VeloBit™ SSD Caching Software Evaluation

Evaluation report prepared under contract with VeloBit Corporation

Introduction

With SSDs taking the enterprise by storm, one way to deploy them is as a cache in front of existing storage. The competition is heating up in the SSD caching marketplace, as vendors both large and small are introducing SSD caching products.

One of these is the VeloBit HyperCache 1.1 SSD caching software. Because it is a host-based software solution, it can work with any SSD in any form factor, including PCIe cards and disk-drive form factor SSDs. HyperCache is based on optimization techniques applied in four areas:

- Content Locality Caching, which caches data blocks based on the "popularity" of their contents
- In-line compression that uses a "similarity detection" algorithm
- A horizontal architecture that allows data that is not cached to bypass the SSD, minimizing SSD writes
- A conservative insertion protocol that places data blocks on the SSD in such a way as to minimize wear, extend the life of the SSD and improve performance

VeloBit commissioned Demartek to evaluate its HyperCache 1.1 SSD caching software in the Demartek lab in Arvada, Colorado, measuring its performance in Linux and VMware environments. These performance tests were conducted with two different PCIe SSDs and a SATA-interface SSD. A comparison was also made against a Linux implementation of FlashSoft SSD caching software (FlashSoft is an SSD caching software solution recently acquired by SanDisk).

Evaluation Environment

The tests were conducted on multi-core servers connected to Fibre Channel storage systems in the Demartek lab with both a PCIe SSD and drive form factor SSDs. The testing included TPC-C workloads using MySQL and VDbench and were run in Linux and VMware environments.

Evaluation Summary

VeloBit HyperCache 1.1 performed very well in our lab tests, especially when it was caching both reads and writes. The performance of VeloBit HyperCache, especially in the read-intensive tests in the native Linux environment, is the highest that we have seen for host-based SSD caching, achieving an 800% performance improvement in TpmC results in the native Linux environment and a 60x performance improvement in one read-intensive test. VeloBit HyperCache also outperformed the competitive host-based SSD caching solution from FlashSoft in the native Linux environment.

We recommend that enterprises obtain the trial version and evaluate it in their environments.

1 – SSD Caching

SSD caching is an excellent way to deploy SSD technology. When SSDs are used as a cache, the following observations can be made:

- The caching solution places a copy of "hot" data into the cache.
- The caching solution decides when to place the copy of the hot data into the cache.
- Multiple applications can gain a performance benefit.
- The aggregate performance gains occur over time as the cache "warms-up."
- Applications do not need to be modified to take advantage of the SSD cache.
- Some caching solutions only cache reads, others cache both reads and writes.
- Management of a caching solution is relatively simple.

Host-based caching solutions such as VeloBit HyperCache can generally take advantage of any SSD form factor available to that host server, while caching solutions supplied with hardware tend to use only the SSD form factors supported by that hardware. In addition, host-based SSD caching solutions generally interoperate with, and require no changes to, back-end SAN or NAS storage.

Host-based SSD caching solution performance will vary depending on the number and type of processors in the host system and the performance characteristics of the SSD or SSDs that are used for the cache. Another major factor is the operating environment. Applications running in operating systems running natively on host system hardware can expect higher performance gains than the same applications and operating systems when running as a guest in a virtual server environment. This is due to the overhead of the virtual server environment and the implementation of the SSD caching solution for that virtual server environment.

For host-based caching solutions, there are two basic decisions the administrator must make. These are to decide how much SSD resource to devote to the cache and to decide which storage should be cached. After these two decisions have been made, generally, the host-based SSD caching solution requires very little direct management.

VeloBit HyperCache

VeloBit HyperCache has taken a different approach towards SSD caching than we have seen in some other solutions. HyperCache creates a caching layer on the server that uses a combination of main memory and the SSD together as the storage cache. The data that is written to the VeloBit HyperCache is compressed, in order to reduce the number and size of write operations on the SSD, extending the life of the SSD. This compression also improves the effectiveness of both the RAM and SSD portions of the cache, providing a substantial acceleration of read and write requests.

VeloBit HyperCache can work with any SSD form factor, including PCIe SSDs and drive-form-factor SSDs such as SATA-interface SSDs. Multiple drive form factor SSDs can be used if software or hardware RAID is employed.

The VeloBit HyperCache solution uses an architecture that VeloBit calls "content locality." This technique does not view data blocks as strictly hot or cold in the way that other caching solutions do. Instead, VeloBit HyperCache categorizes an incoming block based on how similar its contents are to other blocks. This technique is called "line-speed delta compression", and results in three

types of data blocks: reference, associated and independent. Reference blocks have contents that are similar to other blocks, and are called "popular." Associated blocks are blocks that are similar to reference block and include a pointer to the reference block and a list of differences (delta) between that block and the reference block. Compression is applied to reference and associated blocks. Specifically, the deltas are compressed and packed into a single write on the SSD, reducing wear on the SSD. Independent blocks do not have popular contents and are not compressed.

Write-caching with VeloBit HyperCache can be enabled or disabled. If enabled, there is a write-cache-depth parameter that defines the maximum amount of time that elapses between the point at which a block is received into the cache and the point at which it is written to the back-end storage. Blocks written to primary storage are decompressed and written in their original form on the storage device.

Blocks are flushed to disk when the write-cache-depth time parameter is met, the primary storage is idle or the amount of "dirty" data in the cache reaches 15 GB.

VeloBit HyperCache uses a feature known as "scan-resistant caching" or "sequential filtering" to prevent large sequential I/O blocks from being cached. This feature is enabled by default, but can be disabled if desired.

The Linux version of VeloBit HyperCache was released in March 2012. The Windows and VMware versions are currently in Beta testing and scheduled for general availability in July 2012.

2 – Test Environment

In order to test the VeloBit HyperCache solution, we configured different server and storage combinations to highlight the ability of VeloBit HyperCache to run with different types of hardware. The tests were run in a native Linux environment and in a VMware environment with a Windows guest. The tests included running a TPC-C workload and a synthetic I/O workload with vdbench in various configurations.

These tests were conducted with each server connected to a separate Fibre Channel storage system. The tests were run with the following hardware configurations:

- No SSD caching for a baseline
- PCIe SSD used for the SSD cache
- SATA SSD used for the SSD cache (Linux platform only)

Although different SSD devices were used, the amount of SSD capacity allocated for the SSD cache was limited to 100 GB in all test cases, in order to make the tests as similar as possible. This SSD cache was 25% or less of the database size used for the database tests.

Linux Installation

For the Linux environment, VeloBit HyperCache installs as a standard package using the following steps:

- Install the RPM in the standard fashion
- Run the registration application in a command line window (the VeloBit HyperCache license is not bound to any particular SSD)
- Load the kernel module by entering "velobit start" in the command line window

The commands for configuring it are straightforward, and function as documented. There was one minor bit of confusion in that the flush delay is called "cache depth" in the documentation. For our tests, we configured the SSD device and the devices that it should cache.

This installation process was easier than the equivalent process for the FlashSoft SSD caching solution. The FlashSoft license is bound to the SSD that is used for caching. The FlashSoft solution requires more steps to configure the caching for a particular volume. In addition, the FlashSoft solution configuration steps must be performed on un-mounted volumes.

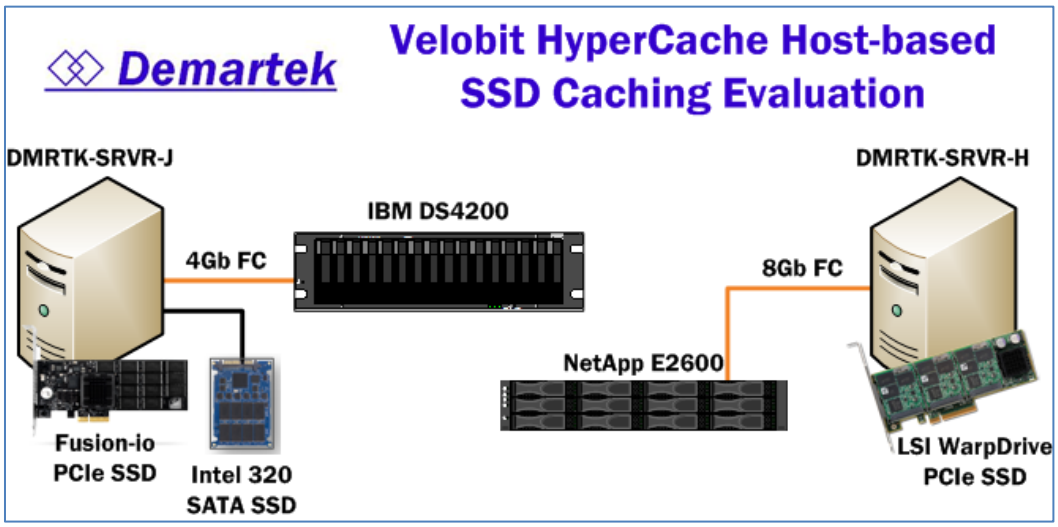
VMware Installation

For the VMware installation, VeloBit HyperCache installs as a virtual appliance. The installation of this virtual appliance consists of the following steps:

- Load the appliance (a pre-packaged guest virtual machine) into VMware
- Add the existing datastore and the SSD to the virtual appliance as virtual disks
- Start the appliance and run the same configuration utility that is used for the native Linux installation
- Set the static IP address of the appliance
- Configure VMware to connect to the iSCSI target that the virtual appliance creates
- Create a new disk on this iSCSI target
- Present the new disk as a datastore for a guest virtual machine

Once installed, the SSD and desired devices are mapped to the virtual appliance. The virtual appliance allocates a portion of the SSD for acceleration for each of the desired physical storage LUNs and makes accelerated iSCSI LUNs available to guests. The guest VMs use iSCSI to access the accelerated LUNs that have been created by the appliance. The caching functions are all controlled within the virtual appliance.

There were some areas where the documentation for the VMware version was a bit unclear.



Server Configuration (Linux)

- Supermicro X8DAH+-F
- 2x Intel Xeon X5680, 3.33 GHz, 12 total cores, 24 total logical processors
- 144 GB RAM
- Internal boot SATA SSD
- 8 Gb/s Fibre Channel Host Bus Adapter (HBA)

Server Configuration (VMware)

- Supermicro X8DTH-F
- 2x Intel Xeon X5540, 2.53 GHz, 8 total cores, 16 total logical processors
- 48 GB RAM
- Internal boot SATA SSD
- 8 Gb/s Fibre Channel Host Bus Adapter (HBA)

SSD Cache Devices

- Fusion-io io-Drive, 160 GB, PCIe
- LSI WarpDrive, 300 GB, PCIe
- Intel 320, 300 GB, SATA

The PCIe SSDs were installed inside the servers. The SATA SSD was connected to a motherboard SATA port.

Storage Systems

- IBM DS4200 with 4 Gb/s host interface
- NetApp E2600 with 8 Gb/s host interface

Operating Environments

- Linux: Centos 5.6, kernel: 2.6.32-220.7.1.el6.x86_64
- VMware: vSphere 5.0
- VMware guest: Windows Server 2008 R2 SP1

3 – TPC-C Workload

TPC-C workloads are representative of online transaction processing (OLTP) systems and perform a mixture of transactions including new-order, payment, order-status, delivery and stocking-level operations. The read-write mixture of this workload is approximately 70% read and 30% write. This workload provides a transaction per minute (tpmC) score at the end of the processing. The data produced by this test is not very compressible.

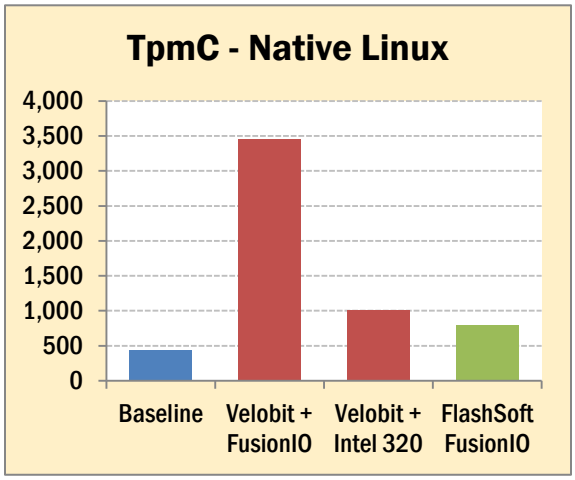
A baseline test was run with the server and back-end Fibre Channel storage without caching, then repeated with SSD caching enabled. As expected, the tpmC score increased when the SSD caching was enabled.

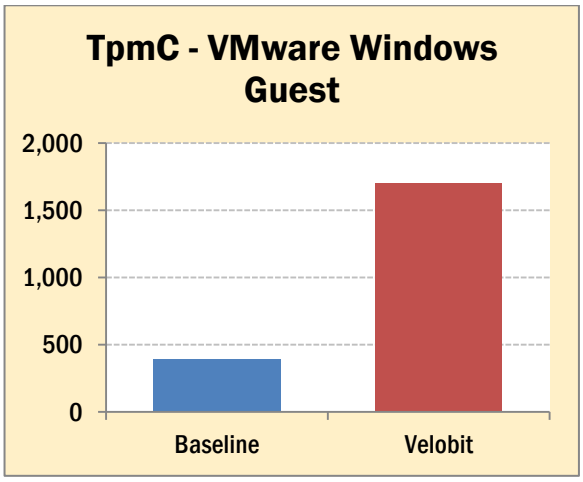
These TPC-C workloads were run with MySQL as the database application and included the following parameters. The database size was approximately 440 GB.

- Warehouses: 4200
- Connections: 96
- Rampup: 6000 seconds (100 minutes)
- Measure: 6000 seconds (100 minutes)

VeloBit HyperCache can operate in read-write caching mode and read-only caching mode. A competitive SSD caching solution was also run to provide an additional point of comparison. VeloBit HyperCache supports native Linux and Windows operating systems as well as VMware and Hyper-V environments. This benchmark was run in Linux and VMware environments.

The following charts show the performance results of the baseline and SSD caching solutions in read-write mode for the TPC-C workload. These results show the performance of the cache after the rampup (or warmup) period listed above was completed.





The TpmC score in the native Linux environment with the VeloBit HyperCache read-write SSD cache enabled was approximately ***8x higher*** than the baseline results. The same test run in the VMware environment with one Windows guest VM was approximately ***4.3x higher*** than the baseline results in the VMware environment.

As expected, greater acceleration was achieved with the PCIe SSD than with the SATA-interface SSD. Compared to the baseline with no SSD caching, VeloBit HyperCache provided ***8x higher*** performance with the PCIe SSD and ***2.3x higher*** performance using a single SATA-interface SSD in the TPC-C test in the native Linux environment. This allows administrators to provide different levels of performance depending on the budget for SSD technology.

As a point of comparison, the VeloBit HyperCache outperformed another host-based SSD caching solution, FlashSoft, for this test. Using the same SSD hardware, the VeloBit HyperCache solution outperformed the FlashSoft solution by providing approximately ***4.3x higher*** TpmC results for the TPC-C workload in the native Linux environment.

Also expected was the difference in performance gains between the native Linux implementation and the performance gains in the VMware environment. VMware incurs additional overhead simply because it is a hypervisor, and there is some additional overhead because the guest must use iSCSI as the access mechanism for the accelerated LUN. However, even with these additional overheads, the VMware version provided ***more than 4x higher*** performance. The increased performance in the VMware environment means that additional guest virtual machines could be run on the same physical VMware server. Further testing would be needed to determine how many additional guests could be run in this configuration, but we estimate that a few more could be run. The number of additional guests that could be run would be affected by the processor and memory configuration of the host system as well as the type of SSD used for the cache.

Costs for TpmC Performance

Higher levels of performance are usually accompanied by higher prices. The following is an analysis of the TpmC results shown above for the Linux and VMware environments. The prices shown for the hardware are for quantities of one and are approximate based on public pricing sources for new equipment. Pricing may vary by supplier, quantity and any negotiated discounts. Prices are in US dollars (US$).

VeloBit HyperCache list pricing: $1250.00 per processor for native Linux (and Windows) and $2,500.00 per processor for VMware. This is price per processor regardless of the number of cores. Quantity discounts are available.

The analysis shown below excludes the cost of the server and any operating system or hypervisor licenses. Each server has two processors. The Linux and VMware tests were conducted with different servers, SSDs and back-end storage, so these two sets of results cannot be directly compared.

The "TpmC Gain" is the relative improvement (multiplier) in performance with the SSD caching enabled. The "$ per TpmC" includes the cost of the SSD, caching software and the back-end storage (excluding any storage networking switches). The "$ per TpmC Gain" is the cost of the added SSD device and SSD caching software divided by the TpmC Gain.

For all of the SSDs, both PCIe and drive-form-factor, the SSD cache was limited to 100 GB. Smaller capacity SSD devices of the same performance would result in a lower $ per TpmC.

Linux Cost Analysis	TpmC	TpmC Gain	SSD cost	VeloBit cost	Storage cost	$ per TpmC	$ per TpmC Gain
Linux, Baseline, no SSD	430	–	$0	$0	$30,000	$69.77	–
Linux, VeloBit, ioDrive	3449	8.02x	$9,000	$2500	$30,000	$12.03	$1433.75
Linux, VeloBit, Intel 320	1012	2.35x	$500	$2500	$30,000	$32.61	$1274.70

For raw performance gain, the single PCIe SSD provided significantly higher performance than the single SATA-interface SSD. However, we believe that some combinations of multiple drive-form-factor SSDs configured in either a software or hardware RAID configuration may provide higher performance at lower cost than a PCIe SSD.

VMware Cost Analysis	TpmC	TpmC Gain	SSD cost	VeloBit cost	Storage cost	$ per TpmC	$ per TpmC Gain
VMware, Baseline, no SSD	391	–	$0	$0	$20,000	$51.15	–
VMware, VeloBit, WarpDrive	1699	4.35x	$8,000	$5000	$20,000	$19.42	$2991.76

Because VMware environments can support multiple guest operating systems, the number of guest virtual machines taking advantage of the SSD cache should be factored into the total cost of the solution. In the tests shown above, exactly one guest virtual machine was running and configured to use the SSD cache.

4 – VDbench Workload

VDbench is a synthetic I/O load generator that can be set with a variety of block size, queue depth and read/write mixes. As with the TPC-C tests, the VDbench tests were run with a baseline then repeated with the SSD caching enabled. The average I/Os per second (IOPS), average megabytes per second (MBPS), average response time and average CPU utilization were recorded.

VDbench writes compressible data patterns. VeloBit HyperCache is designed for optimal performance with compressible data, so it is reasonable to expect higher performance gains for this type of data as compared to the TPC-C workloads shown in the previous section.

A 400 GB test file was configured for VDbench and four different combinations of read/write mix and read hit percentages, as shown below:

- `wd=wd1,sd=sd1,xfersize=512,readpct=100,seekpct=100,rhpct=90`
- `wd=wd2,sd=sd1,xfersize=512,readpct=100,seekpct=100,rhpct=100`
- `wd=wd3,sd=sd1,xfersize=512,readpct=80,seekpct=100,rhpct=90`
- `wd=wd4,sd=sd1,xfersize=512,readpct=80,seekpct=100,rhpct=100`

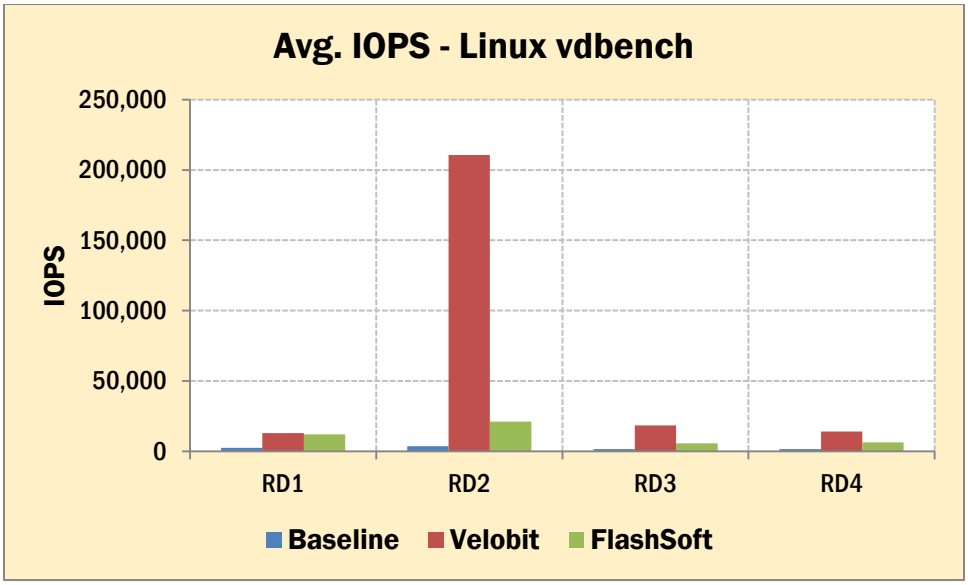
Four tests were run using each of the configurations listed above. Each test was run for 1200 seconds (20 minutes), with 32 threads using the parameters shown below:

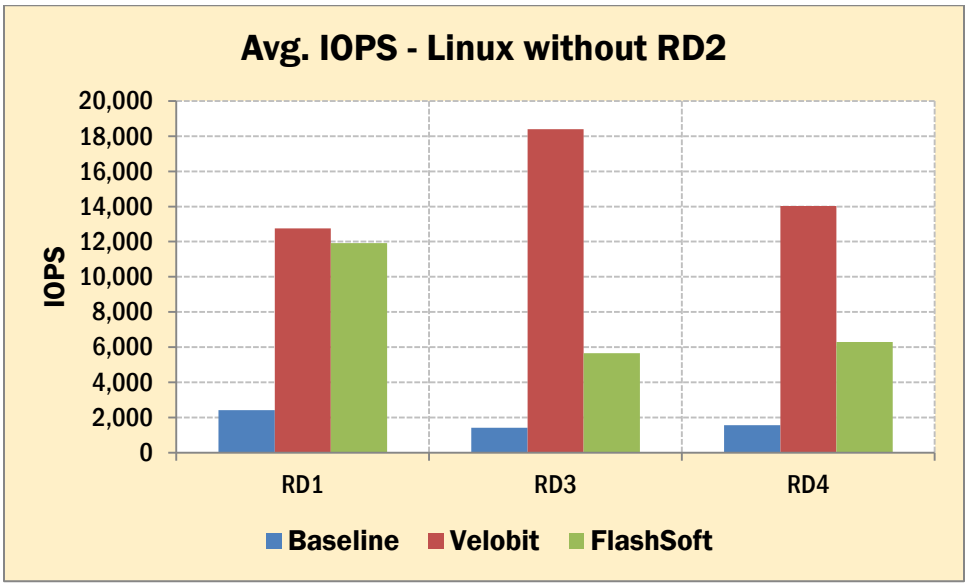
- `rd=rd1,wd=wd1,iorate=max,elapsed=1200,warmup=0,forxfersize=(16k),forthreads=(32)`
- `rd=rd2,wd=wd2,iorate=max,elapsed=1200,warmup=0,forxfersize=(16k),forthreads=(32)`
- `rd=rd3,wd=wd3,iorate=max,elapsed=1200,warmup=0,forxfersize=(16k),forthreads=(32)`
- `rd=rd4,wd=wd4,iorate=max,elapsed=1200,warmup=0,forxfersize=(16k),forthreads=(32)`

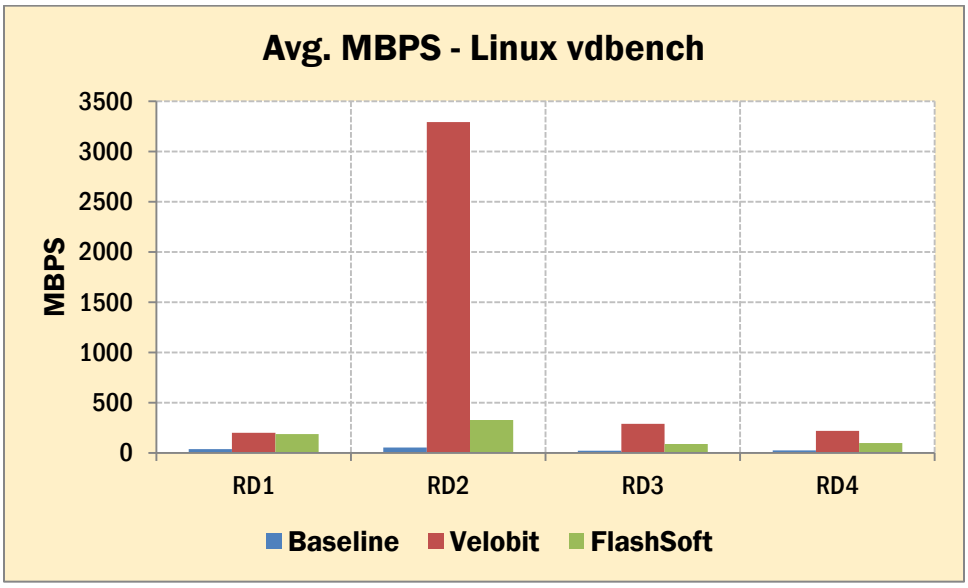
The VeloBit HyperCache software performed especially well for configuration RD2, which was a read-intensive test. Because the performance for RD2 was so high, there are two sets of graphs, one with RD2 included and one without RD2. VeloBit HyperCache provided ***61x higher*** performance than the baseline and the FlashSoft solution provided ***6x higher*** performance than the baseline.

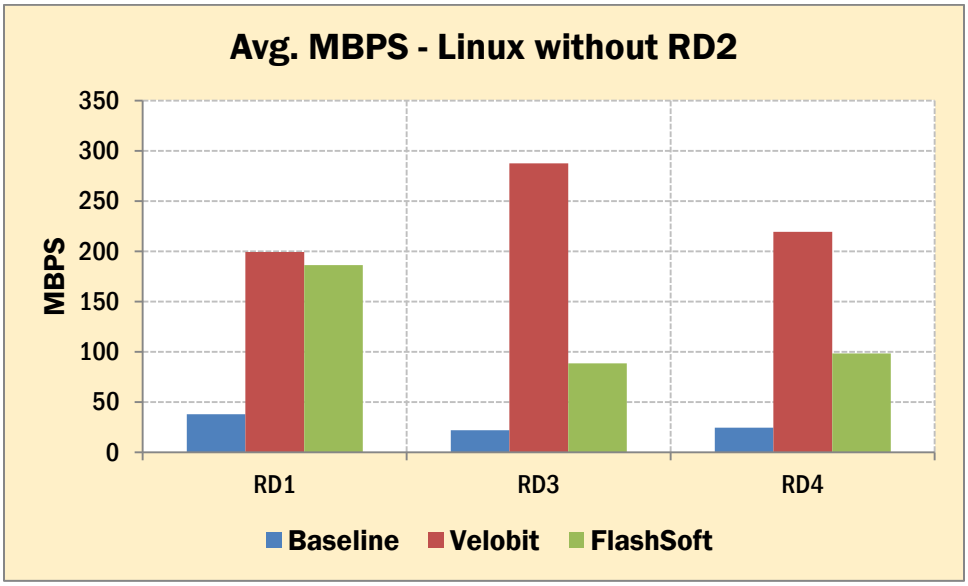
These tests were run in a native Linux environment, and then repeated in a VMware environment with one Windows guest. In all cases, VeloBit HyperCache provided outstanding performance improvements.

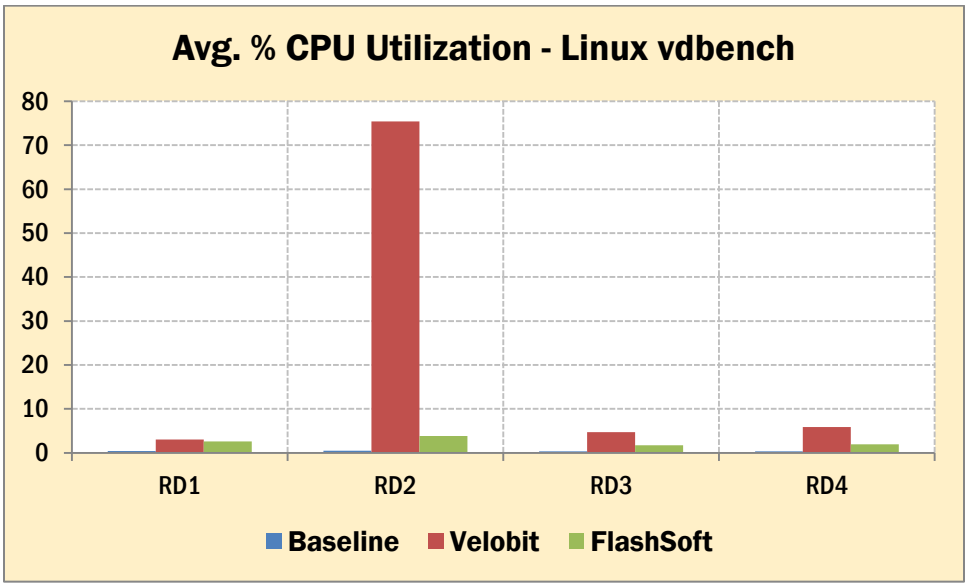
VDbench – Linux Environment

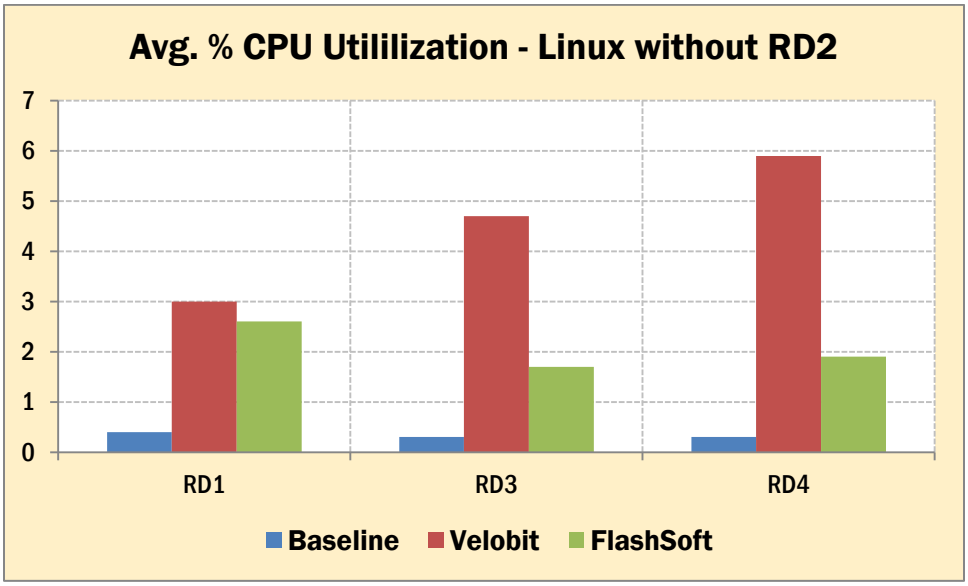


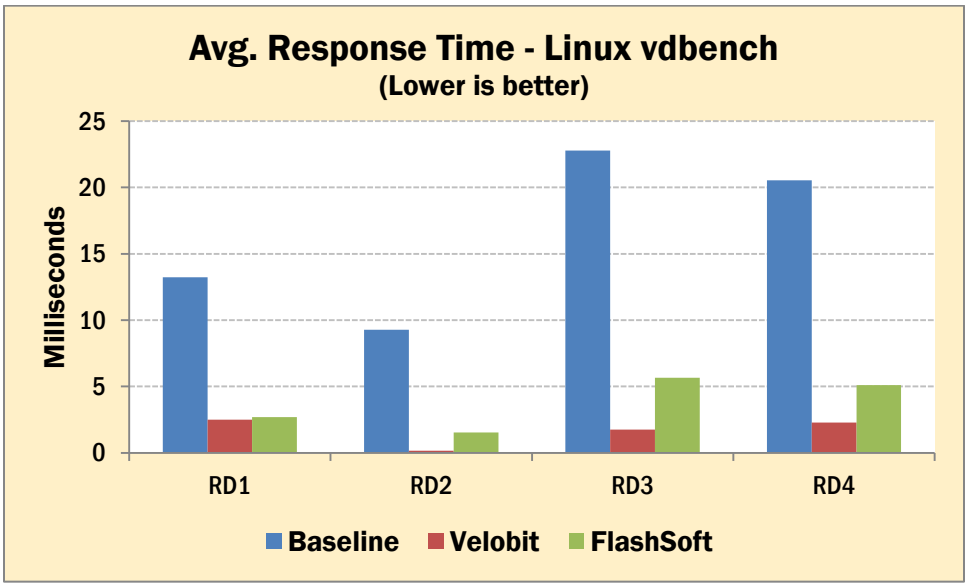



Avg. MBPS - Linux vdbench
MBPS
3500
3000
2500
2000
1500
1000
500
0
RD1
RD2
RD3
RD4
Baseline
Velobit
FlashSoft


Avg. MBPS - Linux without RD2
MBPS
350
300
250
200
150
100
50
0
RD1
RD3
RD4
Baseline
Velobit
FlashSoft


Avg. % CPU Utilization - Linux vdbench
80
70
60
50
40
30
20
10
0
RD1
RD2
RD3
RD4
Baseline
Velobit
FlashSoft


Avg. % CPU Utililization - Linux without RD2
7
6
5
4
3
2
1
0
RD1
RD3
RD4
Baseline
Velobit
FlashSoft



Summary of VDbench results in the Linux environment

IOPS and MBPS Performance Relative Gains (baseline = 1.0)

	RD1	RD2	RD3	RD4
Baseline	1.0	1.0	1.0	1.0
VeloBit	5.3	61.1	13.1	9.0
FlashSoft	4.9	6.1	4.0	4.0

Response Time Relative Performance (baseline = 1.00)

	RD1	RD2	RD3	RD4
Baseline	1.00	1.00	1.00	1.00
VeloBit	0.19	0.02	0.08	0.11
FlashSoft	0.20	0.16	0.25	0.25

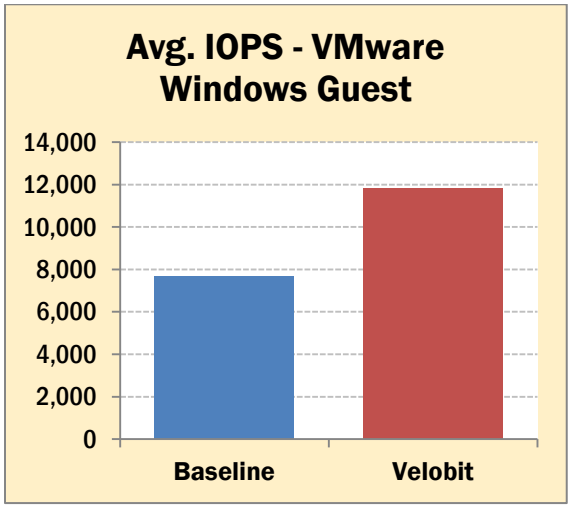
CPU Utilization Relative Performance (baseline = 1.0)

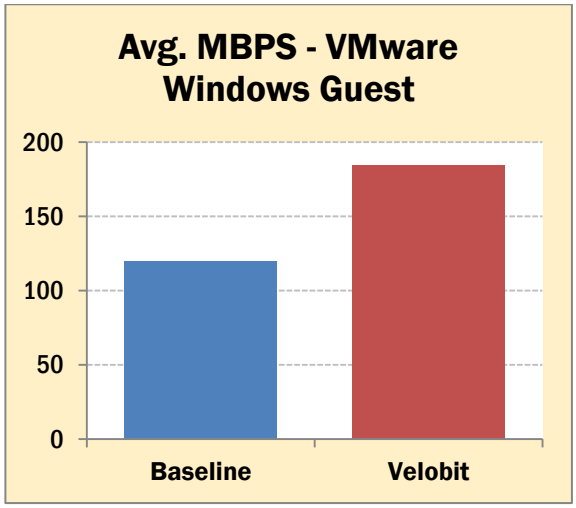
	RD1	RD2	RD3	RD4
Baseline	1.0	1.0	1.0	1.0
VeloBit	7.5	150.8	15.7	19.7
FlashSoft	6.5	7.6	5.7	6.3

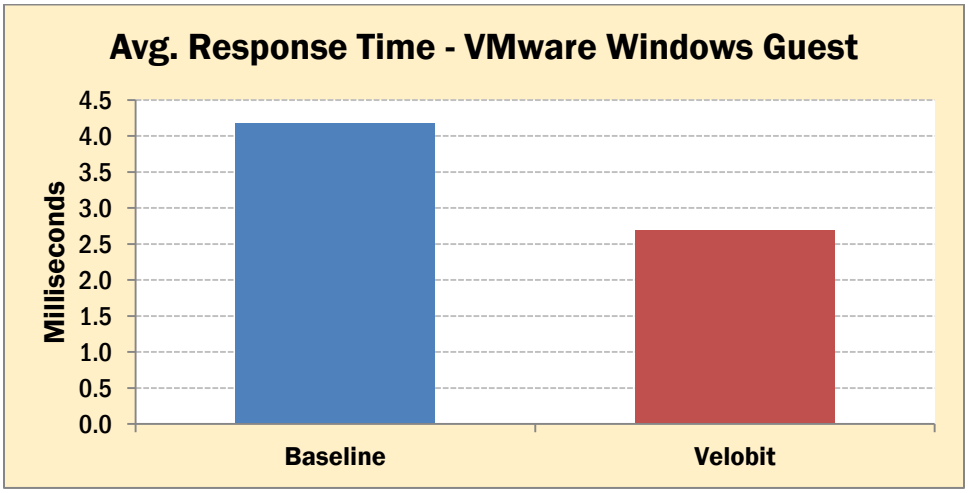
VDbench – VMware with Windows Guest

For the VDbench tests performed in the VMware environment, a single Windows guest operating system was configured and the "RD6" configuration was used (see note below).

The performance gains in the VMware environment were not as high as in the native Linux environment, which was expected.







The "RD6" VDbench configuration parameters used for the VMware Windows Guest tests:

- `wd=wd6,sd=sd1,xfersize=512,readpct=50,seekpct=100,rhpct=100`
- `rd=rd6,wd=wd6,iorate=max,elapsed=1200,warmup=0,forxfersize=(16k),forthreads=(32)`

Summary and Conclusion

The VeloBit HyperCache 1.1 provided excellent performance for the real-world TPC-C workload and the synthetic VDbench workload. We recommend that users get the trial version and test it with their own applications.

The performance of VeloBit HyperCache, especially in the read-intensive tests in the native Linux environment, is the highest that we have seen for host-based SSD caching software.

We found that the installation of the Linux and VMware versions of the VeloBit HyperCache software was very straightforward. Especially in the Linux environment, we were able to get up and running fairly quickly.

Because this is a host-based software SSD caching solution, it can work with any type of SSD and does not require changes to applications or back-end storage infrastructure. The performance of this host-based SSD caching is also partly dependent on type and number of processors in the server. The decision as to the SSD form factor (PCIe vs. SATA-interface) is simply a matter of performance desired and budget available for the SSD technology. Both provide very strong performance.

We believe that multiple drive-form-factor SSDs configured in either a software or hardware RAID configuration may provide higher performance at lower cost than a PCIe SSD.
