

Module 5: Analyzing Symmetrix Cache Performance

Upon completion of this module, you should be able to:

- Discuss the architecture features of the cache and internal data paths
- Measure the effects of hit rate on performance
- Detect write pending issues and propose solutions
- Prepare data storage for proper alignment with Symmetrix cache
- Understand XtremeSW Cache interaction with the VMAX



This module focuses on analyzing the performance of Symmetrix VMAX cache.

Module 5: Analyzing Symmetrix Cache Performance

Lesson 1: Symmetrix Cache Architecture

During this lesson the following topics are covered:

- Identifying the Symmetrix cache architecture
- Categorizing I/O operations



This lesson covers the architecture of Symmetrix cache.

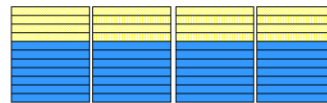
Cache Contents

User Data Slots:

- Data region used for moving data to or from disks
- When allocated for use, a slot represents a fixed address track on the device
 - ▶ All I/O for that track goes through the slot
- Due to mirroring, two physical slots are allocated
 - ▶ Changes are written to both for redundancy

Global System Data:

- Device tables
- Director mailboxes
- Critical data mirrored if cache is mirrored



Global Data striped across physical locations (256 MB stripe size).

EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 3

The main purpose of the cache is to buffer I/O being serviced by the disks. The “user” area of cache is divided into a large number of “slots” that are used to buffer the I/O. A slot is allocated when it contains data for a device. This will happen when a read or write request requires a slot to hold the data, and the data is placed in the slot. Until it is de-allocated and re-used, the slot represents that track of the associated device. All reads and writes to that track will use the same slot. This improves the allocation and performance of the internal operations, as we will see throughout this module.

While the cache is primarily used for user data, it also contains some global system data that is used by the directors. Device tables list the contents of each slot so that any director can find data in cache. Director mailboxes are used to communicate commands between directors. Think of this as a command queue for the directors. Other overhead data is also stored in the global cache. All VMAX arrays stripe the global data across all cache cards to balance the traffic on the Matrix channels.

User data is mirrored, and any data is written to two different physical locations (two directors) for redundancy.

Cache Region Sizes

- Sector

- ▶ Smallest unit referenced on disk
- ▶ CRC calculated per Sector when transferring within array

8 KB

- Slot / Track

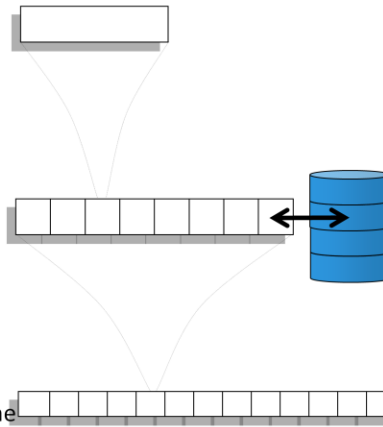
- ▶ One Slot = Eight sectors
- ▶ Slots in cache represent Tracks on disk

64 KB

- Cylinder

- ▶ One Cylinder = Fifteen tracks
- ▶ Many data sizes within the Symmetrix are measured in Cylinders

960 KB



EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 4

The sector, track, and cylinder are the three fundamental sizes for disk storage in the Symmetrix array, and those sizes impact the cache architecture.

The sector is the smallest unit referenced on disk. The backend hardware will not transfer data in sizes smaller than one sector. A Cyclic Redundancy Check (CRC) is performed on every sector of data transferred between components within the array. If the check shows that data has been damaged, the array takes countermeasures, such as re-transmitting or marking a disk region as damaged.

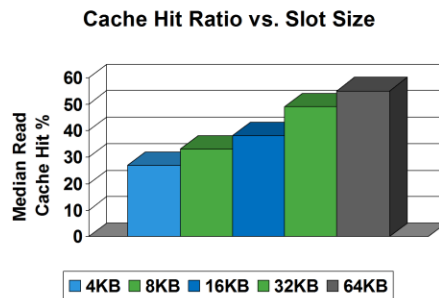
In VMAX arrays, the sector size is 8 KB. A track is composed of eight continuous sectors. When a track of disk data is being represented in cache, the storage location used is called a "slot." VMAX arrays use 64 KB tracks.

The 64 KB size is used to hold all data (Open Systems or Mainframe) on a VMAX array.

A cylinder is composed of fifteen continuous tracks. These virtual cylinders are independent of the configuration of the physical disks in the array. The size of a cylinder appears occasionally when measuring device sizes or meta device stripe sizes, but otherwise do not significantly impact our performance discussion. All cylinders are 960 KB for VMAX arrays.

Locality of Reference Favors Large Cache Slot Size

- Analyzed many customer traces with cache modeling tools
- Varied cache slot size from 4 KB – 128 KB
- 32 KB slot size got ~16% greater cache hit than 8 KB
- 64 KB slot size got ~6% greater cache hit than 32 KB



EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 5

Prior to the DMX-3 arrays, cache slots were 32 KB. The DMX-3 was changed to use 64 KB slots. The key reason that the slot size was doubled in the DMX-3 array, was to reduce overhead. Some cache is used to catalog the contents of every slot; with larger slots, you have less catalog overhead. But, do the larger sizes actually help performance?

To perform the research, EMC engineering gathered I/O traces from hundreds of customer systems. Those traces told us four things:

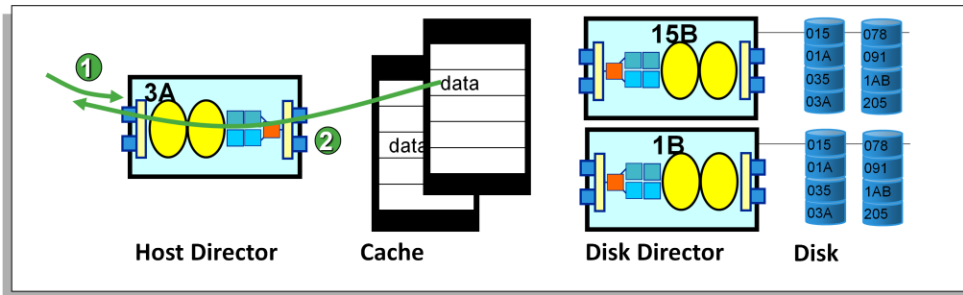
1. When the I/O occurred (with microsecond granularity)
2. Which device was affected
3. Which port was affected
4. What kind of I/O (read/write and block address)

We ran these traces through simulation programs that modeled the Symmetrix cache behavior, and we discovered that a larger track size actually increased the read cache hit ratio, as long as we read to the end of the track. This resulted, then, in improved application performance. Conversely, as we reduced the track size, we reduced the cache hit ratio.

64 KB yielded about 6% greater read cache hit ratio than 32 KB.

You may wonder why this is. The reason is simple: most applications exhibit a strong locality of reference. This means if you read location 5, pretty soon you will read location 3, or 7, or 4. It's not always true, but most applications really do behave this way.

Read Hit



1. Host sends read request and data is found in Cache
2. Host Director pipes data from Cache to host

EMC²

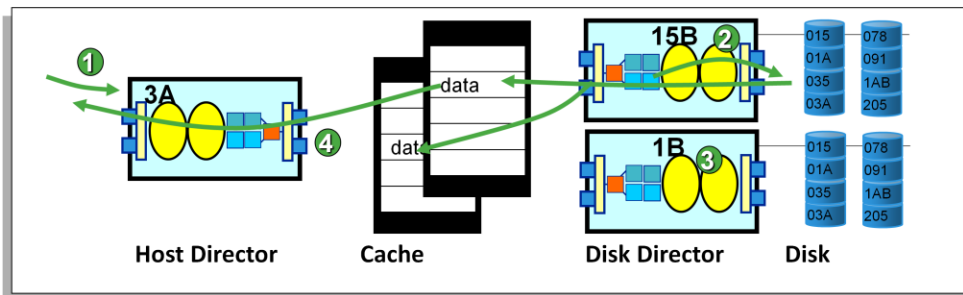
Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 6

In a read hit operation, the host requests data that is found to be already in cache. The data may have been loaded into cache by a previous I/O operation or by prefetching activity. The host director immediately pipes the data from a cache location directly to the host.

Even if the cache slots are mirrored, there is no reason to read the data from cache twice. Duplicate cache accesses are performed only when the slot data changes.

Read Miss



1. Host sends request to Host Director and data is not found in Cache
2. Disk Director sends read request to drive
3. Disk Director pipes data from drive to mirrored Cache locations
4. Host Director pipes data from Cache location to host

EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 7

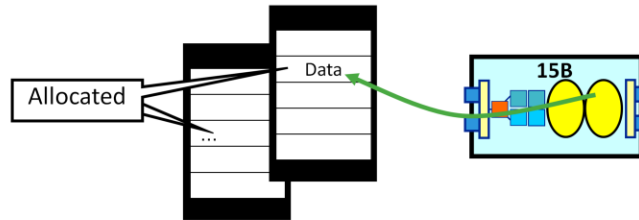
In a read miss operation, the data requested by the host is not already in cache. The appropriate disk director is signaled to retrieve the data from the disk drive. When the drive responds, the disk adaptor sends the data into an available cache slot. The host director can then pipe the data from the cache slot to the host.

The disk director reads the data from the disk and writes it to two cache slots. This may seem excessive since there is no danger of data loss (the data is permanently recorded on the drive), but all slots must be mirrored, regardless of what kind of data is in them. An optimization was introduced in later versions of Enginuity that reduces the number of back-end data transfers. We discuss this optimization in a few pages.

The operation in which the host director reads data from the cache does not need to be performed in duplicate, even when cache is mirrored. Only one operation is required since the cache slot is not being modified.

Read misses take significantly longer to process than read hits since the host must wait for data to be staged from the disk. Backend, as well as front-end resources are required to complete the request.

Selective Cache Mirror Optimization



How it works:

- **Read Miss:** Two cache slots allocated, but load data into only one
- **Read Hit:** Read from valid slot
- **Write:** Always write to both allocated slots

Benefits:

- Pure read streams only cost one backend I/O per read
- Up to 30% improvement for large I/O read miss workloads
- No significant improvement for small I/O or writes
- No benefit for internal replication reads

EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 8

The Selective Cache Mirror Optimization feature was introduced with the 2007 service release of Enginuity 5772. This optimization is only used in DMX-3 and VMAX arrays that have mirrored cache. It allows the disk director to allocate two slots for a read operation, but only load data into one of them. This saves one backend transfer. On a cache failure, the data is simply re-read from the drive.

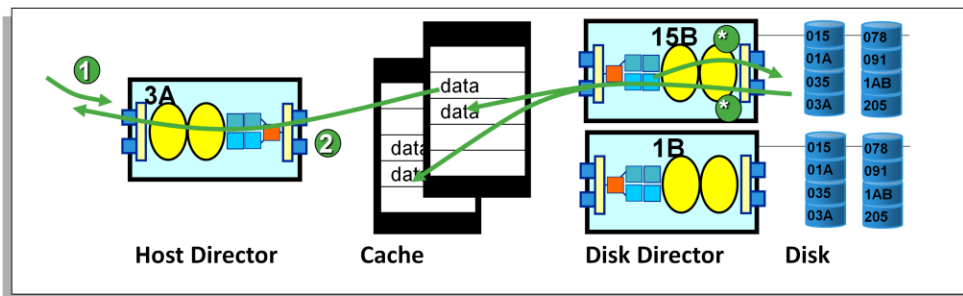
Why are two slots allocated if only one is needed to satisfy a read request?

Once allocated, a slot represents the location on the drive. If a write is made to that location, it will have to be mirrored in cache. By consistently allocating two slots for every data location, the directors have the correct amount of cache space for any future writes.

Benchmark tests show that this optimization can give a 30% improvement for large I/O read miss workloads. It gives no significant improvement for small I/O workloads, or writes. There is no benefit to internal replication reads because the data is always read from one place and written to another. It must be treated as a write, with the data being written to both mirrored locations.

VMAX arrays also mirror cache locations. In a single-engine array, the mirrored locations will be on the two director cards within the engine. On a multi-engine array, the mirrored locations will always be on different engines. This means that you will have to consider the engines in pairs when configuring cache. Each engine must be paired with another engine having the same amount of cache to facilitate mirroring.

Sequential Read with Prefetch



1. Sequential reads trigger Disk Director prefetch task and data is read into cache locations ahead of read stream
2. Host sends request to Host Director and data is found in Cache due to prefetch activity
3. Host Director pipes data from Cache location to host

Note: Disk Director simultaneously uses prefetch job to fill cache ahead of sequential read requests

EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 9

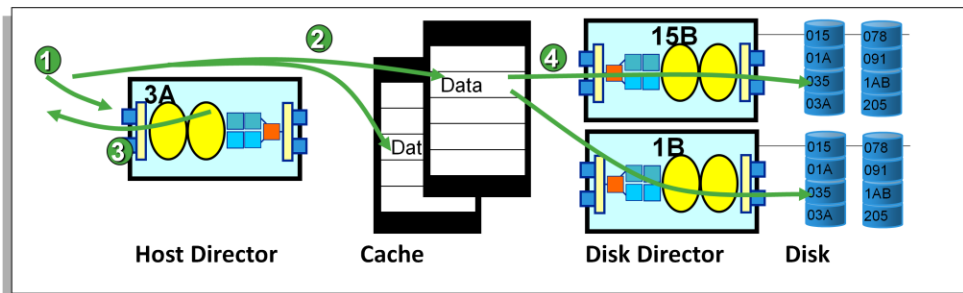
Prefetching is a technique for generating additional read hits. When the Symmetrix detects two reads of data from sequential locations, the appropriate disk director begins a prefetch task. The disk director attempts to read additional tracks of data from the disk into cache, ahead of the data requested by the host. As more sequential locations are read, the amount of prefetched data increases. If the host continues to read data in a sequential fashion, it will find the data already loaded in cache.

Since the disk director cannot tell exactly when the host will stop requesting the data sequence, some of the prefetch work will be wasted. However, since read hits are serviced so much faster than read misses, the performance improvement of prefetching can be significant.

The disk director's prefetch work happens in parallel with host requests. As the host director is servicing requests, the disk director is reading data into cache.

As mentioned previously, two backend director-to-cache operations will be required to load the prefetched tracks if the cache slots are mirrored.

Write Hit (Fast Write)



1. Host sends request to Host Director and available Cache slots are found
2. Host Director pipes data from host to mirrored Cache locations
3. Host Director responds to host to acknowledge successful write
4. Disk Director destages data to disk at some later time

EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 10

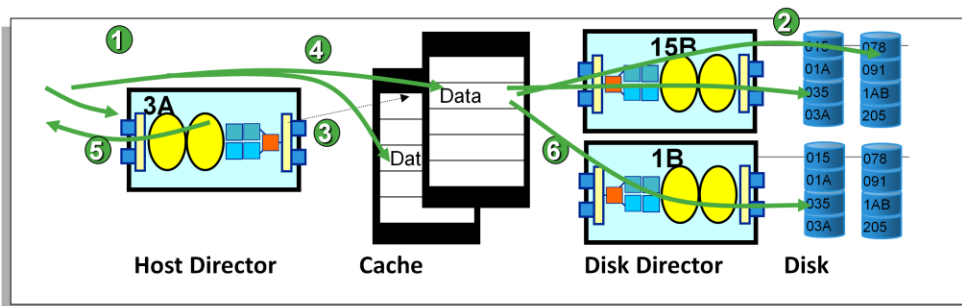
A write hit operation happens when there is sufficient room in cache to store a requested write. The host director finds an available cache slot and pipes the host data to it. The host director immediately finishes the operation with the host, signaling that the data was successfully written. The cache slot used for the write is marked with a special “write pending” code to indicate that it cannot be reused until the data is safely stored on disk.

If the cache slots are mirrored, the host director will have to write the data to two different cache slots.

At some point later, the disk director responsible for the volume will destage the data to disk. Mirroring or parity protection will mean that more than one disk director will need to write to disk before the data can be considered destaged. Once this finally happens, the write pending flag is removed from the slot to indicate that it is safe to recycle for some other purpose.

Destaging is normally assigned a low priority. However, if a large number of write pendings are currently in cache, the disk directors may switch into a higher priority write mode. When this happens, destaging tasks get equal priority with host read requests to help flush the write pendings out of cache.

Write Miss (Delayed Fast Write)



1. Host sends request to FA and no available Cache slots are found due to Write Pending limit
2. DA destages data as a high priority task to make slots available
3. FA finds available slot by polling
4. FA pipes data from host to mirrored Cache locations
5. FA responds to host to acknowledge successful write
6. DA destages data to disk at some later time

EMC²

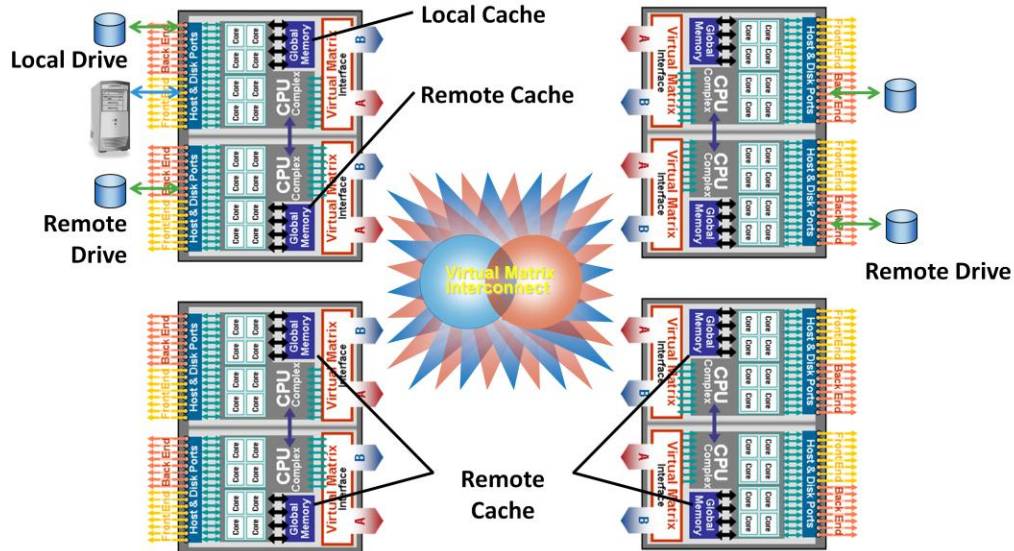
Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 11

Any Symmetrix has a finite supply of cache slots in which to store writes. When they run out, a new write request cannot be serviced as fast as a regular write hit. This operation is commonly called a Delayed Fast Write or Write Miss.

When a request for a write occurs that cannot be serviced because of lack of cache, existing write pending data is immediately destaged by disk directors. When the host director observes that there is space in cache, the write is completed as normal. This is where the term Delayed Fast Write originates: it is a normal Fast Write that must be delayed while destaging is done.

Symmetrix VMAX System and Cache Architecture



EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 12

For I/O operations in a VMAX array, data is always transferred between host and drive through cache. Hits happen when the I/O can be serviced directly from cache; misses happen when an immediate drive operation is necessary.

In a DMX array, the data was stored on cache cards remote to the director, and data had to travel through the Matrix to reach a cache location.

In a VMAX array, cache is stored on the director cards themselves. In some cases, the cache location a director needs might be right on the director card, saving it from having to communicate off-board. In many cases, the cache location is on a remote director, and as the number of engines grows, the probability of this increases.

Compared to how I/O operations in a DMX array worked, you should be able to imagine most of the cache data transfer operations in a VMAX array. Keep these ideas in mind:

- Cache locations for an I/O operation might be stored on any director card or engine. The two mirrored locations will be stored on two different director cards.
- The host director that services the I/O does not have to be on the same director or engine as the disk directors that service the related drives.

Module 5: Analyzing Symmetrix Cache Performance

Lesson 1: Summary

During this lesson the following topics were covered:

- Identifying the Symmetrix cache architecture
- Categorizing I/O operations



This lesson covered the architecture of Symmetrix cache.

Module 5: Analyzing Symmetrix Cache Performance

Lesson 2: Cache Hit and Miss I/O Operations

During this lesson the following topics are covered:

- Reasons for high or low hit rate
- Measuring hit rate



This lesson covers the concepts of cache read and write hit rates.

Issue: Low Hit Rate

- Low hit rate means longer response times
- Typical rates (dependent on application)
 - ▶ Open Systems: 60%-90%
 - ▶ Mainframe: 90%+
- Heavily dependent on application and often difficult to “correct”

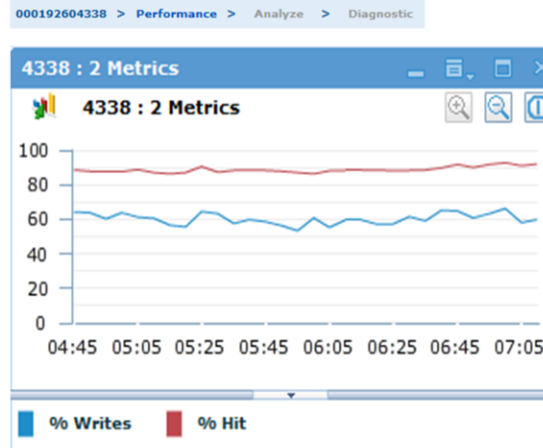


Low hit rate is a performance issue because the response time is longer with miss I/Os than with hit I/Os. Some typical hit rates for Open Systems and Mainframe environments are shown on this slide. Open Systems' data centers are so varied that the general estimate still does not apply to all circumstances.

The hit rate is often much more dependent on the application than it is on the hardware. For instance, an active OLTP database with several terabytes of data might produce a wide ranging random workload. This might produce a very low hit rate, no matter how much caching is happening at the host buffers or the Symmetrix. In this case, there will be no way to significantly raise the hit rate to “correct” the problem. This particular application will have to gain its efficiencies elsewhere.

Measuring Hit Rate by System and Director

- System
 - ▶ % Hit
 - ▶ % Writes
 - ▶ FE Read | Write Hit Reqs/sec
 - ▶ Prefetched Tracks/sec
- Dir-FA
 - ▶ % Hit Reqs
 - ▶ % Write
 - ▶ Slot Collisions/sec



Note: If there is enough cache to absorb all writes, hit rate will typically change with write rate.

EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 16

Several measures are available in Performance Viewer to track hit rate across the entire system and by director. Hit percent (% Hit) is the overall measure, and it is calculated from such measures as “hits per sec,” “reads per sec,” and “writes per sec.” The online help says it is the percent of IO operations performed by all of the Symmetrix volumes, for which the read data was in cache and the write operation could be sent directly to cache without having to wait for data to be destaged from cache to the disks.

The write percentage (% write) is also an important factor in measuring hit percentage. When sufficient cache is available to absorb all the writes, all writes will be hits. This means that an increase in the write ratio will typically cause an increase in the hit ratio as well. This is illustrated in the graph on this slide, which is taken from the system level.

The FE read and write hit requests per second measures are for those reads and writes that are satisfied from cache.

The “prefetched tracks per sec” measure might also be of some interest, since prefetching helps increase the read hit percentage. As we shall see, the “slot collisions per sec” (measured at the director level) is also of interest.

What Causes Misses?

- Read misses caused by:
 - ▶ Requested data not already in cache
 - ▶ Slot collision
- Write misses caused by:
 - ▶ New cache slot required Device or System at write pending threshold; no available cache slot
 - ▶ Slot collision
 - ▶ SRDF synchronous delay



Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 17

Because some of the causes of misses are somewhat obscure, they are all listed here.

Read misses are typically caused when the requested data is not already in cache and has to be staged from a physical disk. This obviously causes a significant increase in response time when compared to a read hit.

A read miss can also be recorded when a slot collision occurs. A slot collision happens when the needed cache slot or the aging (tag data) information is locked for updating by some other process. For instance, the data for a particular read might be already in a cache slot, but the slot is locked by a write operation that happened just prior to the read. This sort of activity can happen when more than one process is sharing the data, as in a multi-threaded database application. In this case, the read cannot be satisfied until the write operation completes and releases the slot.

The main cause of write misses is lack of cache slot availability. As we will see later in this module, there is a limit to the number of slots that can be allocated to hold write data that has not yet been destaged (write pending slots). If this limit is reached, a write miss is registered since the write will be delayed until room can be made by destaging data.

Write misses can also be caused by slot collisions, as described above. As an example, if a write request is made to a data location that is in cache but is currently in the process of destaging, the new write will be delayed until the destaging process finished.

Write misses can also be caused by SRDF synchronous mode delays. In this case, each write to an SRDF device must be acknowledged from the remote site before the next write is allowed. The long delay times that can sometimes be caused by this type of communication will force the next write to delay, and be recorded as a miss.

Effects on Hits: Cache Availability

- Cache Availability: More cache means:
 - ▶ Higher probability of write hits; more room for data
 - ▶ Higher probability of read hits; data remains in cache longer
 - ▶ Hit probability does not increase linearly with increasing cache sizes; for example, doubling cache may only increase read hit percentage from 57% to 68% (a 20% increase)
- Internal Synchronization: initiating BCV, Clone, SRDF, DRV, Dynamic Spare synchronizations occupies cache slots
 - ▶ Fewer available cache slots for other I/Os



Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 18

The probability of both read and write hits can be affected by the amount of cache. More cache means reads remain in cache longer, making it more likely that a subsequent read from the same location will be a hit. More cache also improves the probability that space will be available to store new writes, increasing the write hit probability.

Initiating Symmetrix internal synchronizations also has an effect on cache availability. When mirroring is first initiated between two devices within the Symmetrix, a great deal of data must often be transferred between them. Cache slots are used to buffer transfers between the DAs, reducing the overall number of slots available for other I/Os. A variety of types of devices might require periodic synchronization: BCV (TimeFinder), TimeFinder/Clone, SRDF, DRV (Optimizer), and Dynamic Spares.

Once synchronization is complete, the negative effects largely disappear*. Extra mirrors of a device have minimal impact on cache availability during steady-state, since only one copy of any read/write data are kept in cache no matter how many mirrors are used.

Note*: Extra mirroring to BCV devices does increase DA traffic as write I/Os must be destaged to more destinations. At extreme traffic levels, the extra DA activity will result in a lower destage rate, leaving writes in cache for longer periods, and reducing available cache overall.

Effects on Hits: Sequence

- Sequential Reads: Each read block follows a sequence of addresses
 - ▶ Second sequential read signals DAs to prefetch extra tracks ahead of reads
 - ▶ At first, 2 tracks (128 KB) of data are prefetched
 - ▶ Sustained sequential reads increase amount of prefetch
 - ▶ Reads that actually target the prefetched data are hits

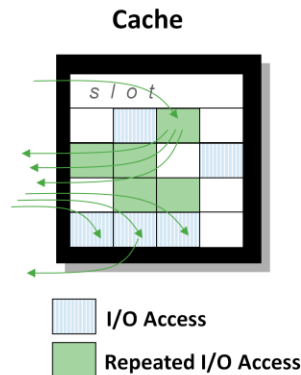
Note: If workload has large amounts of sequential reads (backup, data streaming), then expect a high hit ratio.



Sequential reads trigger the Symmetrix prefetch algorithm. Tracks of data are read into cache ahead of the current track in anticipation of future reads. Longer durations of sequential reads cause larger amounts of data to be fetched. If the read sequence continues, those reads will be hits since the data has already been fetched.

Effects on Hits: Locality of Reference

- **Locality of Reference:** “Hot Spot” of activity in a concentrated area over a short time period
- Repeat access to the same address
 - ▶ Additional reads are hits
 - ▶ Additional writes reuse the same cache slot rather than allocating a new one
- Access patterns grouped within cache slots
 - ▶ Several small writes within a single disk track reuse the same cache slot



EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

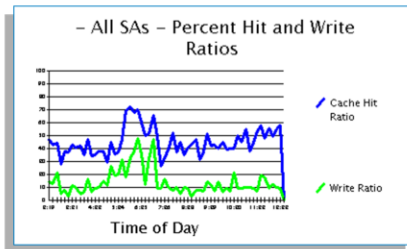
Module 5: Analyzing Symmetrix Cache Performance 20

Locality of reference describes how “random” the I/O access is. Access having high locality of reference is grouped into a small area; many of the reads and writes will be to same locations, or closely grouped locations.

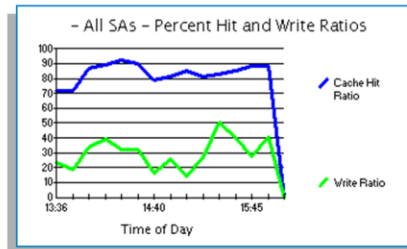
This type of access increases read hits since the same data is being read repeatedly. Write hits and cache availability are also improved since a repeated write will overwrite the old data in cache rather than allocating a new slot.

Closely grouped access patterns take advantage of the large Symmetrix cache slot sizes. Small writes to different addresses within a single track will not cause additional space to be allocated.

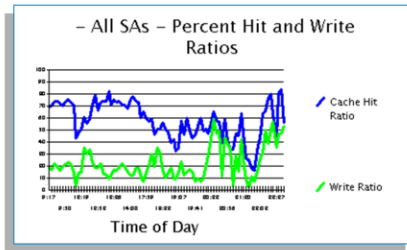
System Hit Percentage Examples



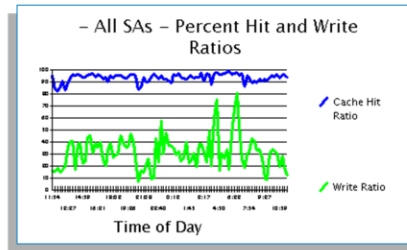
Oracle/SAP (UNIX) OLTP



MS Exchange (NT) Mail



SQL Server/SAP (NT) OLTP



Oracle (UNIX) Banking-OLTP

EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 21

Several real-world examples of hit and write percentage are displayed on this slide. These metrics are often plotted together because an increase in the write percentage will often result in an increase in the overall hit percentage. This is due to the fact that all writes will be hits if sufficient cache is available.

Module 5: Analyzing Symmetrix Cache Performance

Lesson 2: Summary

During this lesson the following topics were covered:

- Reasons for high or low hit rate
- Measuring hit rate



This lesson covered the concepts of cache hit and miss for read and write operations.

Module 5: Analyzing Symmetrix Cache Performance

Lesson 3: System and Device Write Pending Limits

During this lesson the following topics are covered:

- System and Device write pending limits
- Measuring write pending limits
- Reducing write pending issues



This lesson covers the concepts of System and Device write pending limits in a Symmetrix array.

Issue: Hitting Write Pending Limits

- A cache slot is marked as a “write pending” when it contains write data that has not yet been destaged to disk
- Write pending slots cannot be recycled for another purpose until the write is destaged
- Limit on the total number of write pending slots allocated to a device or the whole system
 - ▶ When the limit is reached, all writes to device or system are misses



A write pending cache slot is one that contains write data that has not yet been destaged to disk. These slots cannot be recycled for use until the write is destaged. In a sense, these slots are locked, while any other data in cache may be discarded at will to make room for other I/O activity.

There are limits to the number of write pending slots that can be allocated to a single device or the overall system. When one of these limits is reached, write misses occur for all writes while the write pending data is rapidly destaged to disk. This represents a significant performance issue.

System Write Pending Limit

- Maximum number of cache slots that can be marked write pending at any time for the entire Symmetrix
- On Symmetrix VMAX at Enginuity 5875 or higher:
 - ▶ The System write pending limit is set at 75% of the total number of cache slots
 - ▶ This means 25% capacity is always available for read requests
 - ▶ If all cache were consumed by writes, reads would be blocked!



The System Write Pending Limit marks the maximum number of cache slots that can be used for write pendings in a Symmetrix system. This limit is typically set to 75% of the total cache slot count, though it can be changed with Dynamic Cache Partitioning. Allowing the write pendings to occupy 100% of cache would help write performance somewhat, but there would be no remaining cache for read requests. Since most applications request more reads than writes, reserving at least 25% of the cache space for reads is important for overall application performance.

CAAR: Capacity Available for Active Replication

- CAAR is derived from the number of protection slots available in the system
 - Represents the total capacity of storage that can be actively replicated
 - We need 1 protection slot for each 1.8 GB (marketing) of data
- TF/Clone, TF/VP Snap, SRDF/Star, ORS, and Compatible Flash all use protection slots (SDDF sessions) and count against CAAR
 - The consumption of CAAR varies by replication type
- Replication Cache (the percentage allowed for protection slots) has varied by Enginuity release
 - 5875 introduced an IMPL setting “cache allocation policy”
 - Up to 17% of available cache when set to “default”
 - Up to 30% if set to “replication policy,” but system WP limit is reduced to 60%
 - 5876 lowered these slightly
 - Up to 14% of available cache when set to “default”
 - Up to 26% of available cache when set to “replication policy,” system WP 60%
 - VMAX 10K and 40K default to “replication policy”
 - The 5876 Q412 SR added an additional setting of “Snap Policy” that allowed up to 36% with a 50% system WP limit



Cache is also used for and consumed by replication products. CAAR represents the total capacity that can be actively replicated. This is predicated upon the number of cache slots available for protection slots for replication. The amount of replication cache that is available has been changed with different releases of Enginuity.

Why We Need To Understand This

- Once the CAAR limit is reached, additional replicas cannot be created
- Two choices if new sessions are needed
 - ▶ Terminate existing sessions
 - ▶ Increase CAAR
- Possible ways to increase CAAR
 - ▶ Change the IMPL Cache allocation policy
 - ▶ Add cache
 - ▶▶ Upgrade existing engines
 - ▶▶ Add additional engines primarily just for the cache
- **GOAL** is to know the limits and avoid them
 - ▶ Experience shows dealing with this issue post implementation is difficult and costly



If CAAR limit is reached, then additional simultaneously active replication sessions cannot be started. If sessions are terminated, of course new sessions up to the limit can be initiated. The amount of cache available for replication is dictated by the Cache Allocation Policy in IMPL.bin.

Vault Devices

- DMX-3 Introduced Vault Devices
- Devices configured as Symmetrix System devices
- On power failure, cache is written directly to Vault Devices
- Faster than actually destaging write pendings to drives
- A VMAX array can have up to 75% write pending limit



The DMX-3 array introduced Vault Devices to compensate for truncated system write pending limitations in the DMX-2 family. The Vault Devices are redundantly configured as Symmetrix System Devices; they cannot be mapped to a front-end port. Five gigabytes are reserved from the first four drives on each disk director loop for Vault Devices.

On a power failure, the entire contents of cache is written directly to the Vault Devices. Since this can be done as a large I/O sequential write stream, the entire cache can be written in less time than it would take to destage the individual write pending slots to their locations. When power is restored, the bootstrap program of the directors read the data back into cache and the array resumes operation.

Device Write Pending Limit

- Maximum number of cache slots that can be marked write pending for a single logical device
- All devices have the same Device Write Pending Limit regardless of size or use
 - ▶ Each member of a meta has its own Device Write Pending Limit
 - ▶▶ 4-way meta has 4X limit of a regular device



Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 29

Each logical device has a Device Write Pending Limit that restricts the number of slots that can be marked as write pending for that device. This helps prevent one device from ruining the write performance of the others by occupying too many slots. The threshold is “over-allocated” to the devices: multiplying the device write pending limit by the number of devices always produces a number that is larger than the total number of cache slots. This means that all of the devices cannot allocate their maximum write pendings at once; you will reach the system write pending limit well before this can happen.

Each device gets exactly the same limit, regardless of its size or use (regular devices, gatekeepers, VCMDDB, BCV, etc. are each assigned the same limit). In a typical environment, many of the devices will not be using any write pendings at all since slots, write pending or otherwise, are not permanently allocated to any device.

Symmetrix meta members are treated as individual devices, each having its own write pending limit. If the device write pending limit is 1000 slots, each member of the meta will have its own 1000 slot limit, giving the overall meta a larger total limit.

Device Write Pending Limit Value

- 5% of System write pending limit
 - ▶ Never increases
- For large number of devices, DMX-3+ arrays have much higher device write pending limits than DMX-1 and DMX-2



Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 30

In DMX-2 arrays and earlier, a complicated formula was used to calculate the write pending limit for a device. The formula was based on the number of devices and the overall cache size. The result of the formula was the base write pending limit of a device.

In DMX-3 and later arrays, the device write pending limit is simpler. The device write pending limit is 5% of the system write pending limit. That is, 4% of the total slot count. This limit cannot be increased for any reason.

With a DMX-3 or later, you will hit the system write pending limit if only 20 devices reach their device write pending limits. While the larger limit tends to reduce device write pending issues, you can very quickly get into trouble if you allow several devices to occupy many write pendings.

Cache Stress – DA Write Pending Limit

- DA Write Pending Limit: 50% of the System Write Pending Limit
- This measure is used in signaling “Cache Stress”
 - ▶ High Write/Read ratio is also factored into the complicated decision making on when Cache Stress is signaled
- Disk Directors give equal priority to destage and host I/O tasks during Cache Stress
- Despite the name, DA Write Pending Limit does NOT limit the number of write pendings a DA can have allocated to its devices



EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 31

Cache stress is an important measure used by the Symmetrix to determine if additional write pendings can be allocated to devices or not. One of the chief factors in determining when cache stress occurs is the DA write pending limit, which is one half of the system write pending limit. When the overall write pending count reaches this threshold, the Symmetrix enters cache stress mode. Other factors such as a very high write to read ratio can also signal cache stress. Entering cache stress causes changes in the operation of the Symmetrix. Disk directors go into a high priority destage mode under cache stress, spending an equal amount of time destaging data and servicing read miss requests. Both of these actions tend to reduce performance in the short term: writes to devices are more likely to be write misses, and read misses take longer to service as the disk directors spend additional time on destaging tasks. But, performance is greatly improved in the long term because these activities tend to reduce the build up of write pendings before they reach the system write pending limit.

Reaching Write Pending Limits

If write request is made and a new slot must be allocated to receive the data,

And

Allocating the slot would push the number of write pendings over the System or Device Write Pending Limit,

Then

The request is treated as a Write Miss. The new data is not allowed into cache until existing slots are destaged to make room. The write now moves at

DISK SPEED

EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

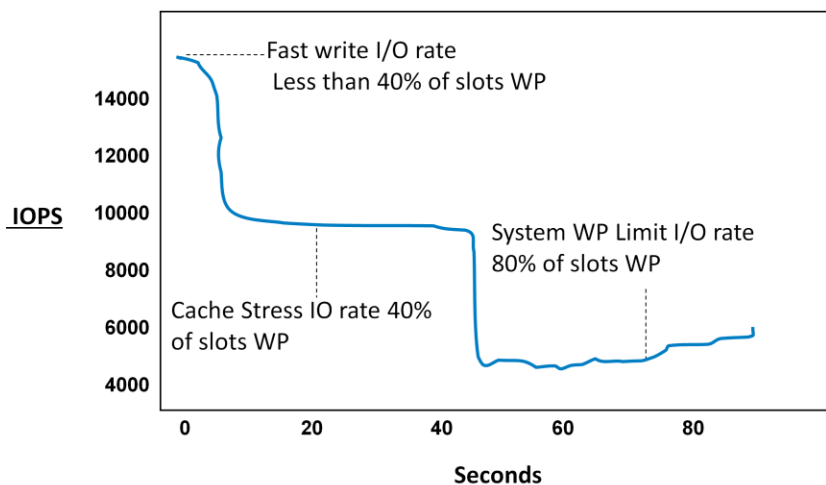
Module 5: Analyzing Symmetrix Cache Performance 32

Hitting a write pending limit has a serious effect on the write performance. If a write request is made and a new slot must be allocated to receive the data, the write pending limit is checked. If allocating the slot would push either the device or system write pending count over the limit, the request is treated as a write miss. The new slot is not allocated until existing write pendings are destaged to make room. The disk directors assign the highest priority to this destage task, since a host is being delayed.

Since an entire slot is marked with the write pending flag, multiple writes or re-writes to the same slot will not require a new slot to be allocated. This means that workloads that have a high locality of reference (frequent repeated access to the same region of the disk) will not suffer as badly when write pending limits are reached, since many of the writes will not actually be write misses.

In a write hit operation, the host only has to wait for the data to be stored in cache (a fast, electronic task). When a write pending limit causes a write miss, however, the host must wait for a slower disk operation. Frequently hitting write pending limits will negatively impact write performance.

Write Pending Limit Effects – Example



EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 33

The benchmark depicted in this slide was performed to study the effects of write pending limits on an actual workload. This was performed on a DMX. The I/O per second was carefully measured as a write intensive workload was applied to 128 Symmetrix devices. The workload was designed to produce more write requests than this particular Symmetrix was able to destage, ensuring a rapid buildup of write pendings. As the graph shows, increased numbers of write pendings had a dramatic effect on the I/O per second, which is due to a dramatic increase in the response time.

As the test first begins, the response time of the I/O was low enough that the workload generator could complete approximately 15,000 IOPS. After only a few seconds the number of write pendings reached 40% of the available cache slots, and the Symmetrix went into cache stress. The disk directors spent extra time destaging write pendings, and devices might have started hitting their device write pending limits. The attempts to deal with the incoming writes increased the response time and reduced the I/O per second to about 10,000.

After 40 seconds into the test, the System Write Pending Limit of 80% of all cache slots was reached. At this point, all writes were treated as misses as the disk directors worked to destage enough slots for any incoming writes. Since the workload was continuing to send writes faster than the Symmetrix could destage them, there was no way to resolve the situation. The response time was reduced enough to produce an I/O per second rate of about 5,000. This represents the worst case I/O rate (the cache was hardly benefiting the workload). Had the heavy write workload been a short spike rather than a constant factor, the cache stress precautions might have prevented the write pendings from ever reaching the system write pending limit.

Determining Write Pending Limits

- Use command line `symcfg` command
- DMX-3 or later:
 - ▶ Device Write Pending Slots is an absolute limit
- Shown below is an example from a Symmetrix VMAX (Enginuity 5876):

```
C:\>symcfg list -v -sid 304|more

-----
Enginuity Build Version           : 5876.251.161
-----
Cache Size (Mirrored)             : 28672 (MB)
# of Available Cache Slots         : 407888
Max # of System Write Pending Slots : 305916
Max # of DA Write Pending Slots    : 0
Max # of Device Write Pending Slots : 15295
-----
```



Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 34

The Solutions Enabler “`symcfg -v list`” command can be used to determine cache size and write pending limits. A great deal of information about the attached Symmetrix is displayed; the cache-related information is:

- **Cache Size:** The raw size of the entire cache
- **Number of Available Cache Slots:** Number of slots in the user area of cache
- **Max Number of System Write Pending Slots:** System write pending threshold. On a VMAX it is 75% of the available cache slots, if using the default cache policy.
- **Max Number of DA Write Pending Slots:** Cache stress limit, or DA write pending limit.
- **Max Number of Device Write Pending Slots:** Base limit for every individual device. If it is a DMX-3 array or later, this is the limit.

In recent versions of Enginuity, the DA Write Pending Slots measure is listed as zero. The high priority destage threshold is still 50% of the system write pending limit. In the example shown here, on a Symmetrix VMAX, the Max # of System Write Pending Slots (305916) is 75% of # of Available Cache Slots (407888). The Max # of Device Write Pending Slots (15295) is fixed at 5% of the Max # of System Write Pending Slots (305916).

Measuring Write Pendings

- System
 - ▶ System Max WP Limit
 - ▶ Local WP Count
- FE Director
 - ▶ System WP Events/sec
 - ▶ Device WP Events/sec
- Volume
 - ▶ Max WP Threshold
 - ▶ WP Count

EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 35

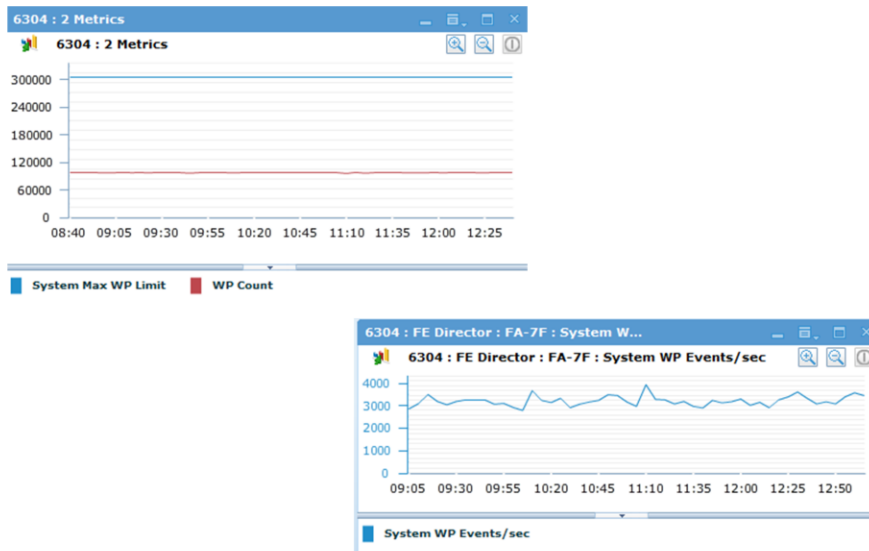
Both system and individual device write pending limits are measurable. The system limit is called “System Max WP Limit,” while the device limit is called “Max WP Threshold.” These metrics show the current maximum write pending capacity. If the limit is a constant, it will always be displayed as a straight line. Device limits in older arrays can vary between the base value and three times the base value, so it is not unusual to see “max write pending threshold” change. They will not change in a VMAX.

The actual number of slots allocated to write pendings can be measured at the system and device level.

The number of write pending events can be measured at the host director level. The “System WP Events per sec” metric measures the number of write misses per second caused by the overall system write pending limit being reached. Any number of these events indicates a serious cache problem that should be addressed immediately.

The “Device WP Events per sec” metric measures the number of write misses per second caused by a single device being at its write pending limit. Heavily accessed devices frequently reach this limit. Immediate action might not be in order, but some attention could be paid to fix the issue if the device is routinely near or at the limit.

System Write Pending Examples



EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 36

Here are two examples of system write pending analysis. On the left is a graph showing a typical write pending situation where "Write Pending Count" is well below the "System Max Write Pending Limit." The graph on the right ("System Write Pending Events Per Second") shows there are more than a few (about 3200 average) system write pending events being registered on a director (some small number always gets logged). This would be a good indicator to investigate further, especially if this has recently risen.

Solutions to System Write Pending Limit

- Spread the workload over more hardware
 - ▶ More cache, if short spike of writes
 - ▶ More backend hardware, if sustained high write workload
- Reduce writes
- Schedule heavy write jobs when no business continuity activity is happening
- Increase locality of reference, if possible
 - ▶ Writing to the same locations reuses cache slots
 - ▶ Writing to disperse locations uses new slots for each write
- Destage faster
 - ▶ Increased backend efficiency or speed



Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 37

If the system write pending limit is being routinely reached, spreading the workload out over more hardware might be a solution. More cache can absorb more writes. If your analysis of the write behavior leads you to believe a short write spike has caused the system limit to be reached, additional cache might help.

Bear in mind that adding more cache will not solve the problem if the backend hardware simply cannot keep up with the continuous rate of writes. Since the outflow of writes (destaging) is lower than the inflow, the limit will eventually be reached, regardless of how big the cache is. In this case, more or faster backend hardware might solve the problem. Since the entire Symmetrix is probably completely utilized in any scenario in which a system write pending limit is being reached, more backend hardware can only be added by moving some of the data to another array.

Make sure that any write-heavy jobs are not scheduled at the same time as business continuity activity. BC actions require a lot of cache and disk activity, and will make the write pending situation worse.

As mentioned in the discussion about cache hits, locality of reference plays a part in write pendings. Repeated writes to the same disk location will reuse the same cache slots. If ten write I/Os are directed at exactly the same track, for instance, only one cache slot needs to be allocated. Random writes to many locations will allocate many different cache slots. If ten write I/Os are directed at a wide variety of addresses, ten different cache slots will be marked as write pendings. Changing the workload to become more concentrated is, of course, not easy to do, but if it is possible, it will help this situation.

Faster destaging will help move write pendings out of cache. Compare the rate of increase of write pendings to the rate of destaging (measured at the Dir-DA level) to determine if speed can be gained there. If the write pending rate is several times higher than the destage rate, almost no change will solve the situation. But, if just a small difference is noted, rearranging the data to make the DAs more efficient, or replacing them with faster models might be a solution.

Solutions to Device Write Pending Limit

- Spread write activity to more devices
 - ▶ Host LVM management
 - ▶ Striped Meta Devices
- Rarely, add more cache if many devices are at the limit



Copyright © 2014 EMC Corporation. All Rights Reserved.

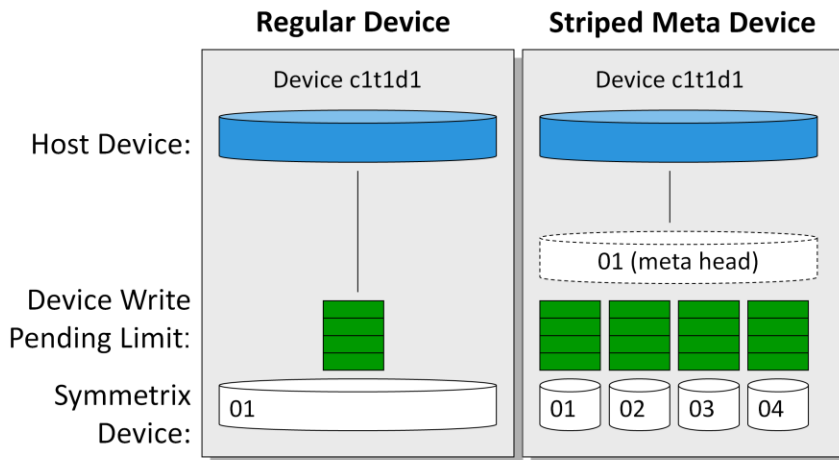
Module 5: Analyzing Symmetrix Cache Performance 38

The first suggestion to resolve a device write pending problem is to spread the data to more devices. Since each device has a fixed write pending limit, doubling the number of devices doubles the number of write pendings for the workload. Remember that this limit is independent of device size or type.

Spreading data across multiple devices can be done using host-based logical volume management tools. Alternatively, striped meta devices might be employed. Both are discussed on the next slide.

Adding more cache will rarely be a solution to this issue because there are typically thousands of devices on the array. Any new cache will be evenly divided among all of the devices, and some of it will be consumed by the system overhead. There is no way to use the added cache just for the devices that need it the most.

Migrate Data to a Striped Meta Device



EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 39

To see why a Symmetrix meta device might be a solution to the device write pending problem, let us examine both regular and striped meta devices. A large regular device is shown on the left. The host device (c1t1d1) corresponds to just one Symmetrix device (01). No matter how big it is, all devices have the same write pending limit, which is represented by the “capacity” green rectangles above the device.

The meta device shown on the right is the same size, but it is composed of four smaller Symmetrix devices. The host identifies it as a single device (c1t1d1) paired with a single Symmetrix device (01), but, internally, the I/O targeted at the head device will be split among the four actual devices (01, 02, 03, 04). Device write pending limits are linked to the actual devices, not the virtual meta. Each device has the same write pending limit as a single large device would have, meaning that this meta has four times the write pending limit as another device.

Performance Viewer reports device activity by the actual devices. Measures for the meta device pictured above could be seen by examining devices 01 through 04. Measures for device 01 would only show the activity directed at that one device, or one fourth of the I/O the host directs at device c1t1d1.

The same effect can be done by presenting four small devices to a host and using logical volume management to group them into a single logical volume. This puts additional load on the host CPU which has to split the I/O requests among the devices, but gives the host administrator greater flexibility in the device size.

Module 5: Analyzing Symmetrix Cache Performance

Lesson 3: Summary

During this lesson the following topics were covered:

- System and Device write pending limits
- Measuring write pending limits
- Reducing write pending issues



This lesson covered the concepts of System and Device write pending limits in a Symmetrix array.

Module 5: Analyzing Symmetrix Cache Performance

Lesson 4: Dynamic Cache Partitions

During this lesson the following topics are covered:

- Use of Dynamic Cache Partitioning to provide a consistent quality of cache service for applications

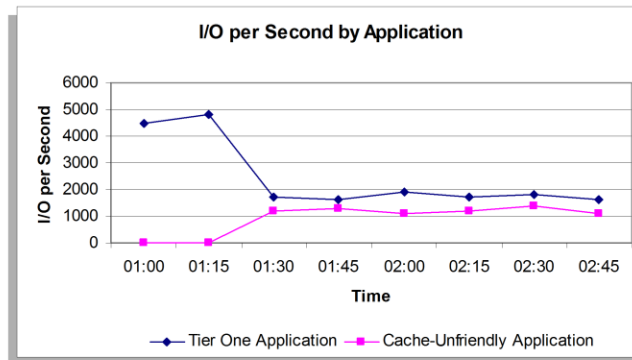


This lesson covers the concept of Dynamic Cache Partition in the Symmetrix array.

Dynamic Cache Partitioning

The Issue:

- Cache-unfriendly application steals cache resources from tier one application



The Solution: Dynamic Cache Partitioning

- Provide some level of service for distinct applications
- Up to fifteen user-defined partitions, plus one default
- Can have static or flexible boundaries

EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 42

Since cache is a globally shared resource, all applications on an array compete for cache slots. One cache unfriendly application might significantly impact the performance of other applications that have a higher hit rate. This might happen when a highly random cache unfriendly application recycles slots that were being used by a tier one application with high locality of reference.

In the past, solutions to this issue were attempted by implementing multiple Least Recently Used groups within the array. Devices added to one Least Recently Used group would recycle slots from each other first, before allocating slots from other groups. This helped isolate the cache unfriendly applications, but it did nothing for write pending issues.

The latest versions of Enginuity allow Dynamic Cache Partitioning, a much more configurable and powerful way of segregating cache. The partitions are logical entities and are spread over all available physical cache resources. Flexible or static cache partitions can be created. One default partition contains all the devices on the array, but you can create fifteen more and move devices into them. You can set static boundaries so the partitions are fixed in size, or you can create flexibility so that the devices in one partition can borrow slots from others when traffic is highest.

Uses for Dynamic Cache Partitioning

- Isolate applications that gain no benefit from additional cache
 - ▶ Very random, read once
- Prevent lower tier applications from stealing cache from higher tier
- Cache-based service level agreements
- Partitions are logical and are evenly distributed across physical cache resources
- Allocation can be changed dynamically



Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 43

The best uses for Dynamic Cache Partitioning are cited here. One use is to isolate applications that gain no benefit from cache, but use up many cache slots just the same. Examples are very random workloads, or any workload that reads data just once. These types of applications tend to allocate many slots without reusing them. This tends to “steal” slots from other applications without gaining much benefit.

Another use of cache partitioning is to prevent lower tier applications from “stealing” slots from higher tier applications. In this case, the lower tier application might actually get some benefit from the additional cache slots, but only at the expense of the more important application.

In both cases, isolating them forces the application to reuse the slots within its partition, instead of taking the slots from another application.

In advanced, performance oriented environments, business units might actually set up cache-based service level agreements. Cache partitioning makes this possible by guaranteeing some minimum level of cache.

Partition Configuration

Min % (0-100)	Target % (10-100)	Max % (10-100)
---------------	-------------------	----------------

- Measured as percent of all cache slots
- Sum of all partition targets must be 100%
- If Min=Target=Max, partition is static
 - ▶ Other partitions cannot borrow slots

Default partition: Min=0%, Max=100%, and Target must be $\geq$ 10%

Donation (fall through) Time (sec)	Write Pending % (40-80)
• Higher decreases chance of donation • Lower increases chance of donation	• Partition-wide write pending limit, measured as percent of Target

Note: Populate partitions with devices. All slots allocated for the device will be within the partition.

EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 44

There are five key settings that can be configured for each cache partition.

The minimum, target, and maximum cache percentages define the limits of the partition as a percentage of the overall cache size. The number of slots in the partition will not go under the minimum or over the maximum. The target is the preferred percentage of slots for the partition. Slots will be added to the partition by borrowing until the target is reached. If additional slots are needed to satisfy an I/O, additional slots might be borrowed, therefore pushing the total above the target, but not higher than the maximum. The sum of the targets of all the existing partitions, including the default, must be 100%.

Slots are passed from partition to partition by borrowing. The Donation Time parameter indicates how long a slot is kept in one partition before it might be borrowed by another. When the slot is not accessed for any I/O for at least the Donation Time, the slot may be borrowed by another partition. A longer Donation Time means the slots will remain in the partition longer, reducing their chance of being borrowed.

If the minimum, target, and maximum are all the same value, the partition is static, meaning the number of slots is fixed.

Each partition can have a partition-wide write pending limit. Measured as a percentage of the maximum partition size, this parameter limits the number of write pending slots in the partition.

All devices start in the default partition. To isolate an application, create a new partition and move the application's devices to it. All devices must be a member of one partition.

Flexible and Static DCPs

- Flexible partitions:
 - Have overlapping minimum and maximum settings
 - Can donate cache slots to other partitions if donation age is reached
 - Can accept cache slots from other partitions when workload varies
- Static partitions:
 - Min%=Max%=Target%
 - Partition size is fixed
 - Cannot donate/accept slots to or from other partitions
 - Provides absolute resource segregation
 - Results in predictable and controlled service levels
 - Donation age is irrelevant
- Flexible and static partitions can coexist



Static partitions are very useful for storage service providers, as these guarantee cache resources, and also simplify chargeback calculations. Flexible partitions optimize cache utilization and can improve overall performance. If a flexible partition has cache slots that have not been accessed within the donation age, it can donate them to another flexible partition that requires it, due to its sudden variation in workload. With flexible partitions, cache slots can migrate between different flexible partitions based on demand and usage, or access. When cache slots are migrated to a different partition, their usage calculation begins anew. So, their time since last access is reset and recalculated in the new partition before further migration. The higher the donation age, the lesser the chance of cache slot migration. This is because it requires a long interval between cache slot reuse before donation can take place. A lower donation age will increase the possibility of migration to a different partition. Cache slots will be accepted by a partition only until the maximum percentage is reached.

Getting Started with DCP

1. Determine the number of distinct service classes for all the applications
2. Start with an equal distribution of Target% among all partitions so that the sum of the Target% equals 100%
3. Set Min%=0, Max%=100, and Donation age=0 seconds
4. Set Analyze mode and enable QoS Monitoring for Cache Partitioning
5. Run the various workloads to get a representative sample
6. Use the graphs in Performance Viewer to determine the exact utilization
7. Enable Cache Partitioning



Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 46

If the number of service classes exceeds eight, then applications must be appropriately grouped. The Symmetrix Logical Volumes used by the applications are the ones that are assigned to different partitions. Assigning SLVs to partitions can be done at a SYMCLI Device Group Level for greater convenience. Group all related SLVs into a SYMCLI DG and assign the DG to a named partition. If precise target allocation is not known, then it is preferable to equally distribute the allocation among all partitions. Minimum=0, Maximum=100, and Donation age=0 are automatically set, if using Analyze mode. Once the cache utilization patterns have been determined, then Cache Partitioning can be enabled. Note that one can enter Analysis mode only if Cache Partitioning is in a disabled state. The typical write pending limit would be 75%, if using the default DCP. When a partition's WP count approaches the WP limit, host throttling will start. When the WP count reaches the WP limit, priority destage will begin.

Common Use Cases for DCP

- Dynamic partitions to optimize workloads
 - ▶ Create Flexible partitions
 - ▶ Allow some partitions to donate slots by setting the same donation age for them
 - ▶ Set a higher donation age for the other partitions to reserve slots for them
 - ▶ Min/Max is not equal to the Target for the flexible partitions
 - ▶ Vary the WP limit if necessary
- Static partitions to fence workloads
 - ▶ Example is a service provider supporting multiple customers
Min/Max equal to the Target
 - ▶ Donation time is ignored
 - ▶ Predictable cache performance for each application class



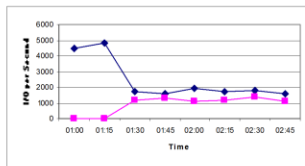
Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 47

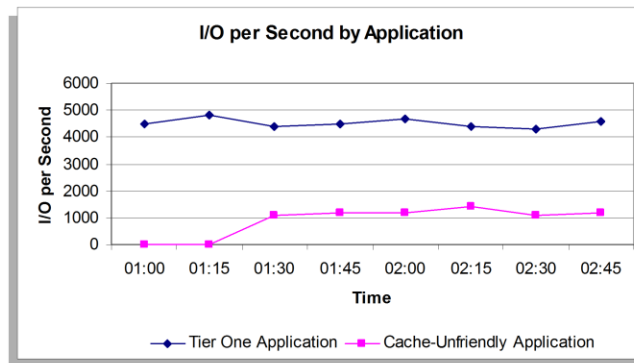
To optimize workloads, flexible partitions can be created. Setting a lower donation age will allow cache slots to be donated between partitions when slots are underutilized. A higher donation age for some partitions will ensure the availability of cache. The write pending limit can be varied as well, based on the type of workload.

In the case where each application should be guaranteed a defined amount of cache, static partitions can be created, and cache slots will not be donated. The applications are segregated and will not interfere with each other's cache availability.

Example



Before



After Partitioning

- **Partition T1:** Tier One Application
 - ▶ Min=30%, Target=40%, Max=80%, Donation=900, Write Pending=80%
- **Partition T2:** Cache Unfriendly Application
 - ▶ Min=0%, Target=30%, Max=50%, Donation=300, Write Pending=80%
- Default Partition automatically gets Target=30%

EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 48

Here is a hypothetical solution to the scenario that started this topic. Create two partitions to segregate the devices from the applications. Set the parameters of the tier one application to be slightly higher than the cache unfriendly application. In particular, it has a higher target and donation time.

By isolating the cache unfriendly application, it cannot “steal” cache from the tier one application. This creates much better results for the tier one application; it is almost unchanged even after the cache unfriendly application starts. The cache unfriendly application performs about the same, since it not reusing cache slots effectively.

Measuring Dynamic Cache Partitioning

- Dynamic Cache Partitions
 - ▶ Write Pending Utilization by Cache Partition
 - ▶ % Hit by Cache Partition
 - ▶ % Cache used by Cache Partition
 - ▶ System Write Pending Count
 - ▶ WP Count by Cache Partition
 - ▶ System % Hit
 - ▶ % Hit by Cache Partition
 - ▶ % Read | Write Hit by Device Group
 - ▶ WP by Device Group
 - ▶ % Read | Write Hit by Storage Group
 - ▶ WP by Storage Group



Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 49

There are many hit ratio and write pending measures recorded for dynamic cache partitions. By clicking on the Dynamic Cache Partition tab, you can show time series and snapshot graphs that compare the performance of any or all of the partitions, or show the total performance across all partitions. By clicking on a single partition, you can show the performance and characteristics of just that partition.

Measures are primarily focused on the cache issues that dynamic cache partitions were created to address (hit ratio and write pendings). You can quickly determine if the devices in a partition have a better or worse hit percentage after adjusting the partition parameters. You can also watch the write pendings of each partition to make sure they are not running out of space.

If dynamic cache partitioning is not enabled on the array, these views will show only the default partition. Every device is considered part of the default partition, so in this case you can use these measures as array-wide indicators. There are many metrics available here that also allow measuring the cache slot aging, fall thru, acquisition, and donation counts for partitions as well.

Module 5: Analyzing Symmetrix Cache Performance

Lesson 4: Summary

During this lesson the following topics were covered:

- Use of Dynamic Cache Partitioning to provide a consistent quality of cache service for applications



This lesson covered the concept of Dynamic Cache Partitioning in the Symmetrix array.

Module 5: Analyzing Symmetrix Cache Performance

Lesson 5: Alignment of I/Os With Cache Slots

During this lesson the following topics are covered:

- Relating I/O misalignment with the inefficiencies it causes
- Measuring I/O misalignment



This lesson covers the issues caused by I/O misalignment and some possible solutions to the problem.

Issue: Cache Misalignment

- Cache slots are tied to strict device addressing
- I/O that is not aligned to the strict addressing can cause:
 - ▶ Additional internal transfers (higher bus/matrix utilization)
 - ▶ Inefficient use of cache resources (cache space wasted)
 - ▶ Physical pre-reads before write (higher bus/matrix and disk utilization)

EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 52

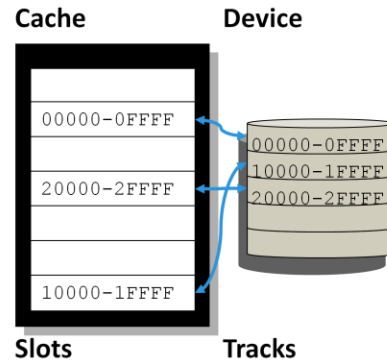
The lesson discusses the strict cache addressing scheme in which cache slots are tied to fixed device address regions. While several performance problems are caused by the strict nature of this addressing, it is designed for rapid stage/destage operations.

As this lesson shows, I/Os that are not aligned properly to the cache addressing scheme will cause:

- Additional internal transfers between directors and cache
- Inefficient use of cache resources; a large number of cache slots might be required to hold a small amount of data
- Physical pre-reads to initialize a cache location with data before a write

Cache Layout

- Each cache slot is a fixed size
- Each *slot* represents a same-sized *track* from a device
- Data stored in a slot must conform to the track addressing scheme
- Designed for fast stage/destage operations



EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 53

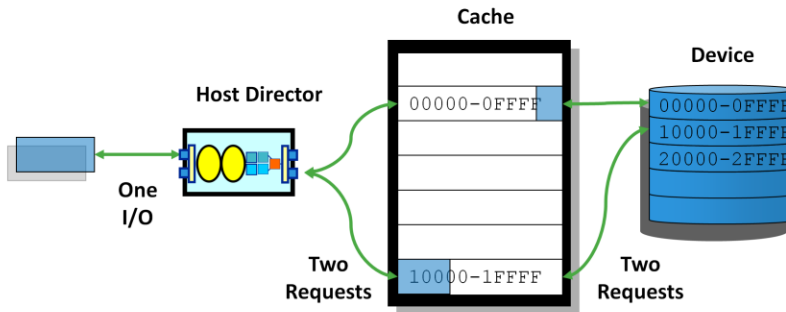
Each cache slot represents a track of an actual device (64 KB on a VMAX). While the slot is allocated for use by the device, all I/O is addressed to the cache slot as if it were the actual device. The address of all data in the slot must line up in exactly the same order as it would if it were stored on the device.

In the diagram on this slide, the first three tracks of the device (starting at addresses 00000, 10000, and 20000 hexadecimal) have been allocated to cache using a 64 KB slot size. Perhaps a read request was made of the first two, and a write was done to the third one. Due to other slots being in use, the three slots reserved for the device are distributed through the cache. From this point until the time that those slots are recycled, all I/O traffic for the indicated addresses of the device will be referenced through those slots. Any data read from or written to the device that falls in the address range 00000-0FFFF will be processed through the cache slot allocated to that address range.

While this scheme appears overly restrictive, it is designed for very fast stage/destage operations.

Track Misalignment

- **Misalignment:** A single host I/O crosses a cache slot/device track boundary
- Inefficiencies:
 - ▶ Causes two requests (internal bus/channel transfer) to satisfy the one I/O
 - ▶ Can waste cache space; ex: 2 Cache slots used to hold an 8 KB write



EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 54

Inefficiencies can result when I/Os do not align to the strict cache addressing scheme. This can cause additional requests (internal bus or channel I/O), increasing the utilization of the Symmetrix internal pathways. In very random workloads, misalignment can also result in additional cache slots being allocated to store small amounts of data.

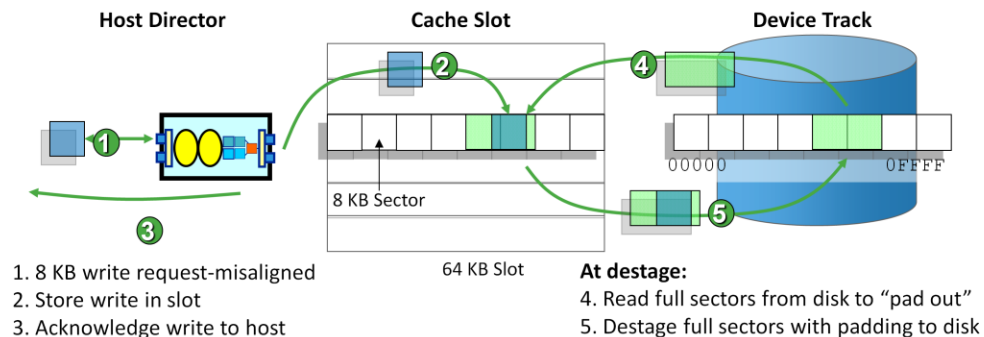
In the example illustrated, a host has written an 8 KB block of data to a device. The starting and ending addresses of the block are in two different tracks of the device, and the I/O crosses a track boundary. Since slot addresses must begin at 64 KB boundaries and the data within them must not “float” around the addresses, two slots will have to be allocated to store the write.

Although the host director received one write from the host, two requests (transfers from the director to cache) are required to store the data into cache. Two more requests will be required to destage the data to disk. This requires more director work and bus or channel activity than if only one request was needed.

The misalignment problem can be especially wasteful for small random blocks. With random I/O, data does not often occur in adjacent addresses, making it unlikely that the rest of the allocated tracks will be used. In the example one this slide, 128 KB of cache space is being used to store the one 8 KB write. More I/O falling within the addresses represented by the two slots (00000 to 1FFFF) will better utilize the space, but if the workload is very random, such I/O might not occur before the slots are recycled.

Sector Misalignment

- **Sector:** Minimum size data block accessed to/from device track
- **Misalignment:** A single host I/O does not begin or end on a Sector boundary
- Inefficiencies:
 - ▶ Can cause one or more pre-reads before destage



EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 55

Misalignment can occur at the sector level. Within the Symmetrix, data is always retrieved from/stored to a physical device on sector boundaries. Sectors are 8 KB on the VMAX array. Writes that do not fall on these boundaries can cause extra data to be read from the device to properly fill the addressing space before the data is destaged.

In the illustration on this slide, a 8 KB write that is misaligned to the 8 KB sector boundaries arrives at the host director. Since this write overlaps two sectors, a full two sector destage will eventually have to be done. But, half of the data in those two sectors did not come from the host or from this device; it is data leftover from some other transaction.

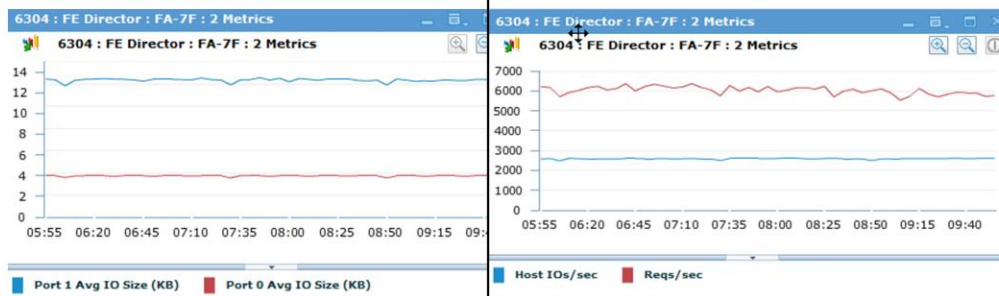
The two affected sectors must first be read from the device to overwrite the leftover data. This "pads" the sectors to their full size. When destaging occurs, the whole 16 KB block (two sectors) can be written to the proper sector locations on the device without corrupting the data.

The Symmetrix is capable of reading/writing blocks in sizes much larger than a single sector. In the example, when a pair of adjacent sectors needs to be transferred, it would happen in a single 16 KB read.

Intelligent Symmetrix algorithms skip the pre-read process whenever it is not necessary. For instance, if the misaligned write is being stored in a cache slot that is already allocated to that device, it could easily be that the "boundary" data is actually valid. It may have been read from the correct device by a previous I/O. In this case, pre-reading data from the physical device would not be necessary. Also, remember that pre-reading is only performed for writes, never for read I/Os.

Measuring Track Misalignment

- FE Director
 - ▶ Host IOs/sec
 - ▶ Reqs/sec
- Port 0|1 Avg IO Size (KB)
- Caveat:
 - ▶ I/O larger than the slot size requires more than one request



EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 56

The misalignment issue is not directly measurable in Performance Viewer, but the number of I/O per second and requests per second can be compared. If the requests are significantly greater than the I/O, then a large percentage of the traffic may be misaligned.

There are two important caveats, however. An I/O greater than the slot size will always require more than one request. The I/O must be placed in multiple cache slots, and each front-end director to slot transaction counts as a request. If the request count is much larger than the I/O count, make sure you check the I/O size as well.

In DMX-3 arrays and higher, each write must be mirrored to two cache locations. However, the two writes are still counted as only one request. This is necessary to preserve the request measure as an indication of misalignment.

The two graphs on this slide show a wide difference in host I/Os and requests.

Is a larger I/O size causing the greater requests for the director?

The average I/O size for both of these ports directors is below 64 KB, so this factor would not make a difference. From this investigation we must conclude that the data may be misaligned.

The “Host IOs per sec” and “Requests per sec” measures are available for backend (DA) directors also. At the backend, “I/Os” means reads or writes from physical disks, while “requests” means a transfer to/from cache. Since the disk directors manage their own I/O operations, analysts usually do not consider these measures.

Solutions

- Align (start) volume groups and logical volumes to 64 KB boundaries
 - ▶ Use volume management software to align the data region
- Align (start) filesystems, data files, etc. to 64 KB boundaries
 - ▶ Many filesystems have configuration information at the beginning of the structure
 - ▶ Make sure the data blocks begin on a track boundary
- Ideal data allocation size is track size (64 KB)
 - ▶ File system minimum allocation size
 - ▶ Database block size (Exchange Server defaults to 4 KB, Oracle to 8 KB, etc.)
- If 64 KB allocation size impractical, at least use a multiple or divisor:
 - ▶ 8 KB, 16 KB, 32 KB, etc.



Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 57

Solutions to the alignment problems are to align reads and writes to sector and track boundaries. This is not such an easy task, however, because of the multiple layers of control that a typical host has over the data access. For example, a database might access a data file in a filesystem stored in a logical volume created from a volume group based on a Symmetrix disk. Each data resource, database, filesystem, logical volume, and volume group, might cause misalignment.

Aligning each of the elements can be done with the software used to create the structure. Use volume management software to make sure the data region of a volume group is aligned to a 64 KB boundary. This can often be done by extending any configuration region to end at a 64 KB boundary. Alignment can be achieved in a similar way when creating filesystems.

Once the data regions are aligned, the I/O sizes should be set to a multiple or divisor of 64 KB to avoid track misalignment. Database preferred block size and filesystems minimum allocation size can be set to enforce this alignment.

If it is impossible to restrict I/O alignment to 64 KB boundaries, at least take time to align the I/O to sector boundaries. This will, at least, eliminate the sector misalignment issue.

A Request-Related Issue

For Directors (front and back), Reads and Writes are based on **different** measures:

- I/Os per second = number of I/Os per second
- Read requests are data transfer requests from cache by the director
- Write requests are data transfer requests to cache by the director

Note: Read or write request doesn't equal an I/O (for directors only)



Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 58

Here is another caveat about the measures presented in Performance Viewer: The read and write requests measures for directors are based on different counts within the Symmetrix itself.

Remembering that a single host read or write I/O can turn into multiple requests due to misalignment or large I/O size, you can see that some unusual math will result. In particular, the sum of the "FE Read Reqs per sec" and "FE Write Reqs per sec" will typically not be equal to the "ios per sec" measure.

This should not cause too much difficulty during analysis, since both reads and writes will match the host I/Os exactly when measured at the Device or System level. It is only when measuring the director numbers that a slight discrepancy may be found.

Module 5: Analyzing Symmetrix Cache Performance

Lesson 5: Summary

During this lesson the following topics were covered:

- Relating I/O misalignment with the inefficiencies it causes
- Measuring I/O misalignment



This lesson covered the issues caused by I/O misalignment and some possible solutions to the problem.

Module 5: Analyzing Symmetrix Cache Performance

Lesson 6: XtremSW Cache

During this lesson the following topics are covered:

- XtremSW Cache
- Integration with VMAX
- Metrics available



This lesson covers the integration of XtremSW Cache with the VMAX and the measurements available.

What is XtremSW Cache 2.0?

- Intelligent software to manage frequently used data
- Previously known as the software part of VFCache (last release v1.5 SP3)
- Accelerates reads
- Passes through writes
- Delivers performance with protection:
 - ▶ I/O latencies of 100µs or less
 - ▶ IOPS requirement in the hundreds of thousands
- XtremSW Cache 2.0 is no longer bundled with a Flash card



EMC²

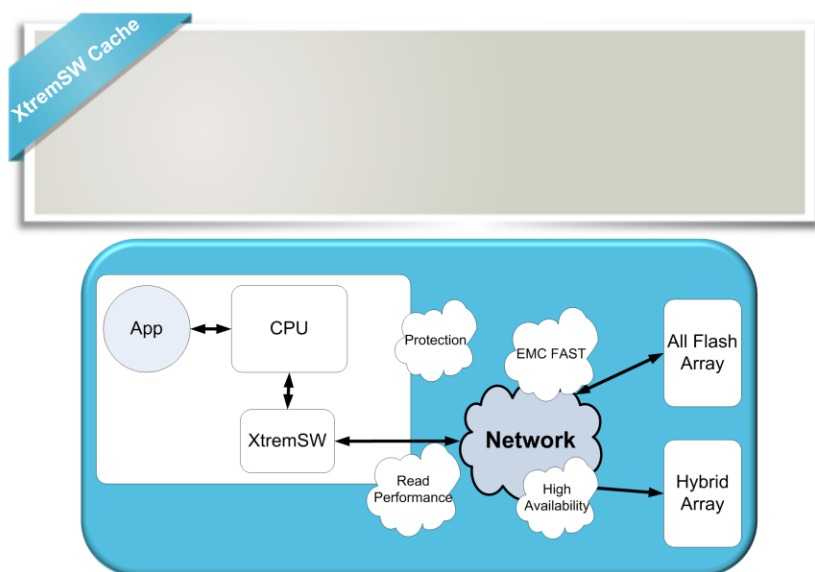
Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 61

EMC XtremSW Cache 2.0, previously known as the software part of VFCache, is intelligent software that is installed on a server to accelerate read I/O requests from a PCIe Flash Card or Solid State Drive (SSD). This dramatically reduces read I/O latency and accelerates application performance. Write I/O requests are passed through to a traditional backend storage array via the server's HBA/CNA. It is considered write-through cache, as any write I/O is not considered complete until it is acknowledged by the backend storage system, making it suitable for the most mission critical data. EMC XtremSW Cache 2.0 provides performance with protection. It can deliver I/O latencies of 100 microseconds or less and hundreds of thousands of IOPs per second for read intensive workloads.

Note: Unlike prior versions, XtremSW Cache 2.0 software is not bundled with PCIe hardware.

What is XtremSW Cache?



EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 62

EMC offers a portfolio of Flash technologies that span the storage spectrum, from the storage array to the host.

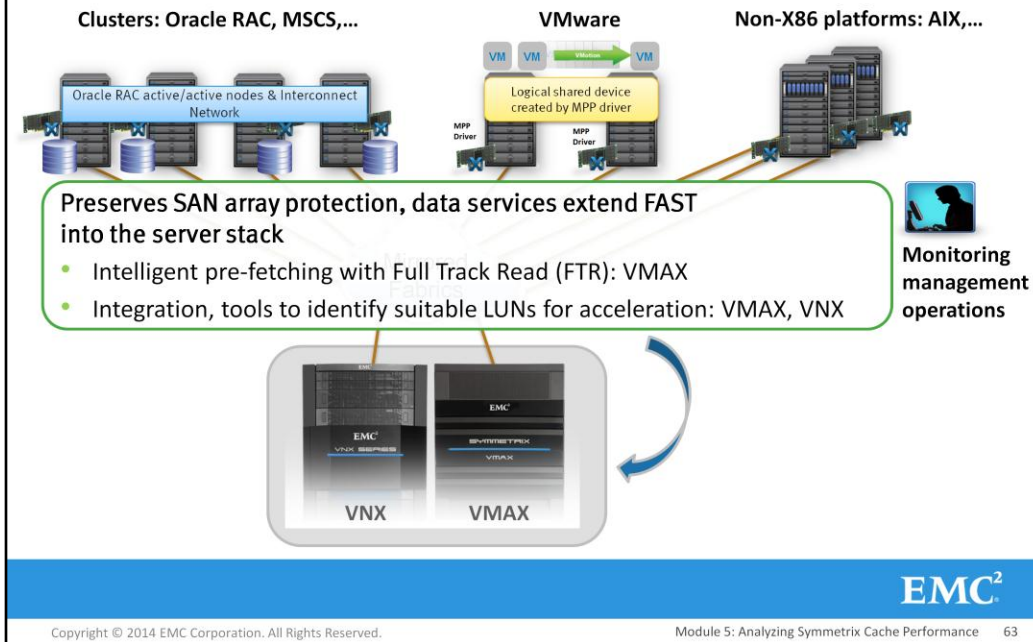
It is important to differentiate between EMC's server-based Flash solutions.

EMC solutions within the server include two options, one is cache-based and the other is a local Store or DAS (Direct Attached Storage) solution.

With XtremSW Cache, previously known as VFCache, we leverage all of the features of EMC's full featured storage arrays by tightly integrating with VNX and VMAX storage array platforms offering hybrid and/or Flash storage, HA, storage tiering (FAST), and native five 9's protection. In addition, XtremSW Cache also offers high-performance metrics that are specific to particular workloads, such as high I/O read characteristics.

While XtremSW Cache offers tremendous performance levels beyond typical network Hybrid and even all-flash arrays, we can go one step further.

XtremSW Cache 2.0: Deeper Integration with EMC Storage



XtremSW Cache 2.0 significantly improves integration with EMC storage arrays, as a key market differentiator. XtremSW Cache extends the FAST Intelligence into the server layer. The VMAX array is aware of accelerated devices and performs intelligent pre-fetching into VMAX cache for these devices, improving the achievable read hit ratio for the XtremSW Cache driver. Both VMAX and VNX provide the integration and tools to help in selecting suitable devices/LUNs for acceleration. Prior investment in the protection and the distinct data services offered on each storage array platform is thus preserved.

VMAX Integration: Intelligent Pre-Fetching

- **Full Track Read (FTR):** VMAX prefetches data to array cache for accelerated devices
 - ▶ New Symmetrix flag changes VMAX behavior for accelerated devices
- VMAX reads a full track for each host I/O request into its cache slot
 - ▶ XtremSW Cache default page size is 8KB
 - ▶ Symmetrix slot size is 64KB
 - ▶ Improves XtremSW Cache read hit %
- VMAX automatically enables Optimized Read Miss for accelerated devices on EFD
 - ▶ Further enhances performance



Customer Benefits

- Improved cache hit ratios
- Better application performance
- More efficient utilization of VMAX resources

EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 64

XtremSW Cache 2.0 provides improved integration with other products within the EMC ecosystem. Prior versions introduced integration with VMAX via SCSI in-band, using vendor-unique mode pages for the VMAX.

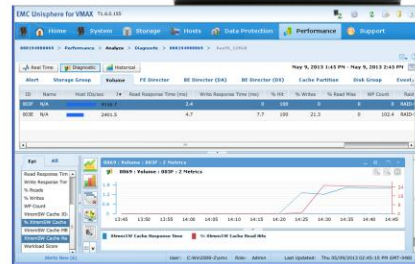
Version 2.0 introduces additional features leveraging this integration mechanism. With VMAX, by default, the array will do intelligent pre-fetching for those devices that it recognizes as accelerated devices. This has been shown to improve performance across the board for a variety of tested applications.

Pre-fetching is implemented using the Full Track Read (FTR) feature on the VMAX. For devices that are accelerated, FTR is automatically enabled. FTR results in the entire track (64 KB) to be read into the VMAX cache, for every host I/O request to an accelerated device. FTR can improve the read cache hits for XtremSW Cache, resulting in better performance for the accelerated devices.

For accelerated devices on EFD, the VMAX also automatically enables the “Optimized Read Miss” (ORM) flag. With ORM enabled, the VMAX will bypass its internal cache on a read miss workload. This significantly improves the service time whenever XtremIO Cache needs to process a read miss on the server.

Improved Intelligence and Reporting with VMAX Integration

- LUN selection feature
 - ▶ Ranking based on VMAX reported data
- Array recognition and reporting of devices under XtremSW Cache control
 - ▶ Key performance statistics reporting
- Alerting and Call Home support



Customer Benefits

- Support for effective selection of VMAX source devices
- Cache aware reporting by storage array management tools
- Centralized management and alerting with potentially large number of XtremSW Cache hosts
- Further leverage existing investment in production VMAX

EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

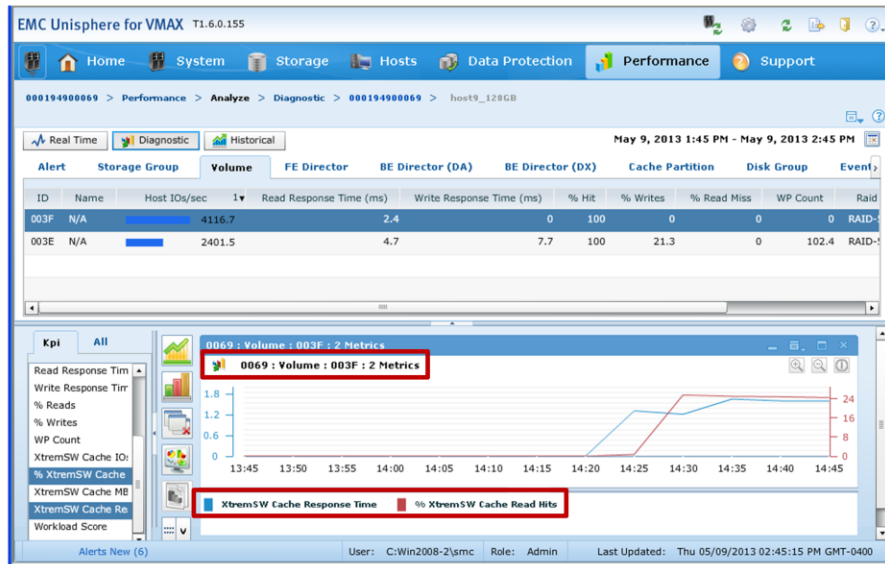
Module 5: Analyzing Symmetrix Cache Performance 65

XtremSW Cache 2.0 offers a LUN selection feature for VMAX devices on the host, i.e., a way to determine which of several VMAX devices are good candidates for acceleration. It does this by querying the VMAX for key I/O statistics on these devices, and using an internal algorithm to rank all VMAX devices with a score from 0-10, with high scores representing promising candidates. This can be particularly useful in large deployments with many hosts, where an exhaustive manual analysis of all the storage devices is not practical. Note that LUN selection can be done based on prior I/O activity by pulling historical data from the VMAX. Alternatively, the selection feature can also be run for a specific time, for example, during a production window, to make recommendations based on current I/O activity.

In addition, VMAX utilities such as Unisphere for VMAX and Symmetrix Performance Analyzer can provide more detailed reporting related to XtremSW Cache. These reports are based on the more detailed information that version 2.0 software sends to the array.

Call Home via VMAX was introduced in a prior version of the caching software, as a means to facilitate centralized reporting of cache-related events for all Flash-enabled servers in the data center. Version 2.0 expands the list of events being reported to the VMAX. Both card-related events (for XtremSF cards used as cache devices) and cache-related states and events are raised to the VMAX. The VMAX filters these events for criticality, and selectively reports them via a configured ESRS gateway to EMC support.

Improved VMAX Reporting – Example



EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

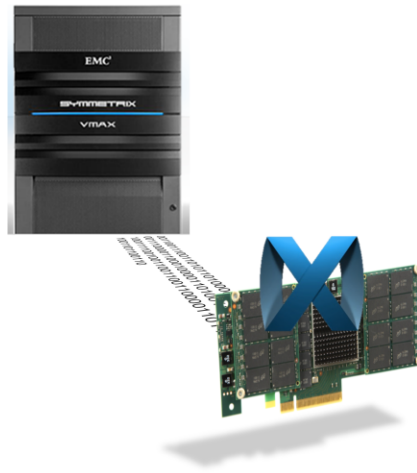
Module 5: Analyzing Symmetrix Cache Performance 66

Shown here is an example of XtremSW Cache statistics. Note that Unisphere for VMAX provides highly customizable filtering and graphing capabilities. For example, it is possible to report the caching rate and latency information array-wide, or on a host-by-host basis, or on any specific set of volumes.

In this example, Unisphere for VMAX has been configured to display two metrics, the cache hit rate and the XtremSW Cache response time for the accelerated device 003F.

VMAX Integration: Intelligent Pre-Fetching

- Full Track Read (FTR): VMAX prefetches data to array cache for accelerated devices
 - ▶ New Symmetrix flag changes VMAX behavior for accelerated devices
- VMAX reads a full track for each host I/O request into its cache slot
 - ▶ XtremSW Cache default page size 8KB
 - ▶ Symmetrix slot size is 64KB
 - ▶ Improves XtremSW Cache read hit %
- VMAX automatically enables Optimized Read Miss for accelerated devices on EFD
 - ▶ Further enhances performance



EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 67

XtremSW Cache 2.0 provides improved integration with other products within the EMC ecosystem. Prior versions introduced integration with VMAX via SCSI in-band, using vendor-unique mode pages for the VMAX.

Version 2.0 introduces additional features leveraging this integration mechanism. With VMAX, by default, the array will do intelligent pre-fetching for those devices that it recognizes as accelerated devices. This has been shown to improve performance across the board for a variety of tested applications.

Pre-fetching is implemented using the Full Track Read (FTR) feature on the VMAX. For devices that are accelerated, FTR is automatically enabled. FTR results in the entire track (64 KB) to be read into the VMAX cache, for every host I/O request to an accelerated device. FTR can improve the read cache hits for XtremSW Cache, resulting in better performance for the accelerated devices.

For accelerated devices on EFD, the VMAX also automatically enables the “Optimized Read Miss” (ORM) flag. With ORM enabled, the VMAX will bypass its internal cache on a read miss workload. This significantly improves the service time whenever XtremIO Cache needs to process a read miss on the server.

VMAX Automatic LUN Selection: Features

- Helps to identify promising candidates for acceleration among VMAX devices
- Computes and displays a score between zero and ten for each candidate device
 - ▶ Devices with the highest scores benefit the most from acceleration
- Uses VMAX statistics for each device to compute the score
- This is used for presales and as a post-deployment tool



EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 68

VMAX automatic LUN selection is a feature that helps identify provisioned VMAX devices that can benefit from acceleration.

The XtremSW Cache software gets several relevant statistics for each provisioned VMAX device from the array and, using these statistics, computes and displays a recommendation score, from zero to ten, for that device. Devices with the highest scores are expected to benefit the most by being configured as source devices.

How the Score is Computed?

The score is computed by
XtremSW CACHE



- The algorithm uses VMAX device statistics as input:
 - ▶ Number of reads
 - ▶ Number of writes
 - ▶ Number of read cache hits
 - ▶ Number of write cache hits
 - ▶ Average I/O size
- Recommendations for a device can be low, medium, or high
 - ▶ Results are based on a computed score between 0 and 10

EMC²

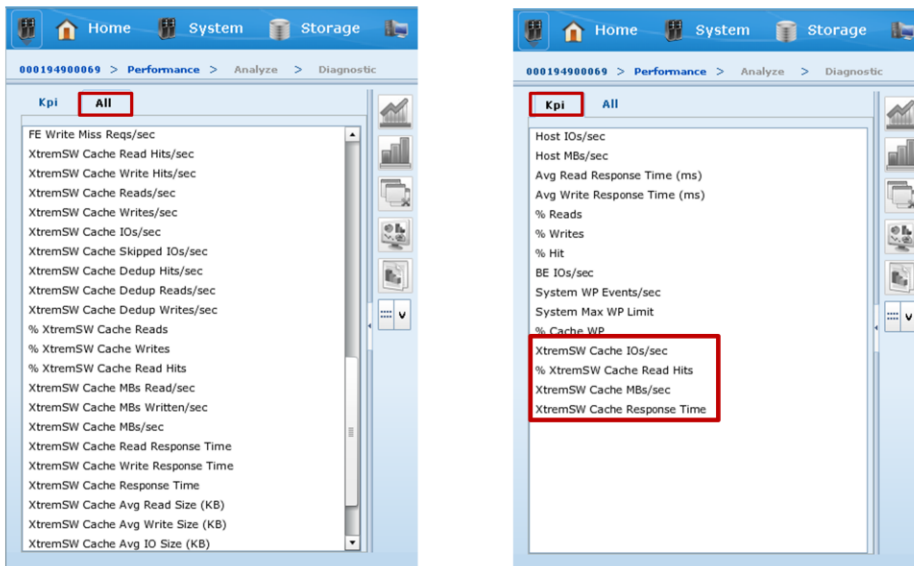
Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 69

The score is computed by XtremSW Cache using an internal algorithm that uses several types of statistics on each VMAX device. These statistics, which the VMAX sends, include: the number of reads, writes, read cache hits, and write cache hits. Additionally, the average I/O size is also considered, since I/Os larger than maxIO will not benefit from caching.

The score is used to provide an overall recommendation of low, medium, or high for each candidate device.

Unisphere for VMAX: Reported Metrics



EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 70

A large number of XtremSW cache metrics are reported by Unisphere for VMAX. Specifically, Performance Viewer can report various key rates such as reads/sec, writes/sec, read hits/sec, dedup hits/sec, etc. It can also provide information about latencies and I/O sizes for accelerated devices.

In this view we can see all the XtremSW Cache metrics that can be selected and reported in the all pane on the left. Four of these metrics are included in the KPI pane, as shown. This enables the administrator to monitor the status and effectiveness of server flash cache for provisioned volumes.

Module 5: Analyzing Symmetrix Cache Performance

Lesson 6: Summary

During this lesson the following topics were covered:

- XtremSW Cache
- Integration with VMAX
- Metrics available



This lesson covered the integration and measurement of XtremSW Cache.

Module 5: Analyzing Symmetrix Cache Performance

Lesson 7: Roadmap and Analysis

During this lesson the following topics are covered:

- Roadmap metrics
- Analysis questions



This lesson covers the metrics and for analyzing cache performance.

Symmetrix General Characterization Roadmap

General Symmetrix Characterization Measures:

Metric	Issue
• % Cache WP • % Hit • System WP Events/sec	Use this to characterize overall effectiveness of cache.
Metric	Issue
• Host IOs /sec • % Writes • Host MB's/sec • Host MB's Written/sec	Use any of these measures (and others discussed) to characterize the overall Symmetrix workload.

EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 73

These are some of the metrics that can be used to analyze cache related issues.

Cache Measures Roadmap

Dynamic Cache Partitions:

Metric	Issue
• % Cache Used • % Hit • % Hit Ratio (%) By Device Group • % Hit Ratio (%) By Storage Group • % WP Utilization	• Are critical partitions getting good cache performance? • Are groups of devices getting good cache performance? • Is hit ratio changing over time?

EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 74

These are some of the metrics that can be used to analyze cache related issues.

Roadmap (Continued)

Write Pendings:

Metric	Issue
- System Max WP Limit - WP Count	Significant performance danger if "System Max WP Limit" approaches "WP Count."
Volume level: - Max WP Threshold - WP Count	Look for devices whose "write pending count" is consistently at "max write pending threshold."

Cache Alignment:

Metric	Issue
- Host IOs / sec - Requests per sec	Do "requests per sec" significantly outnumber "ios per sec?" Possible misalignment.
Port 0 1 Avg IO Size (KB)	Are I/O sizes greater than slot size? If so, request to I/O mismatch may not be an indicator of misalignment.

EMC²

Copyright © 2014 EMC Corporation. All Rights Reserved.

Module 5: Analyzing Symmetrix Cache Performance 75

This slide is a continuation of the available metrics in Performance Viewer as they relate to cache performance.

Analysis Questions

Some additional questions to consider while analyzing:

- Are system-wide or partition write pending counts becoming excessive?
- Are critical partitions reaching write pending limits?
- Does write pending count vary over time?
- Are some groups accumulating excess write pendings?



Listed here are some additional questions you may want to ask during the discovery process.

Analysis Questions (Continued)

Higher hit rate produces better performance:

- Is hit rate changing unexpectedly?
- Does it match workload characteristics (random, sequential, etc)?
- Are misses being caused by sector misalignment or slot collisions?
- Are cache resources being consumed by a business continuity synchronization (covered in a later module)?
- When sufficient cache is available, a change in write rate is reflected in % hit. Is change in % hit caused by change in % write?



Listed here are some additional questions regarding hit rate percentages you may want to ask.

Module 5: Analyzing Symmetrix Cache Performance

Lesson 7: Summary

During this lesson the following topics were covered:

- Roadmap metrics
- Analysis questions



This lesson covered the metrics for analyzing cache performance.

Module 5: Summary

Key points covered in this module:

- Symmetrix architecture, I/O operations, and optimization algorithms
- Factors affecting hit rate
- System and Device write pending issues
- Cache alignment issues
- XtremeSW Cache integration



This module covered analyzing the performance of Symmetrix cache.

This slide is intentionally left blank.



This slide is intentionally left blank.