SNIA TECHNICAL TUTORIAL

Storage Virtualization

The SNIA Technical Tutorial booklet series provides introductions to storage technology topics for users of storage networks. The content is prepared by teams of SNIA technical experts and is open to review by the entire SNIA membership. Each booklet corresponds with tutorials delivered by instructors at Storage Networking World and other conferences. To learn more about SNIA Technical Tutorials, e-mail: snia-tutorialmanagers-chair@snia.org.

SNIA TECHNICAL TUTORIAL

Storage Virtualization

Frank Bunn
VERITAS Software

Nik Simpson
Datacore Software

Robert Peglar
XIOtech Corporation

Gene Nagle
StoreAge Networking Technologies

SNIA
Storage Networking Industry Association

Contents

Preface

"Storage virtualization" has become the buzzword in the storage industry, especially with the increased acceptance of storage networks. But besides all the hype, there is considerable confusion, too.

Many users have been disconcerted by the terminology, the different ways in which vendors present virtualization, and the various technical implementations of storage virtualization. In addition, the two alternative approaches to virtualization based on in-band or out-of-band technologies have resulted in much discussion about the pros and cons of the two technologies. Needless to say, in many cases these discussions have generated more "heat than light" and have done little to clarify the issues for end users needing to address their day-to-day storage management challenges and concerns.

Acting in its role as an educator, SNIA has undertaken to offer clearer definitions and terminology in general, and to convey them to both vendors and users. In order to provide a common frame of reference for discussions about storage technology, SNIA has developed the Shared Storage Model (SSM), which allows us to place different storage technologies within a consistent framework and helps users understand where a particular technology is implemented and how it relates to other storage technologies. Accordingly, we've chosen to use SSM as an overall framework for explaining virtualization in this booklet. The model offers a general structure for describing various architectures for accessing storage systems, and can be considered an architecture vocabulary. It shows differences between approaches without evaluating them per se. As a result, it lets vendors present their solutions—including any architectural differences—more clearly, and gives users a better understanding of what the vendors offer.

Storage Virtualization was the first tutorial of the SNIA Education Committee. Development began in summer 2001 and continues to this day as we incorporate new material and address new developments in the storage virtualization segment. At the time of writing this booklet, the tutorial team consisted of about 35 members from different vendor companies as well as end users interested in this topic. These team members either monitor the activities and review the content, or actively develop the material to reflect new technologies and changes to existing technologies.

This booklet is based on both parts of the content presented at the 2003 Spring Storage Networking World. The two parts of the tutorial "Why, What, Where, and How?" and "Effective Use of Virtualization" address the basic technology used in virtualization and some of the practical applications of that technology. This booklet is in many ways a "scripted version" of the official SNIA

tutorial, but is able to expand on some of the concepts and provide more detailed explanations than we can offer in the one-hour tutorials.

Using the SNIA Shared Storage Model and the SNIA Storage Virtualization Taxonomy, the first part mainly deals with the different terms, types, locations, and technologies of storage virtualization and the data services that can be built on it. The second part of the booklet covers practical application of block virtualization and how to make the most effective use of it. It describes the implementation step by step and aspects of availability, performance, and capacity improvements. The material also discusses the role of storage virtualization within policy-based management and describes its integration in the SNIA Storage Management Initiative Specification (SMI-S).

Besides myself, the following members of the tutorial team have invested a lot of their time to support this booklet project:

Nik Simpson	Datacore Software
Robert Peglar	XIOtech Corporation
Gene Nagle	StoreAge Networking Technologies

All four authors work for companies that represent a different storage virtualization approach. This diverse group of authors ensures that the content of this booklet is well balanced and does not place too much emphasis on a particular approach to storage virtualization. Additionally, we have tried to include information about how we believe storage virtualization is developing and how it may affect readers' storage infrastructure in the future.

Feedback and comments to further improve the quality of the tutorial content and this booklet are always welcome and can be sent via e-mail directly to tut-virtualization@snia.org.

Frank Bunn
VERITAS Software Corporation
SNIA Tutorial Manager, Storage Virtualization
Krefeld, Germany
August 2003

About the Authors

Frank Bunn

Frank Bunn is Senior Product Marketing Manager at VERITAS Software for the EMEA Central Region and a member of SNIA Europe, where he is involved in a number of industry committees. Since starting his IT career in 1982 he has worked as a trainer, post- and presales system engineer, and consultant at Nixdorf Computer, Siemens-Nixdorf, and Network Systems Corporation. Before joining VERITAS Software in April 2001, he was a product manager at StorageTek, managing and introducing the SAN solutions product line to the Central European market.

Mr. Bunn is a frequent speaker at international IT events, author of technical articles and white papers, and co-author of the VERITAS published book *Virtual Storage Redefined.* He also manages the Storage Virtualization Tutorial project for the SNIA Education Committee.

Nik Simpson

As Product Marketing Manager for DataCore Software, a leader in the development of storage virtualization, Nik Simpson has provided competitive analysis, technical briefs, presentations, and white papers for DataCore partners and customers deploying networked storage pools "Powered By DataCore"™. Before joining DataCore, Mr. Simpson was Product Marketing Manager for Servers and Storage at Intergraph Computer Systems and spent time at Bell Labs as a resident visitor.

Mr. Simpson also writes on a variety of topics, including Windows NT and the development of Intel servers, and has contributed storage- and server-related articles to industry publications such as *InfoStor, Computer Technology Review,* and *Windows 2000 Magazine.* In addition, he is a member of the Council of Advisors providing storage industry analysis for the investment banking community.

Mr. Simpson holds a BSc. in Computer Science and has twenty years of experience in open systems, servers, and storage.

Robert Peglar

Rob Peglar is Chief Architect and Chief Market Technologist of XIOtech Corporation. A 26-year industry veteran, he has global corporate responsibility for technology, storage architecture, strategic direction, and industry liaison for XIOtech. He has

extensive experience in the architecture, design, and implementation of large heterogeneous storage area networks (SANs) and virtual storage architectures. He is a frequent speaker and panelist at leading health care, storage, and networking industry-related seminars and conferences worldwide. He has led XIOtech in architecture to its current industry-leading position in virtualized network storage. Prior to joining XIOtech in August 2000, Mr. Peglar held key field and engineering management positions at StorageTek, Network Systems, ETA Systems, and Control Data. Mr. Peglar holds a B.S. degree in Computer Science from Washington University, St. Louis, Missouri, and performed graduate work at Washington University's Sever Institute of Engineering.

Gene Nagle

As Director of Technical Services at StoreAge Networking Technologies, a manufacturer of storage area network management products, Gene Nagle manages all customer service and technical marketing activities for North America. Prior to his employment at StoreAge, Mr. Nagle served as a product line manager for Prism at Quantum|ATL, managing the full line of options for Quantum|ATL libraries, including Fibre Channel and Ethernet connectivity solutions and management features. In addition to those duties, Mr. Nagle represented Quantum Corporation at the Storage Networking Industry Association (SNIA) and worked with partner companies to promote new technologies and new standards for secondary storage.

Having served in the storage industry for 15 years, Mr. Nagle also spent 3 years at Overland Data as a product marketing manager for the LibraryXpress line. Additionally, he has held management positions with Cheyenne Software, Advanced Digital Information Corporation (ADIC), and GigaTrend.

Mr. Nagle holds an MBA in International Business.

Acknowledgments

The authors would like to acknowledge the diverse group of people who've worked to make the virtualization tutorial and this booklet possible. First, we'd like to say thanks to the other members of the SNIA Storage Virtualization tutorial team for their assistance in creating and reviewing the content of the tutorial on which this booklet is based. In particular, we'd like to single out John Logan and Ben Kuo, for delivering new material for the 2003 Storage Networking World event series, and Dave Thiel. Dave has been an active member and tutorial speaker from the beginning and now acts as track manager and referee for the Storage Networking World tutorial day.

Second, we'd like to thank the staff of ComputerWorld, and Nanette Jurgelewicz in particular, who've given us the chance to present the Storage Virtualization tutorial at multiple Storage Networking World conferences worldwide.

Another big thank you goes to the SNIA Education Committee for the sponsorship of projects like this. None of this would have happened without the active support and encouragement of former Education Committee Chair Paul Massiglia and the former tutorial coordinator Paula Skoe, who worked tirelessly to establish and manage the SNIA tutorial series.

We'd also like to thank everyone who attended our tutorials at Storage Networking World or at other storage-related events and provided feedback and suggestions for further improvements.

And finally, we'd like to extend our thanks to the production manager, Leslie Austin and her team, who transformed our raw manuscript into this first-class booklet.

Part I

Why, What, Where, and How?

1 Storage Virtualization—Definition

In 2001, the Storage Networking Industry Association (SNIA) set out to develop a consistent, industrywide definition of the term "virtualization." This was done in order to give consumers (end users) and producers (manufacturers) of storage common ground on which to discuss the topic. The SNIA definition is as follows:

> 1. The act of abstracting, hiding, or isolating the internal functions of a storage (sub)system or service from applications, host computers, or general network resources, for the purpose of enabling application and network-independent management of storage or data.
>
> 2. The application of virtualization to storage services or devices for the purpose of aggregating functions or devices, hiding complexity, or adding new capabilities to lower level storage resources.

Put succinctly, the definition may be termed as "abstraction of detail." This is the essence of virtualization. Virtualization provides a simple and consistent interface to complex functions. Indeed, there is little or no need to understand the underlying complexity itself. For example, when driving an automobile the driver doesn't need to understand the workings of the internal combustion engine; the accelerator pedal "virtualizes" that operation.

The SNIA Shared Storage Model

In an earlier effort, The Technical Council of the SNIA constructed an important document called the "Shared Storage Model." The purpose of this model was to educate end users by illustrating how the layering of technology in modern storage architectures creates a complete range of storage functions. The model is pictured below.

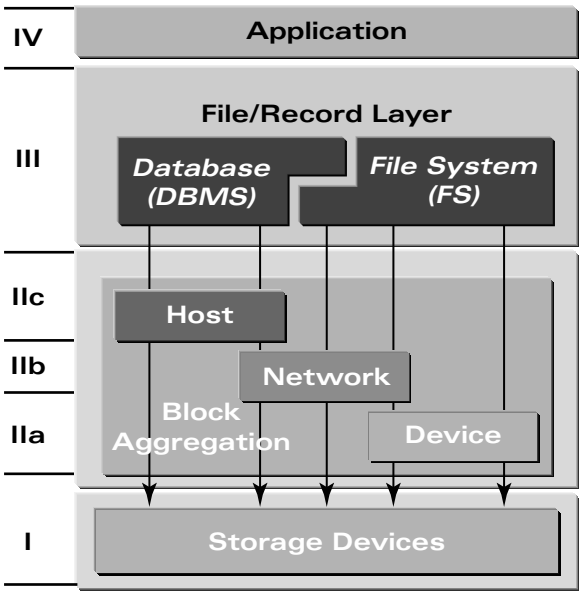


FIGURE 1–1 The SNIA Shared Storage Model

The model is divided into four main layers:

I. The storage devices themselves (e.g., a disk drive)
II. The block aggregation layer
III. The file/record layer
IV. The application layer

Each layer may present virtualized objects to the layer above it. For example, several disk drives (layer I) may be virtualized by layer II, appearing as a single device to layer III. Layer II is further subdivided into device, network, and host layers. In order to reflect the various locations that can perform the block aggregation function on layer II, we'll look at this topic in more detail later in the booklet.

Virtualization Differentiation

Today, several techniques are employed to virtualize different storage functions within the model. These include physical storage (layer I devices), RAID groups, logical unit numbers (LUNs), storage zones, LUN subdivision (a.k.a. "carving"), LUN masking and mapping, host bus adapters, logical volumes and volume management, file systems and database objects (such as a table space, rows, and

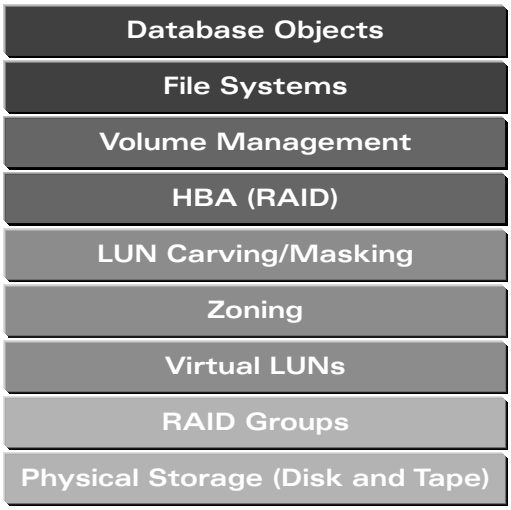


FIGURE 1–2 Typical Storage Functions Mapped to the SNIA Shared Storage Model

columns). The devices responsible for these virtualization functions include disk arrays and array controllers, storage switches and routers, discrete virtualization appliances, host bus adapters, operating systems, and application-layer software. This diverse group of functions and virtualization providers reflects a growing interest in storage virtualization as the key technology in solving common storage management problems.

2 Storage Virtualization—Why?

Problem Areas for Virtualization Solutions

Storage virtualization as a technology is only useful if it offers new ways to solve problems. The most severe problem that virtualization can help solve today is the overall management of storage. Today, storage infrastructure represents one of the most heterogeneous environments found in modern IT departments, with a multitude of different systems at all levels of the stack—file systems, operating systems, servers, storage systems, management consoles, management software, etc. This complexity has become a hindrance to achieving business goals such as 100% uptime. The problems of data access impact several areas that can be improved by virtualization:

- Single points of failure within the SAN, whether in a single array, a switch, or some other component of the path from application to storage, are always a problem and without virtualization, eliminating them can be prohibitively expensive.
- One of the key metrics for judging quality of service (QoS) in the SAN is performance. Today, maintaining deterministic storage performance for applications is a very complex task. Virtualization offers the opportunity to improve performance, but more importantly it offers the chance to manage performance in real time to maintain guaranteed and measurable QoS.
- The data stored in the SAN is critical to the day-to-day operation of a company. Losing access to it, even for a couple of days, can be disastrous. Virtualization is a key enabling technology in making more affordable options for disaster recovery and data archiving available to customers that can't afford today's high-end solutions.

In addition, the ever-increasing demand for "raw" storage (i.e., level I devices that supply capacity) is partly driven by poor utilization. Poor utilization inevitably leads to unnecessary expenditures in both hardware and management costs.

Industry surveys indicate that capacity utilization in open systems environments is at 30–50% for disk devices and 20–40% for tape. This means that companies are on average buying 2–3 gigabytes (GB) of disk capacity for every gigabyte of stored data. Clearly, these rates are not viable for any ongoing business. The root cause of poor utilization is the way in which storage is allocated and bound to hosts. Because the process is complex and prone to error, storage administrators tend to allocate large chunks of capacity simply to minimize the number of times they have to repeat the procedure during the operational life of the host. Storage virtualization offers new ways to address this problem through automation, just-in-time provisioning, and tiered storage architectures in which different quality levels of storage (i.e., low-cost storage for noncritical data) are allocated to meet specific requirements using predefined and automated policies.

Traditional Architectures versus Virtualized Architectures

In traditional storage architectures, all storage is managed and deployed as discrete physical devices (layer I in Figure 1–1). Very little, if any, functionality is performed at layer II to abstract detail. Paths, connections, access and configuration, provisioning, and all other aspects of storage must be individually addressed and managed. Clearly, this is not a model that scales, and it cannot be controlled in an orderly fashion.

The benefits of moving toward a virtualized architecture are numerous. Many businesses today are realizing a significant reduction in downtime through virtualization, because of reduced management complexity. Simply stated, it is easier to manage virtual resources than to manage physical resources. There is less vendor-specific detail and thus more scope for automation. In addition, utilization of physical storage, scalability, and flexibility are greatly enhanced. By virtualizing the components and storage services, we get much closer to the "nirvana" of automated storage management in which the common and tedious tasks that burden storage administrators are handled automatically. Virtualization is also the key to providing manageable and predictable quality of service in the SAN.

Attempting to solve these problems without virtualization requires understanding unique and vendor-specific characteristics of each component within the stack—an overwhelming task for any policy or service engine trying to coordinate all business activity. Virtualization provides the necessary abstraction of detail so that the "universe" of managed entities appears ordered, simple, and uniform.

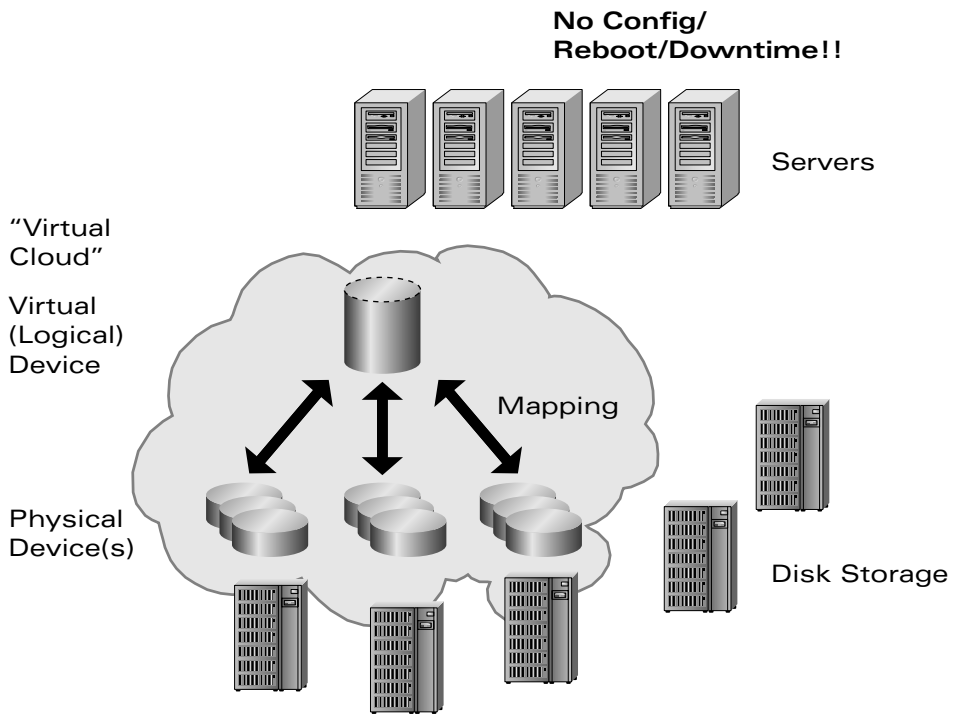


FIGURE 2–1 A Virtual Storage Connection

A Basic Virtual Storage Connection Model

A very basic model of virtual storage may be depicted by simply changing the physical devices (layer I) into virtual devices (layer II). The servers accessing these virtual devices do not "know" the difference—the interface to the virtual devices is identical to the interface to a physical device. Operations to the layer I devices such as adding, changing, or replacing components may be performed without disrupting the layers above.

In summary, there are three main reasons why customers should consider using storage virtualization:

- Improved storage management in heterogeneous IT environments
- Improved availability, and elimination of downtime with automated management
- Improved storage utilization

3 Storage Virtualization—What?

The SNIA Storage Virtualization Taxonomy

The SNIA storage virtualization taxonomy describes five different types of storage virtualization: block, disk, tape (media, drive, and library), file system, and file virtualization (see Figure 3–1 on the following page). This booklet deals primarily with block virtualization, but for the sake of completeness, it also provides a brief overview of the other virtualization technologies.

Disk (Drive) Virtualization

Disk—or, to be more precise disk drive—virtualization is one of the oldest forms of storage virtualization and has been implemented for decades in disk drive firmware.

At the lowest level, a location on a magnetic disk is defined in terms of cylinders, heads, and sectors (CHS), but each disk is different in terms of the numbers of cylinders, etc. (That's why capacity varies.) Clearly, a form of addressing that is different for each disk is completely unsuited to today's operating systems and applications, since they would have to know the exact physical property of every single magnetic disk—an impossibility given the wide range of hardware and the fast pace of developments in this sector.

Instead, the physical properties of the disk are virtualized by the disk firmware. This firmware transforms the CHS addresses into consecutively numbered logical blocks for use by operating systems and host applications. This is known as logical block addressing (LBA), and it has revolutionized the way host computers deal with disks. Now, the size of the disk is determined simply by the number of logical blocks; for example, a 36-GB disk has twice as many logical blocks as an 18-GB disk.

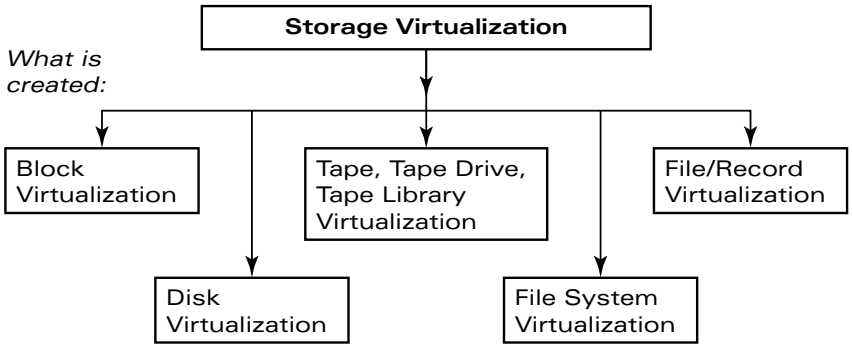


FIGURE 3–1 Types of Virtualization in the SNIA Storage Virtualization Taxonomy

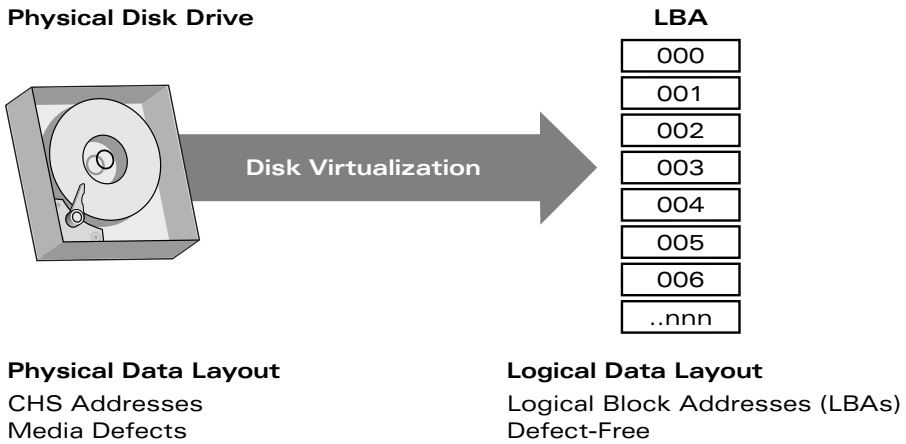


FIGURE 3–2 Disk (Drive) Virtualization

Disk virtualization also ensures that the magnetic disks always appear defect-free. During the life of a disk, some of the blocks may go "bad"—i.e., they can no longer reliably store and retrieve data. When this happens, the disk firmware remaps those defective blocks to a pool of spare defect-free blocks. This relieves the operating system of the task of keeping track of bad blocks and ensures that the host just sees a well-behaved, defect-free device.

Tape Storage Virtualization

Tape storage virtualization is utilized by several tape library components, and falls into two basic areas: virtualization of tape media (cartridges) and virtualization of the tape drives.

Tape Media Virtualization

Tape media virtualization uses online disk storage as a cache to emulate the reading and writing of data to and from physical tape media.

Using disk storage to virtualize tape in this way improves both backup performance and the service life of the tape drives. Performance is improved because the disk acts as a buffer to smooth out the fluctuations caused by the network or excessively busy hosts. These fluctuations in the data stream may cause the tape drive to switch from streaming mode to start/stop mode (also known as "shoeshining") in which the tape drive must write a small amount of data, stop tape media motion, and rewind (backup) before writing the next small parcel of data. This back-and-forth motion causes sharp drops in recording speed and increased wear and tear on the tape media and tape drive recording heads.

One of the overall goals of virtualization is improved utilization of storage capacity. Many mainframe users suffer from the fact that, in many cases, only 15–30% of their overall tape media capacity may be used due to variances in specific applications and access methods. Tape media virtualization improves the level of capacity utilization by emulating the existence of large numbers of small tape media volumes as expected by some operating environments, while actually accumulating the data on disk. The data on disk is then written out to tape at streaming speeds, filling each volume of media to its optimum level, as it allows improved "compression" of files in the disk cache before storing them on the tape medium. In addition, this form of virtualization improves restore performance and mean time to tape data by avoiding the time-consuming mounting and unmounting of tapes. The mount and unmount operations are emulated via intelligence at the disk cache layer, returning status to the requestor just as if a physical tape medium had been physically mounted or unmounted to or from a physical tape drive.

Tape Drive Virtualization

Open systems users generally do not have the tape media utilization problems described above. Their biggest challenge lies in attempting to share physical tape drives in tape libraries among as many host systems as possible, saving significant hardware resources. Prior to SANs, tape drives were directly attached to individual servers. But SAN technologies like Fibre Channel have allowed tape drives to

be shared by multiple servers, by controlling access to the drives through the network instead of through an individual server. When tape drives are networked, however, the problem lies in how to ensure that different applications or servers don't clash when accessing the tape drive and corrupt each other's data. Tape virtualization in this context makes it possible to establish tape drive pools with guaranteed data integrity. A single physical drive may appear as several virtual drives, which can be assigned to individual servers that treat them as dedicated resources. When a server attempts to use its virtual tape drive, a request is sent to the tape drive broker that reserves and maps a physical tape drive to the host's virtual drive. When the operation is complete, the media broker returns the physical tape drive back to the tape drive pool so that it can be assigned to another host. The virtualization intelligence ensures controlled access to the drives and prevents conflicts between applications and servers as the data is written to or read from tape media.

Defective tape drives may be nondisruptively replaced by other physical drives from the pool, thanks to virtualization techniques ensuring that backups are completed without being impacted by hardware failures. All these functions run in the background, and storage consumers are unaware of them.

Another form of tape drive virtualization is closely related to the RAID technology used with disk storage. Often referred to as RAIT, or Redundant Array of Independent Tapes, the concepts are very similar to those delivered in RAID controllers. Several tape drives are grouped into a logical unit; the benefits are similar to those associated with RAID, such as improved performance and greater reliability. Going one step further, entire tape libraries may be virtualized and distributed by the technique known as RAIL, or Redundant Array of Independent Libraries. In this method, entire physical tape libraries are emulated to present the applications and servers of a (set of) virtual library resources that are allocated and controlled as if a single library were reserved by the application or server.

File System Virtualization

The simplest form of file system virtualization is the concept of a networked remote file system (often referred to as network attached storage, or NAS) such as NFS or CIFS. In this form of virtualization, dedicated file servers manage shared network access to files in the file system. That file system is shared by many hosts on the network that may run different operating systems. For example, Windows and Unix hosts can access the same file system through an NFS share. Regardless of operating system, the files on the shared file system are accessed through the same high-level mechanisms used to access local file systems. This means that applications and users can access files using the same interfaces regardless of the physical location of the file. This abstraction, or "hiding," of data location is an excellent example of one of the important features of storage virtualization: location transparency.

Another special form of file system virtualization is used to simplify database management. Database table spaces and transaction logs are still often located on raw disk devices in an attempt to maximize performance. Other administrators prefer to implement these entities on a file system to improve management, as raw disk devices are notoriously difficult to administer. However, the additional overhead of the file system degrades performance. File system virtualization in database environments combines the advantages of raw partitions with those of file systems. The file system is visible to the administrator and permits optimal management of the database entities. From the point of view of the database itself, however, its entities are physically located on raw disk devices; the file system remains hidden and therefore its buffered I/O is bypassed, ensuring maximum throughput.

File/Record Virtualization

The most widely deployed example of file virtualization is Hierarchical Storage Management (HSM), which automates the migration of rarely used data to inexpensive secondary storage media such as optical disks or tape drives, or low-cost high-density disk storage such as Serial ATA (SATA) arrays. This migration is transparent to both users and applications, which continue to access the data as though it was still on the primary storage medium. Here again, virtualization results in location transparency. A pointer in the file system combined with metadata from the HSM application ensures that the migrated file can be rapidly retrieved and made available to the requestor, without the requestor having to know the exact physical location of the file.

Block Virtualization

Most of the new work on storage virtualization in recent years has focused on layer II of the SNIA shared storage model, which deals with block-level disk services, and it is this *block virtualization* that most vendors are referring to when the subject of storage virtualization comes up.

Block virtualization is the next logical step beyond the simple CHS-LBA disk virtualization described earlier. Where disk virtualization "manipulates" a single magnetic disk and represents it as logical block addresses, block virtualization goes a step further by virtualizing several physical disks to present a single logical device. This is often referred to as *block aggregation*.

The idea behind block virtualization is simple: Overcome the physical limits of individual devices without requiring any additional intelligence in applications, so that the latter just see a "bigger" disk (a new virtual disk with a larger logical block address range). However, simple block aggregation is just one facet of block virtualization; other services can also be introduced into the block virtualization

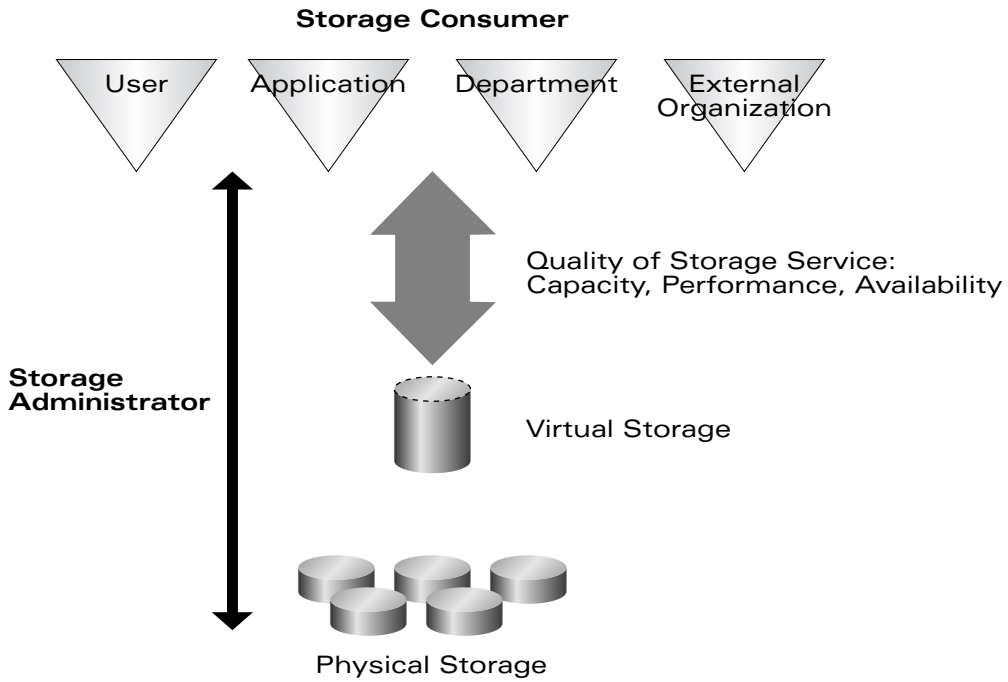


FIGURE 3–3 The Logical View of Storage

layer to deal with performance, availability, and other important storage attributes. In a perfect world, all storage management tasks would be dealt with from the perspective of the storage consumer (the application), not the storage provider (the array). The storage consumer has relatively simple high-level requirements:

1. Capacity: Is there enough space to store all the data generated by the application—is it big enough?
2. Performance: Is the storage provided to the application able to meet the response time requirements of the application—is it fast enough?
3. Availability: Is the storage provided to the application able to meet the availability requirements of the application—is it reliable enough?

The physical aspects of storage, such as disk size and the number of arrays, are irrelevant; storage consumers don't want to deal with technical details, they just want to define the storage services they need.

Storage administrators have the task of meeting the requirements of storage consumers. To do this efficiently in a dynamic, heterogeneous environment, they need powerful tools that offer flexibility and scalability. Storage administrators

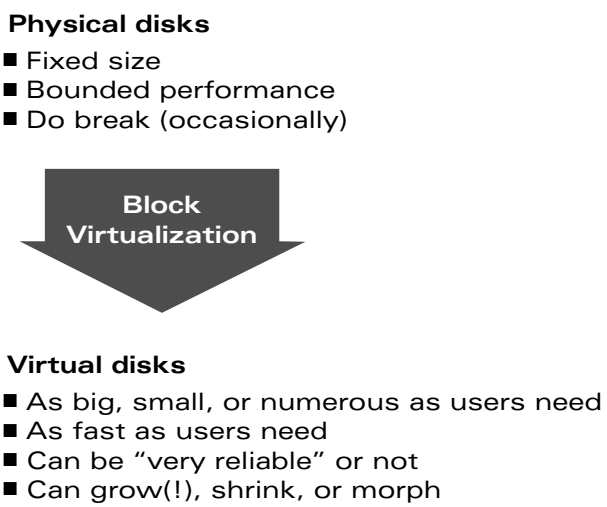


FIGURE 3–4 Virtualization Makes "Devices" from Devices

want to use the same tools to help them manage as many different servers and storage systems as possible; their goal is simplified management. Storage virtualization is a critical tool in the storage administrator's toolbox that can help the administrator meet these goals.

The goal of block virtualization is to control physical storage assets and combine them to provide logical volumes that have sufficient capacity, performance, and reliability to meet the needs of storage consumers without burdening the consumers with unnecessary low-level detail. Instead of consumers being provided with a difficult-to-manage set of physical storage devices, consumers simply see one or more logical volumes that are indistinguishable from conventional physical disks. The consumers don't realize that logical or virtual units are involved; they just see capacity that meets the needs of the application. The virtualization layer is responsible for mapping I/O requests to the logical volume onto the underlying physical storage. Put simply, block virtualization creates virtual storage devices from physical disks, which are as large, fast and available (resilient) as storage consumers require.

The capacity, performance, and reliability requirements of storage consumers are met by combining one or more of the different capabilities of block virtualization:

- If storage consumers need additional disk *capacity,* additional volumes are generated or existing logical volumes are enlarged. Other free physical disk resources are added in the background without affecting access to data. Of

course, the reverse is also possible. Several smaller logical volumes can be created from a single large physical disk; this is known as "slicing" and can be very useful where an underlying array may have limits in terms of the number of LUNs it can present.

- If storage consumers need greater *performance,* then the simplest approach is to stripe the data across multiple disks, or even multiple arrays. This achieves an increase in throughput approximately proportional to the number of physical drives that the data is striped across. As we'll see later in this document, striping is not the only option offered by storage virtualization for improving I/O performance.
- If storage consumers need enhanced *availability,* there are a number of options, including clustering, RAID, synchronous mirroring to another array(s), and/or asynchronous data replication over long distances. All of these functions can be implemented as functions in the block virtualization layer.

  With multiple copies of data in different locations, configuration changes can take place online. Storage consumers continue to work without interruption because their data is still available even while an entire storage array or cluster of arrays is replaced.

4 Storage Virtualization—Where?

Block-level virtualization isn't new. In fact, most systems in production today are utilizing some form of block-level virtualization in which a logical block address from the host is mapped in some way to physical storage assets. The major change is that block-level virtualization is increasingly seen as a key technology for the implementation of new services and the reimplementation of some older but previously proprietary services in the storage network.

To understand the reasoning behind various implementations of storage virtualization, it's worth stepping back for a moment to look at how an I/O request passes through the execution chain from application to storage.

1. An application makes a read or write request to the operating system.
2. The request goes either through a file system or directly to a disk (usually managed by a database). At this point, the I/O request has been transformed into a logical block address(es).
3. The next task is to convert the logical address into a real physical disk address (i.e., a CHS.) This transformation can take place in the host, somewhere in the network or at the storage. It's actually quite likely that some part of the conversion takes place in all three places. For example, on the host a volume manager may be involved, while a RAID controller in the storage device does additional transformation, before the final LBA-CHS translation takes place in the firmware of the disk.
4. After the address transformation is complete, an address on a particular disk is accessed and the results passed back up the chain.

It's convenient to think of the storage infrastructure as a stack of technologies with each layer in the stack responsible for some part of the transformation of the I/O request. Vendors have responded by positioning their virtualization solutions at different layers in the stack, so that the market operates with three approaches:

- Host-based virtualization
- Storage-based virtualization
- Network-based virtualization

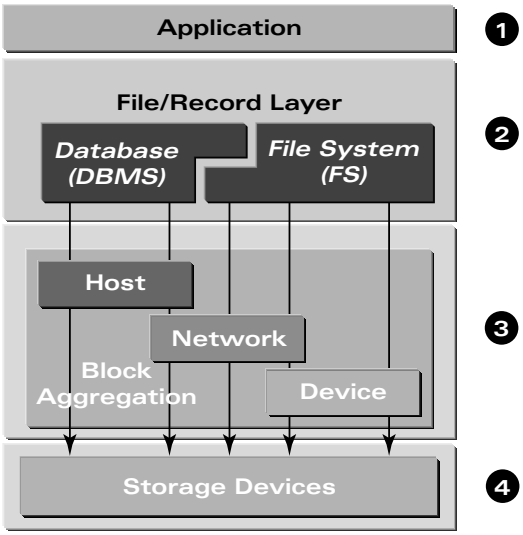


FIGURE 4–1 I/O Request Execution from Application to Storage

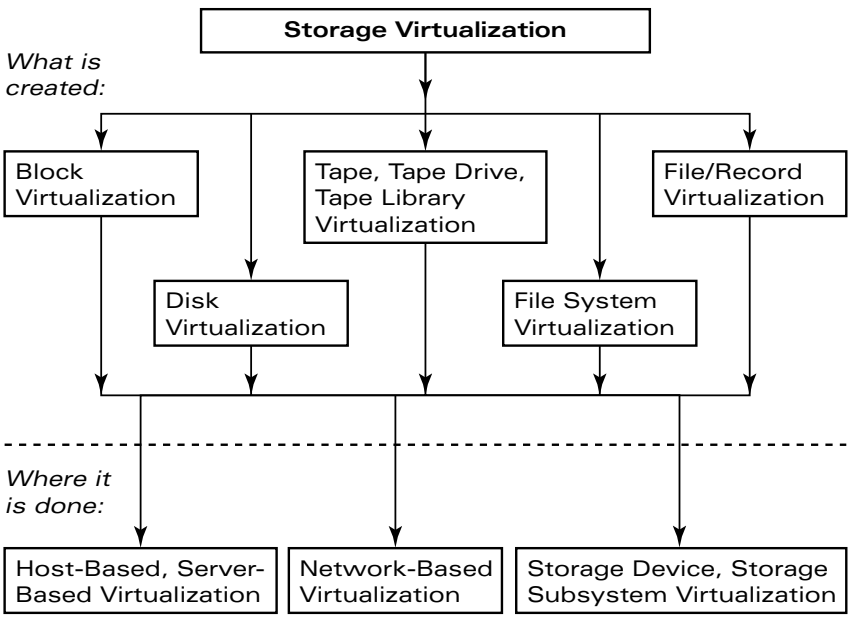


FIGURE 4–2 Three Locations of Storage Virtualization

Host-Based Virtualization

This type of virtualization is generally associated with logical volume managers, which, with varying levels of sophistication, are commonly found on just about every computer from the desktop to the data center. As is the case with storage-based virtualization, logical volume managers are not necessarily associated with SANs. Yet they are still the most popular method of virtualization because of their history and the fact that direct attached storage (DAS) is still very widespread. Increasingly, the logical volume manager (LVM) is a standard part of the operating system, but more advanced third-party implementations are also quite common. The most common uses of host-based LVMs are:

1. Aggregating physical storage from multiple LUNs to form a single "super-LUN" that the host OS sees as a single disk drive
2. Implementing software RAID and other more advanced functions, including snapshots and remote replication
3. Managing the health of disk resources that are under the control of the operating system

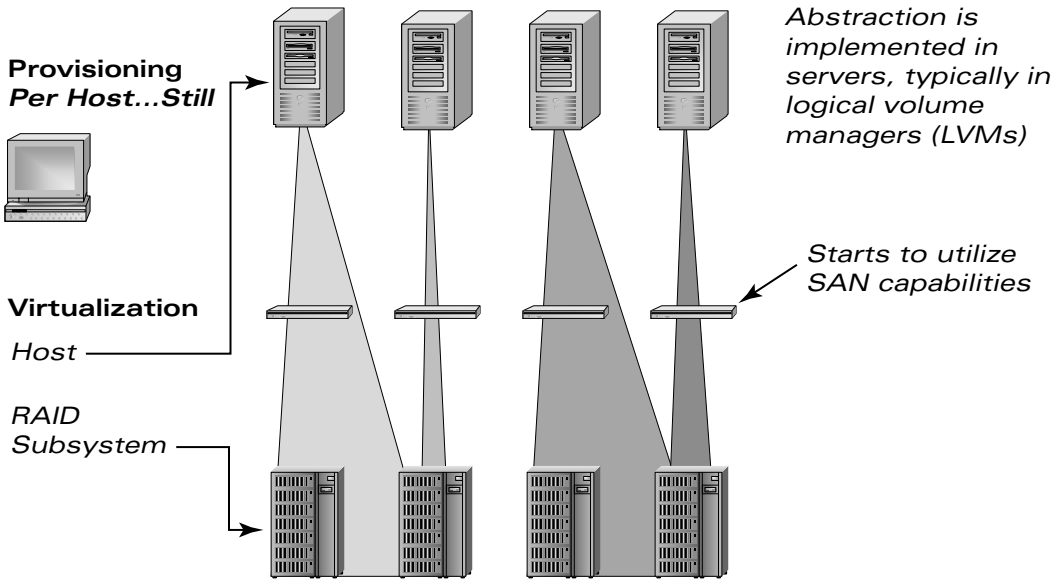


FIGURE 4–3 Host-Based Virtualization

The great advantages of host-based virtualization are its stability after years of use in practice, and its openness to heterogeneous storage systems. The proximity to the file system, which is also on the host, makes it possible to join these two components tightly for efficient capacity management. Many LVMs allow volumes and the file systems on them to be enlarged or reduced without having to stop applications.

The downside of a host-based approach is that the LVM is server-centric, which means that the storage provisioning must be performed on each host, making it a labor-intensive task in a large, complex environment. Therefore, some vendors are offering so-called cluster volume managers in a homogeneous server environment to ease storage management by allowing multiple servers that share access to common volumes to be managed as one.

Storage-Based (Subsystem-Based) Virtualization

Even though you may not have known it, your storage arrays may have been performing storage virtualization for years. Features including RAID, snapshots, LUN masking, and mapping are all examples of block-level storage virtualization. Storage-based virtualization techniques are equally applicable in SAN and DAS environments.

Storage-based virtualization is typically not dependent on a specific type of host, allowing the array to support heterogeneous hosts without worrying about variance of host operating systems or applications. Also, storage-based RAID systems deliver optimum performance in relation to their hardware because features like caching can be tuned to the specific hardware. The downside to this approach is that the storage virtualization functions are typically confined to a single array; for example, the source volume used for a snapshot and the snapshot itself are maintained on the same array, making the snapshot useless in case of hardware failure. In some cases, virtualization functions extend across multiple arrays, or a cluster of arrays or controllers; however, these solutions are typically restricted to a single-vendor implementation.

Frequently, host- and storage-based virtualization is combined, adding the flexibility of the host-based LVMs to the performance of hardware-assisted RAID. For example, the host-based LVM can use multiple RAID-5 LUNs to create virtual volumes spanning multiple disk arrays. In addition to simply striping across LUNs from multiple arrays, LVMs in the host can also mirror (with striped mirrors or mirrored stripes) volumes across multiple arrays. Host-based LVMs are also used to provide alternate path fail-over in most environments because the host is the only device in the I/O chain that knows for certain whether its I/O completed. Similarly, LVMs may also implement load balancing on the access paths to the storage to increase performance.

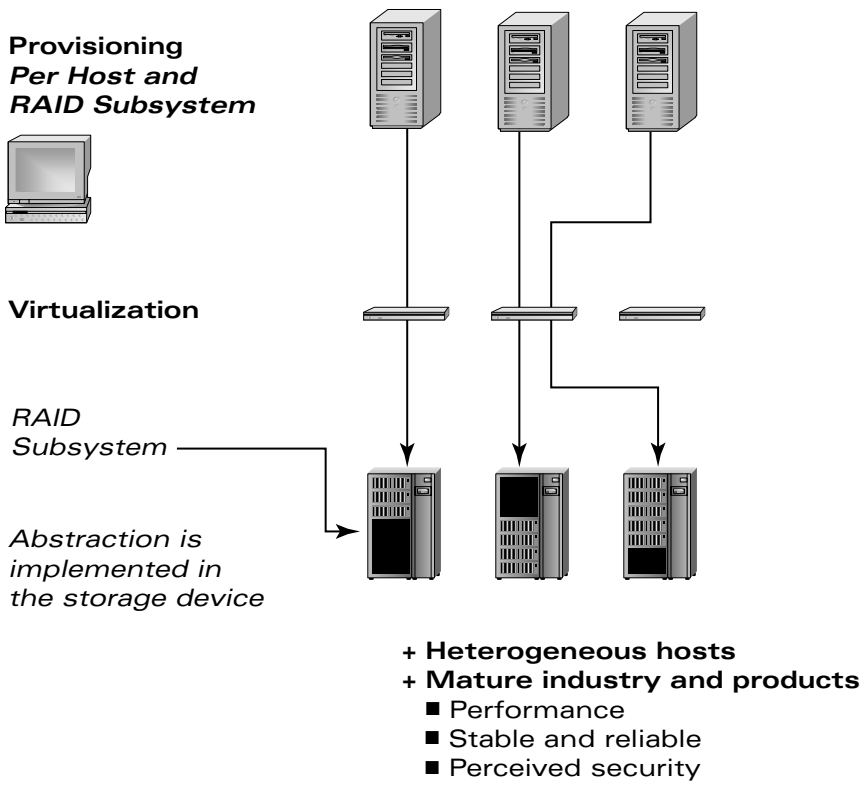


FIGURE 4–4 Storage-Based Virtualization

Network-Based Virtualization

The advantages of host- and storage-based virtualization can be combined into a storage management layer existing within the SAN fabric. This network-based virtualization represents the latest development in the field. Network-based virtualization has the potential to provide the foundation for the automated storage management needed to contain and manage the explosive growth in storage capacity.

A network-based approach to virtualization supports data center-wide storage management and is able to accommodate a truly heterogeneous SAN with a diverse range of host platforms and storage resources. Typically, network-based virtualization is implemented using "black-box" appliances in the SAN fabric and potentially some agent software installed on the host (the need for and the extent

of those host-based agents depends on how virtualization is implemented in the appliance). The appliance itself can be anything from an off-the-shelf server platform to a dedicated, proprietary hardware design. Typical functions offered by network-based virtualization include:

- Combining several LUNs from one or more arrays into a single LUN before presenting it to a host
- Taking a single LUN from an array and slicing it into smaller virtual LUNs to present to the hosts
- Synchronous and asynchronous replication within the SAN as well as over WAN links
- Device security to ensure that access to a LUN is restricted to specified hosts

Other functions include caching, advanced volume management, storage on demand, and QoS functions, but the availability of these more advanced features varies from one vendor to another.

SAN appliances are available on the market as proprietary solutions, but more commonly as standard Windows, Unix, and Linux servers with corresponding virtualization software. All of the vendors with products available today recognize the importance of avoiding single points of failure within the SAN and offer solutions that provide redundancy and failover.

More recently, switch vendors have announced intelligent switches with embedded virtualization intelligence. In many ways, these virtualizing switches fit within the basic concept of a SAN appliance; the main difference is that the intelligence is part of the switch rather than being outside of the switch in one or more dedicated appliances. At the time of writing this booklet, none of these products is generally available or likely to be before late 2003. A more detailed description of intelligent switches will be provided at the end of the next chapter.

5 Storage Virtualization—How?

SAN appliances can be integrated into the storage infrastructure in two different ways:

1. In-band virtualization: The appliance(s) is directly in the data path (in-band) between the servers and storage devices. With an in-band approach, both I/O and command and control metadata pass through the appliance(s).
2. Out-of-band (OOB): The appliance(s) only sees command and control metadata; the actual I/O goes directly to the storage.

In reality, virtualization implemented in the host or storage system is also a form of in-band virtualization as these components are located at the end points of the data path. Additionally, even with OOB approaches, some element must be in-band because it's necessary to see the I/O. For example, if remote replication is a function of the OOB virtualization scheme, there must be some component that sees every write operation in order to replicate it.

Some early industry sources preferred the terms *symmetric* and *asymmetric* instead of *in-band* and *out-of-band.* SNIA is trying to standardize terminology in order to improve understanding, and therefore recommends the exclusive use of the terms *in-band* and *out-of-band virtualization.*

In-Band Virtualization

With an in-band appliance(s) located in the data path between hosts and storage, all control information (metadata) and data pass through it. Another term used to describe this process is store and forward. To the host, the appliance looks and behaves like a storage array (e.g., an I/O target) that presents logical volumes. To the storage, the appliance looks and behaves like a host, issuing read and write requests (e.g., an I/O initiator) that are indistinguishable from I/O requests generated by conventionally attached hosts.

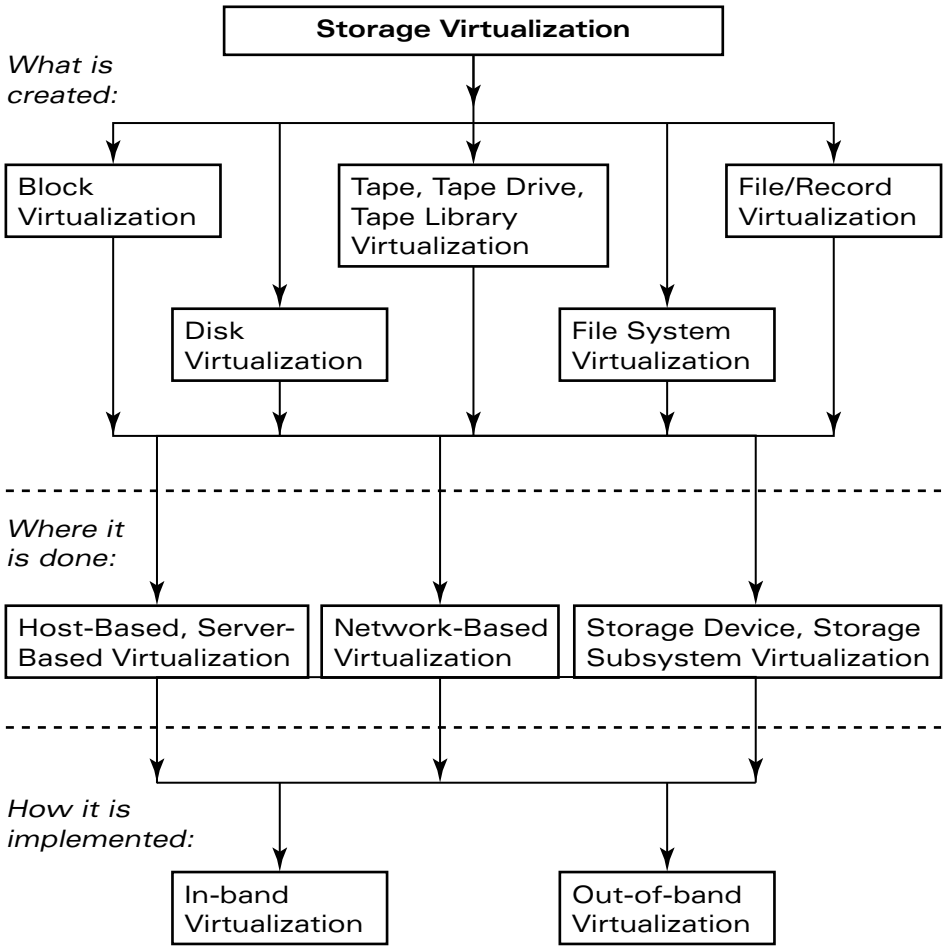


FIGURE 5–1 Two Implementation Methods: In-Band and Out-of-Band

Because the in-band appliance has complete control over access to storage resources (typically the SAN would be zoned so that hosts only see the appliance(s)), there is a high degree of security when volumes are accessed. Effectively, the appliance acts as a storage firewall: Because the appliance is the device that responds to hosts when they ask for a list of available LUNs (e.g., a Fibre LIP or fabric log-in request), hosts don't even see resources that are not specifically assigned to them. As a result, the problem of unauthorized or unintentional access to LUNs is minimized, if not eliminated altogether.

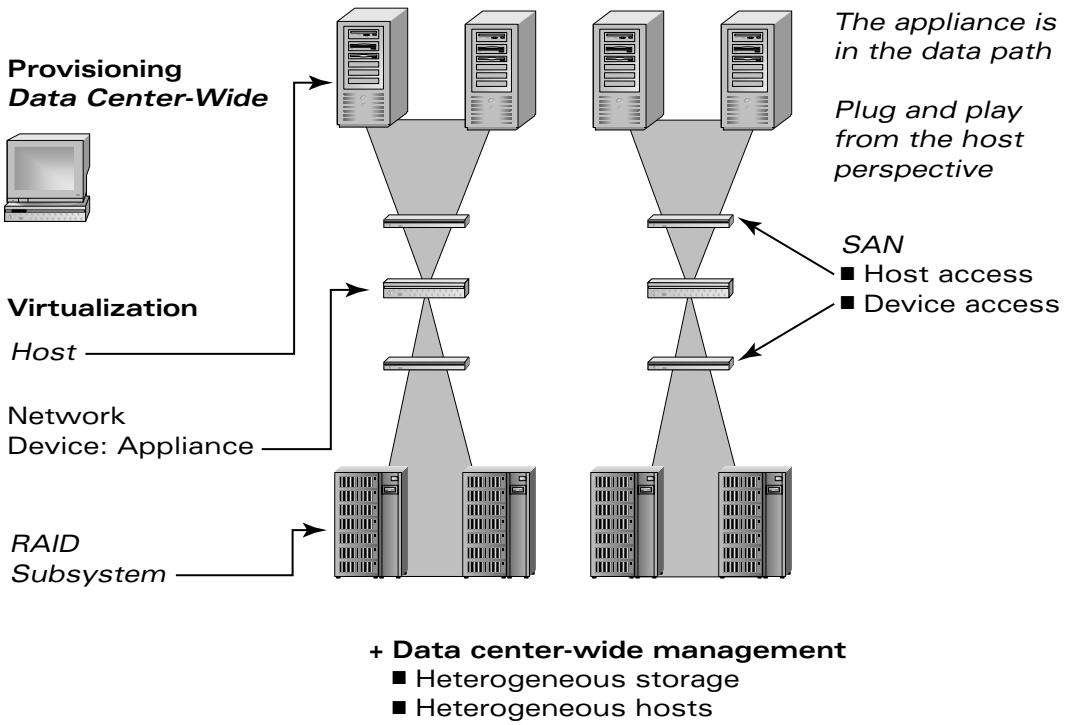


FIGURE 5–2 In-Band Network-Based Virtualization

Because the in-band virtualized LUN is presented to the host as a standard device and discovered through standard mechanisms, no special drivers on the host are required, which simplifies implementation and management of the SAN. Because no host-based agent is required, there are few limits on supported host operating systems; as a general rule, the in-band approach supports any host that can attach to the SAN. (Note that host agents are still used for fail-over, as this is the only possible location for this function.)

Because the in-band appliance acts as an I/O target to the hosts and as an I/O initiator to the storage, it has the opportunity to work as a bridge. For example, it can present an iSCSI target to the host while using standard FC-attached arrays for the storage. This capability enables end users to preserve investment in storage resources while benefiting from the latest host connection mechanisms like iSCSI and InfiniBand. Although an in-band appliance is another element in the data path, there is no automatic performance penalty if the in-band appliance offers caching. With caching, in-band approaches can actually improve overall I/O

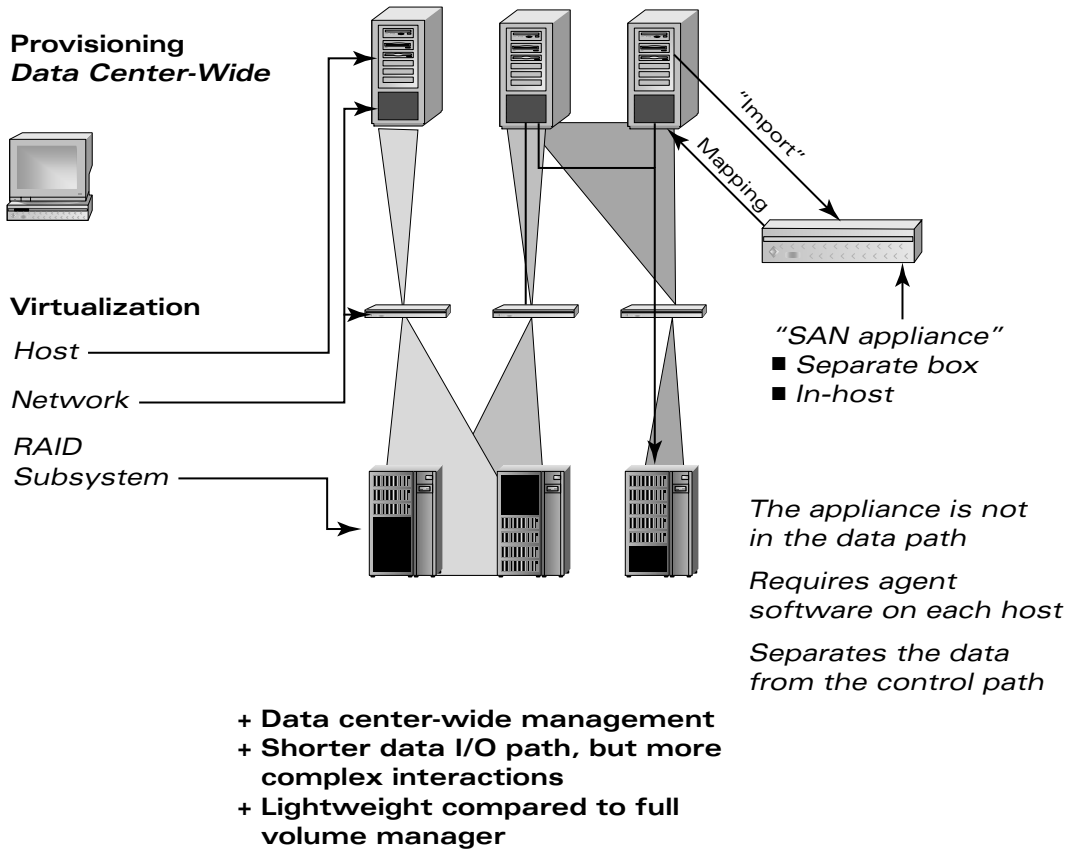


FIGURE 5–3 Out-of-Band Network-Based Virtualization

performance by offering faster cache response times and offloading work from arrays. Depending on throughput demands, there is always some sort of limit to the number of hosts that can effectively use the services of each in-band appliance, so multiple appliances may be called for.

Out-of-Band Virtualization

As the name suggests, an out-of-band appliance sits outside the data path between host and storage, and like in-band network-based virtualization, this approach is now available on the market. The OOB appliance communicates with the host systems via Ethernet or Fibre Channel in order to resolve requests for LUN access, volume configuration, etc. Because hosts must contact the appliance in order to gain

access to storage, they must be aware that storage is being virtualized, unlike in-band approaches in which the virtualization function is transparent to the hosts. As a result, OOB virtualization architectures require a virtualization client or agent in the form of software or special HBA drivers installed on each host. This agent receives information on the structure and properties of the logical volumes as well as the corresponding logical/physical block mapping information from the appliance. Thus, enforcement of storage security is performed at the agent. Additionally, the agent will be involved in replication and snapshot operations since the OOB appliance, which doesn't see I/O, can't directly perform these functions.

The appliance is responsible for storage pool and volume configuration and control information. The host uses this information to address the physical blocks of the storage systems on the SAN. The OOB approach has proven somewhat more complicated to implement than in-band solutions because of the need to deliver agent support for a diverse group of operating system platforms. Any OS without the agent will not be able to take part in the virtualized SAN. On the other hand, as the data flow doesn't pass through the out-of-band appliance(s), the technical requirements regarding hardware platform, HBAs, CPU, memory, and cache obviously are much lower than those of an in-band appliance. Increasing "virtualized" data flow in the SAN environment doesn't have a direct impact on the out-of-band appliance. This reason, and the fact that additional Fibre Channel cabling effort is avoided, can lead to good scalability in large enterprise SANs.

Because of the low system requirements, the implementation of this appliance type can be done theoretically as a software-only function within a normal clustered application server in the SAN (on-host implementation), avoiding the need for extra appliance hardware.

Switch-Based Virtualization—In-Band or Out-of-Band?

Although full details of the various switch-based implementations have yet to emerge, they appear to represent a hybrid approach in which some functions are effectively in-band and some are out-of-band. In essence, switch-based virtualization will behave similarly to in-band virtualization appliances in that it will not require agents on the host.

At first glance, switch-based approaches appear to be inherently in-band, as the switch as a whole is located in the data path. But a closer look inside the "smart switch" reveals a more hybrid function, with some characteristics of an out-of-band approach combined with some characteristics of an in-band approach. A management blade inside the switch (or an external OOB appliance) acts as the "out-of-band meta-controller." That controller is responsible for device discovery, volume configuration, and I/O error handling. For most operations this component is therefore not in the data path; only configuration and I/O error management data will pass through the meta-controller. The meta-controller works in

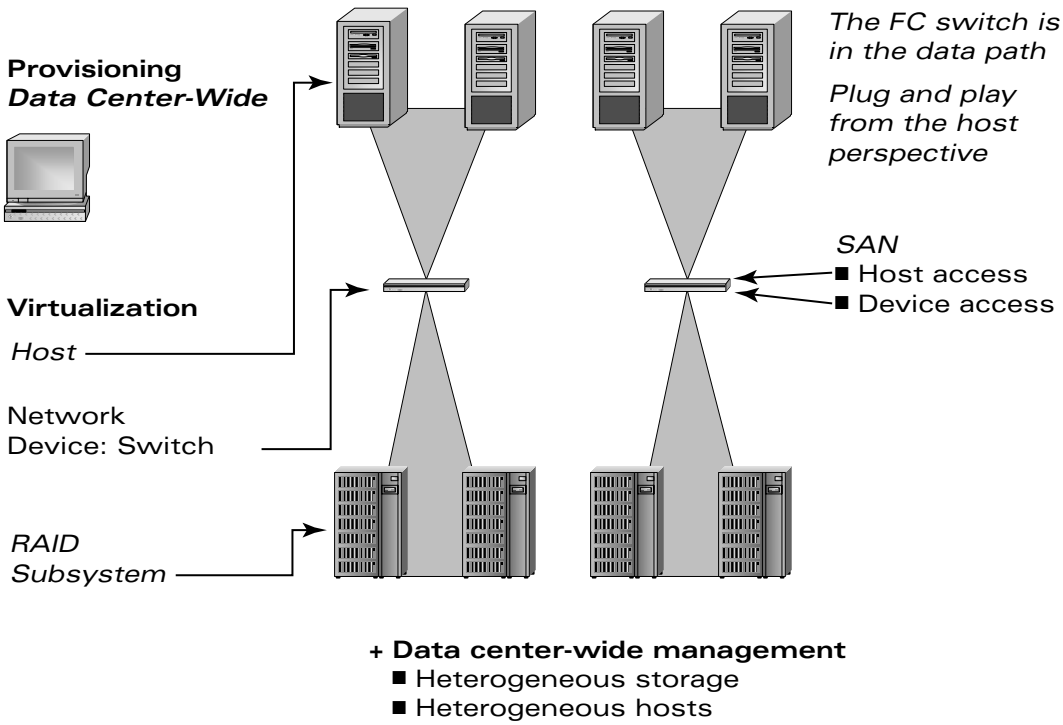


FIGURE 5–4 Switch-Based Virtualization

conjunction with intelligent ports that perform the in-band operations like replication. These act as the virtualization clients: Once they have received the volume information from the blade controller, they work mainly independently, providing the virtual/physical I/O translation and forwarding the data to the correct targets.

There are two approaches that can be implemented in the switch:

1. Command termination and redirection: In this method (also used by in-band appliances), the switch acts as an I/O target for hosts and as an I/O initiator for storage. Each request from the host terminates at the switch, where it is transformed and reissued as a new request to the storage. Once the storage responds, the switch responds back to the requesting host.
2. Packet cracking: In this approach, the switch is semi-transparent and I/O requests are sent from host to storage, but when they pass through the switch, the packet headers and payload are examined and desired transformations applied. For example, if replication is required for a particular LUN, the switch would take the payload from a write and send a copy of it out of another switch port for replication.

TABLE 5–1 Comparison between In-Band Appliances and Intelligent Switches

Comparison	Appliance Based	Switch Based
Multivendor fabric	Independent functionality	Interoperability mode
Switching	Separate*	Integrated
Performance	Read and write caching	No store and forward
Functionality	Rich feature set possible	Cost and footprint limits
Availability	Fail-over mechanisms	Fabric topology
Connectivity	Usually HBA/NIC ports	High-density switch ports
Scalability	Implementation specific	Implementation specific
Storage ROI	Leverage legacy storage	SAN-attached storage
Maturity	Stable products in 2002	First generation in 2003

*Some in-band appliances can also perform the switching function.

To date, most of the information about early intelligent switches suggests that the "packet cracking" approach is more popular.

Because in-band appliance-based and switch-based approaches are somewhat similar, it's worth looking at the typical feature sets side by side (see table above).

6 Enhanced Storage and Data Services

Regardless of the location of virtualization functions, storage virtualization overall forms a base on which to build several important data services. Services such as volume management, clustering, LUN control, snapshots and data replication are significantly less complex when virtualization is present than when it is not.

For example, virtualization makes clustering much easier. When an application fails over to another server, the associated volumes are logically transferred (imported) to another server. The cluster remains unaware of the physical structure of the volume, which may be very complex. This simplifies all import operations.

As snapshots or point-in-time copies and data replication become more and more important for improved data protection and other storage management applications, it's worthwhile to look into these two in more detail.

Snapshots

A good example of the type of service implemented through virtualization is device-independent snapshots. When data in a typical volume or file system is continuously changing, a snapshot provides a view of that volume that represents the data at a point in time. If it has data integrity (for example, with database files quiesced and all buffers flushed to disk), that view can be used for a tape backup by mounting the view at a backup server. Of course, snapshots have been implemented in arrays for many years, but virtualization changes the requirements as the table on the next page demonstrates.

These may seem like small differences, but they can have a dramatic impact on the costs associated with the use of snapshots. A tiered approach to storage acquisition can be employed, with secondary functions like storing snapshots offloaded to less costly arrays, freeing up both storage and I/O performance on the source array that can be used for more performance-critical applications.

TABLE 6–1 Comparison between Conventional and Virtualized Snapshots

Conventional Array-Based Snapshot	Virtualized Snapshot
Consumes storage within the source array	Can redirect the snapshot to another device in another array
Puts additional load on the source array to manage the snapshot	Offloads all snapshot functions from the source array
Requires the array to implement all snapshot functions	Can use non-array (e.g., JBOD) storage for both source and target

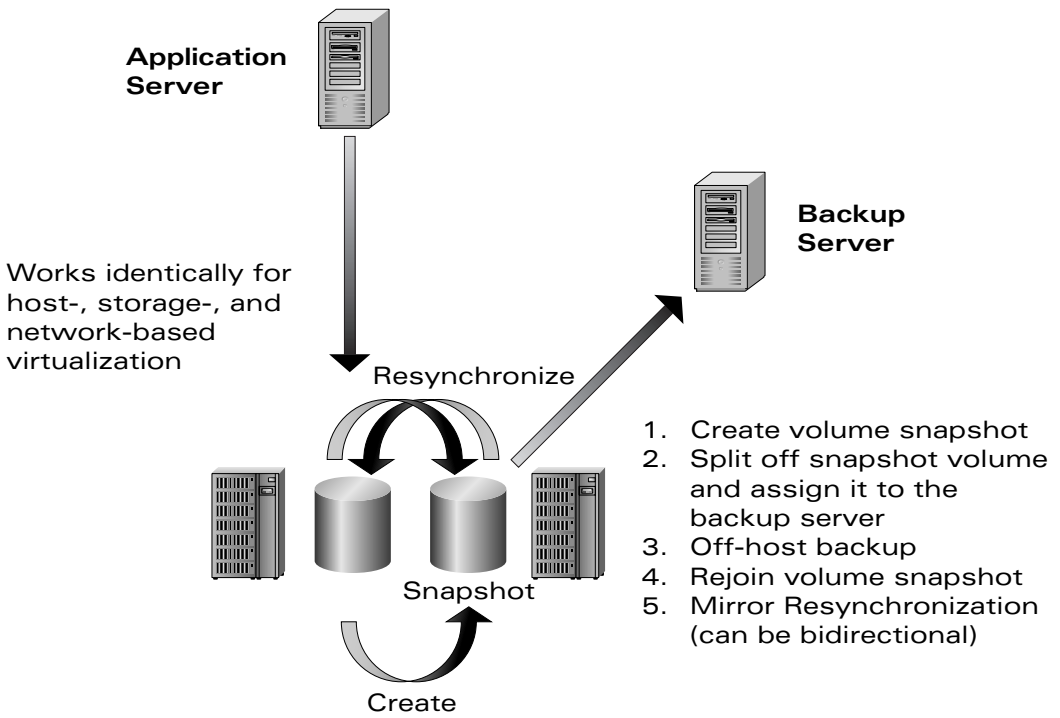


FIGURE 6–1 "Split Mirror" or "Full Copy" Snapshot

A snapshot-based replica can also be used both for later file restoration or disaster recovery (though note that effective disaster recovery, or DR, depends on moving the replica to a remote location), and for providing real data to a test

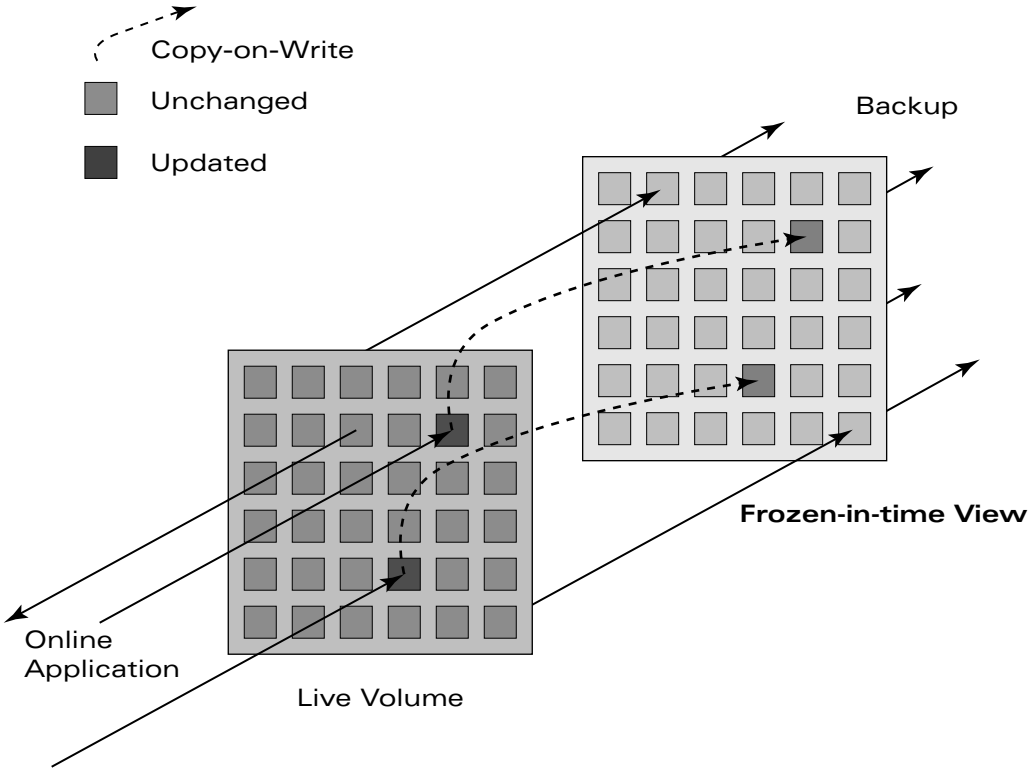


FIGURE 6–2 Copy-on-Write Snapshot

environment without risk to the original data. A local or remote asynchronous mirror can be established by updating the replica on a regular basis, based on what changes have occurred at the original file system since the snapshot, or since the last update of the mirror.

In most implementations, there are two distinct types of snapshot:

1. The "split mirror" or "full copy" snapshot, which holds exactly the same amount of data as the original volume. Figure 6–1 illustrates the creation and lifetime workflow of such a full copy snapshot used for an off-host backup solution.
2. The "copy-on-write (CoW)" snapshot, in which the snapshot target (work space) only holds original data for blocks that have changed, while the live volume holds the changes.

In order to be useful, a copy-on-write snapshot has to be combined with the original source volume in order to present a point-in-time view to an application (Figure 6–2). In a variation of copy-on-write, "redirect-on-write" snapshots write

the changed data out to the snapshot work space, leaving only frozen-in-time original data in the original blocks.

As only changed blocks are considered for copy-on-write snapshots, they don't require the same amount of disk capacity as the primary volume. The capacity really depends on the number of changed blocks during the period the snapshot is used. The downside is that, depending on the type of implementation and the number of block updates, CoW snapshots generate more workload compared to a split mirror snapshot, and cannot be used for data restoration if the snapshot source is unavailable.

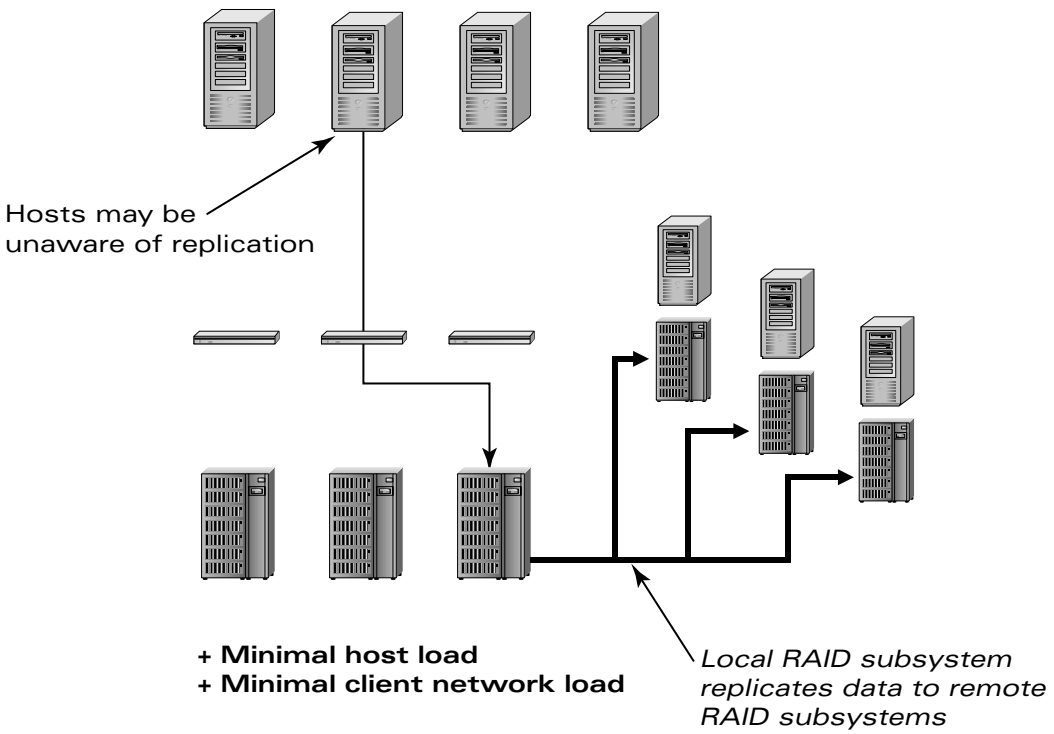


FIGURE 6–3 Storage-Based Data Replication

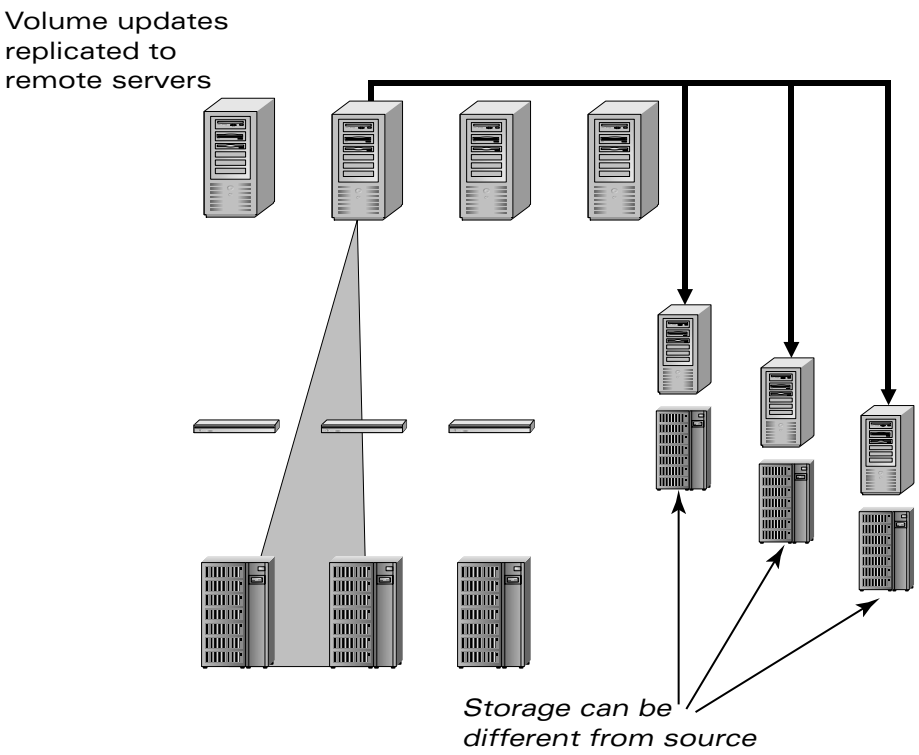


FIGURE 6–4 Host-Based Data Replication

Data Replication

When it comes to replication, virtualization is a key technique in providing cost-effective solutions regardless of whether replication is synchronous (i.e. the source and target remain in lockstep) or asynchronous (the target lags in time behind the source). The three diagrams below provide a comparison of replication methods at the storage device, at the host, and in the network fabric itself. Although these services have been provided at the disk arrays and at the host servers over a longer period of time, the provision of these services in the network, using in-band appliances or intelligent switches, or using out-of-band appliances, is gaining greater favor. The reason for this is apparent from these diagrams: Network-based services are independent of both host servers and storage devices, allowing for changes and scalability of these elements to be made more freely.

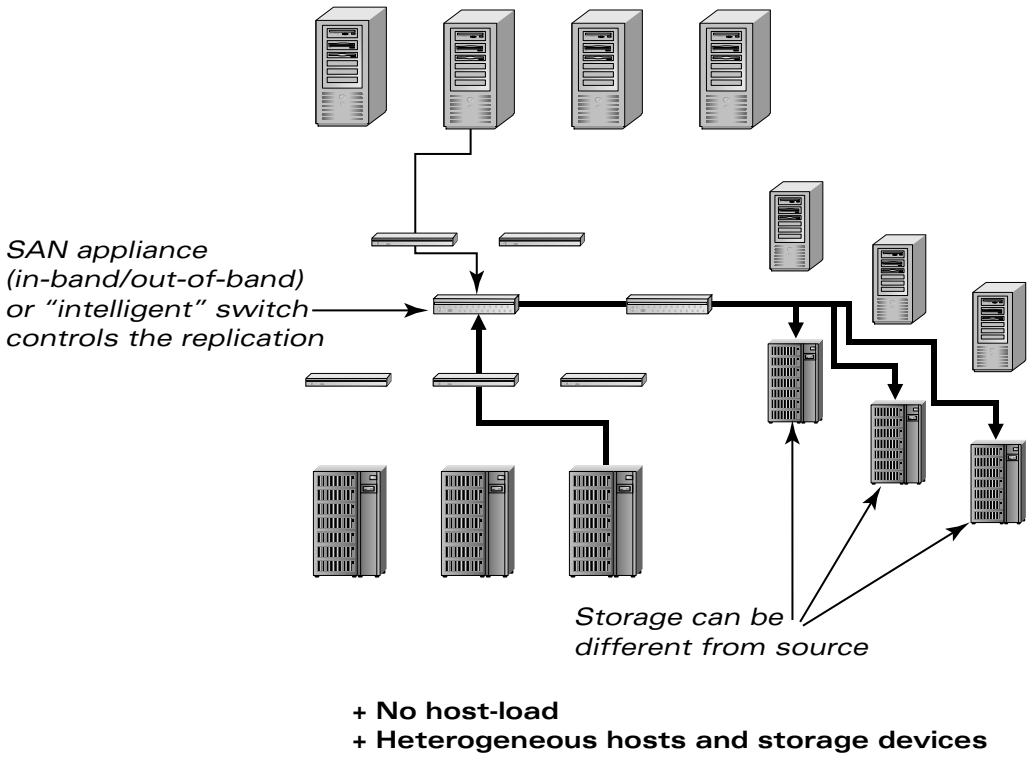


FIGURE 6–5 Network-Based Data Replication

With storage-based replication (see Figure 6–3 on page 36), the function is implemented within the array as firmware; typically, this approach only works between arrays of the same type. With a host-based solution (see Figure 6–4 on page 37), an agent installed somewhere in the I/O stack on the host is responsible for the replication. This type of replication is storage device independent and application transparent. The benefit of moving replication to the host is that it eliminates the problem of replication between different array types. Host-based replication typically shows a slight increase in server CPU and memory utilization—about 4–5% in typical configurations.

Network-based replication (see Figure 6–5 above) offers the chance to combine the best features of host- and storage-based approaches; as with storage-based replication, the work is offloaded from the host while maintaining the storage independence of the host-based approach.

Part II

Effective Use of Virtualization

7 Implementing Storage Virtualization

For many potential users of network-based virtualization, the key question is "How do I get there from here?" They want to know how to migrate an existing DAS- or SAN-based environment to a fully virtualized environment with minimum disruption to application availability and existing data. In this chapter we'll answer that question with a step-by-step analysis of the techniques used to migrate from a conventional storage implementation to a virtualized storage implementation. The initial question actually breaks down into several simpler questions:

1. Can I reuse existing storage assets even if they are currently directly attached to my application servers?
2. Will I need to do a complete backup/restore of data from drives that I want to virtualize?
3. How do I actually go about designing and implementing a virtualized SAN?
4. How do I ensure that my newly virtualized SAN is reliable, without any newly introduced single points of failure?

To a degree, some of the answers to these questions will be specific to a particular implementation of virtualization, so it's important to consult with the chosen vendor before making any final decisions, but that said, there are many common concepts that can be applied to the various solutions on the market today.

Reusing Existing Storage

For many customers, the first question is how much of their existing equipment can be reused. The equipment in question typically breaks down into several categories:

- SAN infrastructure. If the customer has existing infrastructure in the form of switches, hubs, HBAs, etc., are there any restrictions on its use in a virtualized SAN? The short answer is that the solutions on the market today should be able to work with existing SAN infrastructure. The long answer is that

individual vendors may have found problems in interoperating with particular devices (often a firmware upgrade issue), so it's always worth checking with the vendor as part of any initial implementation study.

- SAN attached storage. Again, this should all work without problems in a virtualized SAN, but the same caveat applies: Check with the vendor to see if there are any known problems with any of the storage in the SAN.
- Direct attached storage (DAS). With many of the in-band solutions available today, DAS resources such as SCSI-attached arrays can be migrated into the SAN simply by attaching them directly to the virtualization engine using a suitable HBA. The virtualization engine handles the FC-SCSI (or FC-SSA, etc.) bridging within the virtualization engine. For an out-of-band virtualization system, it will be necessary to use stand-alone FC-SCSI bridges to perform this function.

In short, most of the existing equipment can be reused, though particularly with older DAS, it's worth looking at the economics of reuse when migrating to a SAN environment. Older storage may well not be cost effective, given the maintenance charges and other recurring expenditure associated with it.

Backup/Restore Requirements

Regardless of the circumstances, a backup is always recommended when performing a migration to a virtualized SAN. That said, it may not be necessary to perform a restore of the data. For out-of-band virtualizers, the data on the disks should not be touched at all by the virtualization process. For in-band virtualizers, the answer is implementation dependent; most of the solutions on the market today offer the ability to take an existing LUN and simply layer the virtualization services onto the LUN without impacting the partition table or file systems on the LUN. However, using in-band virtualization with existing LUNs will prevent the virtualizer from offering a full set of advanced functions for that LUN, so it's worth discussing this issue with the vendor(s) in question.

Another option to consider is the use of host-based mirroring to mirror an existing LUN to a fully virtualized LUN as part of the implementation process. We'll look at this approach in more detail later.

Implementing a Virtualized SAN

For the purposes of this discussion we'll assume the following (see Figure 7–1):

1. The object of the exercise is to move existing hosts to a Fibre Channel-based, virtualized SAN.
2. The hosts all have direct attached storage in which the data must be migrated to the SAN with a minimum of downtime and disruption.
3. All the hosts have some form of host-based mirroring available, either built in to the operating system or from a third-party application installed on the host.

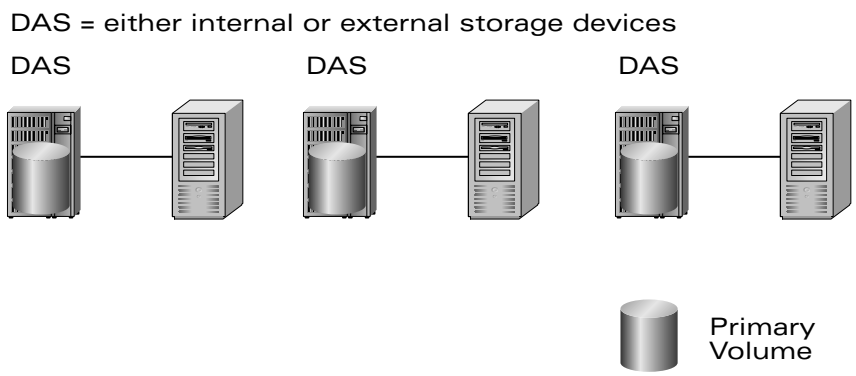


FIGURE 7–1 Existing Environment with Direct Attached Storage

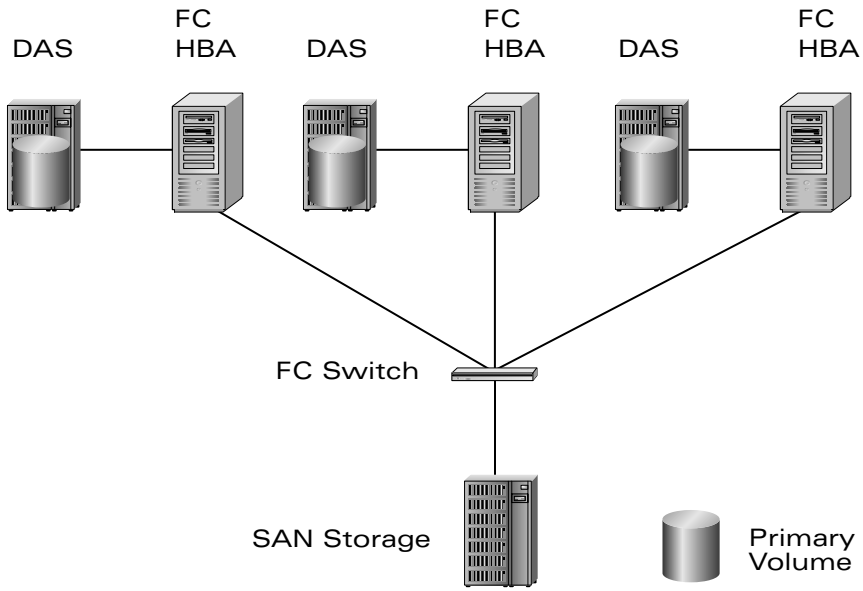


FIGURE 7–2 Adding the SAN Infrastructure

The first step is to introduce basic SAN infrastructure in the form of a Fibre Channel switch and Fibre Channel host bus adapters in each server to create the environment shown in Figure 7–2.

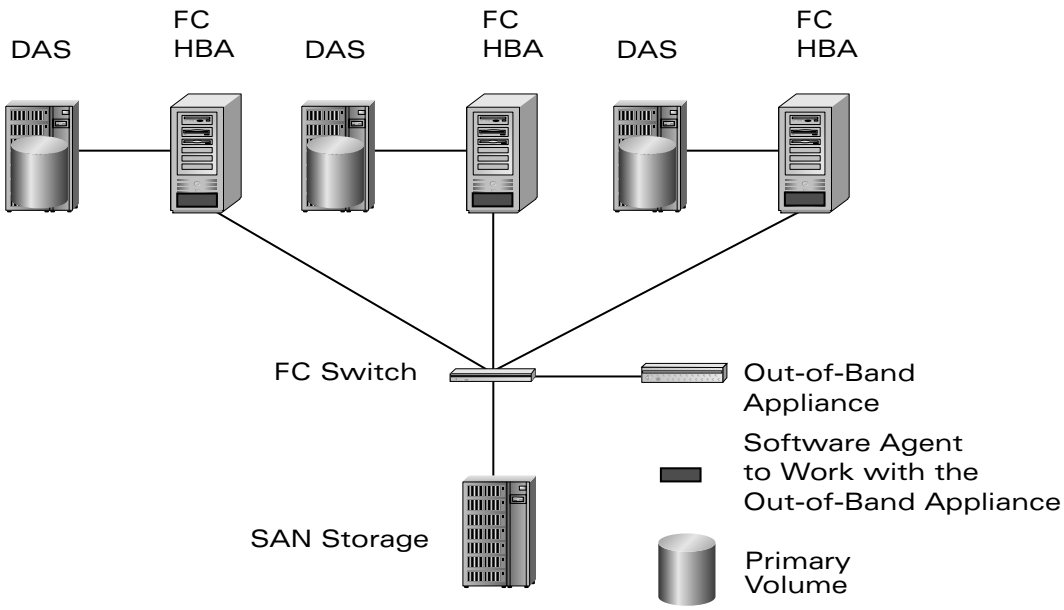
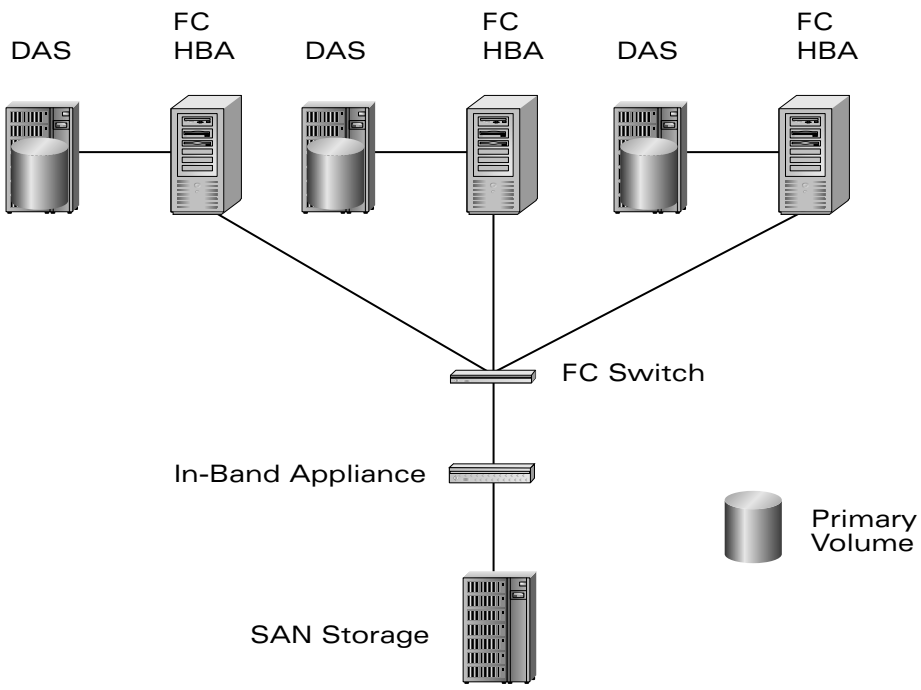


FIGURE 7–3 Introducing Out-of-Band Virtualization into the SAN

With the basic SAN infrastructure in place, the next step involves introducing the virtualizer into the environment, and this is where the difference between in-band and out-of-band approaches has the greatest impact. Figure 7–3 illustrates the SAN after the introduction of an out-of-band virtualizer.

With an out-of-band approach, the virtualizer is not in the data path between the application servers and the storage; this means that each application server must have either a specialized HBA or software installed to make the application server aware of the virtualization layer.

In Figure 7–4, the virtualization appliance is in-band, sitting directly between the hosts and the storage, appearing as storage to the hosts and as an I/O initiator to the storage.



Note 1: In some SAN implementations, the role of the switch can be replaced by the in-band appliance, which may be able to offer sufficient FC ports to perform both roles.
Note 2: With the introduction of smart switches, the role of the appliances can be augmented or replaced by intelligence in the SAN infrastructure itself. As these solutions were generally not available at the time of writing this booklet, subsequent figures discuss the appliance-based approach.
Note 3: Once the virtualizer has been introduced to the SAN and the necessary SAN and host configuration changes have been applied, there is little difference in the actual steps needed to migrate the data on the direct attached storage to the newly acquired SAN storage, so the rest of the figures in this section will not distinguish between the two approaches.

FIGURE 7–4 Introducing In-Band Virtualization into the SAN

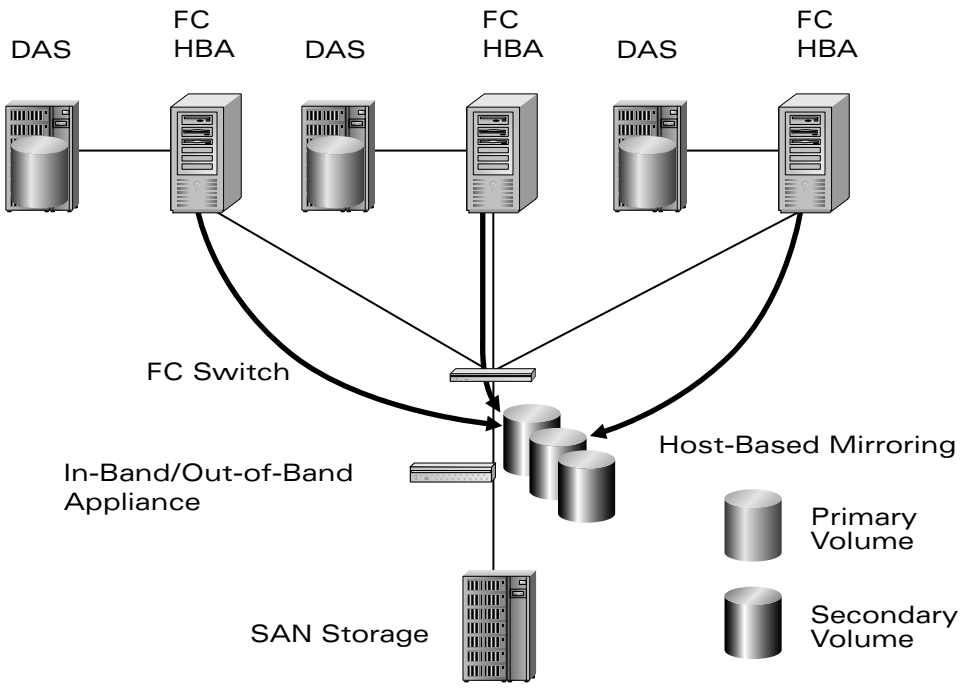


FIGURE 7–5 Using Host-Based Mirroring to Migrate Data

Migrating Data

Once the application servers are connected to the SAN and the virtualization engine is in place, the next step is to get a copy of the data on the direct attached storage into the SAN. There are several steps in this process:

1. A LUN equal in size to the locally attached storage is served up to the application server.
2. Using the host-based mirroring functions on the host, a mirror is created between the local storage and the SAN storage. At this point, there will be some performance degradation while the mirror is created, but scheduling the operation during off-peak hours can minimize the impact.

In this method (see Figure 7–5), the original storage (mirror source) remains on the application server, and is mirrored over to the SAN to create the secondary copy (mirror target).

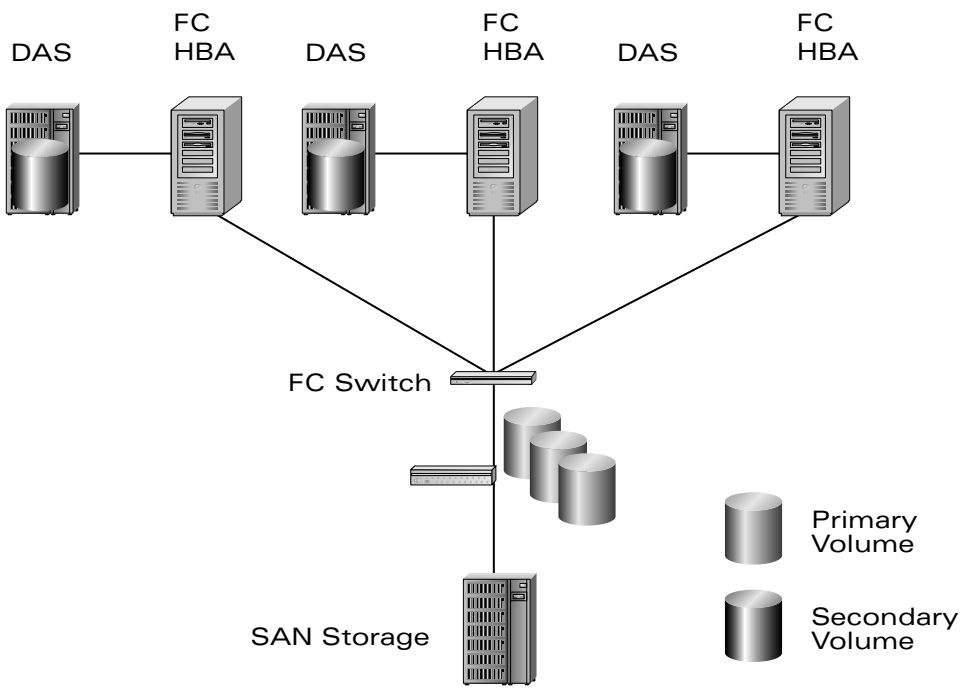


FIGURE 7–6 Swapping the Primary/Secondary Mirror Relationship

With mirrors synchronized, it's time to make the SAN copy of the data the primary source for the host. This is achieved either by using the mirror software on the host to switch the primary/secondary relationship, or, if the mirroring software doesn't support such a switch, by simply taking the direct attached storage resources offline and forcing the application server to switch over to the secondary data source.

This produces the situation shown in Figure 7–6, in which the application servers no longer rely on the locally attached storage for primary storage access.

Removing Direct Attached Storage

Ideally, the next step is to eliminate locally attached storage altogether, but before this can be done safely (i.e., without introducing additional points of failure), it's necessary to make the SAN robust by adding at least one more virtualization engine to the SAN.

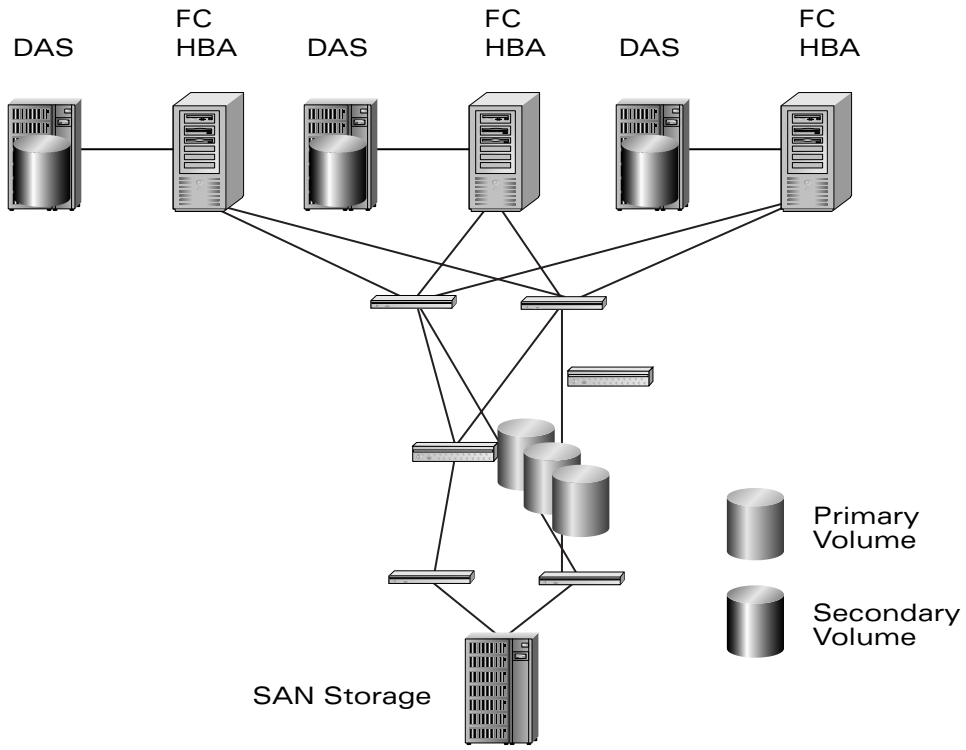


FIGURE 7–7 Adding Redundancy to the Virtualization Layer

The exact method used to add redundancy in the virtualization layer is implementation dependent and should be discussed in detail with the virtualization provider. At the very least it will require the addition of some form of alternate path software on each host unless the host will continue to perform the mirroring, in which case no alternate path software will be required. With redundancy added to the virtualization layer, it's now safe to migrate the secondary mirror copy from the host to SAN attached storage, as shown in Figure 7–8. Ideally, this process can take place in several steps:

1. Use the virtualization layer to create a mirror in the SAN without eliminating the host-based secondary. If this is possible, it eliminates any window of vulnerability when the host attached storage is removed.
2. Shut down the hosts and remove the locally attached storage. You may need to leave the boot device on the server. There are some advantages to having the boot device in the SAN, but it's not always possible; make sure its practicality is discussed with the storage, OS, and virtualization vendors involved.
3. If possible, reattach the storage removed from the hosts directly to the SAN to form part of the new storage pool.

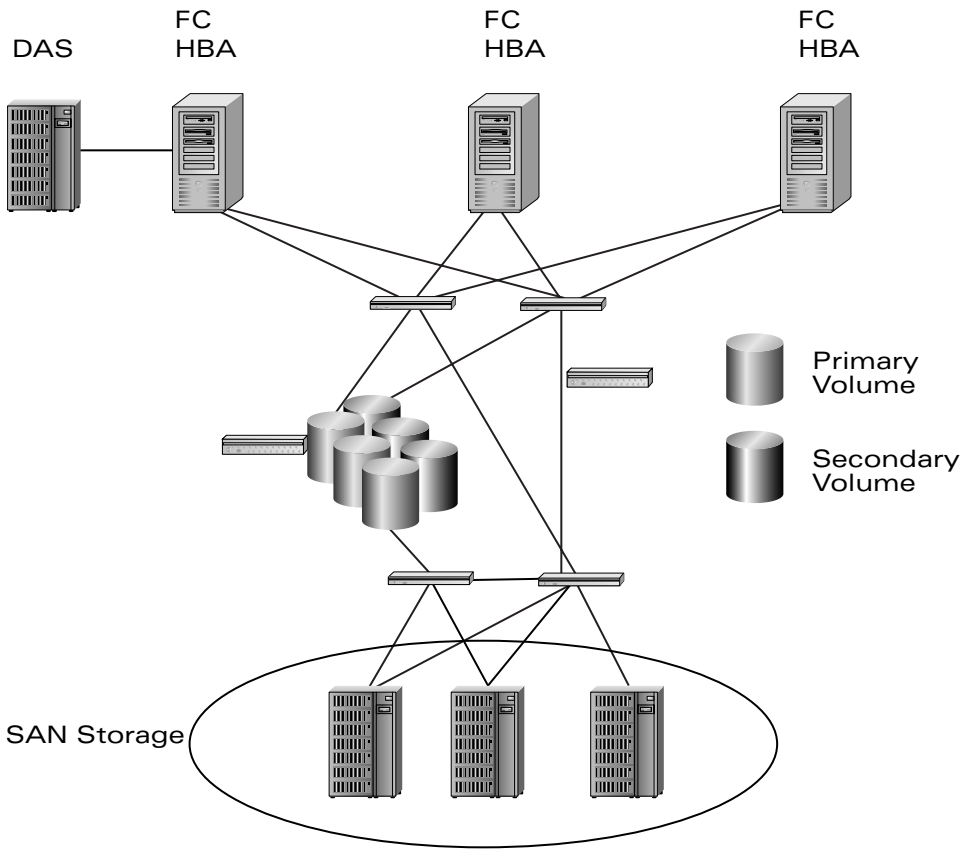


FIGURE 7–8 Eliminating Direct Attached Storage

Managing Growth

The final issue to deal with is how to expand the SAN as storage, host attachment, and performance requirements grow. Figure 7–9 shows how the SAN has evolved to include multiple virtualization appliances and additional storage and hosts.

The next chapter deals with the issue of scalability and the different approaches to it in more detail.

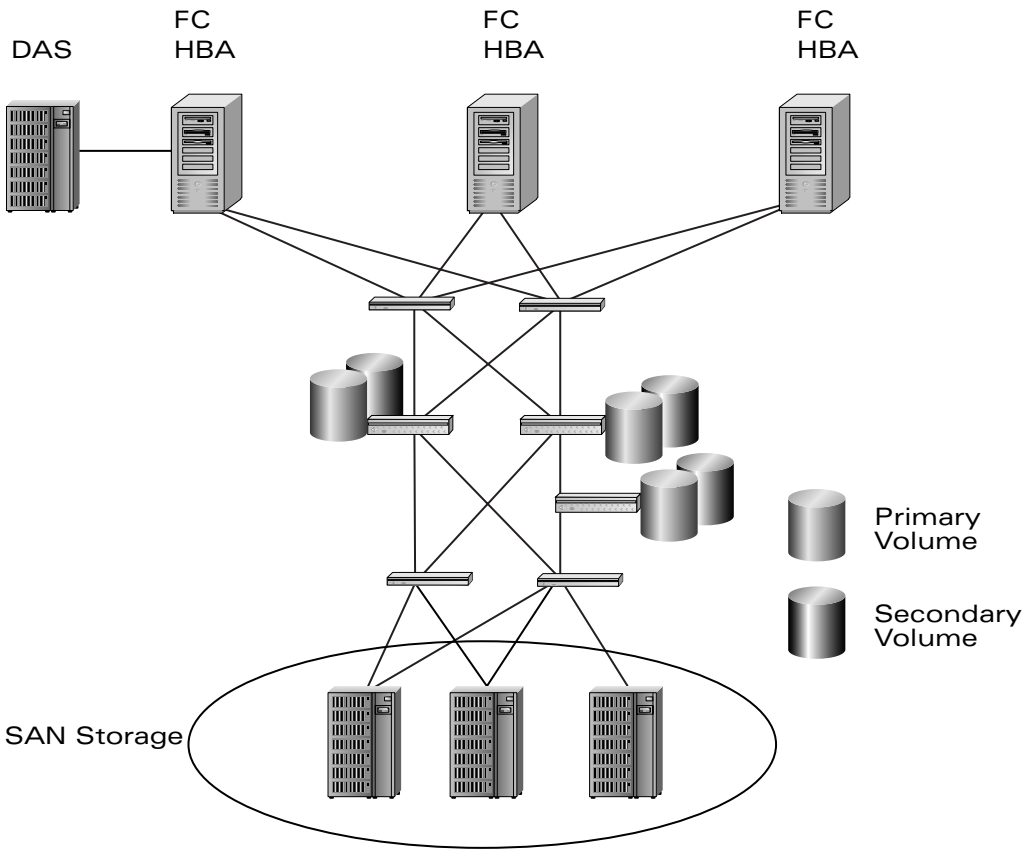


FIGURE 7–9 Managing Growth

8 Achieving High Availability

Basic storage virtualization provides several functions that are important in addressing data availability requirements, including:

- The ability to combine physical drives using RAID techniques like RAID-1, 5, and 1+0 overcomes the potential loss of access and data integrity posed by the data being held on a single physical disk.
- Mirroring and/or synchronous replication between arrays eliminates the actual array as a potential point of failure.
- Advanced data services like remote asynchronous data replication allow copies of data to be created off site and provide a solution to the problems posed by a more serious event that can close down an entire data center.

However, these functions are worthless if the virtualization method introduces additional areas of vulnerability, such as single points of failure in the network. Accordingly, it's important that the virtualization method itself be reliable.

There are several aspects to consider when striving for very high levels of availability (for example, greater than 99.99% uptime, or "four nines") in a virtualized storage network. In network-based virtualization, one should never rely on a single virtualization engine, appliance, or switch in the topology of concern. A dual-engine or clustered-engine approach is called for, ideally with a high degree of transparency to the host layer in case of network-layer failure. In addition to clustered-engine network approaches, the methods of deploying the engine(s) in a highly available topology include active-active, active-passive, and true N-way (N+1, N+M) techniques.

For example, in a dual active-passive in-band approach, there would be two virtualization engines, each able to control a particular set of virtual volumes and present those volumes to the host layer. In the case of engine failure, the passive engine must transparently assume the presentation and I/O flow to the affected virtual volumes. The exact technique to achieve this highly available volume presentation is vendor specific, so careful consideration of each vendor's approach is called for. The engines must also perform subsystem-independent mirroring or replication to ensure that the volumes are not endangered due to subsystem failure.

In an active-active approach, the concept is similar, but each engine performs a portion of the total I/O activity to the virtual volume set, as well as performing replication. As in the active-passive case, upon engine failure the remaining engine must transparently assume the presentation and I/O load. Again, the engines in the active-active approach may be appliances or switches. In either the active-passive or active-active approach, the engines must have a reliable, private (not shared by hosts or subsystems) method of interengine communication, also known as a "heartbeat."

For high availability with regard to the host level, a cluster file system approach requires multiple subsystems and virtualization engines, as well as multiple paths from each host to the set of engines, coordinated by a cluster file system. Typically, cluster file systems are of an *N*-way design, such that a failure of any particular host only impacts the overall capability in a 1/*N* manner. In other words, given *N* hosts, if one of those hosts fails, $N - 1$ of the hosts remain and continue to provide coordinated access to the cluster file system.

In summary, a complete high-availability virtualization solution must include multiple-host support, cluster file system capability, host-based multipathing software (for access to the network layer), multiple virtualization engines via appliances or switches, replication capability, and multiple subsystems, each capable of subsystem-level virtualization (in the absence of a virtualized network layer) and/or subsystem multipathing to present volumes to the network layer for further virtualization and host presentation if necessary.

9 Achieving Performance

Although the use of storage virtualization does not necessarily compromise the storage performance of a SAN and may even enhance it, a prospective user of virtualization is well advised to examine the performance aspects of the systems under consideration. Several methods can be used by a virtualization system to enhance the speed with which storage is accessed by the servers on a storage area network.

In the case of network-based virtualization, the host and the array are relieved of most of the burden associated with virtualization. Additionally, an out-of-band virtualization appliance inherently provides wire-speed performance, or very nearly so, as the appliance only manages a transformation layer (agent) at each host, which in turn presents a direct connection between the host and its assigned storage.

Other methods of performance enhancement by virtualization systems include striping and caching. Striping by a virtualization system is done across multiple physical disks within an array, or between multiple storage arrays, a method that can be used by network-based virtualization (in-band, including switch based, and out-of-band) or by host-based virtualization. In general, the more physical devices the data is striped across, the better the performance in terms of throughput and I/O per second.

Caching is a well-established approach to potentially improving performance when there is a large difference in speed between different components in the data path. Traditionally, it's been offered in two places:

1. On the host. The vast majority of modern operating systems perform read-ahead and write-behind caching for disk I/O, allowing some of the I/O requests to be satisfied directly from the high-performance host memory.
2. In the array. Most storage arrays also perform caching functions within the array itself, allowing the array to better manage the order in which I/O operations are presented to the disks.

The addition of network-based virtualization offers another opportunity to cache I/O operations and further offload the array cache by moving frequently accessed

data closer to the host. With network-based caching, we have to consider the need for reliability within the new cache layer, particularly if writes are to be cached. However, this is a well-understood problem and can be addressed using the same cache-mirroring methods pioneered in dual-controller cached storage arrays.

Intelligent switches may not need cache at all to perform well, as they are not based on the store-and-forward technology. As already described earlier, internally they are using an out-of-band architecture. The virtualization client or agent, responsible for the virtual/physical I/O transformation, is implemented in high-speed ASICs on each port. These hardware components, combined with the switch backplane, ensure minimum delay times, providing the potential for wire-speed performance.

As mentioned in the previous chapter on high availability, virtualization systems often offer mirroring, and locally mirrored storage can be set up to be read simultaneously, providing two reads from disk simultaneously and reducing the tendency for contention for common disk resources. Another high-availability feature associated with virtualization on the SAN, dual (or multiple) pathing, also offers the capability for load balancing.

This can occur either at dual or clustered (N-way) storage controllers, or at dual (or multiple) host bus adapters, or both, and it is generally controlled by host-based agents that automatically use the path that is least busy for a given I/O request. Either mirroring or load balancing can be done without virtualization present, but if storage virtualization is in use, the mirroring should be done by the virtualization system.

In summary, virtualization systems inherently possess several ways in which storage performance can be enhanced. In some cases, there are features like mirroring and load balancing that are available without virtualization, but must be integrated with it when virtualization is used. In other cases, such as striping across storage arrays, this type of enhancement is only available with virtualization.

10 Achieving Capacity

From the moment the very first disk drive spun in a computer, storage administrators have faced the challenges of managing capacity in a cost-effective manner. Common problems associated with conventional approaches to capacity management include:

1. Fixed size: Typically, disks are either too small or too big for the proposed application.
2. Locked-in location: Capacity is not necessarily where it's needed.
3. Finite capacity: Disks run out of space, arrays reach maximum capacity, and it usually happens at the worst possible time.
4. Management complexity: Complex environments make adding new storage resources time consuming and error prone. Some companies talk of taking days just to ensure that adding new capacity to a host doesn't break anything.
5. One size doesn't fit all: Frequently, a host has different requirements for different volumes. In such cases, it would make sense to use expensive high-end storage for some volumes and lower cost storage for other volumes. Currently, few companies do that because of the complexity involved with serving storage from multiple arrays to a single host.

If you look at the way storage is managed today, there is a fairly consistent "flowchart" for the life of storage associated with an application server.

1. When the host is installed, the storage administrator assigns storage to it.
2. Once the application is running, it starts to create data and consume storage space.
3. As long as there is sufficient capacity on the application server, everything continues to operate smoothly.
4. At some point, the available capacity on the application server is insufficient to meet predicted demand and action must be taken.
5. New capacity must be found, assigned to the application server, and made available to the application.

This continuous cycle impacts IT organizations in two areas:

1. The process for adding storage and integrating it with applications (stretching file systems, moving data, etc.) is largely manual, complex, and prone

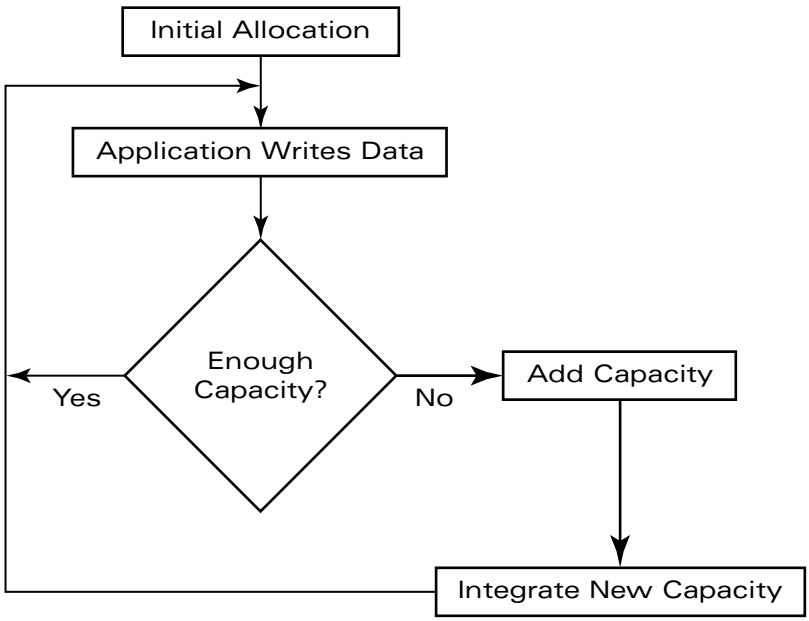


FIGURE 10–1 Life Cycle of a Conventional Logical Volume

to error. This is particularly true in large complex SANs with many hosts, storage arrays, and zones. As a result, not only can the process take several days (mostly in planning and preparation), but also the cost of a mistake can be incalculable due to downtime and lost business opportunities.

2. In order to minimize the number of times an application server requires additional storage, many IT organizations pursue a strategy of overprovisioning of storage to application servers in a futile effort to avoid the problems caused by running out. This results in unnecessary storage purchases and large amounts of capacity sitting idle waiting for applications to consume it.

Storage virtualization can improve many aspects of the problem by:

- Pooling storage: The problems caused by not having space in the right place are reduced when available storage assets are pooled so that storage from any array can easily be assigned to any host.
- Simplifying allocation: The process of actually assigning storage to application servers is simplified by providing a common management interface and simplifying zoning requirements within the SAN.
- Expanding LUNs: By their very nature, disk drives come in fixed sizes; virtualization removes the limits of fixed disk drives and also allows LUNs to be expanded after they've been assigned to the host.
- Being a platform for automation: Many capacity management tasks are being automated in virtualized SANs without having to deal with the idiosyncrasies of each array.

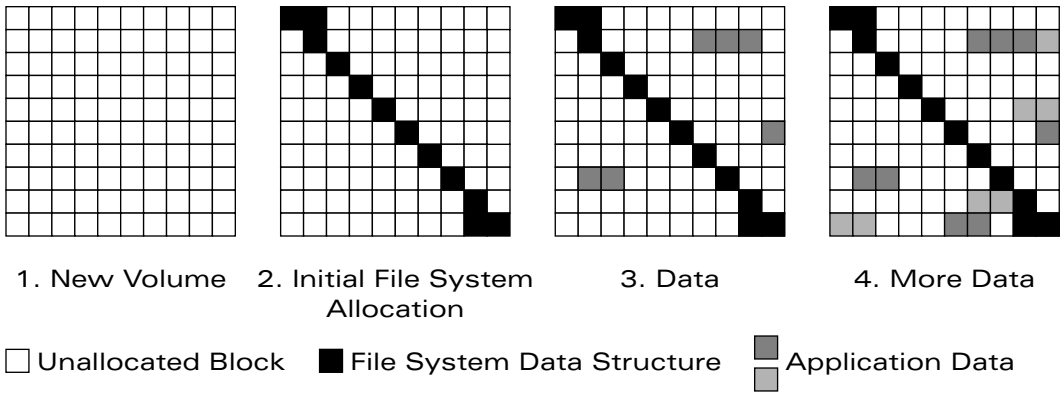


FIGURE 10–2 Allocating Storage on a Sparse Volume

Adding Capacity

The issue of expanding LUNs is worth examining in more detail because at least two distinct approaches to the problem are emerging. The first approach is based on automating conventional techniques and the other is based on a truly virtual LUN with capacity allocation handled in ways analogous to those used in virtual memory.

The more conventional approach utilizes (with varying degrees of automation) a relatively simple sequence:

1. Agent software (or a system administrator) on the application server detects that free space on a particular volume has fallen below a specified threshold or high-water mark.
2. Additional capacity is assigned to the application server (with a new LUN, by resizing or the existing LUN). With all storage resources pooled and a simple interface to hide the complexity, virtualization dramatically simplifies this process.
3. The new capacity is integrated with existing capacity. How this integration takes place depends on the sophistication of the host's volume manager and file system. On some platforms, the operation is entirely seamless and there is no interruption to the application. Other platforms may need the application to be taken offline to complete the process.

For this approach to adding capacity, virtualization offers the promise of widespread automation of the provisioning process through a single interface (or API) independent of the storage array.

The conventional approach to adding capacity is not the only approach; solutions implementing a "sparse volume" are also emerging. The sparse volume approach attacks the problem in a completely different fashion. A sparse volume is a truly virtual device in which the 1:1 correlation between capacity seen by the host and capacity used on the array is broken—the host might see a 2-TB LUN,

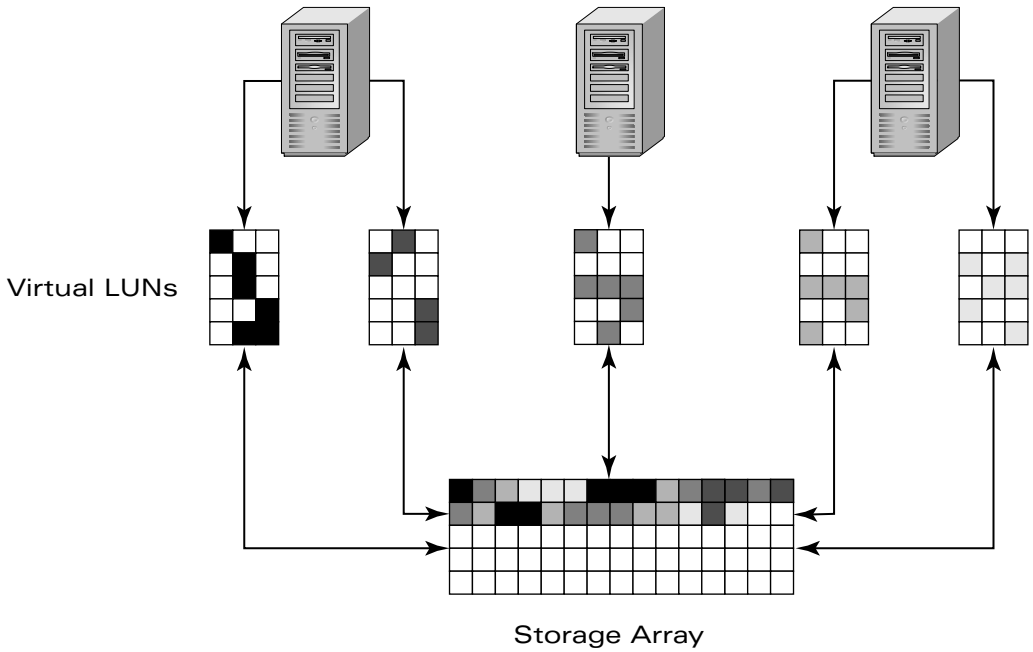


FIGURE 10–3 The Limitations of Sparse Volumes within a Single Array

but until data is written to that LUN it consumes no physical storage. This approach is analogous to that used in virtual memory operating systems where a process may be given 4-GB of virtual address space, but only consumes physical memory resources for those addresses in which it actually stores data or instructions.

In a storage context, the use of this kind of virtual address is relatively simple to understand, as we can see from Figure 10–2 that shows how storage is allocated to a new volume.

- In (1) a new sparse volume has been created. When it is first assigned to the application server, it requires no physical storage.
- In (2) the volume has been partitioned and formatted, and physical storage has been allocated to hold the device partition table and the data structures created by the file system (superblocks, etc.).
- Stages (3) and (4) represent stages in the life of the volume at which applications have written data to the disk, resulting in additional allocations of storage from the storage pool.

Although sparse volumes have been offered with some array products, they've always been limited by the capacity of the array. For example, if an array has a

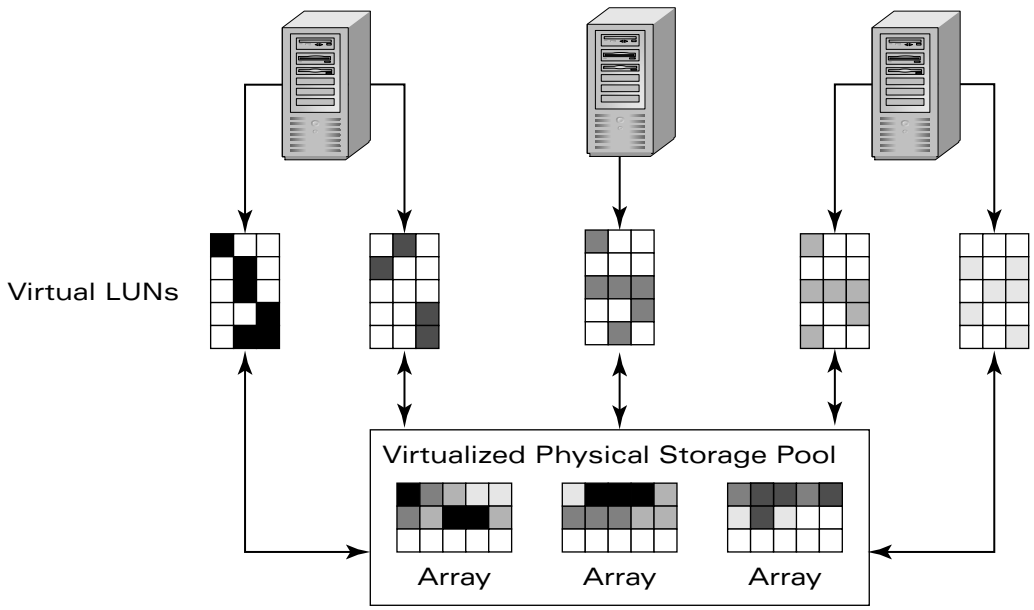


FIGURE 10–4 Overcoming Array Capacity Limitations by Storage Pooling

maximum capacity of 5 TB and five 2-TB sparse volumes are allocated from it, there is a chance that one or more of the volumes may reach a point at which the host thinks that additional capacity is available, but all the capacity in the array has been consumed.

The solution to this is the pooling of storage resources enabled by virtualized storage. The pool of storage used for satisfying physical storage requests can be expanded by adding additional arrays.

Perhaps the greatest benefit for many administrators is the fact that the sparse volume approach requires no additional software on the host—it doesn't matter which OS the host uses so long as it can access volumes on the SAN.

When the sparse volume is combined with a volume manager that can aggregate several LUNs into a single volume for the OS, virtualization offers an end to the allocate-use-expand lifecycle. Combining these technologies allows the administrator to allocate as much logical capacity as the server could conceivably use during its life without having to buy that capacity until it's actually needed.

Virtualization of storage allows new solutions to old problems and can reduce or even eliminate many of the more time-consuming and error-prone tasks associated with managing the storage capacity assigned to each application server.

11 Storage Virtualization and the SNIA Storage Management Initiative

August 12, 2002, saw the public announcement of the SNIA's Storage Management Initiative (SMI), SNIA's strategic initiative for solving interoperability problems and providing heterogeneous storage management solutions. SMI is a development of work that originated in a working group of 16 SNIA members, under the former code name "Bluefin," which defined a set of rules to ensure uniform storage management. Its recommendations, based on the standards known as Web Based Enterprise Management (WBEM) and Common Information Model (CIM), were accepted by SNIA and became the foundation of SMI, which is being developed further by various technical committees and vendors.

WBEM is a generic term used to describe a management protocol based on the transport mechanism familiar from the Web: HTTP (HyperText Transport Protocol). WBEM uses the Common Information Model (CIM) to describe the storage objects to be managed. CIM is a hardware- and implementation-independent information model that describes physical and logical storage objects and their interdependencies. It can be extended quickly using the eXtensible Markup Language (XML) encoding function CIM-XML, and works with various frameworks. In addition, WBEM offers the possibility of converting SNMP to CIM using customized software extensions, with the result that older storage systems could also be managed using this gateway functionality.

In the SNIA's Shared Storage Model, the management function is implemented on all layers using embedded agents in the objects to be managed. If no embedded agents are available, proxy agents or external providers can also act as the management interface. The agents or providers communicate with the management stations (SMI-S "clients") by means of CIM-XML via HTTP, and provide status and reports and monitor information. Clients send SMI-S commands for active management to the agents that convert these into vendor-specific proprietary instructions to be executed for the individual physical and logical storage objects.

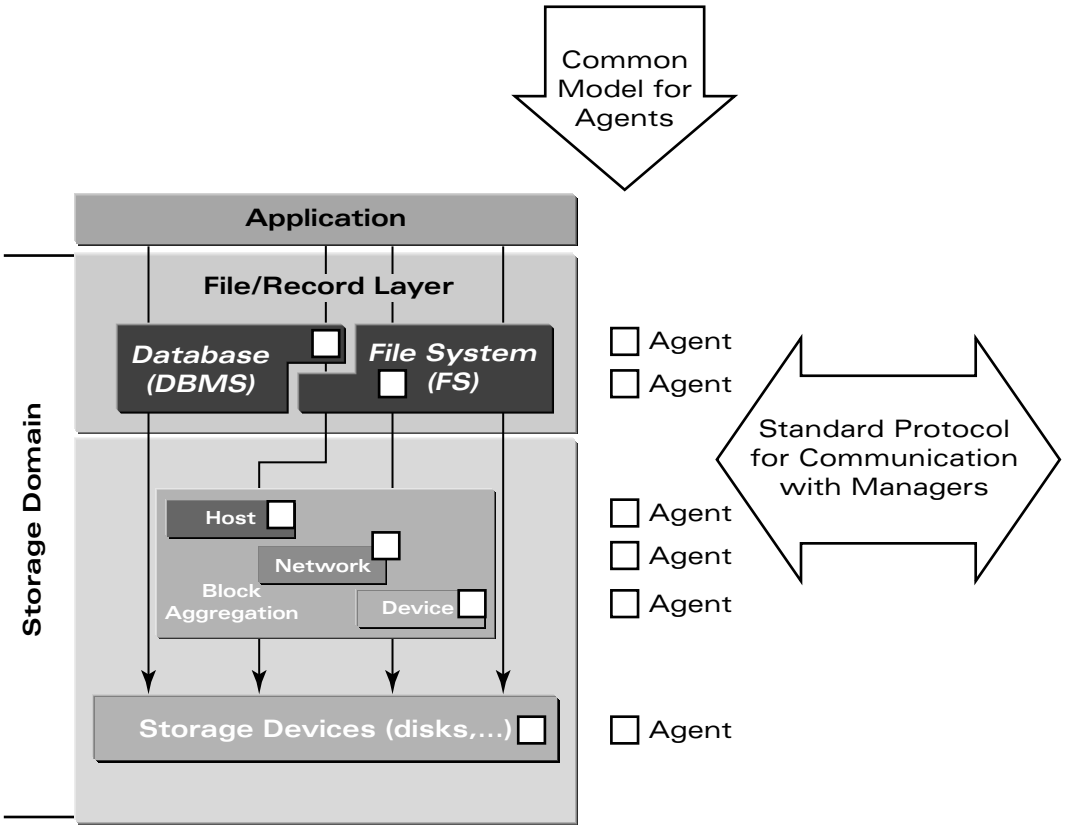


FIGURE 11–1 Management Agents Within the SNIA Shared Storage Model

SMI-S is based on the WBEM/CIM services and also describes how they are to be implemented in a complex storage environment. As a result, SMI-S contributes technologically significant innovations in the following five areas:

1. A common interoperable transport protocol is defined by WBEM/CIM.
2. Discovery challenges are solved with the Service Location Protocol (SLP).
3. Locking mechanisms (to be implemented in later SMI versions) using a common lock manager ensure stability and data integrity.
4. Storage security techniques (authorization, authentication, encryption, etc.) can be integrated.
5. Simple installation and implementation in customer environments is possible.

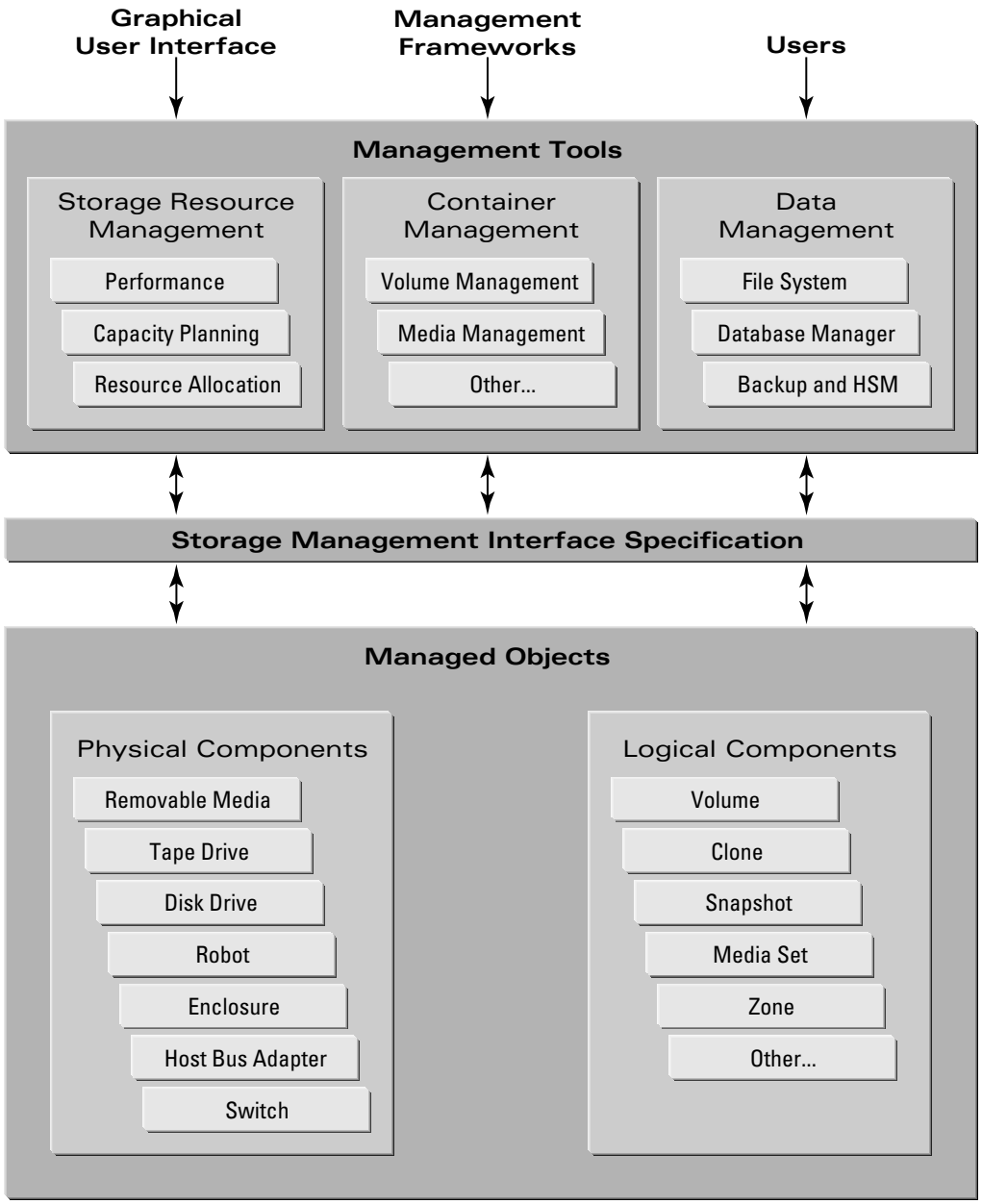


FIGURE 11–2 SNIA SMI-S Architectural Vision

The SNIA is set to complete the definition of the SMI Specification V1.0 by 2003. However, that does not automatically mean that this storage standard will then be implemented everywhere overnight. Nonetheless, the SNIA's aim is to ensure that 50–60% of all new products shipped in 2004 will support and may be managed via the Storage Management Initiative Specification (SMI-S) interface, and that in 2005, all new products support SMI-S.

So, the obvious questions are:

1. Does SMI-S represent a "virtualization" technique?
2. How will widespread adoption of SMI-S impact existing and future storage virtualization efforts?

The answer to the first question is a qualified "yes"—SMI-S is not a virtualization of storage per se; it is a virtualization of the management APIs for the different vendors' components.

For example, SMI-S may define a method for creating a volume, and this method can be used by any management application. Each array vendor then creates an SMI-S-compliant management layer that accesses device-specific commands and procedures to turn the generic SMI-S volume creation instructions into the specific commands needed by a particular array.

The long-term impact of SMI-S on virtualization products is profound. Today, if an SRM management application or a virtualization engine wants to create LUNs or manage snapshots or data replication, it must do so using the device-specific, proprietary, and frequently undocumented interfaces of each component. Such a reverse engineering approach represents a tremendous burden for the developer of the solution and can't be guaranteed to work from one release to another because undocumented or unsupported interfaces can be changed by the vendor of the component at any time. If SMI-S becomes widely adopted, this all changes: Management applications can monitor and control any SMI-S compliant component without having to develop device-specific routines or worry about the vendor changing API calls or other interfaces within a device.

12 Policy-Based Service Level Management

Virtualization provides real-time management for quality of service (QoS). In today's IP networks, we take QoS management for granted; network administrators can define policies to ensure that the right systems get the necessary network traffic priority at the right time. This sort of capability can be added to SANs using the latest generation of virtualization products. For example, automated QoS policies for different application servers will ensure that a critical end-of-month inventory database run gets priority for I/O traffic over less critical functions.

Today, most of the tasks to meet the defined service levels for storage capacity are still performed manually. This chapter describes how a policy-driven approach helps to automate the process of capacity planning and provisioning and further improve service levels for storage consumers.

As a result of increasing quantities of data, the number of people involved in managing storage is by necessity increasing, and as a result, storage management costs are increasing. A good example of why this is happening can be drawn from the complex, labor-intense process of storage provisioning.

Because of the complex procedures involved in assigning new storage resources, many data center managers are fearful of not being able to respond to storage capacity bottlenecks in a timely fashion, and so they still allocate excessively high capacity to business-critical applications. It's this overallocation approach that needlessly ties up storage resources and capital and leads to the poor utilization rates seen in typical data centers. On the other hand, if too little storage is allocated to the server, it soon reaches a point where more storage must be allocated, and if new storage resources are not added promptly, the application stops, causing unacceptable downtime—with all the financial consequences that downtime entails for critical applications.

The key to avoiding these problems is Automated Storage Resource Management (ASRM), which, together with storage virtualization, can reduce the human (and therefore error-prone) element in these repetitive tasks and allow automation to do the "heavy lifting." Ideally, all the storage administrator should have to do is define rule- or policy-based responses that allow an automated system to detect problems before they become critical and solve those problems without human intervention.

The virtualization that is the subject of this book is just one component of a comprehensive storage provisioning and management solution. There are other tasks as well: Storage can only be virtualized for servers if it is discovered and made physically accessible to them. A large and complex storage network that is growing requires additional functions, particularly when the newly added storage must first be secured by access mechanisms such as zoning and/or LUN mapping and masking.

The heterogeneous nature of open systems storage confronts administrators with the need to master a variety of additional management functions and interfaces. But because of the subtle interface differences from one vendor to another, use of these interfaces is prone to error. This is where modern heterogeneous SAN virtualization and storage resource management (SRM) tools, with their ability to integrate devices from different vendors, gain particular importance. They provide passive and active functions and simplify the workload in a heterogeneous world by promising a single consistent interface to storage management functions and the underlying hardware. Their role in IT operation is increasing in importance, especially in ensuring that service level agreements (SLAs) are met.

Although glossy marketing brochures portray storage provisioning automation as a simple concept, from a technical point of view it is based on many preconditions and intermeshed functions within the storage resource management system. The following steps represent a rough summary of what it takes to provision new storage for a business-critical database function in an enterprise.

1. *Monitoring of storage capacities* The basis for an intelligent decision-making process is to correctly identify utilized and free storage capacity as absolute figures and as percentages. The SRM tool has to analyze the storage capacities of various storage quality classes (based on attributes such as high performance, resilience, price, etc., as presented by the virtualization system) in conjunction with certain applications, users, departments, and other requirements. Very detailed storage information, up to the file system and application level, are required; otherwise, real end-to-end management from the storage consumer's point of view is not possible.
2. *Event and threshold management* Here, the utilization of logical storage units, such as file systems and table spaces, is compared with predefined limits, and specific responses are triggered if they match. These responses range from simple notification of the administrator to immediate automatic functions that will run on their own without external intervention.
3. *Discovery of free storage capacities with the right attributes* It is important to discover and report exactly on free storage capacities in the storage network. However, as noted, not all storage is identical or meets all the requirements for the specific applications. Storage is classified differently on the basis of attributes such as high availability, performance, cost, or other SLA parameters. One means of making things easier is to create storage pools with different attributes beforehand using virtualization and have administrators assign each new storage system to one of them as it is brought on line.

4. *Transferring the storage to the server's zone* If the new storage resources are already in the server's zone, this step can be skipped. However, if the storage pool or the newly discovered storage is in another fabric zone, the SRM tool must make changes to the configuration in the storage network. The zoning of the fabric switches is adapted using vendor-specific APIs or with the aid of the standard SNIA SMI-S interfaces, so that the server and storage can be united physically.
5. *Permission for the server to access the new storage* Many disk arrays in a SAN use security functions such as LUN binding and LUN masking for access protection. LUN binding firmly assigns access to the LUNs via specific Fibre Channel ports of the disk arrays. LUN masking uses WWN tables to control which server is given access rights to the LUNs. The WWN address of the server host bus adapters may need to be entered in the access tables of the storage systems using the LUN security functions—once again, either via vendor-specific APIs or the SNIA SMI-S standard. Only then is the server given access to the additional storage capacities that are to be allocated to the application.
6. *Allocating and changing the size of existing volumes in the server* An application cannot do much with new raw LUNs, so block virtualization is used to integrate new storage in the existing logical volumes that the application is currently using. They must be allocated without interruption to operations ("on the fly"). Any necessary changes to the layout of the volumes pertaining to the RAID level or number of disk mirrors are likewise carried out in online mode with the help of virtualization.
7. *Informing the application of the increased capacity* The last step is to inform the application that it now has more storage capacity. There are various techniques for doing this, as described in Chapter 10 about capacity management. Some vendors of virtualization solutions offer what is termed "sparse allocation." Sparse allocation is a technique that leads the application to believe that it always has a volume of 2 TB, although this may not physically be the case. Physical storage up to this limit is actually assigned to the host only if the host actually writes data to the volume.

   Another technique is to change the size of the file system and that of the underlying logical volume at the same time. Because the file system is the standard interface of the application for managing occupied and free storage blocks, increasing the size of the volume and file system increases the proportion of free blocks that the application can use.

   Applying this principle in reverse, an efficient capacity planning concept should naturally be able to recognize overprovisioning of storage capacities over a lengthy period of time and to respond by retransferring unused storage resources back to the free storage pool. Of course, the size of the file systems and volumes must be able to be reduced in online mode to avoid application downtimes.

   Although charge-back is not always a requirement for storage on demand, it is worthwhile to charge the actual use of resources to the storage consumers. This is not only fair; it automatically leads to savings because

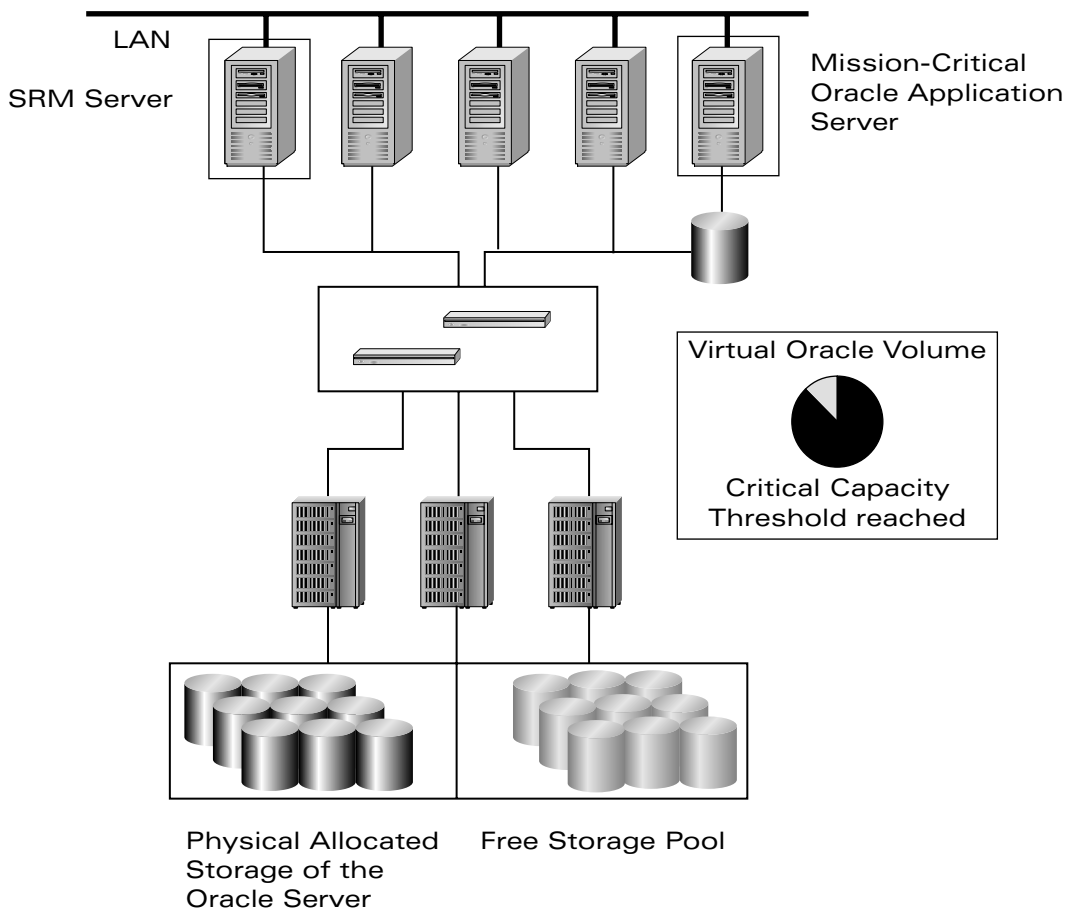


FIGURE 12–1 An SRM Management Application Detects a Critical Capacity Problem

users are more aware of the resources that they use. Here again, the use of virtualization and other tools for organizing and allocating storage is a prerequisite for this type of management.

Today, much of the work that saps the resources of the IT department involves simple and mundane tasks that can and should be automated. This type of automated service level management is inconceivable without storage virtualization. Trying to manage and automate storage tasks while dealing with the individual characteristics of each device is too complex. Through automation, many daily tasks will no longer require human intervention. For example:

- Adding new capacity to a storage pool, or assigning new resources from the pool to an application server, can be performed automatically without impacting service to any application server.

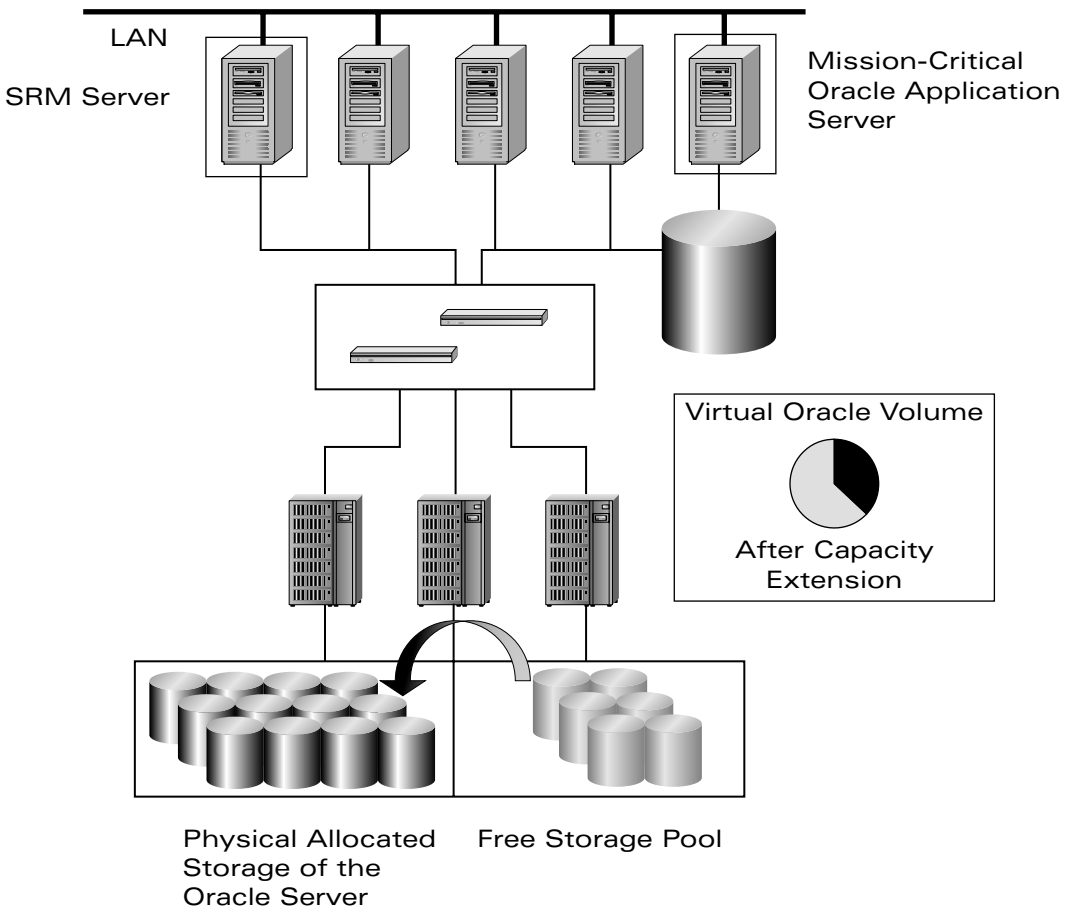


FIGURE 12–2 Policy-Based Storage Management Solves Capacity Problems Quickly and Without Manual Interaction

- Existing volumes can be dynamically resized, migrated, and optimized for performance, all without taking systems or applications offline.

Only storage virtualization promises to do this for us.

Once common procedures are automated, then it's possible to introduce policy-based responses to common conditions that previously would require the intervention of a storage administrator. The advantages of policy-based service level management are complementary to storage virtualization and are briefly summarized again below.

Improved Storage Utilization

According to independent analysts from Gartner and IDC, the average utilization of disk storage systems is currently about 50%. In other words, half of the capacity is wasted. This represents the waste of a great deal of money because the purchase ties up capital unnecessarily. In addition, delaying a decision to buy brings considerable cost advantages because the prices of disk systems continue to fall each year.

Automatic storage provisioning prevents overconfiguration with expensive storage resources. Instead of server-specific alternate pools, one single free storage pool is available for the entire storage environment. This generates tremendous savings by reducing hardware costs. In addition, the use of just one storage pool makes it possible to consolidate storage effectively and simultaneously reduce manual administration activities.

Reduced Downtime

Downtime due to running out of storage is a constant headache, whereas automatic just-in-time storage provisioning guarantees rapid allocation of storage capacity to business-critical applications. Policies define how and when to act in response to an application server running low on storage, ensuring that new storage resources are available before the application fails because of the dreaded "Out of Space" message. Such a system and process standstill causes huge losses in revenue for many companies.

Relief for Administrators

Based on virtualized storage, policy-based management will replace many of the tasks that are performed by storage administrators today. As a result, administrators will be able to manage more storage and respond in a more timely fashion to more complex problems for which responses can't easily be automated. They will no longer be called upon, in haste and at the eleventh hour, to locate new storage systems and assign them to the applications via numerous manual steps.

It should be noted that policy-based service level management is still a very recent discipline on the storage management market. Only a very few enterprises are deploying these techniques in their entirety. However, the continued rapid growth of storage, upcoming new storage standards driven by the SNIA, and the pressure of costs on IT departments will drive the spread of automatic storage provisioning throughout the storage market within the next few years.

13 The Future of Storage Virtualization

Unified Management

The last two chapters dealt in detail with a leading-edge issue: unified management with integration of discovery, reporting, storage virtualization, and storage automation. In a heterogeneous world, this method of intelligent storage provisioning through active management of all levels, from the application to the physical storage devices, is only feasible in the long term with virtualization working together with open standards such as SNIA SMI-S. "Storage as a utility"—heralded years ago when the first SANs appeared—can soon become a reality.

Automatic Data Migration Services

Storage virtualization will also evolve, and automatic data migration services will become commonplace. The reasons for this are technical as well as economical.

From a technical point of view, such services offer fault-tolerant, high-performance access to data. If the virtualization intelligence recognizes that storage systems can no longer cope with requirements, the data is migrated or copied to other, faster storage resources without intervention by the administrator or users even noticing anything. SLAs relating to performance and availability can thus be observed.

From an economical point of view, it should be noted that not all data is of equal importance to a company. That means that it is not necessary for all data to be stored and managed on expensive online storage systems. The older data becomes, the less frequently access is needed, statistically speaking. This is where data lifecycle management comes in. Intelligent data migration services, far in advance of current HSM concepts, can be used to ensure that less frequently used data resources are transferred to cheaper storage systems with several hierarchical levels, such as SATA, tape robots, or long-term archives, and managed there.

Data Center-Wide Volumes and File Systems

This booklet primarily describes block virtualization. Block virtualization creates volumes with a specific capacity, performance, and availability and assigns them to specific servers. If the same volume is assigned to multiple servers that access it simultaneously, certain precautions must be taken to avoid problems with data integrity. An important example of such a precaution is to implement a cluster file system that controls write access by the various servers and management of the metadata of the file system. However, currently volume sharing and cluster file systems are mainly found in homogeneous server environments. At the time this document was created, there was no solution on the market for assigning identical volumes to several heterogeneous servers and at the same time controlling access by means of a heterogeneous cluster file system. The different features of the various operating systems and their footprints in the volumes and file systems must still be "virtualized" for this to be possible. However, several companies are working hard to combine block and file system virtualization. Once this aim has been achieved, there will be no barriers to transparent access to data, irrespective of the application, server operating system, network, or storage system.

A Appendix

Glossary

Rather than expanding this document with many additional pages explaining storage networking-related terms, we strongly recommend that you use the online SNIA Dictionary of Storage and Storage Networking Terminology located at **www.snia.org/education/dictionary**.

In addition to the online access, SNIA provides versions of this storage networking dictionary on mini CD-ROMs.

About the SNIA dictionary: The members of the Storage Networking Industry Association have collaborated to create this Dictionary of Storage and Storage Networking Terminology. The collaboration has been extensive, with many members making substantial contributions. There has also been extensive review and comment by the entire SNIA membership. This dictionary thus represents the storage networking industry's most comprehensive attempt to date to arrive at a common body of terminology for the technologies it represents. The reader should recognize that in this rapidly evolving field, new terminology is constantly being introduced, and common usage is shifting. The SNIA regards this dictionary as a living document, to be updated as necessary to reflect a consensus on common usage, and encourages readers to treat it in that spirit. Comments and suggestions for improvement are gratefully accepted at any time, with the understanding that any submission contributes them to SNIA; and SNIA will own all rights (including any copyright or intellectual property rights) in them. Comments and suggestions should be directed to dictionary@snia.org.

B Appendix

Recommended Links

Join the SNIA:

www.snia.org/join

Participate in the development of the Storage Virtualization Tutorial:

www.snia.org/members/tutorials/virtualization

In order to participate, you must first become a member of the SNIA.

Home page and work area of the SNIA Storage Virtualization tutorial team:

www.snia.org/apps/org/workgroup/snia-edcomm/tut-virtualization

SNIA Shared Storage Model (SSM):

www.snia.org/tech_activities/shared_storage_model

SNIA Storage Management Initiative (SMI):

www.snia.org/smi/home