VirtuCache Case Study

USE CASE

- Transactional Database

LOCATION

- USA

SOLUTION

- VirtuCache with VMware host based SSDs
- Read and Write Caching

The VirtuCache Difference

At the school district, read latencies plummeted from 43ms to 2ms, write latencies dropped to 12ms from 30ms and write throughput increased by over double from 6MBps to 15MBps delivering exceptional performance for a low investment.

500 Users, 1 million transactions per user from 1 VM per host with 4 hosts. VirtuCache delivers over 200,000 transactions Vs 67,000.

VirtuCache Read+Write Caching Solution Competing With a Read Caching Solution at a USA Large School District

VirtuCache is a Server Side Caching software solution that has the below features:

- Kernel only and Partner Verified and Supported by VMware
- Clusters SSDs across Hosts
- Uses such a cluster of SSDs to accelerate Reads and Writes
- Replicates only the writes to 2 other SSDs on 2 other hosts. In case of host failure, it syncs the backup copy of the cache to backend SAN, thus preventing data loss. By replicating only writes, it conserves network bandwidth.
- Dedupes IO at the block level thus minimizing both network bandwidth and SSD usage

Results of TPC-C tests at this school district for three use cases – uncached, with the incumbent software and with VirtuCache for read and write caching.

TPC-C Tests were run (http://www.tpc.org/tpcc/). TPC-C is a benchmark tests that mimic real-life transaction processing workloads using a relational database.

These tests were run using a tool called HammerDB in a Windows 2008 Server R2 VM running MS SQL Server 2008 R2.

- 500 Users, 1M transactions per User from one VM per Host.
- Running simultaneously on 4 VMs on 4 Hosts.
- Datastore was on LeftHandNetworks iSCSI Appliance
- Disk.SchedReqNumOutstanding (DSRNO set to 128) and PVSCSI Driver Used (Both parameters are needed because of high MBps tests)

We ran this test 3 times – Uncached, with the incumbent Read Caching software, and using VirtuCache Read+Write Caching software.

Caching Software	Read Lat (ms)	Write Lat (ms)	Read MBps	Write MBps	Transactions per minute
Uncached	43	30	23	6	65K
Competing READ ONLY Caching Software	1	32	23	9	101K
VirtuCache READ & WRITE Caching Solution	**2**	**12**	**22**	**15**	**200K**

Storage performance bottlenecks within virtualized servers result in inconsistent performance of applications deployed within VMs and lower VM to host ratios. This is especially so if data intensive applications like databases or latency sensitive applications like VDI are deployed within virtual machines. Server-side Caching software caches frequently/recently used data from a networked storage appliance to SSDs installed in physical servers. Subsequently, by serving more and more data from much faster in-server SSD (instead of backend storage appliance), such software accelerates storage performance many times.

VirtuCache Case Study

Host side caching software needs to accelerate both reads and writes especially when competing against from all-flash arrays. Caching writes is important even for workloads that are read intensive. If VirtuCache did not to accelerate writes, the reads behind the writes on the same thread will not be accelerated, thus slowing reads as well.

Using TPC-C benchmark, we showed 2x improvement in performance versus a read caching software vendor at a large district.

Storage performance bottlenecks within virtualized servers result in inconsistent performance of applications deployed within VMs and lower VM to host ratios. This is especially so if data intensive applications like databases or latency sensitive applications like VDI are deployed within virtual machines. Server-side Caching software caches frequently/recently used data from a networked storage appliance to SSDs installed in physical servers. Subsequently, by serving more and more data from much faster in-server SSD (instead of backend storage appliance), such software accelerates storage performance many times.

What are the benefits to customers?

Since storage IO issues are the main bottleneck in enterprise datacenters, by eliminating this issue, Server-side Caching software allows end users to experience improved performance of applications running within VMs and helps increase the number of VMs deployed on each VMware host. Since Server-side Caching software achieves this without requiring customers to replace their existing storage networks or storage appliances, this approach results in considerable cost savings to the customer.

What should the design principles be for server-side caching in light of competitive pressure from SSD based arrays?

As cost of all-flash arrays is dropping, server-side caching software is increasingly competing with such SSD based arrays. To effectively compete with these appliances, server-side caching software should be able to do all of the below:

–It should be **kernel based**. Most server-side caching software currently is VM based, deployed either as an agent installed in VMs or a separate VM per host. VM based software causes storage IO from the kernel to be redirected to the VM, and caching decisions are made in the VM. Because of multiple kernelspace-userspace context switches, VM based software is less efficient than kernel-only software for the same SSD. As a result, kernel-based Server-side Caching software can compete with all-flash arrays using less expensive SATA, SAS, PCIe and NVMe SSDs installed in VMware hosts.

— **No manual intervention.** Automatic caching of data with no human involvement is the ideal way to do caching. There are caching solutions on the market that let administrators define caching policies at the application and file level. In theory, such granularity for defining caching policies look impressive, however it becomes a significant management burden for the server administrator to intelligently assign caching policies on a per VM and per application basis, even in a small datacenter with hundreds of VMs. The server administrator will need to understand the applications deployed within VMs, their relative importance and I/O patterns, for devising intelligent caching policies.

— Needs to accelerate writes. To deliver performance that rivals SSD based storage arrays, server-side caching software needs to accelerate writes as well (in addition to reads). The common misconception with server-side caching software is that since IT workloads are 80%+ reads, accelerating reads would suffice. This would be true if reads and writes were on separate threads. Since in real world applications, reads and writes are on the same thread, by not accelerating writes, writes that are in front of the reads, choke even reads.

— **VMware certified.** Since kernel mode software needs to interoperate with other 3rd party software in the kernel (like drivers, anti-virus software etc), it is imperative that the VMware kernel not be hacked. The only way for the customer to verify this is if VMware has certified the vendor's software.

Why is it hard to support write caching with kernel-only implementation for VMware ?

The reasons why most Server-side Caching software accelerate only reads is because accelerating writes requires writes to be written to only the local host based SSD without synchronously writing to the backend SAN. This opens up the possibility of data loss in case the local host fails. To prevent against such data loss, local host-based cache needs to be replicated across hosts. Subsequently if the host fails, the backup copy of the cache from another host needs to be immediately synced with the backend storage appliance. What makes such software hard to develop is that this architecture requires that SSDs be clustered across hosts; such a cluster needs to scale to the maximum number of nodes supported in a VMware cluster; ability to replicate writes across hosts without consuming a lot of network bandwidth; and seamlessly supporting vMotion and other VMware advanced features.

The reason why developing such a solution at the kernel level is hard because VMware is a closed & proprietary kernel, VMware does not provide an API specifically build a Host Side Caching solution, and for a kernel only solution all these features need to be developed with a small software footprint.