IBM XIV® Storage System

Snapshots Reinvented

White Paper

September 2008

This document could include technical inaccuracies or typographical errors. IBM may not offer the products, services or features discussed in this document in other countries, and the product information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. The information contained in this document is current as of the initial date of publication only and is subject to change without notice. All performance information was determined in a controlled environment. Actual results may vary. Performance information is provided "AS IS" and no warranties or guarantees are expressed or implied by IBM. Information concerning non-IBM products was obtained from the suppliers of their products, their published announcements or other publicly available sources. Questions on the capabilities of the non-IBM products should be addressed with the suppliers. IBM does not warrant that the information offered herein will meet your requirements or those of your distributors or customers. IBM provides this information "AS IS" without warranty. IBM disclaims all warranties, express or implied, including the implied warranties of noninfringement, merchantability and fitness for a particular purpose or noninfringement. IBM products are warranted according to the terms and conditions of the agreements under which they are provided.

info@xivstorage.com
www.xivstorage.com

Contents

Introduction

The ability to create and manage snapshots of existing volumes is among the most fundamental features commonly offered by enterprise-class storage systems today. The IBM XIV® Storage System has taken this concept one step further, offering a completely innovative approach to snapshot creation and management. The XIV system offers clear advantages over alternative approaches currently available in the industry, including the following:

- Unlimited number of snapshots in the system.
- Snapshot creation in virtually zero time, regardless of the size of replicated volumes.
- Unaffected performance levels in a system that supports snapshots, regardless of the number of snapshots currently defined in the system.

In this white paper, we describe the XIV snapshot architecture and explain its underlying advantages in terms of performance, ease of use, flexibility and reliability. We explain how these advantages are achieved by harnessing the power afforded by the system's unique overall grid architecture.

Why Snapshot Volumes?

A snapshot is a logical volume whose contents are identical to that of a given source volume at a specific point in time. Whenever a snapshot is created in a storage system, the source volume continues its normal I\O activity without affecting the snapshot.

Nowadays, snapshots are used in the industry for various types of tasks, including the following:

- **Data Backup to External Media.** Snapshots enable hot backups that can be saved in external media and eventually used for recovery purposes. Using the snapshot for backup enables a consistent point-in-time copy of the whole volume without stopping any running application.
- **Recovery from Logical Failure.** Data that is safely stored in the system is sometimes mistakenly modified or deleted by users or administrators. Logical failures of these kinds derive from problems in the application running in the host (such as database corruption or from users' mistakes (such as accidental file deletion). Snapshot-supported backup mechanisms can easily assist recovery from such situations. The recovery process supported by snapshots can be performed in one of two ways:
    - Restoring the contents of the volume from the snapshot, thus essentially returning to the point in time where the snapshot was created.
    - Retrieving the corrupted data selectively from the snapshot and copying it at the application level to the source volume.

- **Testing.** While developing and testing new software versions, developers need to use full copies of existing sets of data. Snapshots provide the ability to do so. For the purposes of testing, snapshots may be configured as writeable volumes that the host may modify like any other independent volume in the system.
- **Data Mining.** Many IT environments implement data mining procedures that require the use of point-in-time pictures of the contents of a volume, while the latter continues to be modified by the host. Snapshots are used in such cases.

A Customer Wishlist

Today's enterprise customers expect certain features in a snapshot mechanism, such as:

- **Instant, Straightforward Creation of Snapshots.** Requires minimal planning ahead and minimal use of the administrator's time.
- **Multiple Snapshots.** Reduces the time lost in restoring data and improves the fine granularity of restoring capabilities.
- **Writable Snapshots.** Becomes essential when snapshots are used for development and testing. Database systems (e.g. Oracle), as well as other applications, require write access to disk to update and save an application state's even when the operation requested is read only (e.g. table scan).
- **Create a Snapshot of a Snapshot (a tree of snapshots).** The ability to create snapshots of writable snapshots. This enables the continued use of the writable snapshot, with the ability to restore from any read-only snapshot of that writable snapshot.
- **Restore Repeatedly from Snapshots.** The ability to restore from a snapshot into primary copy and, if that point-in-time is not desirable, then restore from another snapshot.
- **Differential Snapshots.** Creates actual copies of data only where the source and the snapshot currently differ, improving performance and the efficiency of storage space use in the system.
- **Minimal Performance Degradation.** Creation and management of snapshots should not affect the overall performance of the system, especially for the customers' production volumes.
- **Scalability.** All the above requirements (instance creation, multiple snapshots, and reduction of performance penalties) should apply equally to both small and large volumes and not be compromised when the overall amount of data in the system continues to grow.

Snapshot Implementation in the XIV System

The IBM XIV Storage System implements a totally revolutionary approach to all aspects of snapshot creation and management, substantially overcoming the limitations of other technologies available in the market (see appendix for details of the challenges of other snapshot technologies).

The most salient features of the XIV system's approach to snapshots include:

- Instant sub-second snapshot creation
- Thousands of snapshots allowable per volume
- Virtually no performance degradation
- Instant sub-second restore of source volume contents from the snapshot
- Performance, creation time, and restore time independent of volume size and number of snapshots
- Writable snapshots with all the above properties
- Snap-on-snap for flexible system development and testing
- Ability to restore repeatedly from any snapshot

These important breakthroughs are made possible through the fundamental architectural features of the XIV system, as a fully scalable grid storage platform built around two main principles:

- The data modules are implemented as standard Intel-based servers comprising, all within the same physical unit, the cache and the disks, with two PCI-X buses interconnecting them.
- Every logical volume in the system is stored as data portions distributed across **all data modules** and **all disk drives** in the system.

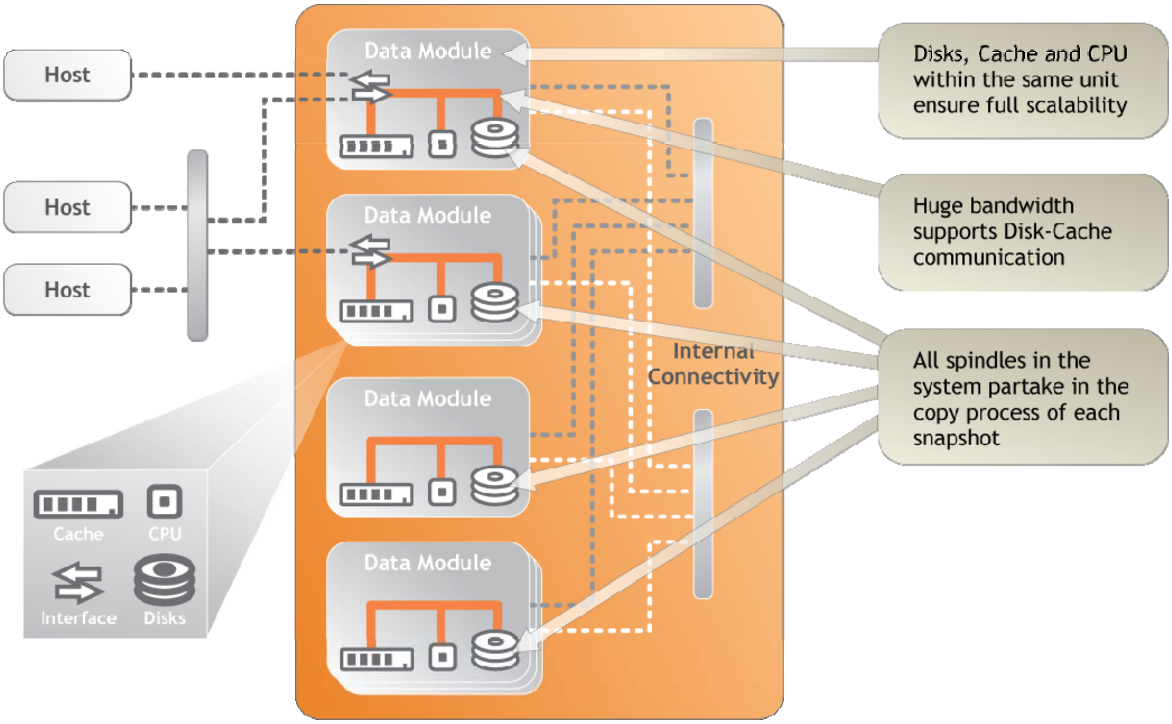
Host
Host
Host
Data Module
Data Module
Data Module
Data Module
Internal Connectivity
Cache
CPU
Interface
Disks
Disks, Cache and CPU within the same unit ensure full scalability
Huge bandwidth supports Disk-Cache communication
All spindles in the system partake in the copy process of each snapshot

Figure 1: XIV system architecture and its inherent advantages in snapshot creation

The architectural advantages that derive from these principles include the following:

- Growth in the system's storage capacity is always accompanied by a parallel growth of its CPU power and its cache memory area, thus ensuring full scalability of all internal processes and, in particular, of snapshot processes
- Data duplication is always performed internally within a system's modules, with a huge bandwidth supporting disk to cache communication and with no need for across-shelf communication
- Each snapshot-related process is concurrently performed over all modules and over all spindles system-wide
- An innovative (patent protected) metadata design supports a practically zero-time process of snapshot creation
- The basic data duplication mechanism is redirect-on-write (rather than copy-on-write) and it drastically reduces the impact of snapshots on the system's performance

The following sections detail these innovations, while indicating their natural incorporation into the overall architecture of the XIV system. The overall architectural advantages of the system are fully harnessed on behalf of its snapshot mechanisms and apply equally efficiently to writable and non-writable snapshots.

Innovative Metadata Design

For all storage systems, creating and handling differential snapshots requires the implementation of complex sets of metadata to determine which data portions associated with a volume are used by which snapshots and/or source volume. However, in the XIV system, snapshot creation is a practically zero-time command, because ***the size of the metadata used in this process is truly minimal at the time of creating the snapshot***.

For example, suppose we have a five TB volume in our system that we want to snapshot every four hours. After one year, we reach a total of 2000 snapshots for that volume. Irrespective of this large number of copies and the large size of the initial volume, the XIV system defines only a very small, fixed-size metadata header at the time of creation of each snapshot. This metadata grows only as physical copies of data portions are added – and only if they are modified in the source volume. Under these conditions, the very concept of 2000 snapshots of even a large volume becomes truly feasible; for that matter, the seemingly large number of snapshots affects only slightly, if at all, the overall performance of the system.

Consider a second example, which illustrates how snapshot creation is a practically zero-time command in the XIV system: Suppose we have a 30 TB volume, for which we want to create one snapshot. As long as the source volume is not modified by incoming write requests, the volume and snapshot continue to share the physical copy of all data. As long as the volume and snapshot are identical, the snapshot has practically no metadata memory requirements.

Redirect on Write

Rather than the standard copy-on-write mechanism implemented when a write request modifies a data slot shared by source and snapshot, the XIV system implements a different approach called *redirect-on-write*. The traditional copy-on-write approach involves three disk-seek operations, whereas the XIV system's redirect-on-write involves only two such operations, as indicated in the following diagram:

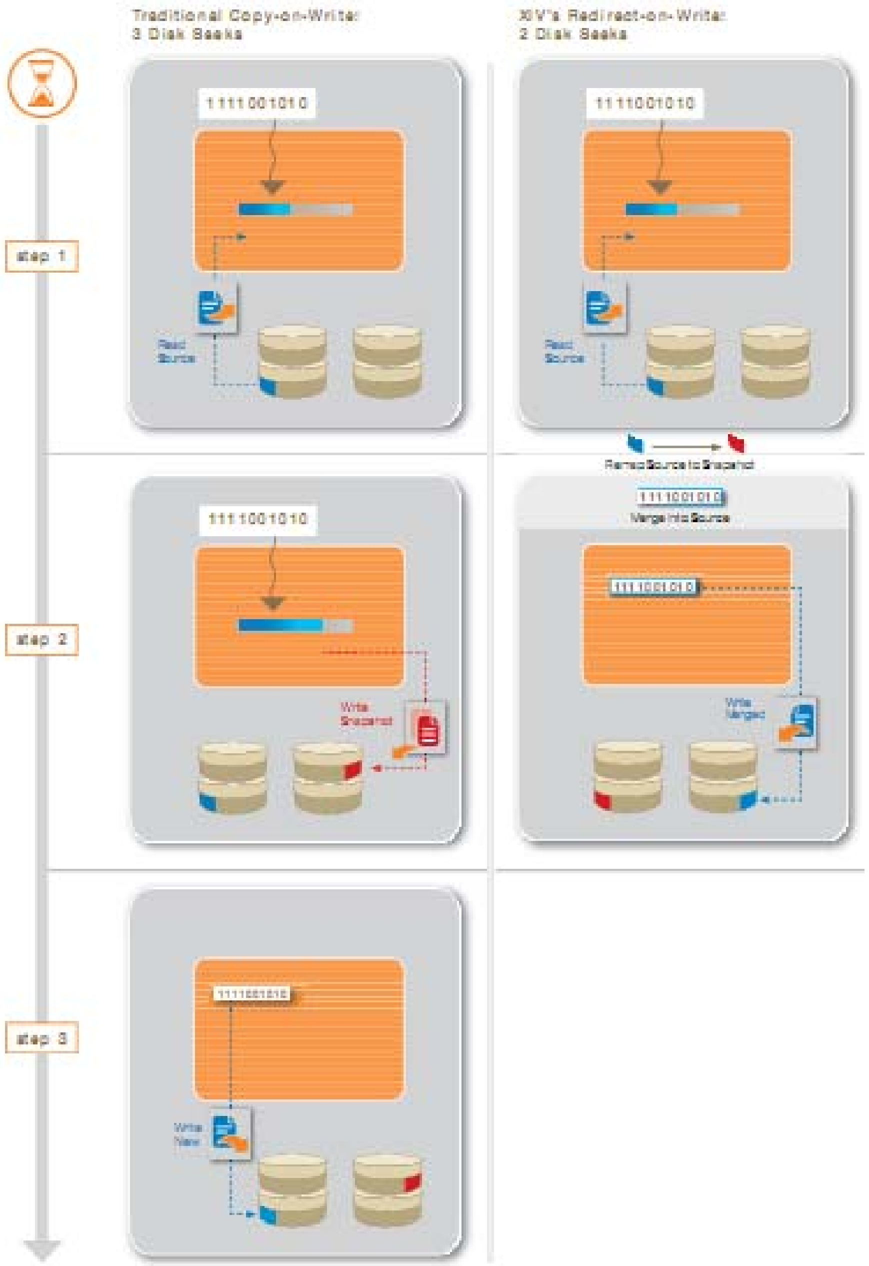

Figure 2: Copy-on-write vs. Redirect-on-write

Intra-shelf Copying and the XIV System's Scalable Grid

The performance penalty incurred by copy-on-write processes is unavoidable in differential snapshot implementations. The redirect-on-write approach sensibly decreases, but does not completely eliminate, the penalty. Still, the XIV system's unique architecture provides unmatched superb performance under snapshot activity, due to its basic architectural principles.

- **Performs redirect-on-write entirely within a module, with no intra-shelf communication.** Given the sophisticated algorithm the XIV system uses to distribute data across the system, the source copy of any specific portion and all physical snapshots of it always reside within the same module. Each module is a standard Intel-based server that comprises, all within the same physical unit, the cache and disks, with two PCI-X buses interconnecting them.

  As a result, a truly huge bandwidth is put to the service of the redirect-on-write operation and no common bottleneck can arise. Consequently, overhead is considerably reduced. In contrast, traditional storage systems perform copy-on-write between disk shelves through a common controller connected via FC or a common bus.

- **Performs redirect-on-write concurrently on all modules in the system.** Indeed, there is no degradation of the system's performance when storage capacity and memory are added. In contrast, traditional systems perform copy-on-write using a limited number of controllers.

Powerful Snapshot Features

Based on the unique architectural advantages described above, the XIV system offers some very important, powerful snapshot features, including:

Writable Snapshots

Any snapshot can be defined in the XIV system as a **writeable snapshot**. When this snapshot is mapped to a host, the host views it as a standard read/write volume rather than read-only, as is the case with snapshots not defined as **writeable** or as implemented by most vendors. The snapshot remains transparent to the host whether it has been mapped to a standard volume or to a snapshot. A writable snapshot continues to share non-modified data with its source volume in the XIV system. Thus, only data that has specifically been written to this snapshot consumes additional physical resources. Moreover, writeable snapshots can even be resized. The additional capacity added to the writable snapshot is not consumed in practice until it is actually written to.

Writeable snapshots are mainly used in two scenarios:

- **For block-level read/write access.** Many applications require block-level read/write access even when the application-level access is read-only. This may be caused by file system metadata, signature block, or other related circumstances. In such cases, it is necessary that snapshots be writeable. As changes to data in situations of this kind are typically minimal, the fact that writeable snapshots in the XIV system consume precious little physical space proves crucial.
- **For testing.** Using snapshots for testing is possible only if snapshots are writeable. Also, in this case, the XIV system's minimal usage of physical space for writeable snapshots makes the testing environment much more resource-efficient.

Snapshot of a Snapshot

Once a snapshot has been defined as writeable, it becomes natural to ask whether one can take a snapshot of a snapshot. Although very intuitive and plainly usable, this feature is very seldom offered by other storage vendors. The XIV system naturally supports producing snapshots of existing writeable snapshots, without losing any of its other remarkable features, including: instant creation, minimal space consumption based on actual usage only, and high performance.

The snapshot-of-a-snapshot feature is extremely advantageous in complex development and testing environments. This feature saves considerable storage space (by eliminating the need for full copies), enables better testing and development (by easily producing multiple snapshots) and provides logical backup, even for applications that use snapshots.

Restoring a Volume from a Writeable Snapshot

A writeable snapshot can even be used as a source in a volume restore operation. At first glance this might seem contradictory, since the writable snapshot has probably been written to and therefore differs from its original contents. Thus, a writeable snapshot does not typically reflect any point-in-time state of the master volume. Nonetheless, in practice, such restore situations are needed. It turns out that restoring from a writeable snapshot is the only way to achieve a fast and convenient restore during application development or testing. With the XIV system, one can essentially create a tree of snapshots in which each branch of the tree is a self-contained development environment. The XIV system provides this unique feature as a natural extension of its simple and powerful snapshot architecture.

Overriding an Existing Snapshot

Side by side with standard snapshot implementation, in which the taking of a snapshot of a volume creates a **new** snapshot volume, the XIV system offers an alternative in which the current contents of the volume are logically copied into an **existing** snapshot. This feature is useful when the existing snapshot is already mapped to a host. In this case, the new content can be used without changes to the volume-mapping and SCSI serial number, either in the XIV system or the host. The host is thus spared the need to run the operating system level rescan process. Of course, the application using the data must be restarted, as the entire contents have been changed. This feature makes the creation of backup environments much simpler and faster.

Snapshot Scenarios

We present here several scenarios that highlight how the XIV system's features and unique architecture solve problems previously unsolvable.

Scenario #1: Testing Remote Mirroring without Interrupting the Mirroring Process

Almost every critical high-end application today is mirrored as a backup on a secondary site. Like any other backup solution, this secondary site has to be tested from time to time or else it would probably fail to work when really needed.

Testing the secondary system means activating the servers and using the data on the secondary site. With traditional storage solutions, this cannot be done without interrupting the mirroring process, consequently creating a window of time in which data is not replicated in the secondary site.

The XIV system provides a simple process for testing the secondary site without interrupting the mirroring process, as follows:

1. Create a snapshot of the secondary volume
2. Make the snapshot writable
3. Map the snapshot to the servers on the secondary site as if it were the source volume

After these simple steps are performed, the environment reaches the following configuration:

- Mirroring to the secondary volume never interrupted; the application remains unaware of the drill in process
- Secondary servers are mapped to a writeable snapshot, fully simulating a disaster scenario

Once the drill is over, snapshots are simply deleted, with no need to recover or synchronize information.

Scenario #2: Testing Volume Size Change

From time to time, one needs to increase a volume's size due to increased application demands. The XIV system's built-in virtualization capabilities make this operation trivial on the storage side, but do not solve the complexity on the application side. Increasing the volume size is a complex operation – and one not supported by all applications-- that should be done carefully. Except for a handful of applications, resizing volumes requires application downtime. Reducing downtime is, by all standards, one of the key metrics for datacenter excellence.

Although the XIV system cannot solve application problems, it can simplify the testing of such processes. The following scheme can be used easily to test volume resizing without interrupting production:

1. Take a snapshot of the production volume.
2. Make the snapshot writeable.
3. Create a new instance of the application for testing and map the newly-created writable snapshot to this instance.
4. Follow the procedure for volume size increase using the new instance, increasing the size of the writable snapshot, and fully debugging the process until all problems are resolved.
5. Follow the procedure for the production volume, while keeping the writable snapshot for reference.
6. Delete the writable snapshot after the operation is complete.

This procedure provides a scheme for off-line testing of the complex procedure, without interrupting production. The testing process is a full replication of the production process: while the application instance used for testing sees a read/write volume being increased, it is unaware that this is a snapshot and not a volume.

Furthermore, the snapshot creation and resizing is performed in practically zero time, without actual data copy, and the process can be tested multiple times without any delay. In addition, since snapshots are differential, the actual space required to increase the volume is not consumed at the testing phase.

Summary

In a truly substantial way, the XIV system delivers a groundbreaking approach to snapshot use that overcomes the most significant performance penalties currently affecting other vendors' products. Its snapshot mechanism harnesses the full power of XIV technology and relies naturally on the advantages provided by the unique XIV grid architecture.

The following two tables summarize the advantages of the XIV system's approach to snapshot creation over that of other leading storage vendors:

Table 1: Applications-based Comparison

Application	IBM® XIV™ Storage System	Other Leading Vendors
Physical backup	Instant creation of snapshot simplifies the backup process High performance with snapshots	Complex snapshot creation results in a complex process Backup snapshots reduce performance
Logical backup	Multiple snapshots with no performance overhead make multi-generation logical backup a reality	No practical solution to multi-generation logical backup
Testing and development environments	Multiple snapshots with instant creation make the process simple and effective High performance with multiple snapshots enables effective usage Writable snapshots enable testing Snapshot of snapshot provides logical backup for development environments	Limited functionality and performance make the process ineffective and wasteful
Data mining	High performance with snapshots enables data mining Instant creation makes the process simpler	Performance problems and complex creation limit effectiveness

Table 2: Features-based Comparison

Feature	IBM® XIV™ Storage System	Other Leading Vendors
Metadata	**Underlying principle:** Size proportional to number of changes **Specific advantages:** • Near zero-time creation regardless of volume size • Unlimited number of snapshots	**Underlying principle:** Size proportional to number of snapshots and volume size **Specific disadvantages:** • Creation time proportional to volume size • Limited number of snapshots
Copying technique	**Underlying principle:** Redirect-on-write **Specific advantages:** Low performance overhead due to less copying of data	**Underlying principle:** Copy-on-write or full copy **Specific disadvantages:** Higher performance overhead due to increased copying of data
CPU	**Underlying principle:** Fixed CPU/capacity ratio **Specific advantages:** • Fixed time for creation. • Fixed performance overhead	**Underlying principle:** Fixed CPU per overall system capacity **Specific disadvantages:** • Creation time does not scale • Performance overhead does not scale
Copying	**Underlying principle:** Within modules **Specific advantages:** High performance, independent of size	**Underlying principle:** Across modules **Specific disadvantages:** Low performance, fixed with size

Appendix: The State of the Art and its Limitations

Leading storage vendors offer incomplete snapshot solutions nowadays and cannot fully meet customers' requirements as specified in the preceding section. This should not come as a surprise, as the basic architecture of their flagship products was designed 15 to 20 years ago. Snapshot creation functionalities were added to these architectures long after their initial design and in many respects they constitute an unnatural match. As a consequence, customers pay full price and receive much less than they deserve.

Snapshot creation mechanisms typically involve two main, successive stages:

- **Stage 1. Establishing the logical relation between a source and its snapshot.** Stage 1 typically comprises the creation of some a set of metadata. This set is subsequently used as the basis for all activity related with Stage 2 of the mechanism. The metadata enables, for instance, the indicating of those data areas that are currently shared between the source volume and the snapshot. As typically implemented in existing approaches to snapshots, the size of this set of metadata is proportional to the size of the source volume.
- **Stage 2. Gradually creating physical copies of the data stored in the source volume, as data in the source volume continues to be modified.** Stage 2 involves implementing a mechanism commonly known in the industry as "copy-on-write." Each time that a data slot in the source volume is modified by a write request for the first time after snapshot creation, copy-on-write mechanisms typically involve the addition of two I/O interactions with the disk and, hence, two disk-seek actions. Thus, after the incoming data has been safely written to cache memory and the write request has been acknowledged to the host, the command is completed with background activity that involves one read and one write operation in order to fully update the new status in both the source volume and the snapshot. This clearly imposes significant performance overhead penalties on the system and these penalties increase as the number of snapshots in the system continues to grow.

Therefore, the main limitations affecting currently available snapshot mechanisms are derived from problems found in each of the two stages and include the following:

- **Slow Process of Snapshot-creation.** In spite of the typical vendor's statement, snapshots are not really created via atomic, zero-time commands but, rather, in processes that span more than negligible times, depending on the size of the snapshot volume
- **Severe Throughput Degradation.** Especially as the number of snapshots continues to grow
- **Increased Latencies of Individual I\O Transactions.** Especially when using a differential snapshot
- **Limited Number of Snapshots: both per Volume and System-wide.** For certain customers, this becomes a serious limitation on the effectiveness of snapshots as a solution for logical failure scenarios

- **Inefficient Use of Storage Space.** Especially when full volume copy is used as a solution to the performance issues
- **One-time restore only.** Some vendors do not support repeated restore from snapshots