



Storage vMotion of a Virtualized SQL Server Database

Performance Study

TECHNICAL WHITE PAPER

Table of Contents

Executive Summary	3
Introduction	3
Factors Affecting Storage vMotion	3
Workload	4
Experiments and Results	4
Migrating Virtual Disks from Older Array to Newer Array	4
Test Setup	4
Performance Metrics	5
Baseline	6
Migrating Virtual Disk Containing Data Files	7
Migrating Virtual Disk Containing Index Files	8
Migrating Virtual Disk Containing Log Files	10
CPU Cost of Disk Migration	11
Migrating Virtual Disks from Newer Array to Older Array	11
Test Setup	12
Baseline	12
Migrating Virtual Disk Containing Data Files	12
Migrating Virtual Disk Containing Index Files	13
Migrating Virtual Disk Containing Log File	14
CPU Cost of Disk Migration	14
Best Practices for an Optimal svMotion Experience	14
Conclusion	15
Appendix	16
Details of Hardware and Software Used for the Experiments	16
Server	16
Storage Area Network	16
Virtual Platform	16
Virtual Machine	16
Workload	16
Storage Layout for the DS Workload in CX 3-40 (Older Array)	16
Storage Layout for the DS Workload in NS 960 (Newer Array)	17
Estimation of the CPU Cost of an Order Request of DS2	17
References	17
About the Author	17

Executive Summary

This paper provides a deep look at the performance implications of the live storage migration of a VM running a large SQL Server database workload. Performance of the workload when migrating different components of the database (data, index, and log) was measured and compared against the steady state performance (when there was no migration). Experiments discussed in this paper also show the impact of I/O characteristics of the database components on the migration time of the virtual disk containing them. Based on the knowledge gained from the experiments, a few best practices are suggested for the live migration of the storage of the virtual machines (VMs) with minimal impact to the performance of applications running in them.

Introduction

The ability to move (migrate) a VM while it is operating is one of the essential requirements for a platform that aims to provide a Cloud-like experience. Live migration (called vMotion in vSphere) is involved when moving VMs for the purpose of load balancing resources, hardware maintenance, high availability (HA), or disaster recovery. Live migration involves copying the memory and device state of a VM from one physical host to another with no perceivable down time. Another important component of a VM that often requires it to be moved is the data accessed by the VM. The data is stored in virtual disks (.vmdk files in the virtual machine file system or logical devices assigned as raw device mappings) that are created on storage local to a host or external in a SAN. VMware vSphere has provided the capability to live migrate the memory and device state of a VM since its early versions. Later versions of vSphere provide the ability to also live migrate the virtual disk belonging to a VM, thus completing the process of liberating the VMs from the physical hardware on which they run.

Live storage migration (called Storage vMotion, or svMotion in vSphere) is a necessity for several reasons:

- For moving VM storage in and out of storage containers (datastores, NFS volumes, or raw devices) for balancing load or capacity of the containers
- For maintenance and upgrades to the file system or storage arrays that host the containers
- For moving VMs and their storage components between different internal clouds or between internal (private) and external (public) clouds

Storage distributed resource scheduler (SDRS), a new storage load and capacity balancing feature¹ introduced in vSphere 5, automatically balances I/O load on storage containers by monitoring and migrating the storage of VMs using svMotion. For an in-depth understanding of svMotion architecture and its evolution, see *The Design and Evolution of Live Storage Migration in VMware ESX*².

Factors Affecting Storage vMotion

svMotion performance, to a large extent, depends on the capabilities of the physical hardware backing the storage that is being migrated. svMotion performance (measured by virtual disk migration time) is determined by the weakest link in the entire storage infrastructure. Possible weak links include HBAs, I/O links between the host and the storage, storage processing components, or the physical disks. To achieve a satisfactory svMotion experience with minimal impact on the performance of the virtualized application whose storage is migrated, it is important to have a well-designed physical storage infrastructure. In spite of a well-designed physical infrastructure, I/O characteristics of the applications whose storage is migrated could impact svMotion in different ways and, in turn, be influenced in different ways by svMotion. A good understanding of these I/O characteristics and their effect on svMotion is useful when planning and configuring manual or automatic storage migration.

The remainder of the paper is focused on experiments conducted at the VMware performance engineering lab. The goal of these tests is to gain a better understanding of the impact of application I/O characteristics on svMotion and the effect svMotion has on application performance. The virtual storage being migrated is that of a VM running a SQL Server database workload.

Workload

DVD Store version 2.1 (DS2)³ was used to generate load in the VM. DVD Store is an open source, e-commerce workload that mimics an online DVD store. For the experiments discussed here, DVD Store was configured to use a SQL server database of 500GB in size.

Experiments and Results

Migrating Virtual Disks from Older Array to Newer Array

In the first part of the study, three experiments were conducted to study the interactions of the application I/O characteristics and svMotion with each other. The experiments included:

1. Migrating a virtual disk containing the **data files** of the DS2 workload from the older array to the newer array
2. Migrating a virtual disk containing the **index files** of the DS2 workload from the older array to the newer array
3. Migrating a virtual disk containing the **log file** of DS2 workload from the older array to the newer array

Each experiment was conducted with the following four load conditions:

- a. VM was powered off
- b. VM was on, but DS2 workload was not active
- c. VM was on; DS2 generated a low load (25% of 4 cores)
- d. VM was on; DS2 generated a high load (>90% of 4 cores)

The migration time of each virtual disk when the VM was powered off was used as the baseline for the comparison.

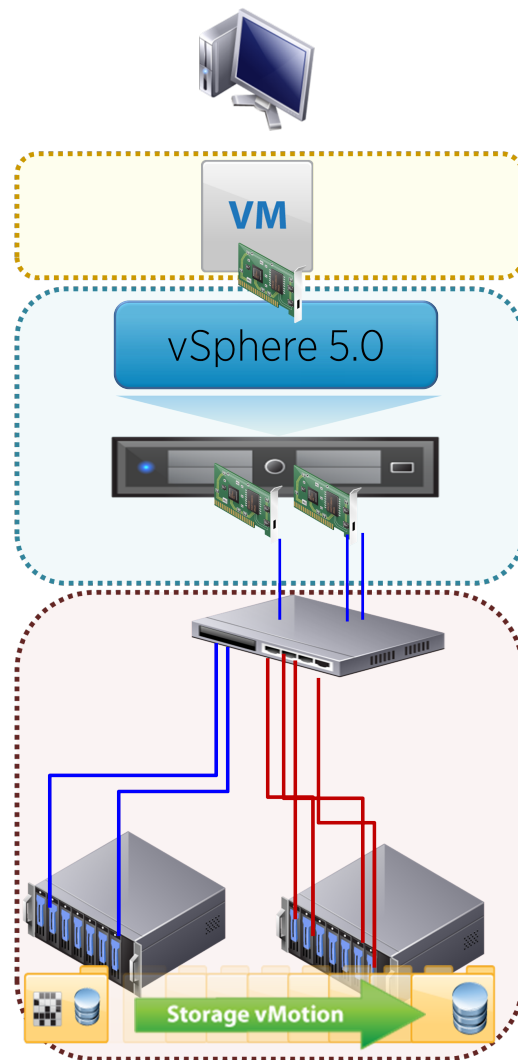
Test Setup

Figure 1 shows the test setup used for the first part of the experiments. A single server was connected to two Fibre Channel (FC) arrays using an 8Gbps FC switch. The host was equipped with four AMD 8370 (2.9GHz, 6 cores) processors and 128GB of memory. VMware vSphere® 5 was installed in the host and a single VM was created. Windows Server 2008 x64 R2 was installed as the guest OS and SQL Server 2005 x64 with Service Pack 3 was installed as the application. In the first phase of the experiments, the VM was configured with 4 virtual CPUs (vCPUs) and 30GB of memory.

Six VMFS datastores were created on six separate logical devices in the source array. A virtual disk was created in each VMFS datastore. The guest OS was installed in a virtual machine on a virtual disk different from the six virtual disks that were created in a separate VMFS datastore. The guest's virtual disk was attached to the VM using an LSI Logic SAS virtual SCSI (vSCSI) controller. The other six virtual disks were attached to the VM using another LSI Logic SAS vSCSI controller. Six identical VMFS datastores were created on six separate logical devices in the destination array. During svMotion, individual virtual disks were moved from the datastores in the source array to the datastores in the destination array.

Inside the guest OS, four virtual disks were chosen to store the index files of the DS2 workload. These were converted to dynamic disks and a single partition was created on them in a striped configuration. Separate partitions were created in the other two virtual disks. All the partitions were formatted with the NTFS file system using a 64K allocation unit as per the recommended best practice⁴.

Full details of the hardware and software versions of the components of the test bed and the configuration details of the virtual disks and the datastores can be found in the appendix.



DVDStore Client

Number	Description
1	4-vCPU, 30GB VMs
2	vSCSI adapters per VM
6	vDisks per VM (1 OS, 1 data, 4 Index, 1 Log)

Number	Description
2	8Gbps HBAs (1*Single port, 1*Dual port)
7	VMFS Datastores (1 OS, 1 data, 4 index, 1 log)
3	8Gbps FC connection

Number	Description
1	CX 3-40 (2 FC Connections)
1	NS 960 (4 FC Connections)

Figure 1. Test Setup used for Phase 1 of the Experiments

Performance Metrics

DS2 provides two key metrics to measure the performance of the workload: orders completed and average time (in milliseconds) to complete an order. For the experiments discussed in this paper, total orders completed was used to calculate the orders per second. This metric was later used when comparing the throughput of the workload under different test conditions.

Also, I/O performance metrics such as reads per second, writes per second, average milliseconds for a read, and average milliseconds for a write on the source and the destination datastores were captured using esxtop⁵.

Baseline

Two types of tests were done to capture the baseline numbers:

- Migration time of each virtual disk when the VM was powered off
- Steady state throughput of the DS2 workload when there were no virtual disk migrations

These tests are described in the following two sections.

Migration Time of Each Virtual Disk when the VM was Powered Off

Initially, the VM was powered off. Then the virtual disks containing the data and log files, and one of the virtual disks containing the index files of the DS2 workload were migrated separately from the source to the destination array. The total time required to migrate each virtual disk was recorded. The read/write throughput and latency at the source and destination array when migrating the data virtual disk are shown in Figure 2.

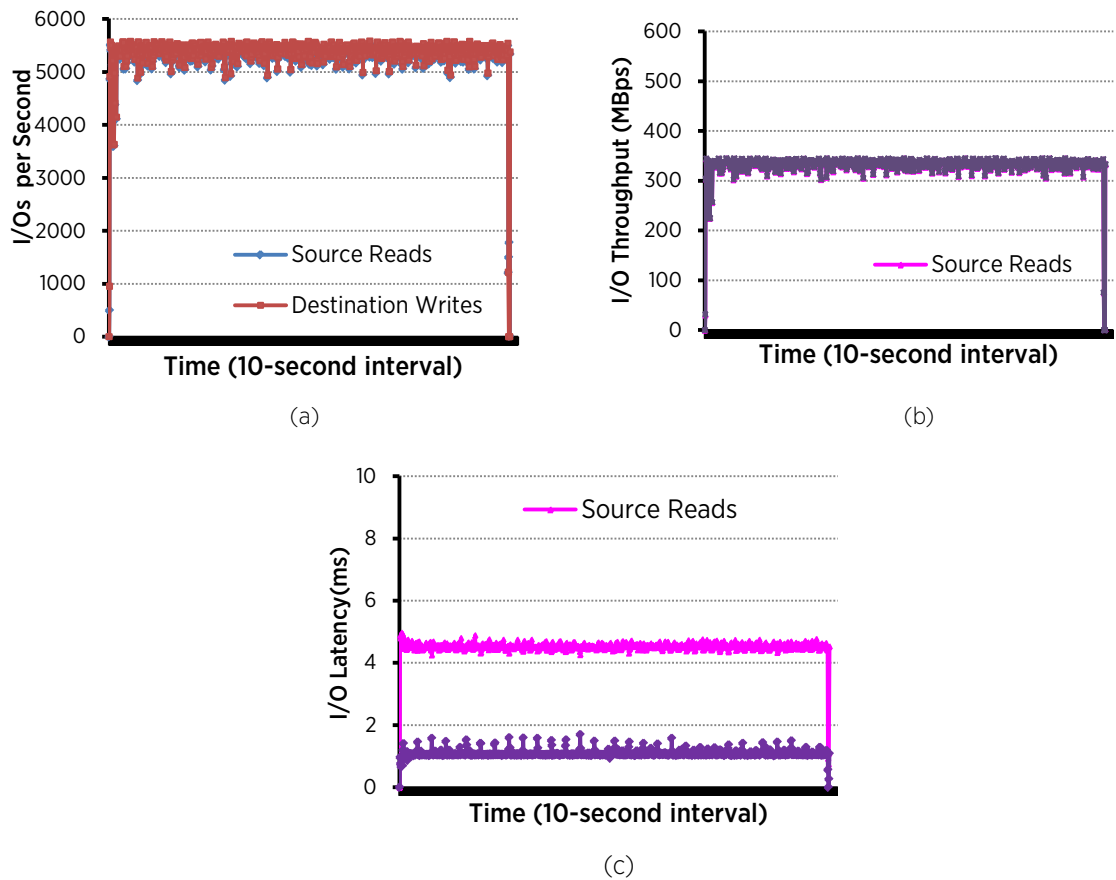


Figure 2. I/O Throughput and Latency at Source and Destination Datastores During svMotion of Data Virtual Disk when the VM was Off

During the migration of the data virtual disk, I/O operations per second on both source and destination arrays was approximately 5,400. I/O throughput was 330MBps. Read latency at the source array was 4.5ms while write latency at the destination array was close to a millisecond. Also note that svMotion moved data from the source to the destination array in 64KB chunks (evident by the ratio of the I/O throughput and the I/O operations per second).

The read/write throughput and latency at the source and destination arrays when migrating index and log virtual disks was very similar to that shown in Figure 2. Hence they are not shown separately. The only difference among all three cases was the total disk migration time.

Steady State Throughput of DS2 Workload when There Were No Virtual Disk Migrations

The DS2 workload was run for 30 minutes and the average number of orders completed per second during the steady state was recorded. This throughput served as the baseline when comparing the DS2 throughput during the migration of different virtual disks. There were not virtual disk migrations during the baseline test.

Migrating Virtual Disk Containing Data Files

In this test case, with the VM powered on, the virtual disk containing data files of the DS2 workload was migrated under different load conditions. The total migration time in each case is shown in Table 1. Also shown is the baseline migration time for moving the data virtual disk when the VM was powered off.

LOAD CONDITIONS	DISK MIGRATION TIME (SECONDS)
VM OFF	1843
VM ON, DS2 inactive	1840
VM ON, DS2 generated low load	2250
VM ON, DS2 generated high load	3731+

Table 1. Time to Migrate Data virtual disk

With the VM on, but DS2 inactive, the virtual disk migration time was similar to that of the baseline. When the DS2 workload became active and started issuing order requests against the database, under a low load condition, the virtual disk migration time increased by 410 seconds. It increased further when the load level became high. The total disk migration time under high load was longer than the total test duration, hence it was not measured accurately. But, it is important to note that the migration time became more than twice that of the baseline.

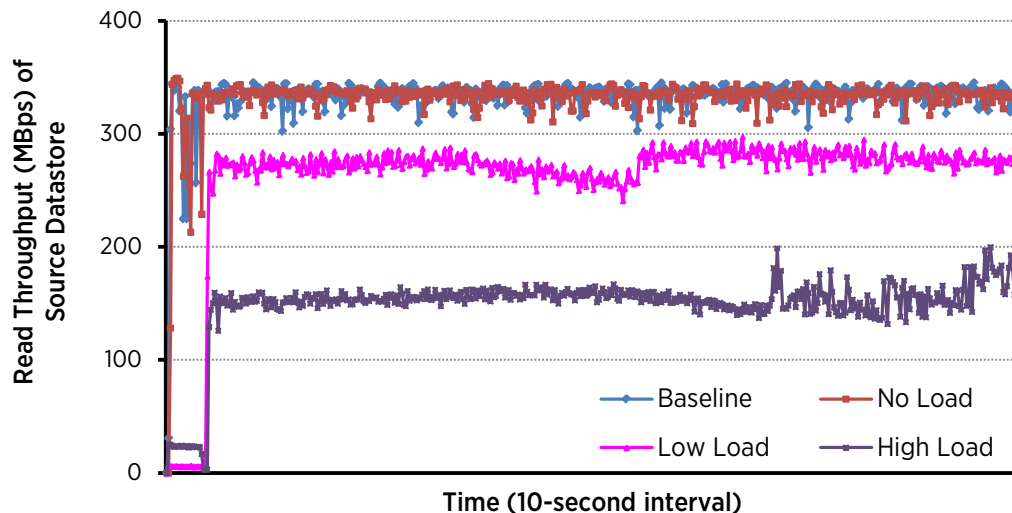


Figure 3. Read Throughput at Source Datastore During svMotion of Data Virtual Disk Under Different Load Conditions

LOAD CONDITIONS	% INCREASE IN DISK MIGRATION TIME	% DECREASE IN READ THROUGHPUT AT SOURCE DATASTORE
DS2 inactive	0	0
DS2 generated low load	22	22
DS2 generated high load	>100	56

Table 2. Comparison of Increase in Disk Migration Time with Decreased in Read Throughput at Source Datastore

Figure 3 compares the read throughput of the source datastore under different load conditions. As the load level of the application increased, the read throughput decreased. Table 2 compares the percentage increase in the disk migration time at different load levels with the corresponding percentage decrease in read throughput at the source datastore. It appears from Figure 3 and Table 2 that the reduction in the read throughput caused the increase in the disk migration time.

It should be noted that the I/O operations on the data virtual disk of the DS2 workload is dominated by random reads of 64KB size. The svMotion traffic is highly sequential, and comprises 64KB read requests on the source datastore. The random read traffic from the application interfered with the sequential read access from svMotion and negatively affected the prefetch algorithms in the array firmware. This resulted in a reduced sequential read throughput which increased the disk migration time. The reason for a small jump in the sequential read throughput during svMotion under both low and high load conditions is not known at this time. This is being further investigated.

LOAD CONDITIONS	AVERAGE DS2 BASELINE THROUGHPUT	AVERAGE DS2 THROUGHPUT DURING VIRTUAL DISK MIGRATION
Low	89.41	87.92
High	455.27	427

Table 3. DS2 Throughput During Data Virtual Disk Migrations

Table 3 compares the DS2 throughput when migrating the data virtual disk with the baseline DS2 throughput. The throughput of the DS2 workload when migrating the data virtual disk was very close to the baseline throughput under low load conditions. Under high load, DS2 throughput reduced by 8% when migrating the data virtual disk.

Migrating Virtual Disk Containing Index Files

In this test case, one of the virtual disks containing index files of the DS2 workload was migrated under different load conditions. The total migration time in each case is shown in Table 4. Also shown is the baseline migration time for moving an index virtual disk when the VM was powered off.

LOAD CONDITIONS	DISK MIGRATION TIME (SECONDS)
VM OFF	258
DS2 inactive	257
DS2 generated low load	273
DS2 generated high load	318

Table 4. Time to Migrate Index Virtual Disk

With the VM on, but DS2 inactive, disk migration time was similar to that of the baseline. When the DS2 workload became active and the client started issuing order requests to the database, under a low load condition, there was a visible increase in the disk migration time. It increased further when the load level became high.

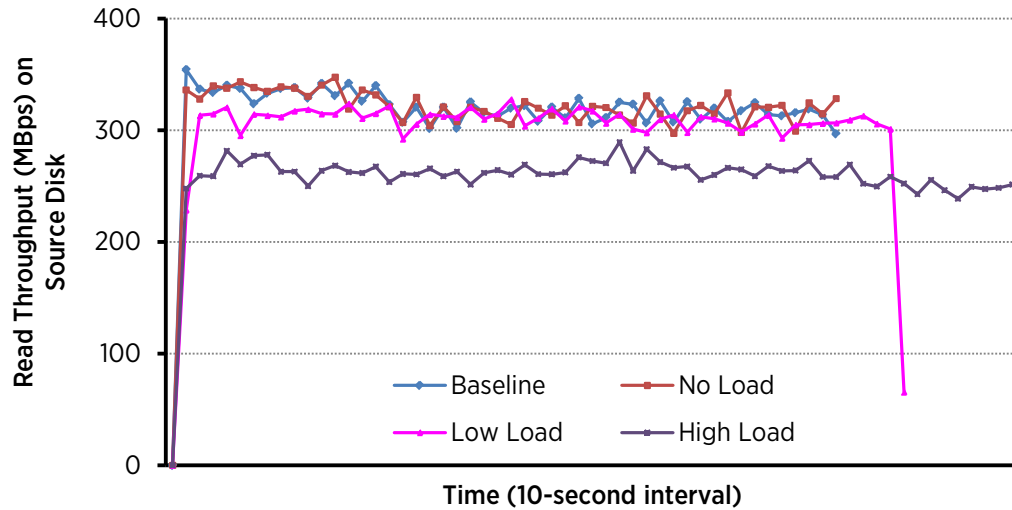


Figure 4. Read Throughput at Source Datastore During svMotion of Index Virtual Disk Under Different Load Conditions

LOAD CONDITIONS	% INCREASE IN DISK MIGRATION TIME	% DECREASE IN READ THROUGHPUT AT SOURCE DATASTORE
DS2 inactive	0	0
DS2 generated low load	6	4
DS2 generated high load	23	18

Table 5. Comparison of Increase in Disk Migration Time with Decreased in Read Throughput at Source Datastore

Figure 4 compares the read throughput of the source datastore under different load conditions when migrating the index virtual disk. As the load level of the workload increased, the read throughput decreased. Table 5 compares the percentage increase in the disk migration time at different load levels with the corresponding percentage decrease in read throughput at the source datastore. It is evident from Figure 4 and Table 5 that the reduction in the read throughput is the main cause for the increase in the disk migration time.

It should be noted that the I/O operations on the index virtual disk of the DS2 workload was dominated by random reads of 64KB in size as is the case for the data virtual disk. The random read traffic from the workload interfered with the sequential read access from svMotion which negatively affected the prefetch algorithms in the array firmware. Hence the sequential read throughput reduced, and the disk migration time increased.

LOAD CONDITIONS	AVERAGE DS2 BASELINE THROUGHPUT	AVERAGE DS2 THROUGHPUT DURING VIRTUAL DISK MIGRATION
Low	89.41	87.44
High	455.27	404.78

Table 6. DS2 Throughput during Index Virtual Disk Migration

Under low load, the throughput of the DS2 workload when migrating the index virtual disk was close to the baseline throughput. Under heavy load, the throughput decreased by 12% compared to that during the steady

state. This can be attributed to the increase in the read latency on the index virtual disk that was migrated. The read latency increased from 5.16ms to 7.8ms (51%) during the migration. Since only one of the four index virtual disks was migrated, 1/4th of all the index seeks could have experienced a near 51% increase in read latency. This increased latency for index seeks probably slowed 1/4th of all the order requests issued by the DS2 client by nearly 51%. In other words, nearly 51% of the DS2 order requests that required seeks from the index virtual disk that was migrated were probably lost. This translates to a total reduction of 12.7%^{*} in the orders completed (throughput) by DS2 which is very close to the 12% throughput reduction observed during the index virtual disk migration.

Migrating Virtual Disk Containing Log Files

In this test case, the virtual disk containing the log file of the DS2 workload was migrated under different load conditions. The total migration time in each case is shown in Table 7. Also shown is the baseline migration time for moving an index virtual disk when the VM was powered off.

LOAD CONDITIONS	DISK MIGRATION TIME (SECONDS)
VM OFF	155
DS2 inactive	156.4
DS2 generated low load	154.17
DS2 generated high load	155.4

Table 7. Time to Migrate Data Virtual Disk

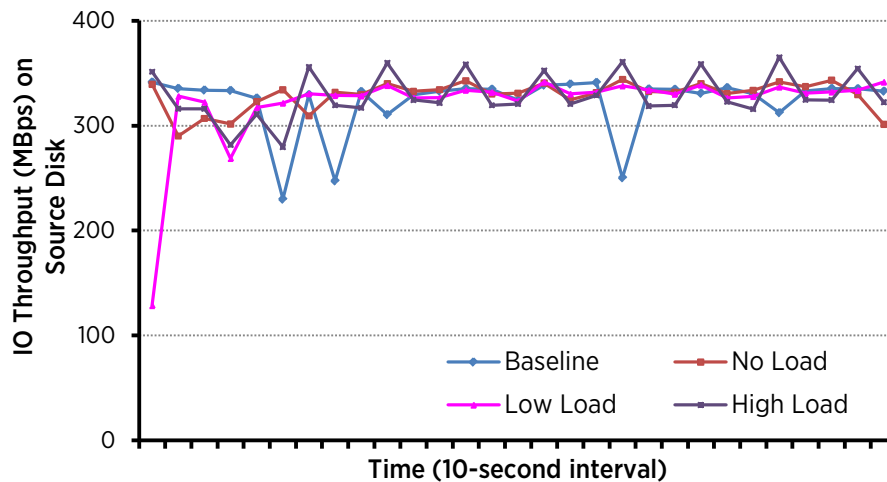


Figure 5. Read Throughput at Source Datastore During svMotion of Log Virtual Disk Under Different Load Conditions

Disk migration time was almost the same in all the test cases as seen in Table 8. The read throughput at the source datastore remained nearly the same when migrating the log virtual disk under different load conditions as shown in Figure 5. Log flush (writes to the log file) is sequential. Two sequential traffic flows on the same datastore clearly didn't interfere with each other and the prefetch algorithms in the array firmware were not impacted at all. Thus, virtual disk migration time remained nearly the same under all load conditions.

* 51% of 1/4th of all order requests

LOAD CONDITIONS	AVERAGE DS2 BASELINE THROUGHPUT	AVERAGE DS2 THROUGHPUT DURING VIRTUAL DISK MIGRATION
Low	89.41	85.88
High	455.27	410.22

Table 8. DS2 Throughput During Log Virtual Disk Migration

It is well known that any increase in the latency of log writes can affect the database performance. An increase in the latency of log writes during log virtual disk migration caused a 4% reduction in the throughput of the DS2 workload under low load and a 10% reduction under high load conditions.

CPU Cost of Disk Migration

svMotion involves a high amount of I/O activity during the virtual disk migrations. Processing large amount of I/O requires CPU cycles. To get a better understanding of the cost of disk migration, the CPU cost of an order request of the DS2 workload, during the baseline test and during each virtual disk migration, was computed. The methodology used is explained in the section “[Estimation of the CPU Cost of an Order Request of DS2.](#)”

DISK TYPE	LOW LOAD		HIGH LOAD	
	BASELINE	DURING VIRTUAL DISK MIGRATION	BASELINE	DURING VIRTUAL DISK MIGRATION
Data	33.68	39.91	24.57	25.71
Index		40.71		27.40
Log		41.34		27.80

Table 9. Comparison of CPU Cost (in Mega Cycles) of an Order Request by DS2 without and with Virtual Disk Migrations

Table 9 shows the CPU cost of an order request issued by DS2 user threads under various test conditions. Under a low load, the CPU cost increased by approximately 18-22% during virtual disk migrations. Under a high load, the CPU cost increased by up to 11.5%. The percentage increase in CPU cost was higher under low load possibly due to the extra wait by DS2 user threads for completing an order request. There weren't enough order requests issued by the user threads to keep the vCPUs busy. Hence, any additional wait increased the CPU consumption. However, under high load, there was enough work for the vCPUs because there were active user threads issuing order requests, in spite of the extra wait by a user thread. Therefore, the percentage increase in CPU cost was lower.

Migrating Virtual Disks from Newer Array to Older Array

In the second phase of the study, the following experiments were conducted to study the impact of application I/O characteristics and svMotion on each other.

1. Migrating the virtual disk containing **data files** of the DS2 workload from the newer array to the older array
2. Migrating the virtual disk containing **index files** of the DS2 workload from the newer array to the older array
3. Migrating the virtual disk containing **log file** of the DS2 workload from the newer array to the older array

Each experiment was conducted only with the following load conditions:

- a. VM was on; DS2 generated a high load

Test Setup

The test bed used for the second part of the study was the same as that in the first part (see Figure 1) except that all the database files belonging to the DS2 workload were placed on the virtual disks created in the VMFS datastores that were located in the newer array (NS 960). The virtual disks were migrated from the datastores in the newer (source) array to the datastores in the older (destination) array.

Baseline

As done previously, two types of tests were done to capture the baseline numbers:

1. Migration time of each virtual disk when the VM was powered off
2. Steady state throughput of the DS2 workload when there were no virtual disk migrations

The migration time of each virtual disk with the associated baseline was compared with the corresponding migration time of the virtual disk observed during the tests discussed in the section "Migrating Virtual Disks from Older Array to Newer Array." Steady state throughput of the DS2 workload with its virtual disks on the newer array formed the baseline for comparing the DS2 throughput during individual virtual disk migrations.

Migrating Virtual Disk Containing Data Files

In this test case, the virtual disk containing data files of the DS2 workload was migrated from the source to the destination array under a high load condition.

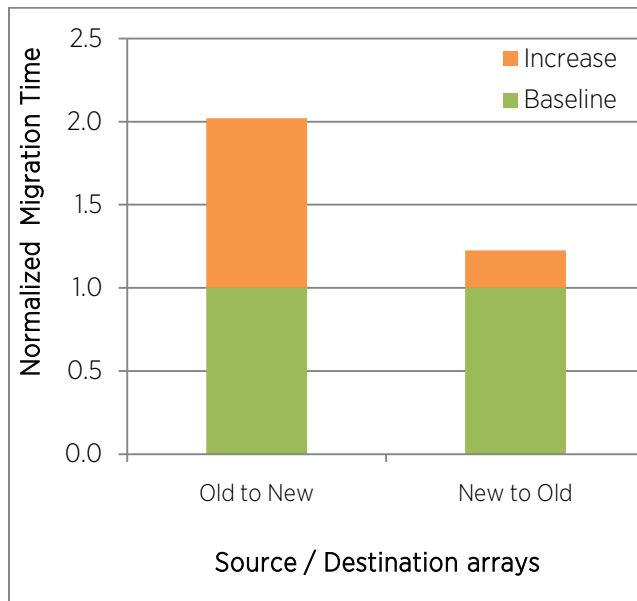


Figure 6. Comparison of Time to Migrate Data Virtual Disk

The orange portion of each vertical bar in Figure 6 shows the increase in time to migrate the data virtual disk under a high load condition compared to that when the VM was powered off. When the data virtual disk was moved from the older array to the newer array, the migration time increased by slightly more than 100%. It increased by 22% when the data virtual disk was moved from the newer array to the older array.

LOAD CONDITIONS	AVERAGE DS2 BASELINE THROUGHPUT	AVERAGE DS2 THROUGHPUT DURING VIRTUAL DISK MIGRATION
High	474.98	431.24

Table 10. DS2 Throughput During Data Virtual Disk Migration

Throughput of DS2 workload reduced by 10% when migrating the data virtual disk.

Migrating Virtual Disk Containing Index Files

In this test case, one of the virtual disks containing index files of the DS2 workload was migrated under a high load condition.

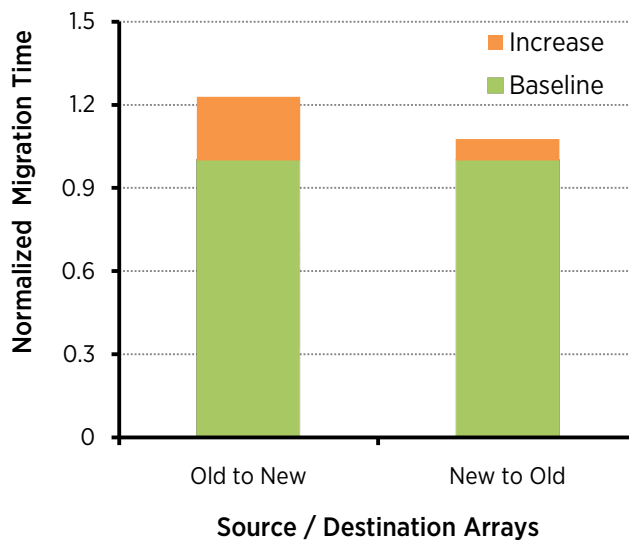


Figure 7. Comparison of Time to Migrate Index Virtual Disk

The orange portion of each vertical bar in Figure 7 shows the increase in time to migrate one of the index virtual disks under high load condition compared to that when the VM was powered off. When the index virtual disk was moved from the older array to the newer array, the migration time increased by 23%. It increased by 7% when the index virtual disk was moved from the newer array to the older array.

LOAD CONDITIONS	AVERAGE DS2 BASELINE THROUGHPUT	AVERAGE DS2 THROUGHPUT DURING VIRTUAL DISK MIGRATION
High	474.98	463.04

Table 11. DS2 Throughput During Index Virtual Disk Migration

DS2 throughput in this case reduced by 3% when one of the index virtual disks was migrated.

Migrating Virtual Disk Containing Log File

In this test case, the log virtual disk of the DS2 workload was migrated under a high load condition. There was no noticeable increase in migration time as it was when the log virtual disk was moved from the older array to the newer array.

LOAD CONDITIONS	AVERAGE DS2 BASELINE THROUGHPUT	AVERAGE DS2 THROUGHPUT DURING VIRTUAL DISK MIGRATION
High	474.98	450.61

Table 12. DS2 Throughput During Log Virtual Disk Migration

In this case, DS2 throughput reduced by 5% when migrating the log virtual disk.

CPU Cost of Disk Migration

As done previously, the CPU cost of an order request of the DS2 workload during the baseline case and during each virtual disk migration was computed using the methodology explained in the section [“Estimation of the CPU Cost of an Order Request of DS2.”](#)

DISK TYPE	HIGH LOAD	
	BASELINE	DURING VIRTUAL DISK MIGRATION
Data	24.41	26.54
Index		26.01
Log		25.58

Table 13. Comparison of CPU cost (in Mega Cycles) of an Order Request by DS2 without and with virtual disk migrations

When a virtual disk was migrated, the CPU cost of an order request increased by approximately 5-7%.

Best Practices for an Optimal svMotion Experience

There are two main factors that can influence the svMotion of virtual disks in a vSphere virtual infrastructure: I/O access patterns on the virtual disk that has to be migrated and the underlying storage infrastructure. While it is hard to expect change to the former, careful considerations given to the latter can help in achieving a better svMotion experience and in minimizing impact to the applications that run in a virtual infrastructure where the VM storage is migrated. Based on the knowledge gained during the tests, the following best practices can help while planning an infrastructure that is capable of allowing live migration of VM storage:

1. Random access patterns on virtual disks interfere with the sequential access of svMotion and negatively affect the I/O throughput of svMotion. This can increase migration time by a significant value. If there is a need to migrate such virtual disks, plan to schedule the migration during periods when there is low or no I/O activity on the virtual disk.
2. Sequential access patterns of a VM on a VM's virtual disk (for example, writes to log files in a virtual disk) generally don't affect the sequential access pattern of svMotion. Even with more than one sequential stream on the virtual disk, most modern arrays are capable of utilizing the I/O prefetching logic in the arrays' firmware to improve the I/O performance. Such virtual disks could be migrated even when there is some I/O activity on them.

However, if the VM's I/O access to its virtual disk is significant, then the svMotion traffic will have to contend with the VM's access for the I/O bandwidth. This could reduce the throughput for both the traffic flows. In such situations, it is better to schedule svMotion during periods when the existing I/O load level reduces.

1. svMotion moves data blocks in 64KB chunks. Any I/O operation from the VM that uses a small request size might see higher access latency due to the large sized blocks of svMotion traffic. If the application in the VM is very sensitive to an increase in the access latency, consider scheduling svMotion during periods of low or no I/O activity on the virtual disk.
2. Most applications that rely on random access patterns to fetch data from a physical storage media may not benefit from cache in the storage array. In such situations, administrators tend to configure (if permissible) a significant chunk of array cache for write access patterns. This may limit the amount of data prefetched by the array for svMotion which can potentially impact the disk migration time. Having enough buffer space to hold the prefetched data for svMotion may help in reducing the disk migration time.
3. In certain situations such as the experiments discussed in this paper, moving a virtual disk from an array with newer hardware, firmware, and larger cache to older arrays could be faster than the other way around.
4. Faster storage media such as solid state disks (SSDs) provide a faster access to the smaller blocks even in the presence of requests for larger blocks. By utilizing the services of SSDs (for example, storage pools built on SSDs and SSD-based secondary cache) in the array, the impact on the performance of the application can be reduced when migrating the application's disks.

Conclusion

Live storage migration is the missing piece in liberating VMs and their associated files completely from the physical hardware on which they reside. Predictable migration times—with minimal impact on the performance of the application accessing the virtual disk that is migrated—are expected from the vSphere's storage vMotion feature. This paper offers a deeper look at the interaction of svMotion with a large, active SQL database workload. The study includes application behavior when migrating individual virtual disks used by the database and the impact application I/O traffic had on the svMotion of a particular virtual disk. The study showed consistent and predictable disk migration time that largely depended on the capabilities of the source and the destination arrays. svMotion increased the CPU consumption of the VM running the test workload from 5% to 22% depending on the load conditions. The I/O patterns of the SQL database workload had noticeable impact on svMotion throughput (and the disk migration time).

Appendix

Details of Hardware and Software Used for the Experiments

Server

- 4 2.8GHz AMD Opteron 8370 processors with 4 cores per processor
- 128GB memory
- 1 single port Qlogic QLE2560 HBA
- 1 dual port Qlogic QLE2562 HBA

Storage Area Network

- CLARiiON CX3-40
 - FLARE OS: 03.26.040.5.031
 - Read cache on each SP: 512MB
 - Write cache: 2500MB
- CLARiiON NS 9-60
 - FLARE OS: 04.30.000.5.512
 - Read Cache on each SP: 1024MB
 - Write Cache: 8736MB
- 1 FC switch

Virtual Platform

- vSphere 5.0

Virtual Machine

- Windows Server 2008 R2 EE x64
- 4 vCPUs
- 30GB memory
- 2 LSI Logic SAS vSCSI controllers
- SQL Server 2005 EE x64 with Service Pack 3

Workload

- DVD Store version 2.1

Storage Layout for the DS Workload in CX 3-40 (Older Array)

COMPONENT VIRTUAL DISK	NUMBER OF VIRTUAL DISKS	VIRTUAL DISK SIZE (GB)	RAID TYPE	NUMBER OF RAID GROUP	NUMBER OF DISKS IN EACH RAID	DISK TYPE
Data	1	600	RAID 0	1	7	FC, 15K RPM
Index	4	80	RAID 0	4	8	FC, 15K RPM
Log	1	50	RAID 0	1	4	FC, 15K RPM

Storage Layout for the DS Workload in NS 960 (Newer Array)

COMPONENT VIRTUAL DISK	NUMBER OF VIRTUAL DISKS	VIRTUAL DISK SIZE (GB)	RAID TYPE	NUMBER OF RAID GROUP	NUMBER OF DISKS IN EACH RAID	DISK TYPE
Data	1	600	RAID 0	1	6	FC, 15K RPM
Index	4	80	RAID 0	1	4	SATA II, SSD
Log	1	50	RAID 0	1	4	FC, 15K RPM

Estimation of the CPU Cost of an Order Request of DS2

To estimate the CPU cost of an order request issued by a DS2 user thread, the average number of orders completed per second was computed from the DS2 performance metrics during a desired test interval (steady state or during virtual disk migration). For the same interval, the average host %processor utilization on was recorded using esxtop. Then the CPU cost of a single order request was computed using the following equation:

$$\frac{\text{Average \%Processor Utilization of the host} \times \text{Rated CPU Clock Frequency} \times \text{Number of Physical Cores}}{100 \times \text{Average Orders per second}}$$

References

1. *Storage Distributed Resource Scheduler (SDRS)*. VMware, Inc., 2011. <http://www.vmware.com/products/datacenter-virtualization/vsphere/vsphere-storage-drs/features.html>.
2. Mashtizadeh, Ali, Emr  Celebi, Tal Garfinkel, and Min Cai. *The Design and Evolution of Live Storage Migration in VMware ESX*. USENIX, 2011. http://www.usenix.org/events/atc11/tech/final_files/Mashtizadeh.pdf.
3. *Dell DVD Store Database Test Suite*. Dell, 2011. <http://linux.dell.com/dvdstore/>.
4. "Disk Partition Alignment Best Practices for SQL Server [2008]," *MSDN Library*. Microsoft, 2009. [http://msdn.microsoft.com/en-us/library/dd758814\(v=sql.100\).aspx](http://msdn.microsoft.com/en-us/library/dd758814(v=sql.100).aspx).
5. Pan, Zhelong. "Interpreting esxtop statistics." *VMware Communities*. VMware, Inc., 2008. <http://communities.vmware.com/docs/DOC-9279>.

About the Author

Chethan Kumar is a senior member of Performance Engineering at VMware, where his work focuses on performance-related topics concerning database and storage. He has presented his findings in white papers and blog articles, and technical papers in academic conferences and at VMworld.

