White Paper

HP 3PAR Adaptive Flash Cache: A Competitive Comparison
# Table of Contents

Executive Summary ................................................................................................................................. 1  
Introduction ............................................................................................................................................. 2  
  Objective ............................................................................................................................................... 2  
  Audience ................................................................................................................................................ 2  
  Contents of this Report ............................................................................................................................. 2  
Data Caching in Primary Storage ............................................................................................................. 3  
  Cache Versus Tier ................................................................................................................................. 3  
  Method of Operation ............................................................................................................................. 3  
  The Long Tail ...................................................................................................................................... 4  
  Using SSD to Mitigate Cost .................................................................................................................... 4  
HP 3PAR Adaptive Flash Cache: Deep Dive ............................................................................................. 5  
  How Adaptive Flash Cache Works ....................................................................................................... 5  
Competitive Analysis ................................................................................................................................. 7  
  EMC VNX and VNX2 .............................................................................................................................. 7  
  NetApp Data ONTAP ............................................................................................................................ 9  
  Nimble Storage CASL ......................................................................................................................... 11  
Conclusions and Recommendations ........................................................................................................ 12  
References ............................................................................................................................................... 13
Executive Summary

Data caching is a technique used by storage array vendors to improve application I/O performance on systems deployed with hard-disk drives (HDDs). A cache stores commonly accessed data in fast system memory (such as DRAM), serving both read and write I/Os to the host and reducing the requirement to service requests from physical spinning disk. A cache holds a copy of data, with the original remaining on spinning media, also known as the backing store). This contrasts with tiering where data exists in only one place.

System DRAM is an expensive commodity and as only a portion of data typically generates the majority of I/O, DRAM is sized to the workload requiring acceleration. However application workloads are variable and insufficient DRAM has a direct impact on both the storage system and application performance.

Adaptive Flash Cache (AFC) is a feature of HP 3PAR StoreServ systems that allows solid-state disks to be used as an extension to DRAM cache. AFC acts as an “overflow” to main system cache, moving data that is less active out of DRAM cache and into the Adaptive Flash Cache. Using this method, the HP 3PAR StoreServ system is able to accelerate I/O for a greater majority of I/O that would typically need to be serviced from disk.

Adaptive Flash Cache was compared with implementations from other leading vendors, including Multicore FAST Cache on EMC VNX, Flash Cache on NetApp Data ONTAP and Dynamic Caching on Nimble Storage CS systems.

The HP 3PAR AFC solution was found to provide greater flexibility and lower cost in how hardware resources could be deployed (for example not requiring dedicated hardware). It also provided greater simplicity, integrating fully with existing platform features.

Overall, HP 3PAR Adaptive Flash Cache provides the most comprehensive and cost efficient use of flash storage, continuing to extend leadership of HP 3PAR StoreServ Flash-Optimizations.
Introduction

Objective

This report looks at the implementation of Adaptive Flash Cache on the HP 3PAR StoreServ storage platform and compares the features and functionality offered to equivalent products in the marketplace today.

Audience

Decision makers in organizations, looking to deliver highly efficient deployments of centralized storage will find this report provides an understanding of the benefits of leveraging flash SSDs as a secondary cache to cost-effectively improve system performance.

Contents of this Report

- Executive Summary – A summary of the background and conclusions derived from Edison’s research and analysis.
- Data Caching in Primary Storage – A discussion of the benefits of caching data within shared storage systems.
- HP 3PAR Adaptive Flash Cache: Deep Dive – a detailed discussion on how enhanced caching is implemented in HP 3PAR StoreServ systems.
- Competitive Analysis – A comparison with competing vendor products from EMC, NetApp and Nimble Systems.
- Conclusion and Recommendations – a summary of the findings comparing HP 3PAR with the competing platforms.
Data Caching in Primary Storage

Caching has been a feature of shared external storage arrays for over 20 years. The aim of caching is a simple one; to provide increased system performance by placing more frequently accessed data in fast system memory. It is well known that hard-disk drives (HDDs) have both unpredictable and slow performance characteristics due to the mechanical nature of spinning media and movement of read/write heads. The use of a cache helps mitigate these issues to provide faster and more predictable I/O response to the application.

Cache Versus Tier

One important aspect that needs to be made clear is the difference between caching and tiering. Tiering places data on differing levels of storage dependent on performance, providing a cost savings by ensuring I/O is delivered to the host in a timely fashion for the most efficient cost. Data is stored uniquely on each tier and system algorithms are used to move data between tiers as workload profiles change.

Caching provides the ability to accelerate workloads by maintaining a copy of the data in fast system memory. The original copy of data still remains on physical media (called the backing store). System algorithms determine which data is best to reside in cache based on metrics such as “least recently used” (LRU) or First in First Out (FIFO). The data in a cache typically does not need additional protection such as RAID, however many write caches based on system memory are protected by battery backup units (BBUs) in case power is lost before data is committed to the backing store.

Method of Operation

Caches typically provide two modes of operation: read caching and write caching, or both.

Read Cache

A read cache stores data in memory (usually system DRAM) that is predicted to be used frequently by the host. Where possible, the storage system delivers an I/O request from the DRAM cache, eliminating the need to access backend disk and so delivering the request with a much faster response time. Eliminating backend I/O also has the additional benefit of reducing the contention on backend disk for other operations such as write I/O.
**Write Cache**

A write cache stores write I/Os in system memory and immediately acknowledges I/O completion back to the host. The data is then written to physical media asynchronously at a later time. From the application perspective, the I/O completes quickly, allowing greater throughput. From the perspective of the array, the data in cache can be “batched” and de-staged more efficiently, for example writing data blocks sequentially to disk.

Although caching seems like a simple process, in fact the implementation of caching can be quite complex. Read caches need to ensure that data in cache is replaced or removed if the cached data is updated by the application. Write caches need to be able to protect data in memory that has not already been committed to disk, in case of hardware failure or power loss. Because of these requirements, we see additional hardware (such as battery backup units) and software features (such as cache mirroring) being used to protect the integrity of cached data.

**The Long Tail**

In typical shared storage environments, a small portion of data will be responsible for the majority of I/O, the so-called 80:20 rule or Pareto Principle. The specific split will be dependent on the workload mix, but in general the principle applies regardless of the exact ratios. The remainder of the data in an array will account for progressively fewer I/Os, resulting in a “long tail” distribution, where when ordered by I/O count.

What the long tail distribution indicates is that the majority of data does not justify being placed in cache (as DRAM is expensive), however the specific ratio of cache to persistent storage is not consistent across differing types of workloads. As a result, on some occasions there will be a surplus of cache and on others a deficit of cache, leading to a direct impact on application performance. It is impractical to add additional DRAM in a running system and any upgrade would incur a cost, even if the demand were only short term. This does not prove to be an efficient use of storage resources.

**Using SSD to Mitigate Cost**

If cost were no restriction, most storage systems would implement large amounts of cache in proportion to the percentage of persistent spinning media. However DRAM is significantly more expensive per GB than spinning disk or solid state disks and therefore there is a practical tradeoff in the amount of cache in a system compared to the amount of persistent media. Features such as HP 3PAR Adaptive Flash Cache allows NAND flash storage to be used as a true extension to DRAM cache and so deliver a more cost effective way to increase performance without the associated cost.
HP 3PAR Adaptive Flash Cache: Deep Dive

HP 3PAR Adaptive Flash Cache is a feature of HP 3PAR StoreServ systems that uses a portion of solid-state disks as an extension of DRAM cache. Using SSD as an extension of DRAM allows a system to cater for increased caching requirements without incurring significant additional cost. Although SSD devices do not provide I/O latency levels as low as DRAM, they are orders of magnitude faster than hard disk drives, enabling their use to improve overall system response times.

How Adaptive Flash Cache Works

Adaptive Flash Cache acts as an extension of the read cache features of HP 3PAR StoreServ system memory. This means the system caches random read I/O requests. The caching process uses solid-state disks (SSDs) as a secondary cache layer, allowing 16KB of cache data to “overflow” from DRAM to SSD when DRAM cache is heavily loaded.

HP 3PAR StoreServ systems do not require SSDs to be dedicated to the Adaptive Flash Cache (AFC) feature. A portion of the flash capacity of a system can be assigned to AFC, other features such as Adaptive Optimization (AO) or as a storage tier. This provides customers significant flexibility and efficiency in how flash is used within a HP 3PAR StoreServ storage array.

AFC Operation

In normal operation, an HP 3PAR StoreServ system will deliver I/O from system DRAM cache wherever possible. As DRAM cache occupancy reaches 90% and with Adaptive Flash Cache enabled, the system will evict data from DRAM cache into SSD cache. The eviction process incurs no penalty in accessing backend disk, as the data is moved directly from DRAM to SSD – there is no backend I/O cost for populating the cache.

The system will place in AFC data that was a result of a small random I/O request. This could be a previous random read request that was between 0-63 KB, or a random write I/O that will be flushed to HDD, while also keeping a copy in AFC in order to increase the likelihood of read hits.

If an I/O request is received for data that has been moved to Adaptive Flash Cache, then this will generate a “cache miss” in DRAM cache. The I/O will then be serviced from Adaptive Flash Cache, with significantly lower I/O latency than would have been achieved by directing the I/O to physical disk.
Adaptive Flash Cache does not handle write I/O caching, but it does have to manage the scenario of updates to data cached by AFC. In this instance, the write I/O is received into DRAM and the old Adaptive Flash Cache copy is simply invalidated, allowing the space to be reused. The data in DRAM is then asynchronously written to backend disk and eventually may become eligible for de-staging to AFC.

**Adaptive Flash Cache Considerations**

Adaptive Flash Cache is used to cache small random I/O type data and volume metadata pages, however data that has a low likelihood of generating a cache hit is generally excluded. This includes data brought into DRAM cache as the result of a read-ahead process, large I/O requests (>64KB), sequential read and write data and of course data that is kept on an SSD backing store. Data on an SSD tier naturally sees no benefit in being also cached on SSD with the same performance characteristics.

Adaptive Flash Cache is provided as a feature included in the HP 3PAR Operating System release 3.2.1 and later. It can also be enabled on systems in “evaluation mode” to see how AFC could have benefited a system with solid-state disks deployed.

Adaptive Flash Cache can be deployed at either a system or VVset level. A VVset is a logical grouping of LUNs with similar operational requirements, for example devices assigned to a single host, cluster or application. The ability to enable AFC on a VVset means customers can ensure data that does not benefit from Adaptive Flash Cache is automatically excluded from the caching process. AFC will be extended to other logical constructs such as VMware VVOLs, once the technology is generally available.
Competitive Analysis

HP 3PAR StoreServ Adaptive Flash Cache has been compared to three storage platforms in today’s shared storage market. The platforms are EMC’s VNX and VNX2, NetApp Data ONTAP and Nimble Storage’s CS series hybrid arrays.

**EMC VNX and VNX2**

Both the previous and current generation of EMC’s mid-range VNX platform have been included in this comparison due to the relative newness of the VNX2 platform. The VNX2 platform (models VNX5200, VNX5400, VNX5600, VNX5800, VNX7600 and VNX8000) refer to the SSD caching feature as EMC VNX Multicore FAST Cache (and all feature architectures under MCx brand). The previous generations refer to the feature simply as VNX FAST Cache. When referencing the features, the term “FAST Cache” will cover both implementations.

**FAST Cache Operation**

FAST Cache uses a Policy Engine to manage data moved into SSD. The policy engine identifies data that is accessed frequently at a 64KB level of granularity and schedules a task to copy data from persistent storage into the FAST Cache SSDs. The policy engine is also responsible for maintaining statistics on data access patterns that is used to determine the data to be included or excluded from the cache.

The FAST Cache Policy Engine determines data eligibility for caching based on I/O activity over a fixed period of time. Eligible data is copied to FAST Cache SSD as a background task and system metadata is updated to flag that the data is now available in FAST Cache. When an I/O request occurs, the system checks to see if the data is already cached in DRAM. If it is not located, a cache miss occurs and a check is made in the Fast Cache Memory Map to see if the data resides in FAST Cache. If located, the data is returned from FAST Cache, a process known as a FAST Cache hit.

Write I/O operations are handled in two different ways, depending on whether Write Cache is enabled on the system and the LUN. If Write Cache is enabled, the I/O request is simply handled by DRAM cache in a normal fashion. If Write Cache is disabled, the system effectively uses FAST Cache like DRAM cache. An I/O write request is held in DRAM cache while FAST Cache is checked for the data. If it exists in FAST Cache, the data is copied to FAST Cache and the I/O acknowledged to the host. If the data is not in
FAST Cache, the I/O is written to persistent storage and completion confirmed back to the host. In this instance FAST Cache acts as a fallback for the failure of DRAM cache or the loss of a controller, which would cause the VNX system to revert to write disabled mode. The cost of this is double the amount of SSD, as write I/O must be protected against hardware failure.

**Fast Cache Cleaning**

Fast Cache uses a cleaning process that periodically writes “dirty” cache pages to disk in times of minimal backend disk activity. A dirty page is classed as one that has been updated in cache but has not yet been committed permanently to disk. Cache cleaning in times of low activity allows pages to be re-used in Fast Cache without having to commit data to physical disk. The system will use free pages, followed by “clean” pages (those where the cache contents match the disk contents) and only use “dirty” pages as the last option when other pages types aren’t available.

**FAST Cache Restrictions and Considerations**

FAST Cache SSD disks are dedicated to the FAST Cache process and cannot be used for persistent data. This can result in an underutilization of SSD capacity. FAST Cache disks are deployed in mirrored pairs (RAID-1) and are global for the whole VNX platform. A minimum of two SSD drives is required to enable FAST Cache (however spare drives should be considered to cater for hardware failure). From this point, the FAST Cache feature is enabled for either an entire storage pool or individual LUNs.

The policy engine excludes certain data types from the cache. This includes:

- Data already on a flash tier of storage. This data would see no benefit from being moved into FAST Cache.
- Small block sequential I/O is excluded and managed in DRAM cache, as this type of I/O is managed by the system in other ways: small-block sequential writes are aggregated in memory for writing as a large I/O block and small-block sequential reads will trigger data prefetching.

Other considerations for the use of FAST Cache and Multi-core FAST Cache include:

- FAST Cache consumes a portion of DRAM for metadata, so DRAM cache is reduced when using FAST Cache.
- Enabling the data at rest feature requires flushing FAST Cache first. After D@RE is enabled, the FAST Cache has to be re-warmed.
• Drive failure on a FAST Cache drive forces the drive pair to degraded mode. This mode causes FAST Cache data on the remaining mirror pair to be flushed to HDD and the drive is marked as available for only read operations.

• EMC recommends Fast Cache be disabled for MirrorView and SnapView private LUNs as the feature can have detrimental effect on performance. Fast Cache does not improve performance for Reserved LUNs and EMC recommend it be disabled.

**EMC Best Practices**

• With limited flash drives, create a FAST Cache pool first then use the remainder in FAST VP – because FAST Cache is global; FAST VP is by storage pool.

• Manually choose drives to balance across backend buses more effectively.

• Do not use FAST Cache with system utilization > 80% (in fact systems are recommended to be sized for 70%).

**NetApp Data ONTAP**

NetApp’s caching implementation is based on dedicated hardware cards, previously known as PAM (Performance Acceleration Module) that are delivered in a PCIe form factor. Multiple generations of card have been produced. The original PAM (which supported only 16GB of capacity) was superseded by PAM-II, now simply dubbed Flash Cache, offering capacities of 256GB, 512GB and 1TB using SLC NAND flash. Flash Cache 2 offers capacities of 512GB, 1TB and 2TB based on an undisclosed NAND type, which is most likely to be MLC NAND. Each FAS filer model selectively supports the range of Flash Cache card models and the specific number of cards supported varies by the availability of PCIe slots, which are shared with other storage networking components. This puts restrictions on filer configurations and impacts scalability. Note that Flash Cache 2 cards are only supported with Data ONTAP 8.1.3 and higher.

NetApp also has another technology known as Flash Pools. This uses solid-state disks to create a caching tier per aggregate in a Data ONTAP system. Flash Pools provides read and write caching, however SSDs have to be dedicated to a specific aggregate.

**Flash Cache Operation**

Flash Cache is enabled in both Data ONTAP 7-mode and cluster mode as a licensed feature. Reads are processed by the system and cached in DRAM for potential future access. As DRAM becomes full, data is evicted from DRAM cache and cascaded down to Flash Cache, which acts as a secondary cache.
Flash Cache provides three modes of operation; Metadata Mode, Normal Data Mode and Low-Priority Mode, all of which are enabled with the FlexScale feature.

- Metadata Mode uses Flash Cache for storing metadata only. Actual user data is evicted from DRAM cache as it would be if Flash Cache was not running. This mode of operation is not recommended for Flash Cache 2 deployments where sufficient capacity is available in the larger PCIe PAM cards to cache all data.

- Normal Mode evicts both metadata and user data into Flash Cache as DRAM cache fills up.

- Low Priority Mode allows data that would normally be excluded from Flash Cache to be retained. This includes write I/O and long sequential read operations. Retention of write I/O data can be beneficial with certain workloads where data is read back almost as soon as it is written.

It should be noted that the Flash Cache operating modes are globally applied to the system and affect all data. There is no ability to apply a more granular approach to multiple types of workloads that are active on a single system. There is however a legacy feature for 7-mode deployments called FlexShare, which does allow cache policies to be applied at the volume level. This feature has not been carried through to the strategic cluster-mode version of Data ONTAP and so is not discussed in any further detail in this paper.

**Flash Cache on Clusters**

For clusters of systems deployed with Data ONTAP cluster mode, it is possible for data requests to be received on nodes other than the node hosting an aggregate and volume. In this instance, the node receiving the I/O request will pass the data cross the cluster interconnect to be serviced by the volume-owning node. Flash Cache data is unique across each node (each node caches its own data) and so the I/O request may be serviced from Flash Cache on the node owning the aggregate/volume. There is therefore no sharing of Flash Cache capacity across nodes.
Flash Cache Restrictions and Considerations

NetApp applies a number of recommendations on the use of FlashCache.

- In a Data ONTAP HA pair, each node is recommended to have an identical amount of flash to ensure consistent performance in the case of a takeover.
- When a takeover occurs, the partner node in the HA pair becomes responsible for caching, resulting in the need re-initialize original node’s cache on giveback. The giveback process may result in lower performance as cache is rewarmed.

Nimble Storage CASL

Nimble storage is a relatively new entrant into the storage market, releasing their first products in 2010. Nimble’s architecture is known as CASL or Cache Accelerated Sequential Layout, which uses a combination of flash and traditional hard, drives to provide improved read and write throughput compared to a standard hard-disk based array. The CASL architecture divides data across the two types of media to optimize both read and write requests. Write I/O is optimized through coalescing many small random writes in NVRAM and writing them out to disk in large sequential stripes. This writes data to disk in a way more suited for that media, that is in sequential format.

The CASL architecture uses flash as a read only cache (a feature known as Dynamic Caching), caching data into SSD in real time as data is received by the system. Flash acts entirely as a cache rather than a tier of storage, with the persistent copy of data remaining on physical disk. As a result, the cache layer is not protected against device failure, reducing the latency and the wear leveling overhead that would be experienced by RAID protecting the data.

Dynamic Caching Operation

The Dynamic Caching feature receives data from two sources; either directly from NVRAM on write, or as a result of a read cache miss. For write requests, data is stored in system NVRAM and mirrored to system DRAM before being compressed and written to disk. If data is determined to be “hot” at this point, a copy is also made into flash cache. The specific algorithm for this process isn’t explained by Nimble, however they describe this process as identifying data that is “cache worthy”.

When data is read from disk, a copy is also stored in SSD cache, allowing it to be available for subsequent I/O read requests.
Conclusions and Recommendations

Caching is a technique that provides significant improvements in performance on systems using hard disk drives through the ability to deliver I/O faster from DRAM and SSD. This is achieved by avoiding physical disk I/O during the process of completing a host I/O request, either pre-empting a read or deferring a write operation. System DRAM is an expensive component cost, however solid-state disk (SSD) can be used to augment DRAM and ensure as a resource this is fully utilized. In this comparison, HP 3PAR Adaptive Flash Cache was compared to EMC VNX (Multi-Core) FAST Cache, NetApp Flash Cache and Nimble Dynamic Caching. In summary, some of the differences in implementation include the following points.

- **Flexibility** – Adaptive Flash Cache can be used in conjunction with other HP 3PAR StoreServ features such as Adaptive Optimization. Customers can choose how best to deploy their SSD resources within the storage system. By comparison, EMC FAST Cache and NetApp Flash Cache deployments require dedicated SSDs. AFC can be deployed to hosts at a VVset level, allowing the feature to be effectively targeted at only workloads that will benefit from acceleration. NetApp Flash Cache and Nimble Dynamic Caching are globally enabled features that cannot be targeted at specific volumes. EMC FAST Cache can be enabled per LUN or storage pool but not at logical entities that address a specific host or cluster.

- **Simplicity** – Adaptive Flash Cache is simple to enable and configure within an existing HP 3PAR StoreServ system. By comparison, NetApp Data ONTAP requires customers to choose between two different features (Flash Cache & Flash Pools) depending on the cache type. Each solution uses different hardware that cannot be interchanged between both features.

- **Cost** – Adaptive Flash Cache is included within the HP 3PAR Operating System at no additional cost and can be used on any solid-state disks supported by the system. Both EMC FAST Cache and NetApp Flash Cache are license options that require dedicated hardware equipment (including more costly SLC NAND flash). Hardware components are recommended to be deployed equally in each node of a configuration, resulting in significant cost implications on multi-node configurations.
References

- Nimble Storage CASL Architecture Deep Dive (YouTube)
- H12208 - EMC VNX Multicore FAST Cache (EMC White Paper)
- H8046 – VNX FAST Cache – A Detailed Review (EMC White Paper)