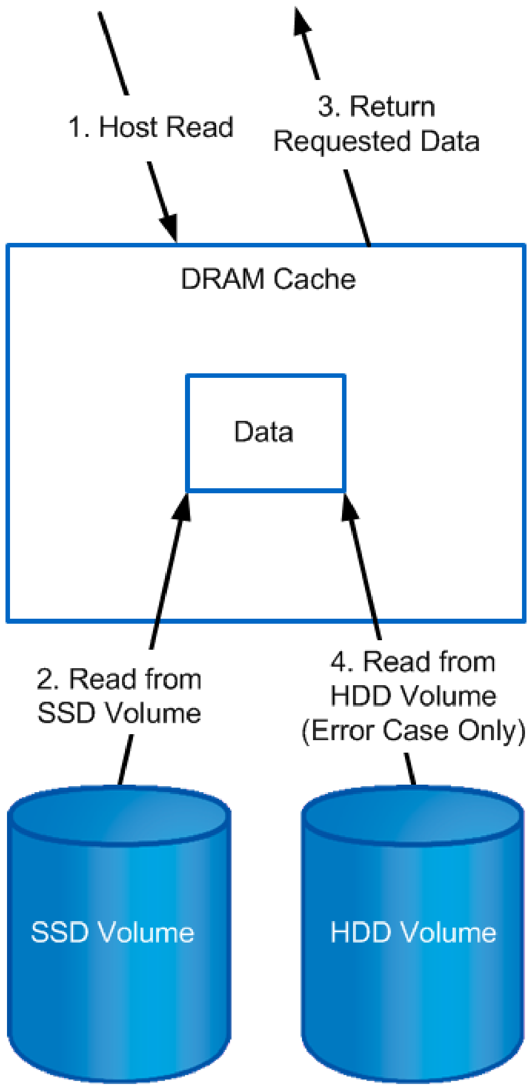
1. Receive the host read.
   - The SSD cache metadata is searched.
   - Any LBAs missing in the metadata indicate an SSD cache miss.
   - If the read data is not to be cached, the host read is passed to the HDD volume (step 2).
   - If the read data is to be cached, the host read is passed to the HDD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

4. Populate the cache.
   - Data is read from the HDD volume and then is written to the SSD volume.
   - If successful, the subblock bitmap in the DRAM metadata is updated to indicate that the LBAs are now present.
   - If an error occurs, the subblock bitmap in the DRAM metadata is updated to indicate that the LBAs are missing.
   - In either case, no additional action is required.

Host Read with Cache Hit

Figure 7 shows how the controller handles a host read request when all of the data is in the SSD cache.

Figure 7) Host read with cache hit.



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

1. Receive the host read.
   - The SSD cache metadata is searched.
   - All LBAs present in the metadata indicate an SSD cache hit.

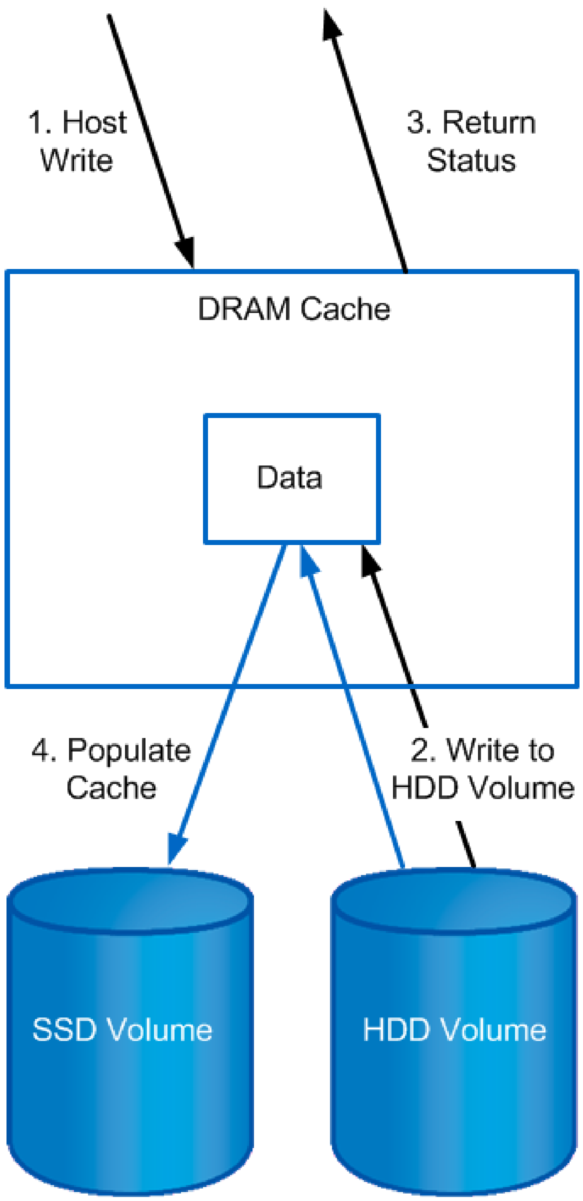
- The translated host read is passed to the SSD volume (step 2).

2. Read from the SSD volume.
   - If the read is successful, the requested data is returned to the host (step 3).
   - If there is a read error, the subblock 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 4).
3. Return the requested data and good status to the host.
4. Read from the HDD volume (error-handling case only)
   - If the read is successful, the requested data and good status are returned to the host (step 3).
   - If a read error occurs, the error is returned to the host.

Host Write

Figure 8 shows how the controller handles a host write request.

Figure 8) Host write.



The following steps provide details about a host write:

1. Receive the host write.
   - The subblock bitmap in the DRAM metadata is updated (prior to write) to indicate that the LBAs are now missing.
   - Determine whether the write data must be cached.
2. Write to the HDD volume.
   - If the write is successful, a good status is returned to the host (step 3), and if the write data must be cached, a background cache populate operation is scheduled (step 4).
   - If there is a write error, the error is returned to the host (step 3). 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.
   - Data is read from the HDD volume and then is written to the SSD volume.

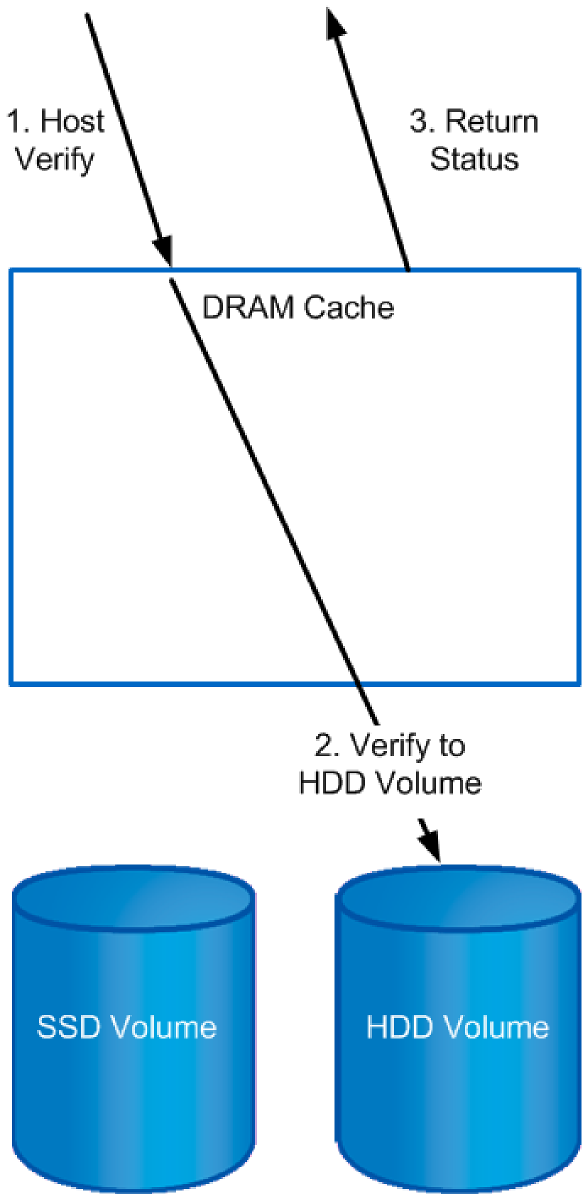
   Note: These operations are performed only if write data is to be cached and the host write to the HDD volume is successful.

   - If successful, the subblock bitmap in the DRAM metadata is updated to indicate that the LBAs are now present.
   - If an error occurs, the subblock bitmap does not need to be updated.
   - In either case, no additional action is required.

Host Verify

Figure 9 shows how the controller handles a host verify request.

Figure 9) Host verify.



The following steps provide details about a host verify request:

1. Receive the host verify.
   - The bitmap in the DRAM metadata is updated to indicate that the LBAs are now missing.
2. Send verify to the HDD volume.
   - If successful, a good status is returned to the host (step 3).
   - 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.

2.8 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.

2.9 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 subblock 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.

2.10 Sequential Streams

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

2.11 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.

Table 5 shows which PI fields are checked for volumes with and without DA enabled.

Table 5) Protection information field checks.

	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

2.12 Full Disk Encryption Drive Support

SSD cache with full disk encryption (FDE) capability is not supported in SANtricity 10.84 because FDE-capable SSDs are not available.

3 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:

- SSD cache hit percentage
- Average latency

The performance modeling tool works with an existing SSD cache in SANtricity 10.84.

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.

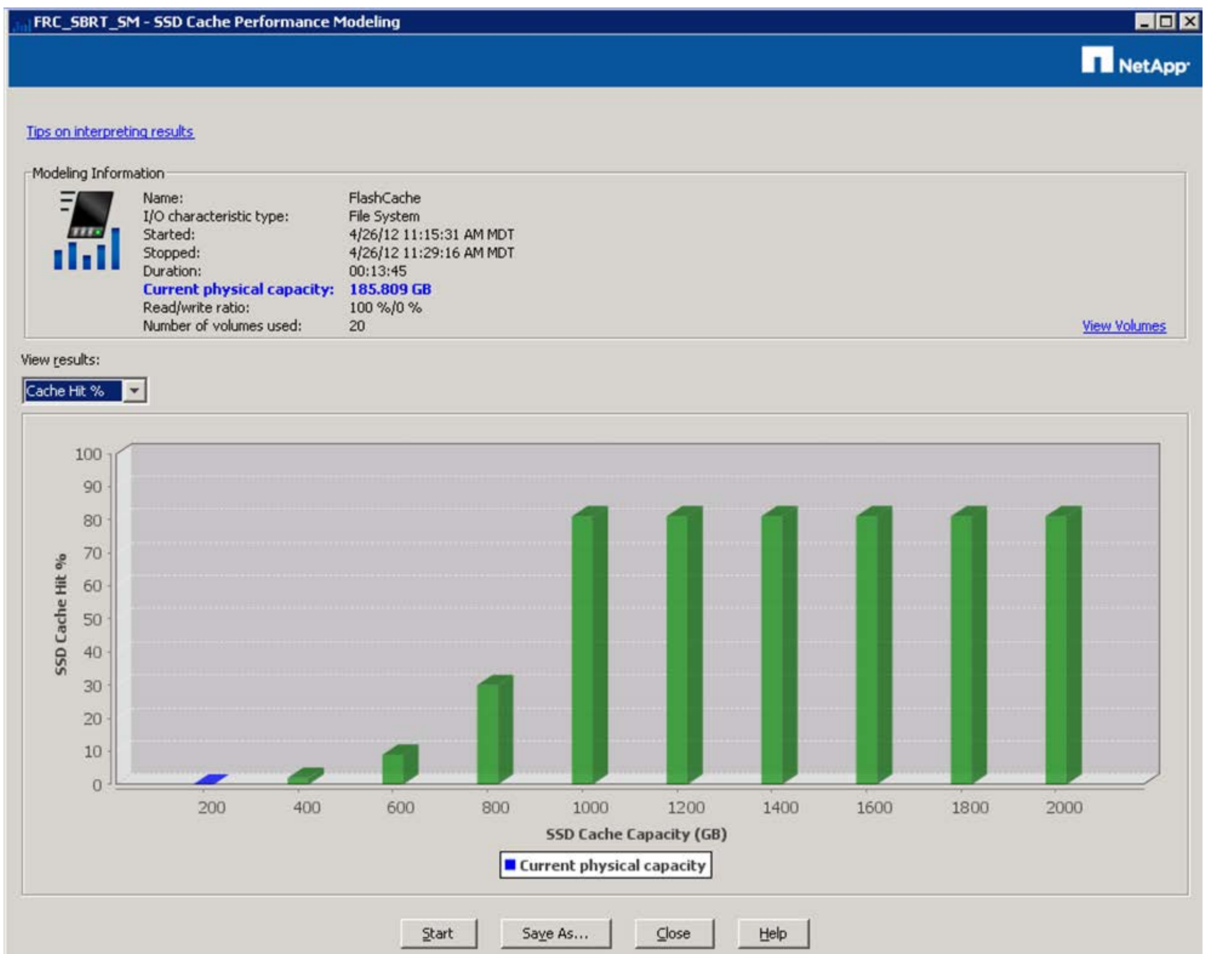
The latency chart uses calculated latencies from the operation of the SSD cache to estimate the time it takes to execute external and internal I/O operations. The tool uses these latency measurements and the measurements from the I/O operations performed during the run of the user's workload to estimate 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 10 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).

Figure 10) 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:

- I/O type (file system, database, or Web server)
- Set of volumes enabled for SSD caching
- Number of SSDs

  Note: 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, “Cache Statistics.”

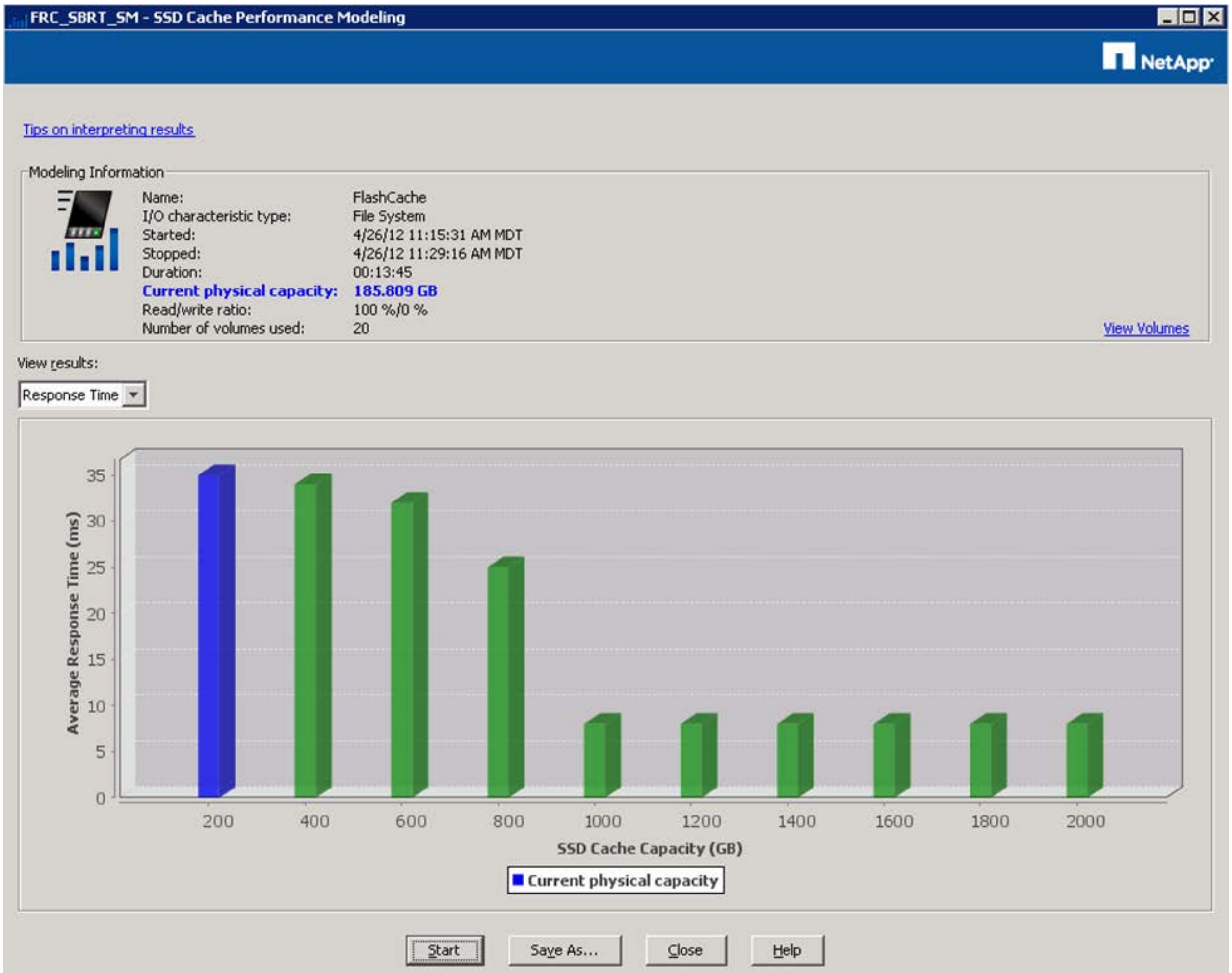
If only one read occurs into a cache block, the SSD cache process does not populate the SSD cache. Therefore a purely random read workload does not cause thrashing or unnecessary overhead with any working set size.

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.

Figure 11 shows the actual average latency (the blue bar) and the predicted average latencies for different cache sizes (the green bars).

Figure 11) Performance modeling tool: average latencies.



4 Cache Statistics

Table 6 describes the statistics that are available from the SSD cache.

Table 6) SSD cache statistics.

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.

Statistic	Description
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 7 describes the information that can be derived from the statistics.

Table 7) Information derived from 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.

Information	Calculation	Description
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.

4.1 Interpreting Statistics

Note: 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:

- Ratio of reads to writes is too small
- Reads are not repeated
- 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.

Note: 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 subblocks 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 Capabilities

The following workloads or benchmarks were run on an E2600 system and an E5400 system:

- Web server
- File server
- SPC-1
- Database (sysbench)
- Microsoft® Jetstress
- NetApp's standard saturation benchmarks

The following configurations were tested:

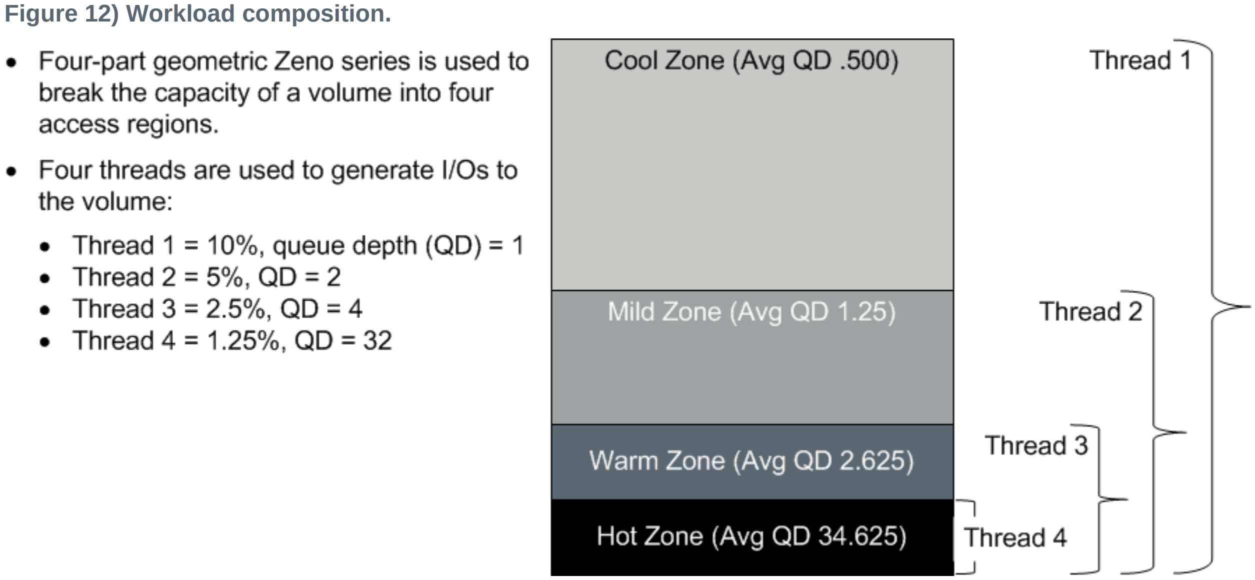
- E2600
  - 2GB memory per controller
  - SAS host ports
  - HDDs: 10 x 500GB 7.2K NL SAS
  - SSDs: 2 x 200GB 2.5" MLC 6 Gb/s SAS
- E5400
  - 12GB memory per controller
  - Fibre Channel (FC) host ports
  - HDDs: 100 x 300GB 15K7 SAS
  - SSDs: 20 x 150GB 3.5" SLC 3 Gb/s SAS

Figure 13 through Figure 18 show examples of Web server, file server, and database workloads running on E2600 and E5400 controllers that were configured according to the specifications listed in this section. Each workload was run four times, once without an SSD cache and once for each I/O type (Web server, file server, and database). Performance measurements were captured at one-hour intervals to show the difference between using an SSD cache and not using an SSD cache with each of the I/O types. The performance differences are the result of the workloads and the time it takes to warm the cache.

5.1 Web Server, File Server, and Database Workload Composition

Figure 12 shows the composition of the Web server, the file server, and the database workload.

Figure 12) Workload composition.



5.2 Web Server Workload

The Web server workload consists of 100% random reads. Table 8 shows the breakdown of I/O sizes.

Table 8) Web server workload I/O size breakdown.

I/O Size	Percentage of Workload
0.5KB	22%
1KB	15%
2KB	8%
4KB	23%
8KB	15%
16KB	2%
32KB	6%
64KB	7%
128KB	1%
512KB	1%

Figure 13 and Figure 14 show sample Web server workloads for an E2600 controller and an E5400 controller, respectively.

Figure 13) E2600 performance with Web server workload.

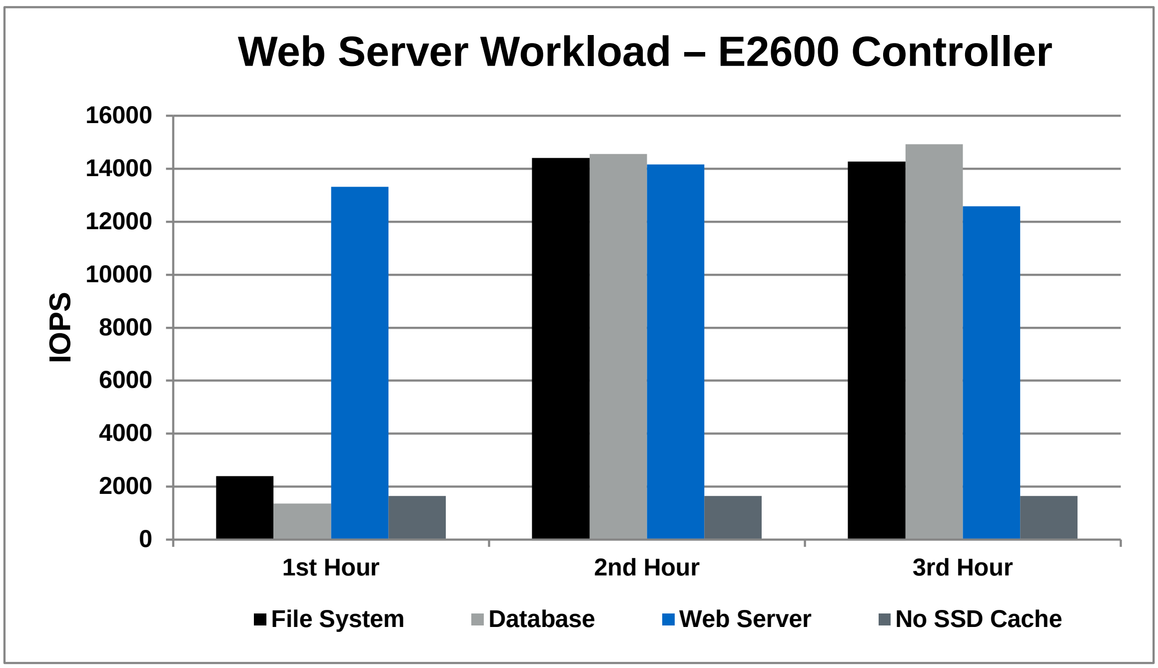
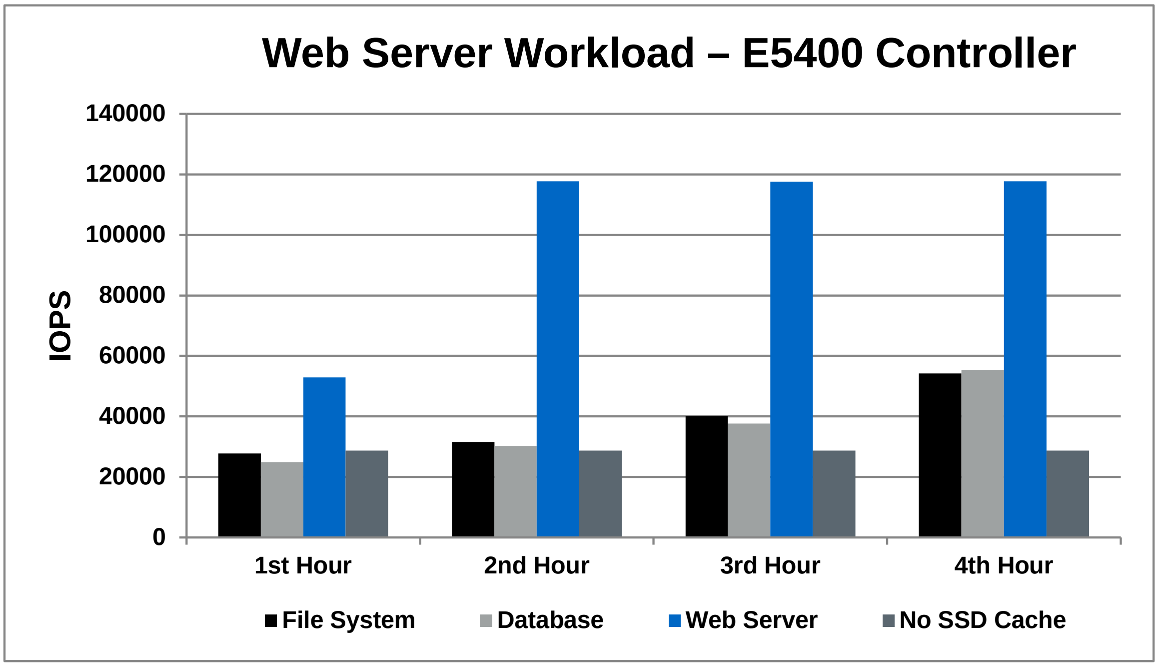


Figure 14) E5400 performance with Web server workload.



5.3 File Server Workload

The file server workload consists of 80% random reads and 20% random writes. Table 9 shows the breakdown of I/O sizes.

Table 9) File server workload I/O size breakdown.

I/O Size	Percentage of Workload
0.5KB	10%
1KB	5%
2KB	5%
4KB	60%
8KB	2%
16KB	4%
32KB	4%
64KB	10%

Figure 15 and Figure 16 show sample file server workloads for an E2600 controller and an E5400 controller, respectively.

Figure 15) E2600 performance with file server workload.

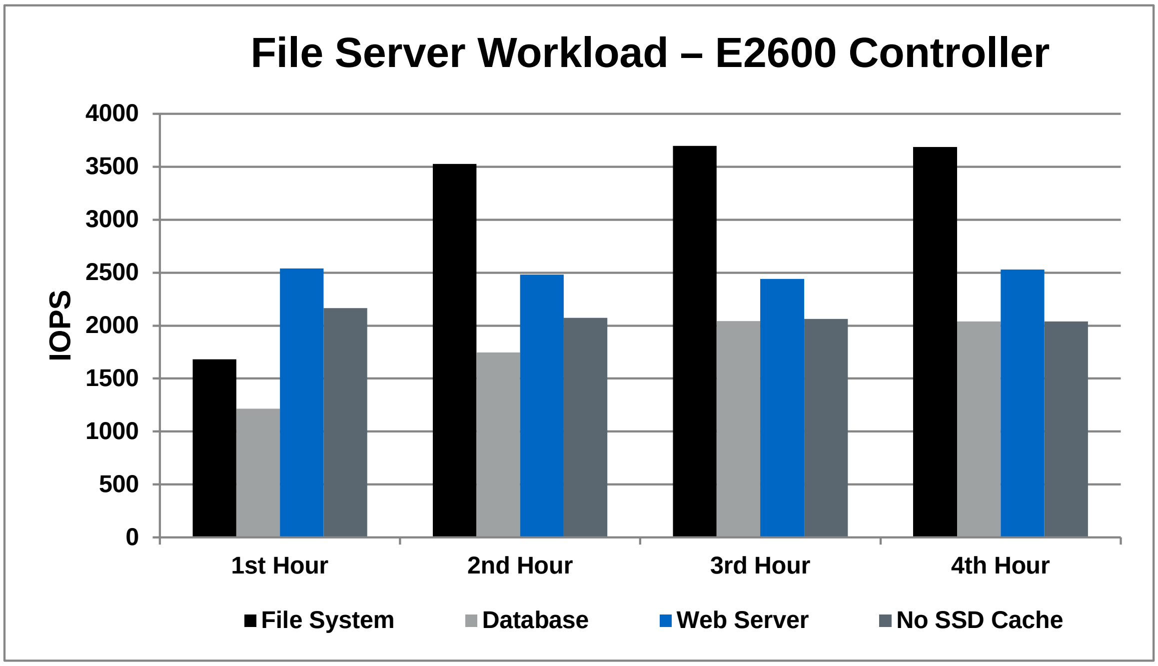
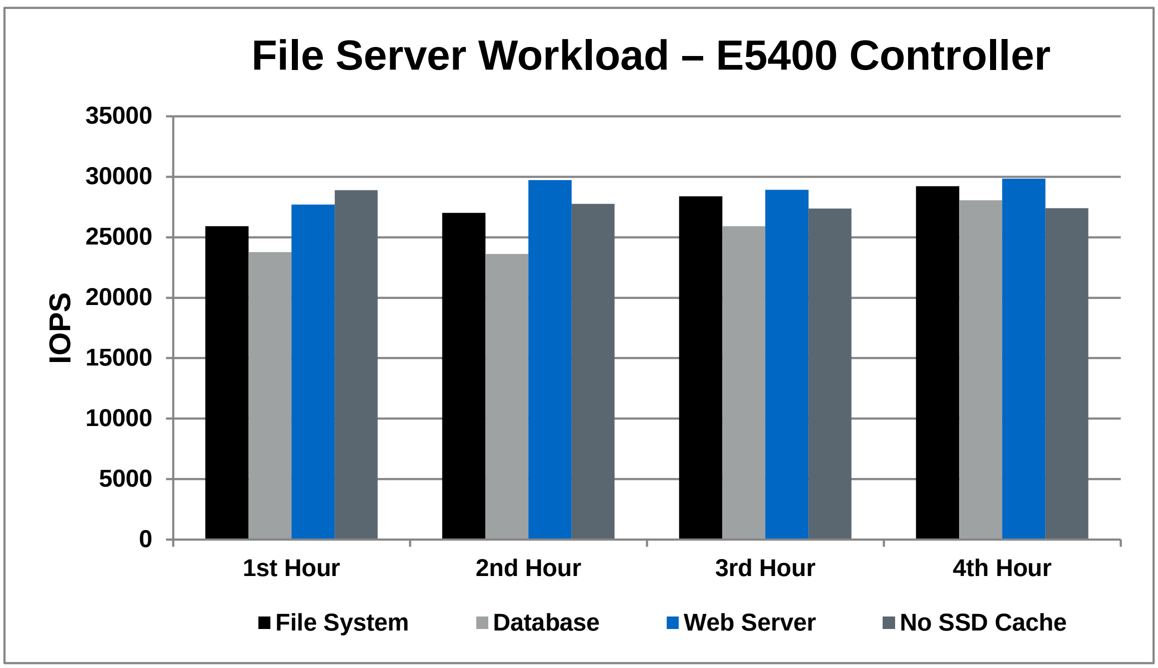


Figure 16) E5400 performance with file server workload.



5.4 Database Workload

Figure 17 and Figure 18 show sample database workloads in transactions per second (TPS) for an E2600 controller and an E5400 controller, respectively.

Figure 17) E2600 performance with database workload.

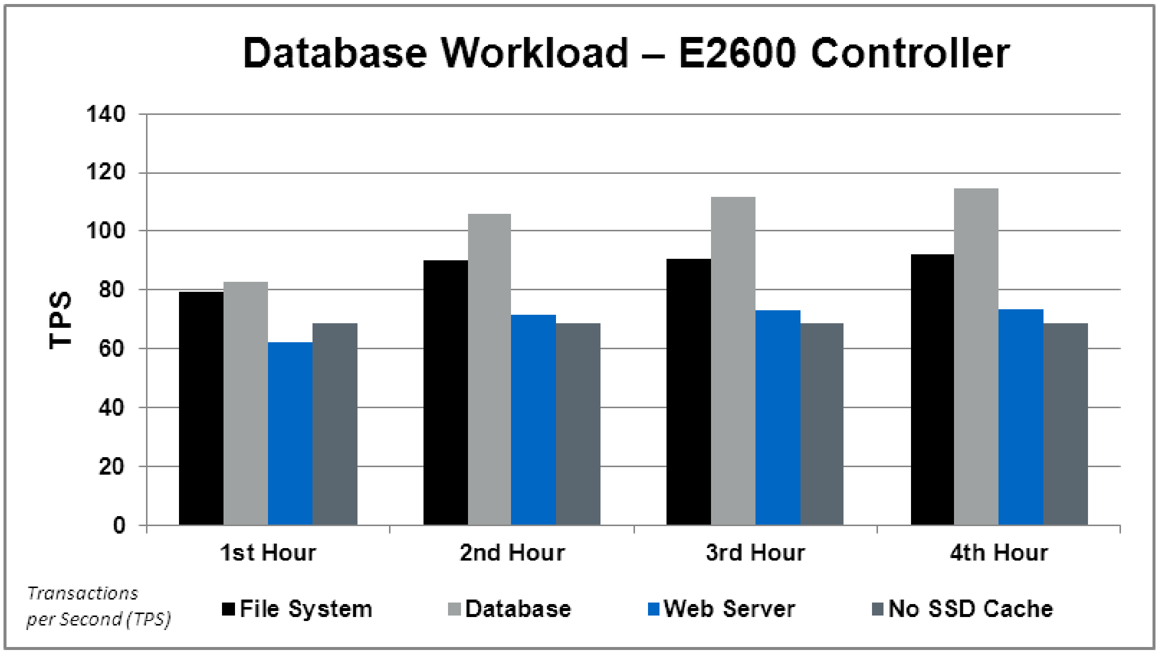
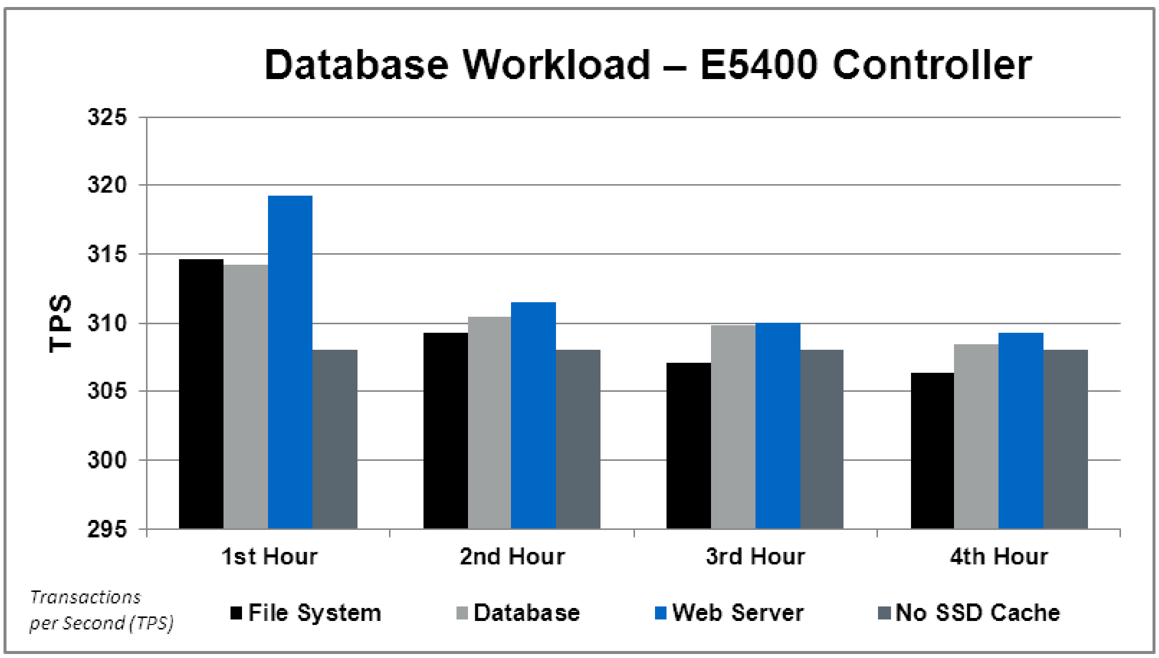


Figure 18) E5400 performance with database workload.



6 Recommendations

Workloads with the following characteristics benefit from an SSD cache:

- Read performance limited by HDD IOPS
- High percentage of reads relative to writes
- Large number of reads that are repeat reads to the same or adjacent areas of the drive

- Working size set is smaller than the SSD cache capacity

NetApp recommends using an SSD cache if:

- An increase in the DRAM cache cost is not feasible, or it is already at the maximum supported amount
- The cost of dedicated SSDs for RAID volumes is prohibitive
- A balanced, cost-effective approach that uses a mix of HDDs and an SSD cache is desired
- A built-in SSD cache performance modeling tool can be used to determine an appropriate SSD cache size

SSD cache is not an effective option if:

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

Version History

Version	Date	Document Version History
Version 1.0	September 2012	First version

Refer to the Interoperability Matrix Tool (IMT) on the NetApp Support site to validate that the exact product and feature versions described in this document are supported for your specific environment. The NetApp IMT defines the product components and versions that can be used to construct configurations that are supported by NetApp. Specific results depend on each customer's installation in accordance with published specifications.

Go further, faster®

NetApp®

www.netapp.com