CHAPTER 5

Backup and Restore Technologies

Backup is the process whereby a coherent copy of data is made. Backup has become more important as the amount of data has exploded, not just in importance, but in volume as well. One study estimates that more data will be created in the next few years than has been created since the dawn of history! It is interesting to compare the growth in data storage with the more widely known and appreciated growth in electronic chip density. Recall that Moore's law implies that the amount of electronics on a given chip area doubles every 18 months. A lot of industry analysts believe that the growth in digital storage is actually handily beating Moore's law in the sense that the amount of data doubles in much less than 18 months.

Historically, tape has been used as a medium for backing up data. Initially tape was a much cheaper medium than disk. Subsequently it was argued that optical media would become the media of choice, but for various reasons this vision never came to fruition. Although the medium of choice (for backup) remains predominantly tape, regular disk drives are increasingly becoming the medium of choice for an initial backup and system mirror. This trend is due mainly to the falling prices of disk storage, which reduces the cost advantage of tape over disk storage. Another reason for the increasing use of disk-based backup is the higher speed, which ensures minimal downtime for server-based applications.

Note that both disk and tape as media for backup have their advantages and disadvantages, and both will continue to be used. Tape-based backup/restore offers a very high-density medium that can easily be transported for off-site archive or disaster recovery purposes. When an initial copy of data is made to disk, very often a secondary backup operation to traditional tape media is made from that disk-based copy.

This chapter explains the technical challenges that need to be solved in order to back up and restore data in a timely way. The chapter also

explains the various ways in which backup/restore techniques can be classified. Also included are discussions of new developments in Windows Server 2003 for accomplishing snapshots (volume shadow copy service) and of the Network Data Management Protocol (NDMP) and how it fits into the Microsoft vision of storage management.

5.1 Reasons for Backup and Restore

Backup is performed for various reasons, and those reasons very often dictate the investment made in accomplishing the backup. The assurance of data availability is the primary reason for creating backups. The higher the data availability requirements, the more investment needs to be made. For example, one form of backup that runs continuously is disk mirroring, in which every data write operation is reflected to another disk to guarantee extremely high data availability.

Data archival also is used to meet legal and other needs in which the data does not have to be accessible immediately but can be produced on demand in a reasonable amount of time, measured in hours, days, or weeks.

Backups are sometimes used to transport data—for example, when one decides to create another data center at a distant geographical location. A similar motivation is to migrate the data to new hardware or, more rarely, a different server platform.

5.2 Backup Problems

Before diving into the various ways that backup and restore operations are accomplished, it is advisable to understand the problems that need to be solved to accomplish the desired objective. The prominent issues are these:

- An ever decreasing amount of time, called the *backup window*, in which the backup operation must be accomplished
- An ever increasing number of APIs that backup applications need to support

■ An inability to back up files that are open and actively being used by an application

Sections 5.2.1 through 5.2.3 consider each of these issues in more detail.

5.2.1 The Backup Window

Historically, server applications were run during regular business hours only. Backup operations were typically done in the wee hours of the night when the applications could be stopped without causing user distress. Once the applications were stopped, the server would typically be taken offline and the data backed up. There are two problems with this old approach:

1. The huge explosion of data has meant that the backup is hard to accomplish within the given amount of time. Difficult as it may be to believe, writing to tape is an expensive operation, especially in terms of time and man-hours consumed. The correct tape has to be located, mounted, and then positioned. Once positioned, tapes are much slower to write to than disk. Whereas most hard drive interfaces are able to transfer data at a sustained rate well over 80MB per second, the fastest tape drives available today see maximum transfer rates of under 30MB per second. Robotic silos can be used to manage the multiple-tape media units, but they are expensive, and of course they can only alleviate the time spent in locating and loading a tape, not make the tape read or write any faster.

2. The second problem is that more and more applications, as well as the data they access, control, create, or modify, are considered important if not mission-critical. This means that the amount of time when the server could be taken offline to accomplish the backup is shrinking.

5.2.2 Explosion of APIs

Customers are deploying more and more enterprise applications that can be stopped rarely, if ever, for backup. Recognizing this fact, each application vendor has resorted to providing APIs for backing up and restoring the application data files in a consistent manner that ensures that no data

is lost. Although the creation of these APIs sounds great, a closer inspection shows that the problem is rapidly worsening as a result.

Figure 5.1 illustrates the problem of an ever increasing need to support more APIs in the backup/restore application. As this example shows, customers typically have multiple applications, and very often multiple versions of the same application. Each backup vendor now must write code to use the APIs provided by each enterprise application. Because many backup application vendors choose to separately license the agents that deal with a specific enterprise application, just keeping track of the software licenses and the costs can make any IT manager dizzy. Furthermore, even all this does not take into account the deployment of the infrastructure, personnel, and discipline to accomplish these backups.

5.2.3 Open Files Problem

Another problem with doing a backup is that it can take a considerable amount of time. If a tape device has a rated throughput of 10GB per minute, it will take 10 minutes to back up a 100GB disk. During these 10 minutes, applications will need access to the disk and will also be chang-
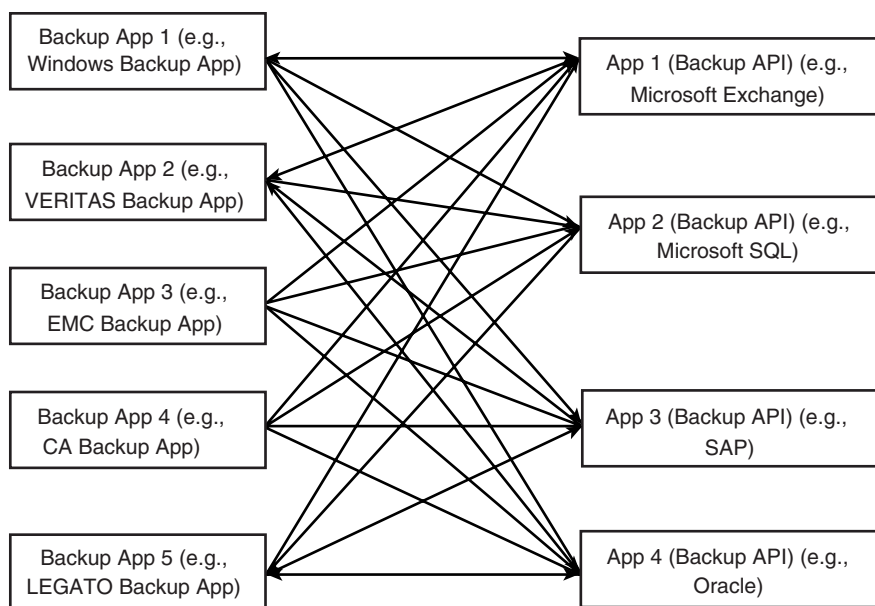
Figure 5.1 Backup API Explosion

ing data on the disk. To ensure that the backup is consistent, three approaches are possible:

1. To prohibit applications from accessing the disk while the backup is in progress. Blocking simultaneous user access to the application during backup was commonplace in the early days of PC computing, when 24x7 operations didn't take place. The backup was done in times of light load—for example, during night hours. Now this approach is not feasible, for a couple of reasons:
   - Operational requirements now often call for 24x7 application uptime, so there is no good time to do the backup.
   - The amount of data needing to be backed up has grown, so the operational hours have increased, and the backup window is often not long enough to accomplish the backup.
2. To back up the data while applications are accessing the disk, but skip all open files. The problem here is that typically, only important applications run while a backup executes, which implies that the important data will not be backed up!
3. To differentiate between I/O initiated by the backup application and other applications. Backup vendors came up with solutions that partially reverse-engineered operating system behavior. In particular, the solutions depend on the implementation being able to differentiate judiciously, a method that can break fairly easily. The implementations have generally used a varying degree of undocumented features or operating system behavior that is liable to change in new versions of the operating system. The solutions also depend on the availability of a sufficient amount of disk space. Another variation that applies to both techniques is whether the implementation works with one file at a time or is simultaneously applied to all files at once.

Three approaches have been tried to allow open files to be backed up yet have a consistent set of data on the backup media corresponding to the open files.

The first approach is to defer application writes to a secondary storage area, allowing the backup operation to back up all files. The approach must also be selective in operation; for example, it must allow the paging file writes to pass through but defer the writes to application data files or put them in a predefined secondary cache (often called a *side store*), ensuring that the data backed up is in a consistent state. I/O to or from the secondary storage area also needs to be treated specially depending

on whether the backup/restore application or a different application is doing this I/O. Once the backup application is done, the data must be copied from the side store to the regular file area.

The second approach is to implement copy-on-write for the benefit of backup applications. When a backup application opens a file, other applications are still allowed to write to the file. To avoid a mix of old and new data in the backup application, the data being overwritten is copied to a side store. If regular applications request this data, the read is handled by the regular Windows file system drivers. When a backup application requests this data, the data is retrieved from the side store. St. Bernard Software is one example of a vendor that has implemented this approach to backing up open files.

Consider Figure 5.2 and notice the layering of drivers (a detailed explanation of Windows drivers, device objects, and so on is available in Chapter 1). The file system filter driver is layered over the NT file system (NTFS) driver, which itself is layered over the disk filter driver. The disk filter driver in turn, is layered over the disk class driver. There are other drivers below the disk class driver (as discussed in Chapter 1), but these are not relevant to the discussion here. When an application opens a file,
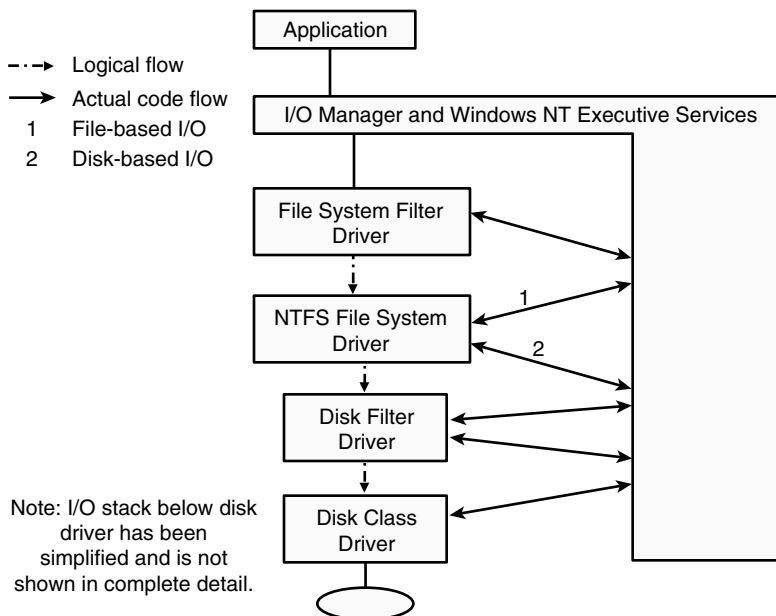
Figure 5.2 Windows NT Filter Drivers

NTFS (in response to application requests) issues a series of commands to read metadata (the location of the file in the disk) and then issues reads or writes to the logical blocks on which this particular file is stored.

The upper filter driver (above the file system driver) shown in Figure 5.2 is ideally placed to intercept file operations and divert the call, if that is what is desired to solve the problem of open files. Microsoft sells a product called the Windows Installable File System (IFS) Kit, which provides information needed to write such a filter driver. A backup vendor may choose to work at a lower level; for example, an image level would typically use a solution that involves writing a lower filter driver (above the disk class driver), as illustrated in Figure 5.2.

The I/O operations shown in Figure 5.2 operate at a file system level to begin with, as denoted by the path marked with the number 1 in the figure. The NTFS file system driver manages the mapping of file data to disk blocks; subsequently, the I/O operation is done at a disk block level, below the NTFS file system, as denoted by the path marked with the number 2. Microsoft conveniently ships the diskperf.sys filter driver as part of the Windows Driver Development Kit (DDK), which is exactly such a driver. Several backup vendors have started with this sample as their building block for a snapshot solution.

The third approach is to take a snapshot of the data and back up the snapshot while the applications unknowingly continue using the original volume. The snapshot may be created by means of a variety of hardware or software solutions. This is the approach Microsoft favors with Windows Server 2003.

5.3 Backup Classifications

Various types of backup schemes exist, and they can be categorized in different ways. In an actual data center, one typically uses multiple types of backups. In short, the categorization of backups should not be taken to be mutually exclusive. Backups can be classified on the basis of

- Architecture
- Functionality
- Network infrastructure

Sections 5.3.1 through 5.3.3 take a look at each of these types of classification.

5.3.1 Backup Classifications Based on Architecture

One way of classifying backups is based on the architecture. That is, backups are classified in terms of the objects they deal with and the amount of awareness the backup application has of these objects. The available types of architecture-based backups, described in Sections 5.3.1.1 through 5.3.1.3, are

- Image- or block-level backup
- File-level backup
- Application-level backup

5.3.1.1 Image- or Block-Level Backup

The backup application in this case deals with blocks of data. Typically, this kind of backup scheme needs all applications on the server to cease accessing the data that is being backed up. The application opens the disk to be backed up as a raw disk (ignoring the file locations) and literally does logical block-level read and write operations.

The advantages of this kind of backup are that the backup and restore operations are very fast, and it can be a good disaster recovery solution. One disadvantage is that applications and even the operating system cannot access the disk while the backup or restore is happening. Another disadvantage is that image-level backups of a sparsely populated volume can result in a lot of unused logical blocks being copied for the backup. Some backup applications provide the logic necessary to detect and skip unused logical blocks. These are called *sparse image backups*.

Finally, it is hard to retrieve just a particular file or a few files rather than restore all the data to a disk. To do so, the restore software must understand the file system metadata as it exists on the tape, retrieve this metadata, and from there, compute the location on the tape where the data for the particular file resides. Some vendors provide the ability to restore a particular file from an image-level backup, but these offerings are available on only certain operating system platforms and not others. Some restore applications do attempt to optimize restoring a file from an image-level backup. These applications write file metadata such as the file allocation table for FAT16 to the tape.

The version of NTFS included with Windows 2000 already keeps all metadata in files—for example, the bit map that represents logical block allocation. The restore application locates the required metadata. From this the software calculates the positions on tape of each of the required

logical data blocks for the file being restored. The tape is then spooled in one direction, and all the relevant portions of the tape are read while the tape is moving in a single direction, thus providing the file data for restoration. The tape is not moved forward and backward at all, so not only is the restore time reduced, but the life of the tape is extended as well. Legato Celestra is one example of such a backup application.

Note that sometimes the choice of backup is limited. Consider the case in which a database uses a raw disk volume (without any kind of file system on that volume). In this case the only two choices are an image-level backup or an application-level backup (the latter is described in Section 5.3.1.3).

5.3.1.2 File-Level Backup

With this type of backup, the backup software makes use of the server operating system and file system to back up files. One advantage is that a particular file or set of files can be restored relatively easily. Another is that the operating system and applications can continue to access files while the backup is being performed.

There are several disadvantages as well. The backup can take longer, especially compared to an image-level backup. If a lot of small files are backed up, the overhead of the operating system and file and directory metadata access can be high. Also the problem of open files described earlier exists and needs to be solved.

Another disadvantage is related to security. This issue arises irrespective of whether the backup is made via a file-level backup or an image backup. The problem is that the restore is typically done through an administrator account or backup operator account rather than a user account. This is the only way to ensure that multiple files belonging to different users can be restored in a single restore operation. The key is that the file metadata, such as access control and file ownership information, must be properly set. Addressing the problem requires some API support from the operating system and file system involved (NTFS) to allow the information to be set properly on a restore operation. In addition, of course, the restore application must make proper use of the facility provided.

5.3.1.3 Application-Level Backup

In this case, backup and restore are done at the application level, typically an enterprise application level—for example, Microsoft SQL Server or Microsoft Exchange. The backup is accomplished via APIs provided by the application. Here the backup consists of a set of files and objects that together constitute a point-in-time view as determined by the application. The main problem is that the backup and restore operations are tightly associated with the application. If a new version of the application changes some APIs or functionality of an existing API, one must be careful to get a new version of the backup/restore application.

Applications either use a raw disk that has no file system associated with the volume/partition or simply have a huge file allocated on disk and then lay down their own metadata within this file. A good example of an application that takes this approach is Microsoft Exchange. Windows XP and Windows Server 2003 introduce an important feature in NTFS to facilitate restore operations for such files. The file can be restored via logical blocks, and then the end of the file is marked by a new Win32 API called SetFileValidData.

5.3.2 Backup Classifications Based on Functionality

Yet another way of classifying backup applications is based on the functionality that is achieved in the backup process. Note that a data center typically uses at least two and very often all types of the backups described in Sections 5.3.2.1 through 5.3.2.3: full, differential, and incremental.

5.3.2.1 Full Backup

In a **full backup**, the complete set of files or objects and associated metadata is copied to the backup media. The advantage of having a full backup is that only one media set is needed to recover everything in a disaster situation. The disadvantage is that the backup operation takes a long time because everything needs to be copied. Full backups are very often accomplished with the image- or block-level backup architecture.

5.3.2.2 Differential Backup

A **differential backup** archives *all changes since the last full backup*. Because differential backups can be either image block based or file based, this set of changes would represent either the set of changed disk

blocks (for image-based backup) or the set of changed files (for file-based backup). The main advantage of differential backup is that the backup takes a lot less time than a full backup. On the other hand, the disadvantage is that recovering from a disaster takes longer. A disaster recovery operation involves running at least two restore operations, one corresponding to a full backup and one corresponding to a differential backup.

With low-end storage deployed, file-based differential backups are used when the applications by nature tend to create multiple small files and change or create just a few of them since the last full backup. In addition, when low-end storage is deployed, file-based differential backups are not typically used with database applications, because database applications, by their very nature, tend to make changes in small parts of a huge database file. Hence a file-based backup would still have to copy the whole file. A good example here is Microsoft Exchange, which tends to make changes in small parts of a huge database file.

With high-end storage deployed, image-based differential backup can be used in any situation, including with database applications. The reason for this flexibility is that the high-end storage units can track a lot of metadata and thus quickly identify which disk blocks have changed since the last full backup. Thus, only this small number of disk blocks needs be archived, and the large number of unchanged disk blocks that are present in the same database file can be ignored. Even though the backup with high-end storage is more efficient, APIs that start the backup at a consistent point and allow the I/O to resume after the backup has been accomplished are still needed. The efficiency of high-end storage simply minimizes the time during which all I/O must be frozen while the backup is being made.

5.3.2.3 Incremental Backup

An **incremental backup** archives *only the changes since the last full or incremental backup*. Again, the obvious advantage is that this backup takes less time because items not modified since the last full or incremental backup do not need to be copied to the backup media. The disadvantage is that a disaster recovery operation will take longer because restore operations must be done from multiple media sets, corresponding to the last full backup followed by the various incremental backups.

In the absence of high-end storage, file-based incremental backup is used only when a different set of files is typically created or modified. With high-end storage that can provide the required metadata tracking, block-based incremental backup may be used.

### 5.3.3 Backup Classifications Based on Network Infrastructure

One way of classifying a backup scenario is based on the network topology used, and how that topology lends itself to achieving the best method for backing up the attached hosts. The network infrastructure–based backup types—direct-attached backup, network-attached backup, LAN-free backup, and server-free backup—are described in detail in Sections 5.3.3.1 through 5.3.3.4.

5.3.3.1 Direct-Attached Backup

Direct-attached backup was the first form of backup used, simply because it emerged in the era when storage devices were typically attached directly to servers. Despite the advent of network storage, direct-attached backup remains a very popular topology for backing up Windows-based servers. Direct-attached backup is illustrated in Figure 5.3.

The advantage of direct-attached backup is that it is fairly simple. An application running on the server reads data from the appropriate disk volume and writes it to the tape device. The biggest problems with direct-attached backup are these:

- Tape devices are duplicated (one per server that needs backup), which is expensive. To put it differently, sharing the tape device between servers is difficult.
- The total cost of ownership is high because you need more administrators doing tape backups using multiple tape devices.
- Storing multiple tapes can be confusing.
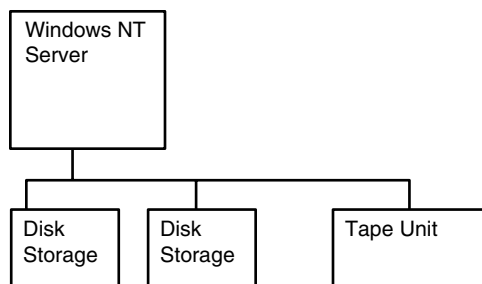
Figure 5.3 Direct-Attached Backup

■ Because the data on different servers is often duplicated, but slightly out of sync, the tape media reflects duplication of data with enough seemingly similar data to cause confusion.

■ Last, but not least, the server must be able to handle the load of the read/write operations that it performs to stream the data from disk to tape.
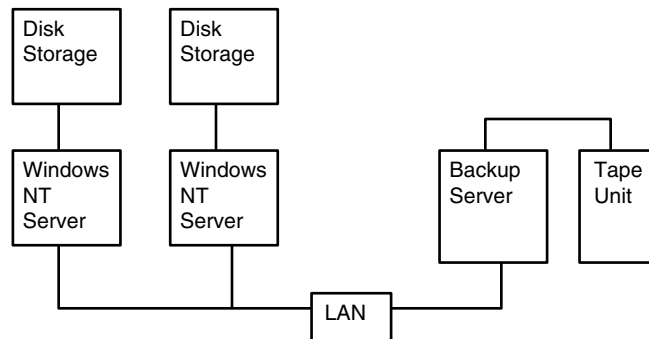
5.3.3.2 Network-Attached Backup

As Chapter 3 discussed, the era of direct-attached storage was followed by the client/server era with a lot of clients and servers sharing resources on a LAN. This LAN environment facilitated the possibility of having a server on the LAN with a tape backup device that could be shared by all the servers on the LAN.

Figure 5.4 shows a typical deployment scenario for network-attached backup. The left side of the diagram shows a couple of servers. These could be application or file-and-print servers, and there may be more than just a couple. The right side of Figure 5.4 shows a backup server with a tape unit attached. This tape device can be used for backing up multiple file-and-print or application servers. Thus, network-attached backup allows a tape device to be shared for backing up multiple servers, which can reduce costs.

The problems that network-attached backup introduced are these:

■ The backup operation consumes LAN bandwidth, often requiring careful segmentation of the LAN to put the backup traffic on a separate LAN segment.

Figure 5.4 Network-Attached Backup

■ Host online hours (i.e., operating hours) increased; that is, the amount of time servers needed to be available for transactions and user access grew. In addition, the amount of data on the servers (that needed to be backed up) started increasing as well.

Increasingly, these problems led to the use of backup requirements as the sole basis for network design, determining the exact number of backup devices needed, and the selection and placement of backup devices.

5.3.3.3 LAN-Free Backup

The advent of storage area networks introduced new concepts for backup operations. The new functionality is based on the fact that a storage area network (SAN) can provide a high bandwidth between any two devices and also, depending on the topology, can offer multiple simultaneous bandwidth capability between multiple pairs of devices with very low latencies. In contrast, using Fibre Channel loop topology with many devices—that is, more than approximately 30—cannot offer multiple simultaneous high-bandwidth connections with low latencies, because the total bandwidth of the loop must be shared among all attached devices.

Figure 5.5 shows a typical SAN-based backup application. Note the FC bridge device in the figure. Most tape devices are still non-FC based (using parallel SCSI), so a bridge device is typically used. In this figure, the Windows NT servers have a presence on both the LAN as well as the SAN.

The backup topology in Figure 5.5 has the following advantages:

■ The tape device can be located farther from the server being backed up. Tape devices are typically SCSI devices, although FC tape devices are now more readily available. This means that they can be attached to only a single SCSI bus and are not shared easily among servers. The FC SAN, with its connectivity capabil-ity, neatly solves this problem. Note that one still needs a solution to ensure that the tape device is accessed properly and with appro-priate permissions. Here are some possibilities:
  ■ One solution is to use zoning, allowing one server at a time to access the tape device. The problem with this solution is that zon-ing depends on good citizen behavior; that is, it cannot ensure compliance. Another problem with zoning is that it will not ensure proper utilization of a tape changer or multitape device.
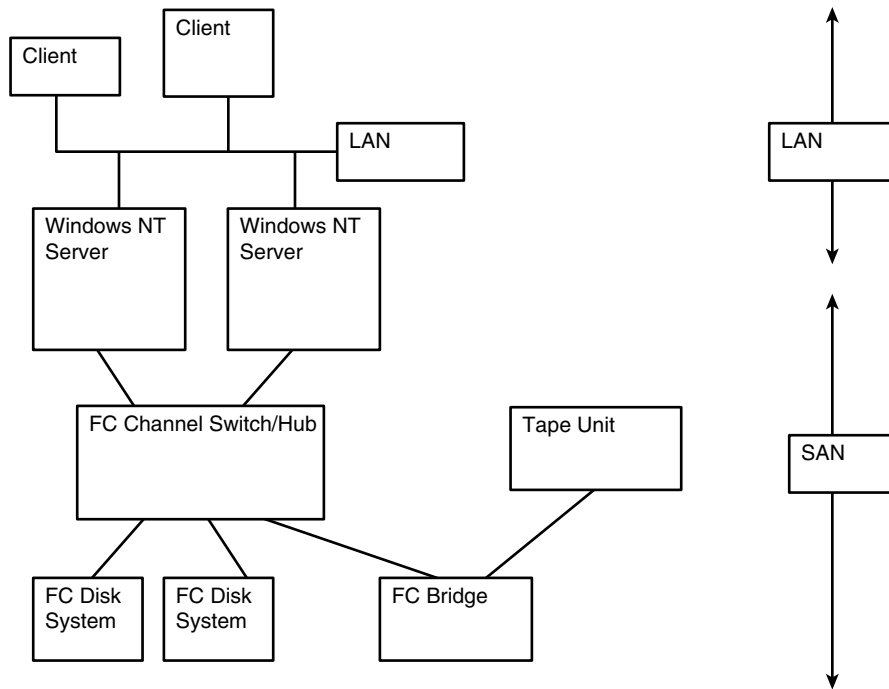
Figure 5.5 SAN-Based Backup

- ■ Another solution is to use the SCSI Reserve and Release commands.
- ■ Yet another solution is to have the tape device connected to a server, allowing for sharing of the tape pool by having special software on this server. Sharing of a tape pool is highly attractive because tape devices are fairly costly. IBM's Tivoli is one example of a vendor that provides solutions allowing the sharing of tape resources.
- ■ The backup is now what is often referred to as a **LAN-free backup** because the backup data transfer load is placed on the SAN, lightening the load on the LAN. Thus, applications do not get bogged down with network bandwidth problems while a backup is happening.
- ■ LAN-free backup provides more efficient use of resources by allowing tape drives to be shared.

- LAN-free backup and restore are more resilient to errors because backups can now be done to multiple devices if one device has problems. By the same token, restores can be done from multiple devices, allowing more flexibility in resource scheduling.
- Finally, the backup and restore operations typically complete a lot more quickly, simply because of the SAN's higher network speed.

5.3.3.4 Server-Free Backup

Server-free backup is also sometimes referred to as *serverless backup* or even *third-party copy*. Note that server-free backup is also usually LAN-free backup—LAN-free backup that also removes the responsibility of file movement from the host that owns the data. The idea is fairly simple, consisting of leveraging the Extended Copy SCSI commands.

Server-free backup began as an initiative placed before the Storage Networking Industry Association (SNIA) that evolved into the SCSI Extended Copy commands ratified by the International Committee for Information Technology Standards (INCITS) T10 Technical Committee (ANSI INCITS.351:2001, SCSI Primary Commands-2). Note that SCSI already supported a copy command, but the problem was that all SCSI devices required attachment to the same SCSI bus to use this command (the Copy command has since been made obsolete in the SCSI standards; see http://www.t10.org). The Extended Copy command adds features such that the data source and data destination may be on different SCSI buses and yet still be addressable because the syntax of the command allows for this.

In server-free backup, the backup server can remain relatively free to handle other work while the actual backup is accomplished by the data mover agent. The data is moved directly from the data source to the destination (backup media) (instead of being moved from the source to the backup server to the destination).

While appreciating the advantages of server-free backup, one should not forget that server-free restore is a very different issue. Server-free restore operations are still relatively rare; that is, backups made using server-free backup technology are very often restored via traditional restore technology that involves the use of a backup software server.

Server-free backup is illustrated in Figure 5.6. In the interest of simplicity, the figure shows the minimum number of elements needed to discuss server-free backup. In practice, however, SANs are much more complex. The figure shows a Windows server connected to an FC switch via an FC HBA. An FC-to-SCSI router is also present, to which
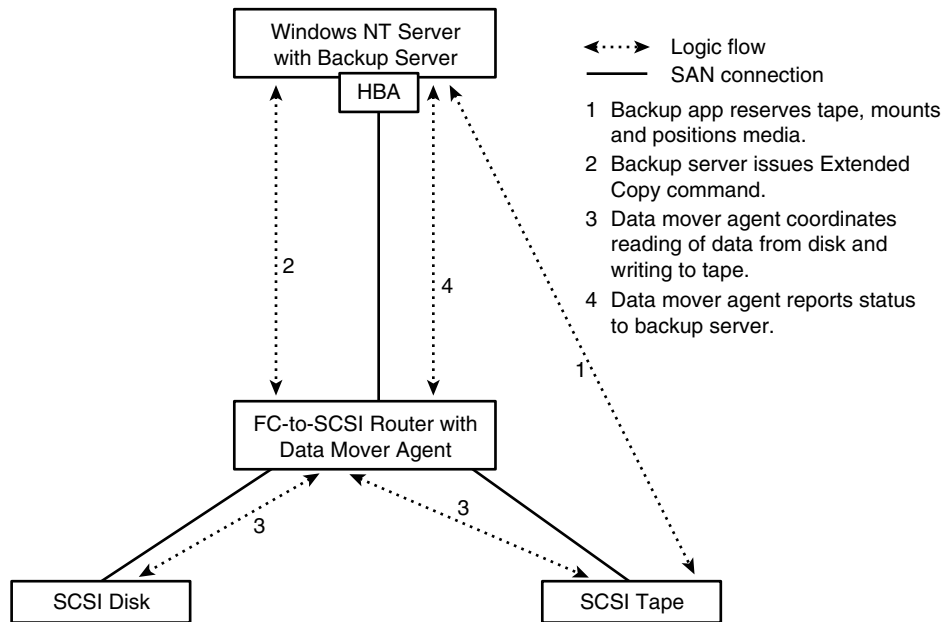
Figure 5.6 Server-Free Backup

are connected a SCSI tape subsystem and a disk device. The disk and tape devices need not be connected to the same router.

A backup server application on the Windows server discovers the data mover agent on the router, through Plug and Play. The backup application determines the details of the backup needs to be accomplished (disk device identifier, starting logical block, amount of data to be backed up, and so on). The backup server software first issues a series of commands to the tape device to reserve the tape device and ensure that the correct media is mounted and properly positioned. When that is done, the backup server software issues an Extended Copy command to the data mover, resident in the router, which then coordinates the movement of the required data. When the operation has been accomplished, the data mover agent reports the status back to the backup software on the Windows server.

Several different entities play a role in server-free backup architecture, including the data source, data destination, data mover agent, and backup server.

The **data source** is the device containing the data that needs to be backed up. Typically a whole volume or disk partition needs to be backed

up. The data source needs to be directly addressable by the data mover agent (described shortly). This means that storage devices connected directly to a server (or cases in which the server and the storage device have exclusive visibility) cannot be data sources for server-free backup because they cannot be addressed directly from outside the server.

The **data destination** is typically a tape device where the data is to be written. The device may also be a disk if one is backing up to disk instead of tape. Tape devices are typically connected to a fabric port to avoid disruption of the tape data traffic upon error conditions in other parts of the SAN. For example, if the tape were connected to an FC arbitrated loop, an error in another device or, for that matter, the occurrence of a device joining or leaving the loop, would cause loop reinitialization, resulting in disruption to the tape data traffic.

A **data mover agent** typically is implemented in the firmware of a storage router because the data mover agent must be able to act on the SCSI Extended Copy command, which is sent to the router in an FC packet. Switches and hubs that examine only the FC frame header are not readily suited to house data mover agents, though this may change in the future.

The data mover agent is passive until it receives instructions from a backup server. Most tapes connected to SANs are SCSI devices, so a storage router (that converts between FC and SCSI) is typically required and provides a good location for housing the data mover agent. Fibre Channel tapes are now appearing on the scene, and some vendors, such as Exabyte, are including data mover agent firmware in the FC tape device itself. In addition, native FC tape libraries are usually built with embedded FC-to-SCSI routers, installed in the library, providing the ability for the library to have a data mover built in. Note that the data mover agent can also be implemented as software in a low-end workstation or even a server. Crossroads, Pathlight (now ADIC), and Chaparral are some examples of vendors that have shipped storage routers with data mover agents embedded in the firmware. A SAN can have multiple data mover agents from different vendors, and they can all coexist.

Of course, to be usable, a data mover agent needs to be locatable (via the SCSI Report LUNs command) and addressable (the WWN is used for addressing) from the backup server software. The data mover agent can also make two simultaneous backups—for example, one to a geographically remote mirror to provide a disaster recovery solution—but the two commands must be built by the server that issued the third-party copy command.

The backup server is responsible for all command and control operations. At the risk of being repetitious, it is worthwhile noting all the duties of the backup server. The backup server software first ensures availability of the tape device, using appropriate SCSI Reserve and Release commands as appropriate. The backup server software then ensures that the correct tape media is mounted and positioned. It is also responsible for identifying the exact address of the data source and the data's location in logical blocks, as well as the amount of data that needs to be backed up. Once the backup server has all this information, it sends an Extended Copy command to the data mover agent. The data mover agent then issues a series of Read commands to the data source device and writes the data to the data destination.

Computer Associates, CommVault, LEGATO, and VERITAS are some examples of vendors that ship a server-free backup software solution. Storage router vendors that ship server-free functionality routinely work with backup independent software vendors (ISVs) to coordinate support because many of the implementations use vendor-unique commands to supplement the basic SCSI Extended Copy commands.

Note that although server-free backup has been around for a while, there is very little support for server-free restore.

5.3.3.5 The Windows Server Family and Server-Free Backup

A lot of the trade press and vendor marketing literature claims that a particular server-free backup solution is Windows 2000 compatible. It is worthwhile examining this claim in more detail to understand what it means. The following discussion examines each of the four components that constitute the elements of a server-free backup solution: data source, data destination, backup software server, and data mover agent.

In most cases a data mover agent outside a Windows NT server will not be able to directly address data sources internal to the Windows NT server. The HBAs attached to servers usually work only as initiators, so they will not respond to the Report LUNs command. If the Windows NT server is using a storage device outside the server—say, a RAID array connected to an FC switch—it will be visible to the data mover agent. So rather than saying that storage used by a Windows NT server cannot constitute the data source for a server-free backup, one needs to state that storage internal to a Windows NT server cannot constitute the data source.

Having the data destination internal to the Windows server is also not

possible, because the data destination also needs to be directly address-able from outside the Windows box (by the data mover agent).

Having the backup software run on the Windows server is certainly feasible. The HBA attached to the Windows server can issue a series of Report LUNs commands to each initial LUN (LUN 0) that it discovers. The backup software then enumerates the list of visible devices and LUNs, and checks which ones are capable of being third-party copy agents. The backup software would have to deal with some minor idio-syncrasies; for example, some products report extra LUNs that need to be used when Extended Copy commands are being issued. Many backup applications that use these devices go through an additional discovery process to verify the data mover's functionality.

The Windows NT SCSI pass-through (IOCTL) interface is capable of conveying the Extended Copy command to the data mover agent (from the Windows NT backup server). Windows NT does not have native support for data movers; Plug and Play can discover them, but drivers are required to log the data mover into the registry.

That leaves the last case—that is, whether a Windows NT server or workstation can be used to run the data mover agent software. One advantage is that such an agent would be able to address and access the storage devices visible to the Windows server. The backup server, how-ever, which might be outside the Windows NT box, would not be able to see these storage devices inside the Windows NT server. The data mover agent needs to be capable of acting as an initiator and target for SCSI commands. Because the HBA connected to the Windows NT server rarely acts as a target, the Extended Copy command may not get through to the data mover agent.

Note that in Windows NT, an application uses the SCSI pass-through interface (DeviceIoControl with an IoControlCode of IOCTL_SCSI_PASS_THROUGH or IOCTL_SCSI_PASS_THROUGH_DIRECT) to issue SCSI commands.

5.4 Windows 2000 Backup Utility

Windows 2000 ships with a backup program that is really a light version of the VERITAS Backup Exec program. The bundled backup utility in Windows 2000 is well integrated with other components of Windows 2000; for example, it integrates with the encrypting file system and also

hierarchical storage management. The backup utility offers support for backing up and restoring the encrypting file system (EFS) included with Windows 2000. Chapter 6 provides information about the EFS. The bundled backup utility is also well integrated with the Removable Storage Manager (RSM, described in Chapter 7). RSM provides support for operations essential to backup such as

- Enumerating media loaded in tape libraries
- Loading and ejecting media in tape libraries
- Providing secure access and preventing data corruption in the mounted media
- Performing housekeeping functions for managing media and tape libraries—for example, cleaning a tape drive or media library

Full-fledged backup utilities offer features that the bundled backup utility in Windows 2000 does not offer. Included are features such as

- Backup agents for enterprise applications such as SQL and IIS
- Support for backing up open files
- Higher performance
- Centralized administration capabilities, including a centralized database that includes a directory and control software for all backup devices and backup catalog(s)
- Support for Extended Copy or third-party copy data movers

Note that the backup utility bundled with Windows Server 2003 has the capability to back up open files as well, because the backup is snapshot based.

5.5 Techniques to Create a Volume Snapshot

A **snapshot** is simply a consistent point-in-time copy of a volume. Consistency in this case is defined as the original application's ability to recognize the data; for example, a Microsoft Exchange server sees a data set as a valid Exchange store, and a Microsoft SQL server recognizes a data set as a valid SQL server database. The data set being referred to here is typically a logical volume.

Volume snapshots are becoming increasingly popular and are used for various different reasons, such as the following:

- To back up a clone of a volume created with snapshot technology while the original volume continues to be used by applications. This is what Microsoft has done with the Windows XP volume shadow copy service, in which a clone of the original volume (at a given point in time) is constructed (assuming that enough free disk space is available) and used to perform backup operations.
- To create a clone of live data for use in a data mining operation.
- To create a clone of live data for testing a newer version of the application in a "live" environment.
- To create a clone of live data as part of a disaster recovery operation.

So how is a volume snapshot created? There are several possible ways, all of them essentially duplication of write operations. The only real difference in the solutions is where the write operations are duplicated. There are four possible locations:

1. **In hardware**. The first obvious way to create a volume snapshot is to have the volume mirrored by hardware and then split the mirror. With low-end hardware and purely host-based (software-based) volumes, each write operation is split into two duplicate write operations—one targeted at the original volume and the other targeted at the mirror. This approach is fairly resource intensive because both of the write operations must be completed before the write response can be sent. Write error operations also need to be handled. The advantage is that although it is resource intensive to keep a mirrored volume, breaking the mirror and thereby creating a volume snapshot is extremely fast. The high speed and reliability of hardware mirroring comes with a cost, though: the expense of the duplicate disks needed for the mirror. With high-end storage units, there is an extensive amount of per-track metadata because instead of the writes themselves being split, after the mirror split the writes are tracked in the metadata. Further, high-end storage units do not delay the mirrored writes, because the writes are declared complete once they hit the storage unit's battery-backed cache.

2. **Above the file system**. The second way to create a volume snapshot in the Windows Server family is to write a file system filter driver that sits above the file system driver—for example,

NTFS or FAT—and duplicate each IRP (I/O request packet) sent to the file system driver. This process is rather complex and cannot easily take advantage of snapshot technology provided in hardware. Examples of products that implement such filters above NTFS include St. Bernard Open File Manager, Vinca (now LEGATO) Open File Manager, and Cheyenne Open File Agent. Note that these products do not necessarily implement snapshots.

3. **In the file system itself**. Moving down the driver stack chain, the third way to implement snapshots is in the file system itself. Network Appliance's WAFL (Write Anywhere File Layout) and the Linux SnapFS file systems are examples of products that implement such functionality. Obviously these are not products that run on Windows NT. Writing a file system is fairly complex, and writing snapshot technology inside a file system makes it even more complex. There is no such method available in the native file systems included with Windows NT.
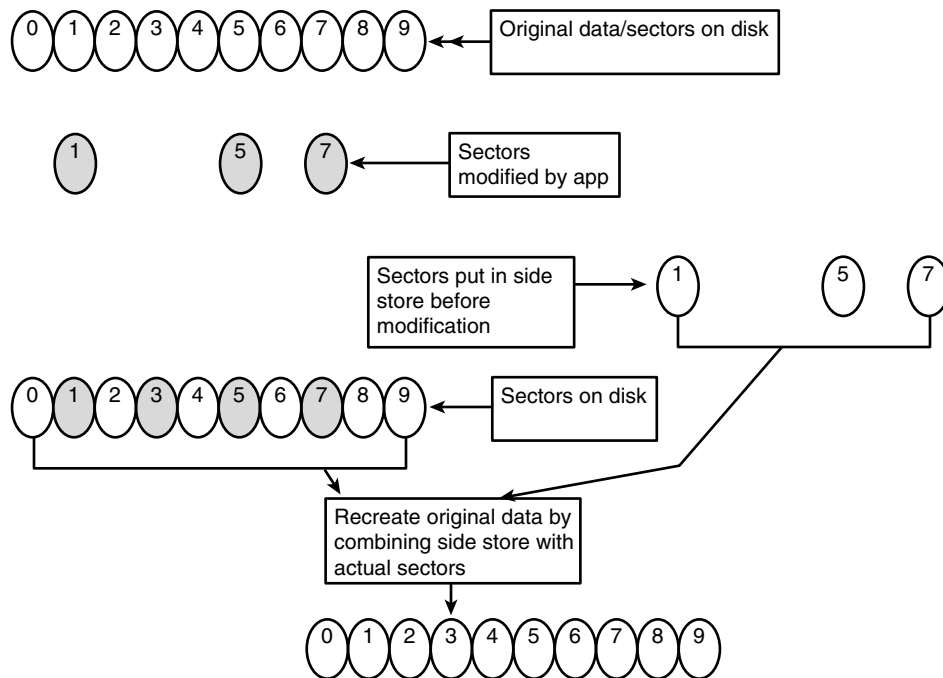
Figure 5.7 Copy-on-Write Snapshots

4. **Below the file system**. The fourth way of creating volume snap-shots in Windows NT is to have a filter driver beneath the file system use copy-on-write technology. The idea is that logical blocks that are changed by an application are first copied to a side store, before the application writes are committed to those logical blocks. This is illustrated in Figure 5.7. This technique is also referred to as *differential snapshot* because only the difference is stored (rather than a complete mirror image/copy being made).

5.6 Windows XP and Windows Server 2003 Volume Shadow Copy Service

With Windows XP and Windows Server 2003, Microsoft has implemented the volume shadow copy service to provide a framework that allows a coordinated and consistent point-in-time copy of disk volumes. For legal reasons, Microsoft has chosen to refer to the functionality as a *volume shadow copy*, which is really no different from the more popularly known term, *snapshots*. Volume shadow copy service has been implemented via a filter driver (called volsnap.sys) beneath the file system.

Microsoft makes a volume shadow copy software development kit available on a nondisclosure basis. This SDK appears to be aimed at three broad categories:

1. ISVs who might wish to develop volume shadow copy writers, including Microsoft Exchange, SQL Server, Oracle, SAP, Sybase, and others.
2. ISVs developing backup and storage management applications. These vendors would develop requestors for the volume shadow copy service.
3. Independent hardware and software vendors (IHVs and ISVs) developing hardware and software for backup, fault tolerance, and data integrity. Examples include VERITAS, EMC, and their competitors in this area. These vendors would develop snapshot providers.

When an application does not have code to support the snapshot service, the application data will still be backed up in a state that will be consistent to the same degree as if the system had failed to shut down gracefully. When an application has code to support snapshot services,

the application is expected to provide those services in a restore operation as well. This is logical because the application is now expected to furnish some data (such as what files need to be backed up, and a backup and restore methodology) when a backup is requested, and it is expected to interpret and act on this same data when presented with the data upon a restore operation.

The major advantage is that backup or restore, as it exists in Windows operating systems, is a tricky business that may not work reliably 100 percent of the time. The new snapshot service will facilitate total reliability while also enabling more complicated scenarios that until now were not possible in Windows.

The volume shadow copy architecture shipping with Windows XP and Windows Server 2003 consists of four types of modules, as shown in Figure 5.8:

   1. Writers
   2. Requestors
   3. Volume shadow copy service
   4. Providers

These modules are described in detail in Sections 5.6.1 through 5.6.4.

As far as the different modules that constitute the snapshot functionality are concerned, snapshot provider 2 has a kernel mode component,
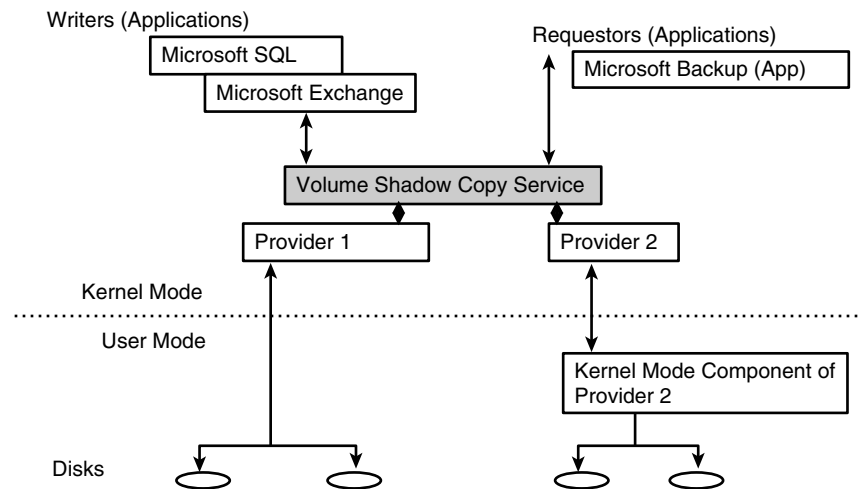
Figure 5.8 Volume Shadow Copy Architecture

and snapshot provider 1 does not. The snapshot provider service is shown in gray in Figure 5.8 to signify that Microsoft supplies this functionality and that vendors do not have to bother writing such a service. Microsoft also provides some of the other components shown, but their functionality is limited either in terms of working with a specific application or in terms of the range of features provided; hence vendors are expected and encouraged to write these components using the snapshot SDK.

Writers and providers need to implement a Component Object Model (COM) out-of-proc provider as described in the volume shadow copy SDK.[1] A provider is typically implemented as a finite state machine. The state machine transitions from one state to another upon receipt of an event that is generated by the shadow copy service. A provider will receive an event (generated by the shadow copy service), but exactly which event depends on the current state of the provider and whether an error has occurred. To state this differently, a provider has a preferred event that it expects to arrive, allowing it to transition to the next normally expected state. However, an error may occur and the provider may receive a different (from the expected) event, and the provider should be able to handle that as well.

The Microsoft shadow copy framework in Windows Server 2003 and Windows XP provides some core functionality needed for storage management, including the following:

- Defining the point in time at which the snapshot is triggered
- Providing synchronization services among applications, databases, operating systems, and file systems to flush cached data, suspend writers, create the snapshot, and provide notification to applications, databases, and operating system elements that they can resume normal mode of operation

---

1. For the benefit of readers unfamiliar with COM, suffice it to say that COM is an extremely important architecture in the Windows world that allows software to be constructed as a series of objects that interact with each other. These COM objects can be written in a variety of programming languages, such as C, C++, and Visual Basic, to name just a few. COM allows two interacting objects to be locally situated on the same machine or to be running on two separate machines altogether. Either these two interacting objects can both be running in the same process (if they are both on the same local machine), or they may be running in two separate processes. In the case of separate processes, COM provides mechanisms for interprocess communication. To summarize, an *out-of-proc* provider is simply a provider, implemented as a COM object, that runs in a separate process context.

- Providing a single API that can be used by backup/restore applications
- Providing a platform for managing the snapshot

The Microsoft shadow copy framework can handle a set of volumes for which a snapshot must be done as a single set. The only behavior supported is "all or none"; that is, if any one operation fails, all of them fail. The Microsoft shadow copy framework also issues a request (to the snapshot provider) to delete the snapshot when the requestor application is done with the snapshot. If the snapshot needs to be available later, the snapshot provider or requestor application must provide the necessary functionality. ISVs can build applications that can create, catalog, and manage multiple snapshots using the shadow copy architecture; however, such applications are not part of the Windows Server 2003 product available from Microsoft.

5.6.1 Writers

Snapshot **writers** are simply the applications that write data (applications simply reading data are not much of a problem). Examples of snapshot writers include Microsoft Exchange, Microsoft SQL Server 2000, SAP, and Oracle. Microsoft and third-party ISVs are both expected to develop snapshot writer–aware applications. Snapshot writers should be developed with the snapshot SDK. In particular, snapshot writers will receive two events from the snapshot service indicating that they should freeze all write activity, and a different thaw event indicating that write activity can now happen (hopefully signifying successful creation of the snapshot). There are other events that the writers can get from the snapshot service, and details can be obtained from the snapshot SDK. Because the applications can define the consistency of data they expect to achieve, they should be able to perform the quiesce operation fairly quickly.

Further, it is worthwhile understanding one big advantage of the volume shadow copy service architecture over traditional hardware-based snapshot mechanisms used with Windows 2000 and previous Windows versions. With these classic versions of creating a snapshot, the hardware-based mechanism had no means of determining the state of the application and of the operating system software and cache in particular. This meant that an appreciable percentage of the snapshots were inconsistent. Further, the only way of determining the health of the snapshot created required running an application data consistency checker, which could run for hours.

By comparison, the volume shadow copy–based architecture not only attempts to flush caches and hold writes, but it can also determine in a matter of a minute or so, whether the created snapshot is consistent or not. Once the thaw event has been signaled, the volume shadow copy service simply interrogates the writers as to whether they successfully managed to hold their write operations between the freeze and thaw events. If any of the writers failed, the snapshot is deemed to have failed.

Snapshot writers are also expected to provide data required for backup and restore—for example, what files need to be copied, what files need to be excluded, what collection of objects needs to be treated collectively as a single set. This data is stored by the snapshot service in a writer metadata document that is in XML format. Writers can also use this document to store data that is of interest to them. To restore data, the application simply hands the collection of data to the writer application to accomplish the restore operation.

Microsoft has announced that it will ship a SQL Server 2000 and Exchange writer, as well as writers for various other components of the Windows Server operating system. Microsoft is cooperating with ISVs to develop writers for other applications, including Active Directory.

5.6.2 Requestors

Requestors are typically backup applications that request the creation of a snapshot by making the appropriate API call to the volume shadow copy service provided by Microsoft. What's interesting here is the fact that the model remarkably simplifies some issues for the backup application writer. The backup application no longer has to solve the difficult issue of where the data to be backed up is, or what part of the data consists of application log files and what special treatment those files merit.

The appropriate writer (e.g., Microsoft SQL Server) is responsible for specifying the files and directories that should be included in the backup. Restore operations also become a lot simpler because the restore application again does not have to locate the data and figure out what files to pass to what API of the application (such as Exchange or SQL). The restore application simply hands the collection of data to the writer (application) and lets it accomplish the restore operation.

5.6.3 Volume Shadow Copy Service

The Windows NT **volume shadow copy service**, written by Microsoft, coordinates the activities for all the snapshot components. In particular, it provides the following:

- A single interface for backup applications or snapshot requestors to deal with, rather than multiple APIs from multiple applications.
- A single interface to produce, manipulate, and delete a crash-consistent volume snapshot or ghost volume.
- A single interface to allow different applications and snapshot providers to register and deregister as snapshot writers or snapshot providers.
- Synchronization and coordination among the various components to accomplish creation, deletion, or movement of snapshots, as well as backup and restore operations. The service prioritizes snapshot providers like this: Hardware providers have the highest priority, software providers are next, and the default Microsoft-provided snapshot provider has the lowest priority.

ISVs do not need to worry about writing a volume shadow copy service. Think of the snapshot service provided by Microsoft as being akin to a print spooler. You need only one print spooler. Some vendors (such as the provider vendors) simply need to write the equivalent of a printer driver. Other vendors need to write the equivalent of a printing application.

5.6.4 Providers

Snapshot **providers** are expected to be written by ISVs and IHVs to create, delete, and manipulate snapshots. As described earlier in this chapter, snapshot providers need to be written as a COM out-of-proc provider, via the snapshot SDK.

The provider may also have a kernel mode component—for example, a filter driver that is located between the file system and the Logical Disk Manager (LDM). This kernel mode functionality may also be optionally implemented in hardware instead. Note that even a hardware-based provider will still leverage the rest of the functionality provided by the framework—for example, definition of the point in time, I/O synchronization, and platform for building storage management applications, including backup, restore, and snapshot management applications.

One prime example of a snapshot provider is the volsnap.sys driver that ships with Windows XP and is also expected to ship with Windows Server 2003. This provider uses copy-on-write technology to create the necessary minimal data in a side store to be able to re-create the volume at a given point in time. The big assumption is that the required amount of free disk space is indeed available. This provider can handle NTFS, FAT32, and raw volumes on Windows Server 2003. However, this snapshot provider can provide only read-only snapshots and handle only one snapshot per volume. This limitation is in the provider itself and not the infrastructure. ISVs and IHVs can build richer functionality in their providers and writers if they so desire.

A complete description of all the events that are signaled to the snapshot provider via its COM provider is available in the snapshot SDK. A couple of important events are discussed here:

- **PreCommitSnapshot**. When the snapshot provider receives a PreCommitSnapshot event, it should start all I/O operations that take a long time—for example, synchronizing of the mirror.
- **CommitSnapshot**. When the provider receives a CommitSnapshot event, it should recognize that the snapshot service will time out the operation within 10 seconds. Hence the functionality here should be extremely fast. Further, until the snapshot is completed, Windows NT will wait to send any write operations to the volumes for which snapshots are being made. This means the provider should not do any I/O on this volume, and if it does, it should not expect the I/O to complete until the snapshot is either complete or is aborted.

Snapshot providers must provide certain mandatory functionality and may provide functionality that exceeds the mandatory functionality. The mandatory functionality is as follows:

- Providers are responsible for locating the storage needed to create the snapshot. The framework provided by Microsoft does not provide any such functionality.
- The provider must mount the snapshot in a different namespace and not have the snapshot mounted as a separate volume. An inspection of Windows XP shows that the Microsoft snapshot provider mounts the snapshot at \Device\HarddiskSnapshotX.

5.6.5 Windows NT I/O Subsystem Modifications

Though the modifications to the Windows NT I/O subsystem are not explicitly part of the snapshot environment, it is worthwhile noting that a fair amount of work was needed in the file systems, I/O stack, and file system filter drivers to accomplish a consistent and reliable point-in-time snapshot. In particular, Microsoft added two IOCTLs that all file systems and file system filter drivers need to implement:

1. **IOCTL_VOLSNAP_FLUSH_AND_HOLD_WRITES**, which should be both chained on and acted upon. The chaining is to allow drivers farther down the stack chain to also act on the IOCTL. The action consists of flushing and holding all file system metadata. Once all data has been flushed, no further writes should be issued until the outstanding IRPs issued to flush all data and metadata have completed.
2. **IOCTL_VOLSNAP_RELEASE_WRITES** also needs to be chained on and acted upon. This IOCTL indicates either successful completion of the snapshot or abandonment of the snapshot operation.

Some relevant portions of the Windows NT operating system have also been modified to trigger these IOCTLs at the appropriate time. Although Microsoft has already modified the file system and file system filter drivers that it ships to provide this functionality, ISV-shipped filter drivers need to do the same.

Section 5.8 describes an industry standard called Network Data Management Protocol (NDMP). But before that topic is discussed, it is worthwhile noting the relationship between volume shadow copy architecture in Windows XP/Windows Server 2003 and NDMP. The shadow copy architecture is a means of creating a clone of the data that needs to be backed up; NDMP can be used to move the data from the clone to tape or other backup media.

5.7 Windows-Powered NAS Devices and Snapshots

Microsoft offers a version of Windows NT that is sometimes referred to as "Embedded NT" and more often as the Server Appliance Kit, or SAK. This offering is based on Windows 2000, which does not have a volume

shadow copy service. For the benefit of original equipment manufactur-
ers that use the SAK to build NAS devices, Microsoft has licensed a snap-
shot solution and included it in the SAK. This snapshot product is the
Persistent Storage Manager (PSM) from Columbia Data Products. This
section provides a quick overview of PSM architecture and functionality.

PSM architecture is shown in Figure 5.9. PSM has a user mode com-
ponent that facilitates snapshot management, including initiation and
scheduling of snapshot creation. The snapshots are created via the ser-
vices of the PSM filter driver that is layered over the disk class driver as
shown in Figure 5.9.

PSM offers the ability to create multiple snapshots and manage them.
Snapshots can be created according to a schedule, and older snapshots
may be saved or written over. One can also "mount" the older snap-
shots and use them for backup or other purposes. Each snapshot has a
date and timestamp associated with it.
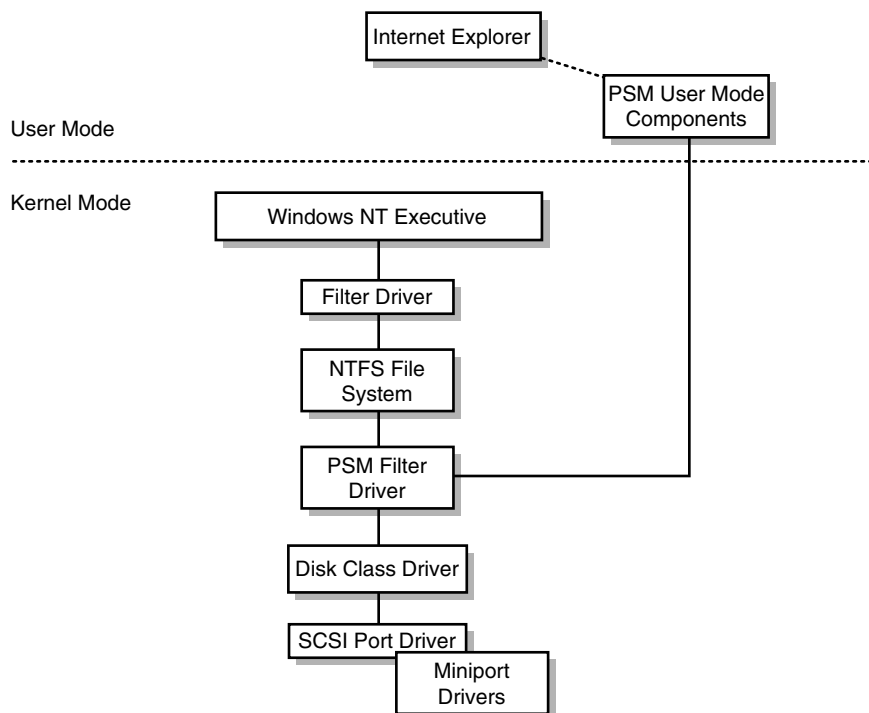
Figure 5.9 Persistent Storage Manager Architecture

5.8 Network Data Management Protocol

NDMP started as an effort primarily by Network Appliance and Intelli-guard (now part of LEGATO) to provide some enhanced functionality in backup/restore applications, such as the following:

- A means to reduce and at least isolate operating system–dependent portions of backup/restore software by dividing up the functional-ity in a modular way
- A standardized means of communication between modules
- A means to separate the movement of data and commands onto separate channels or even networks
- Multivendor software integration

NDMP is now positioned as an open-standard protocol aimed at standardizing backup and restore operations in NAS environments. In the future, NDMP may evolve—for example, to also involve mapping of third-party copy to NDMP.

NDMP can be run in a wide variety of network environments—for example, Ethernet, Gigabit Ethernet, Fibre Channel—as long as the network transport protocol used is IP (Internet Protocol). The NDMP server and agents communicate using IP, and the NDMP server then issues the appropriate block-level SCSI commands.

NDMP is NAS centric and defines a way to back up and restore some data from a device, such as a NAS appliance, on which it is difficult to install a backup software agent. In the absence of NDMP, this data is backed up as a shared drive on the LAN that is accessed via network file protocols such as CIFS or NFS.

NDMP offers several advantages:

- Interfaces are implemented by vendors that have core competen-cies and can concentrate on their core competencies.
- The interfaces are standardized, offering Plug and Play possibili-ties for modules from different vendors.
- NDMP can cut down the LAN bandwidth requirements by offer-ing data flow directly between primary and secondary storage without requiring data to flow to the backup software server and from there to the other device.
- NDMP can have the best of both worlds in that it can be con-trolled centrally via an NDMP control session, yet the data flow can still be local via NDMP data sessions.

5.8.1 NDMP Architecture

NDMP defines a standardized way to break up the backup and restore operations into multiple modules, with the idea that each vendor implements some of the modules. NDMP defines the following entities:

- A data mover agent
- NDMP services
- NDMP sessions

These are described in Sections 5.8.1.1 through 5.8.1.3.

5.8.1.1 Data Mover Agent

A **data mover agent** (**DMA**) is the primary backup application. It establishes NDMP sessions with NDMP service providers (described next) and orchestrates the sequence of steps required to establish a backup or restore operation. The data mover agent is also sometimes referred to as an **NDMP client**.

5.8.1.2 NDMP Services

NDMP defines services that may act as consumers or producers or allow one device to be both a consumer *and* a producer of data streams. NDMP v5 defines three kinds of services:

1. A **data service** that interfaces with the primary storage device (such as a NAS device). This service interacts with the volume or file system that is being either backed up or restored.
2. A **tape service** that interfaces with the secondary storage device, typically a tape device.
3. A **translator service** that performs transformations on data, including multiplexing multiple data streams into one data stream and vice versa.

5.8.1.3 NDMP Sessions

NDMP services interact with each other using NDMP interfaces. The result of this interaction is the establishment of an NDMP session that is termed a **control session** if the session is being used to achieve control

for the backup or restore operation or **data session** if the session is being used to transfer actual file system or volume data (including metadata). There is exactly one control session between each NDMP service and the data mover agent. Control sessions are always TCP/IP based; data streams can be TCP/IP or SAN based. Even though the data streams can be SAN based, the requirement that the control session be TCP/IP based pretty much dictates the presence of a LAN. Data streams may be either between the DMA and an NDMP service or directly between two NDMP services.

The DMA and services can be distributed on two or more computers. Figure 5.10 shows NDMP in operation with just two computers. The NDMP DMA contacts the NDMP server and orchestrates the data movement between the primary and secondary storage devices. The data sessions are established between the NDMP server and the data source and data destination.

Figure 5.11 shows a conceptual implementation of NDMP on Windows NT. The NDMP DMA establishes two NDMP control sessions, one with each NDMP server. The data flows directly between the two NDMP servers and is not "dragged" from one NDMP server to the NDMP DMA and from there to the other NDMP server.
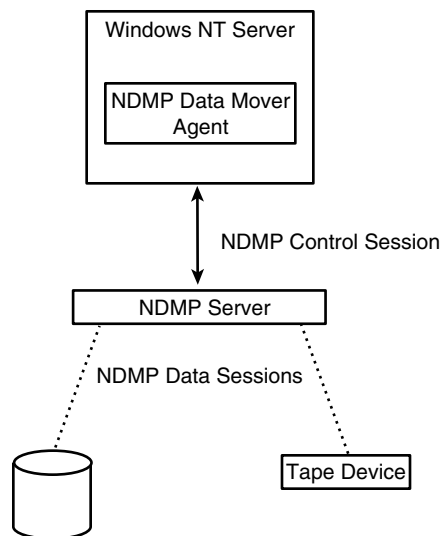
Figure 5.10 NDMP Architecture
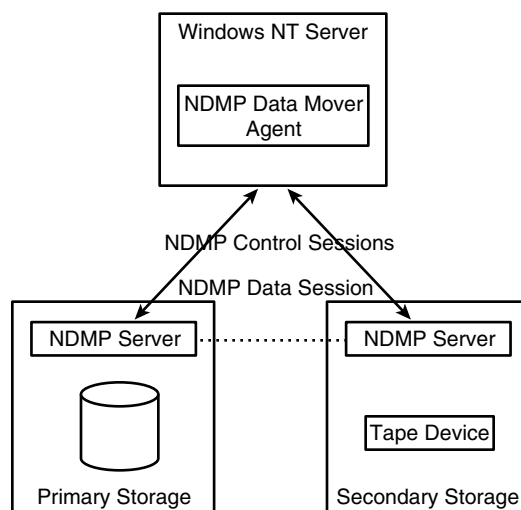
Figure 5.11 Conceptual Windows NT NDMP Implementation

5.9 Practical Implications

High-end storage, which can track metadata on a per-sector basis, allows backup/restore operations to be more efficient by allowing changes to be tracked very closely and having backup operations deal with just changed data.

To be certain that a snapshot is consistent from an application's point of view, it is essential to have the operating system (including file systems) and the application participate in the process of flushing the cache and temporarily suspending write operations while the snapshot is made. The volume shadow copy service shipping with Windows Server 2003 provides the needed operating system and file system support, as well as architecture for the needed application support. Microsoft Exchange and Microsoft SQL Server are two important applications that will take advantage of this architecture.

It remains to be seen how quickly other vendors adopt and support the volume shadow copy architecture.

5.10 Summary

Backup operations have evolved in terms of both user requirements and the technology used to accomplish backups. Usage requirements have dictated that backups be made more frequently, yet without disrupting application access to data. Backup operations evolved from stand-alone backups to backup operations happening across a LAN to backup operations happening in a SAN environment. One problem that backup applications need to solve is backing up open files being accessed by active programs while the backup is being done.

In addition, backup applications have had to deal with a multitude of APIs that are specific to an application version and specific to an operating system version. Yet another trend has been to create the initial backup from disk to disk, via a snapshot operation. Backup to tape is increasingly becoming a secondary backup operation, from the snapshot volume to tape.

The Windows volume shadow copy service provides an efficient way to create snapshots. The architecture provides for all important components, including major applications such as databases and messaging servers to participate in the snapshot creation. Microsoft provides only the infrastructure to create a snapshot. Software vendors may use this infrastructure to build an application that can create and manage multiple snapshots.

Once a snapshot has been created, a backup may be created from the snapshot. Standard protocols such as NDMP may be used to accomplish the backup operation.