

Storage Management and Backup Schemes for Broadcast Video

As videotape gradually disappears from the broadcast plant, storage in a digital file format becomes more important. In most cases, this is a combination of spinning disk and removable media, either data tape or DVD. With that comes the management of material as it moves from video server to nearline to archive and back again. The class of software products that perform this function are generally called hierarchical storage management or HSM systems. This is actually a misnomer in that HSM refers to a specific type of storage management rather than a product category.

HSM is often used as a catchall term for two categories of file management: storage management and file backup. Storage management really consists of four broad subcategories of products, each of which has advantages and disadvantages as it applies to on-air, production, and news environments. Specifically, the subcategories are: shared filesystems, HSM systems, disk extenders, and data movers. Related to but completely different than storage management is file backup, which also often gets lumped in with HSM. Within the category of file backup are subcategories of mirroring, simple backup, and disaster recovery.

Simply stated, storage management is the process of managing how data gets transferred to and from storage devices and how that data is represented to the application or applications using that data. The goal of storage management is to maximize the efficiency of the storage devices while making the use of these devices transparent to the application. File backup is the process of making sure that duplicate copies of important data is kept on different storage locations. Like storage management, the goal is to make the duplication process as automated and transparent as possible. Within a category, each of the choices of storage philosophy are usually mutually exclusive but between categories you can choose a storage management system which will work with a file backup system. For example, a disk extender and HSM would not work with each other but a disk extender and a simple backup system would work with each other.

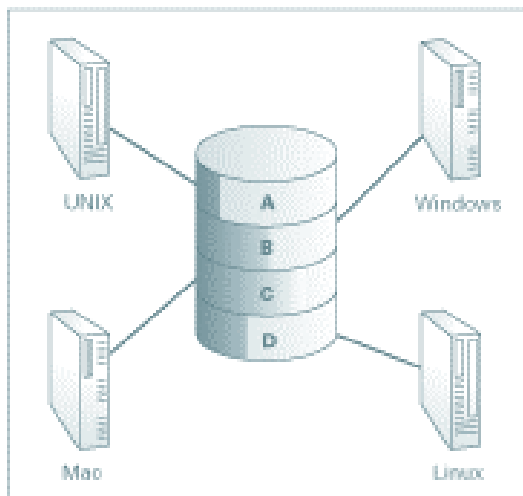
Storage Management Systems

The need to share information between the computer systems (which now are the basis for a lot of the broadcast plant) has driven the advances in storage management. We have gone from an environment where each system was an island of storage to an emerging set of tools and formats like MXF which allow sharing of storage and files between departments. The question is no longer can it be done but instead how to do it most efficiently.

Shared Filesystems

Shared filesystems came about in response to Storage Area Network (SAN) systems being introduced into a heterogeneous computer environment. In the IT world, they allow disparate operating systems to share files seamlessly. This allows common storage to be used for multiple application and computer types. This does not necessarily translate to a broadcast environment where video file formats and real time

processing require a more closed environment. Shared filesystems are more reasonable to implement in either a homogenous (i.e. single vendor) or a production environment where there is less of a real time requirement.



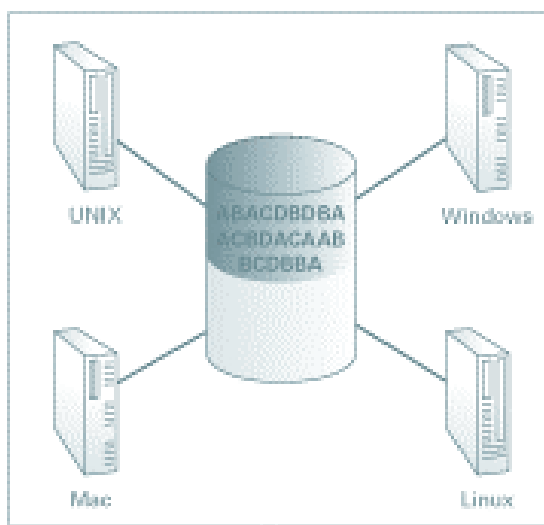
Graphic courtesy of ADIC

Figure 1

A SAN configuration allows computers to share storage physically as shown in Figure 1. This is different than Network Attached Storage (NAS) in that a SAN is directly attached to the individual computers rather than attached over a network as a network node. Generally, SAN implementations are slightly more difficult to manage but offer much greater throughput than a NAS configuration. In a standard SAN configuration, each computer accessing the shared storage is allocated a discrete portion of the storage pool for its own use.

This is where a shared filesystem comes into play. A shared filesystem, as shown in Figure 2, allows the common storage to be shared between computers and applications as a single entity rather than being parceled out to them individually. This is implemented with a translation layer which converts the native representation of each computer to a common representation. This means that files stored by computer A can be accessed and acted upon by computer B and vice versa.

This offers a number of advantages over discrete storage. First, it allows file sharing between different applications. This is particularly



Graphic courtesy of ADIC

Figure 2

important in a production environment where one set of applications may be used for effects and another set for finishing. Rather than cutting a tape to move material, the material is passed as a file. This concept will find its way into broadcasting also as more devices become MXF compliant with common essence. Second, bandwidth is greatly enhanced for the applications sharing files. Rather than moving files across a network, the material appears as direct attached storage to the application. Third, redundant file storage is eliminated. Server A and server B may need the same file at different times but common storage allows it to be stored just once. This is often seen at multi-channel facilities that share spots or program material between networks. Finally, file (i.e. media)

management is greatly simplified in that the material is stored in one place. If material needs to be purged, there is one place (possibly two if a backup is employed) to go to administer this function.

Within broadcast, shared filesystems are implemented with a number of video servers to allow several servers to share common storage. A real world example of a successful implementation of a shared filesystem is at Turner Entertainment Networks. In this case, an ADIC StorNext system was implemented as shared storage for Pinnacle MediaStream servers. TEN has 19 networks sharing 30,000 commercials. The StorNext system allows them to have one copy of each, available to all 19 networks at extremely high bandwidth. The amount of storage, network bandwidth, and administration needed are greatly reduced which translates into real dollar savings in both the up front expenditures and ongoing support of the system.

Shared filesystems are no panacea, however. As alluded to earlier, there are no common file sharing constructs within broadcast although MXF holds great promise. MXF is a great tool but it must be remembered that it is just a descriptive wrapper and does not guarantee common essence implementations between applications. Given that these hurdles can and will be overcome, a secondary issue is control. Sharing of files begs the question of who moves material from production to on-air and under what conditions. For example, having an editor move a spot to the on-air server without QC would not be a prudent thing to do. While the vision exists to share storage, implementation details still need to be worked out.

HSM Systems

HSM systems are designed to migrate material between different levels of storage based on rules defined by the user. This presents storage as a single unit to the application but is, in fact, a hierarchy of storage moving from high performance, high cost media to lower performance, lower cost media. The HSM software tracks files wherever they are in the hierarchy. Figure 3 shows this hierarchy of media and the devices commonly associated with each level of the hierarchy.

There is any number of criteria which can be used to initiate the migration of material within media. Most often, though, migration is based on the fill level of the primary storage that initiates a move to secondary storage. This is implemented by the storage administrator establishing high water and low water marks for migration. The high water mark is the percentage of disk fill level at which the migration algorithm kicks off. The low water mark is disk fill level at which migration ceases. For example, migration could take place when a disk is 80% full and cease when it is 15% full.

The most common migration rule in the IT world is called "least used". As the name implies, the files which have been used the least or not at all in the time period since the last migration are candidates to be migrated to a lower level of storage. Reverse migration also may be applied. If a file is accessed from deep storage multiple times, it may be promoted to higher level in the hierarchy. Other sub-rules can also be applied such as locking a certain set of files onto a level of storage for a period of time or until they are explicitly released.

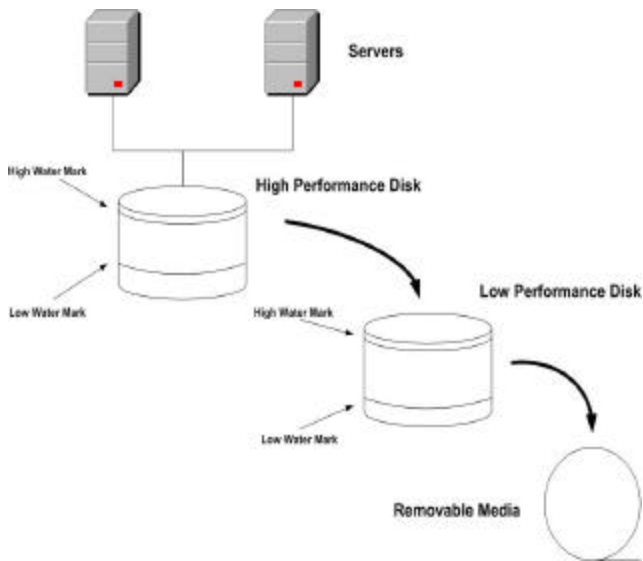


Figure 3

Once the rules are established, the migration occurs unattended. No application or human intervention is needed. This can be thought of as implicit migration as opposed to the explicit migration which is standard practice with data mover applications. The restore path is also implicit in that the application does not have to know or care where the file is stored as the hierarchy looks like one storage pool. HSM packages are common in the IT world with packages from IBM, Sun, EMC, StorageTek and others installed in thousands of sites managing

everything from email to financial records. Packages of this nature have been in use for decades.

HSM mirrors business practices in that information becomes less relevant over time and to a certain extent mirrors news operations in the relevancy of stories over time. It has limited relation to on-air operations, though, in that previous use of a program or spot may or may not have any relevancy to when that material will be needed again. If we look at the on-air video server as the top of the hierarchy and removable media as the bottom, a broadcaster does not want the spot scheduled to play in five minutes migrated to removable media. The movement of media needs to be more explicit.

A second problem with off the shelf HSM packages is that a video file is not a single entity but deconstructs to a number of audio and video files. Figure 4 shows the deconstruction of an Avid sequence. An HSM has no concept of the relationships between files and will apply migration rules indiscriminately with severe unintended consequences.

The most severe of these unintended consequences is that the video file cannot be reconstructed. Without the metadata to make sense of the files in place, the files themselves are useless.

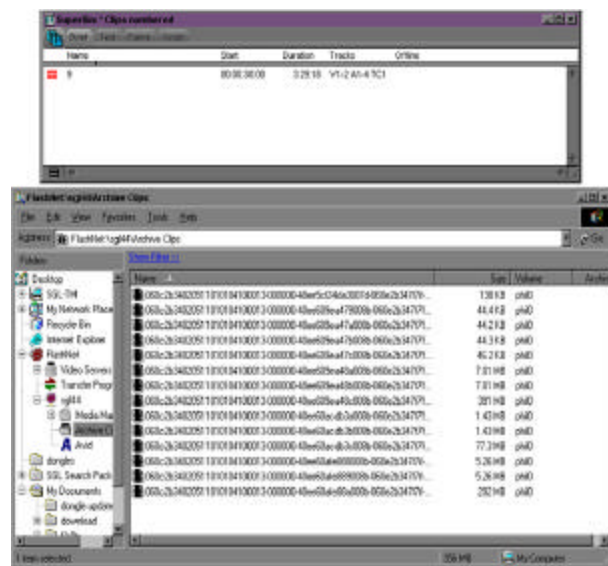


Figure 4

In real world examples, the only practical implementation is to isolate the hierarchy of storage from the application storage as shown in Figure 5. In this scenario, the video files are moved manually or under data mover control to the disk in the hierarchy and the migration applied from there. While workable, this tends to be somewhat clumsy and slow from the standpoint of multiple applications being used to accomplish the migration and the restore path from deep archive being back to the top of the hierarchy and then onto the storage of the application server.

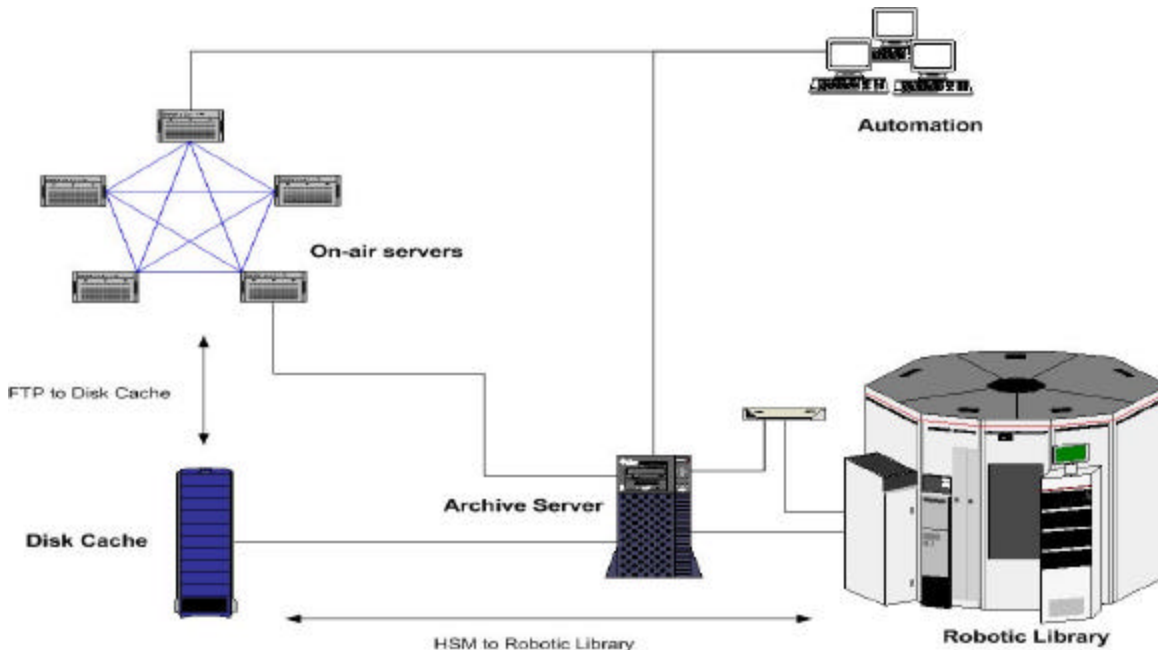


Figure 5

Disk Extenders

Disk extenders operate under the premise of making removable media look like a part of the spinning disk system. The robotic system looks like just another drive letter. This is implemented with a disk subsystem in the front end that provides the first level cache. The disk extender keeps a “stub” of the file on disk and moves the majority of the file off to cheaper removable media. The file looks like it is still on disk and when the application requests a file, the data blocks on removable media are restored to the cache and then transferred to the application. Figure 6 illustrates the data flow.

The migration associated with a disk extender product is initiated by a high water mark much like an HSM. The concepts are very similar with the difference being in the implementation details. HSM systems are usually implemented in self-contained systems rather than integrating with third party applications as is the case with disk extenders. A number of data storage companies offer disk extenders including ADIC, EMC and CA.

The concept of making a library look like removable media is very attractive to application developers looking for expanded storage. The disk/removable media combination looks like a drive letter that the application is writing to. This makes

integration very easy because the application thinks it is just talking to a disk. This does have unintended consequences, though.

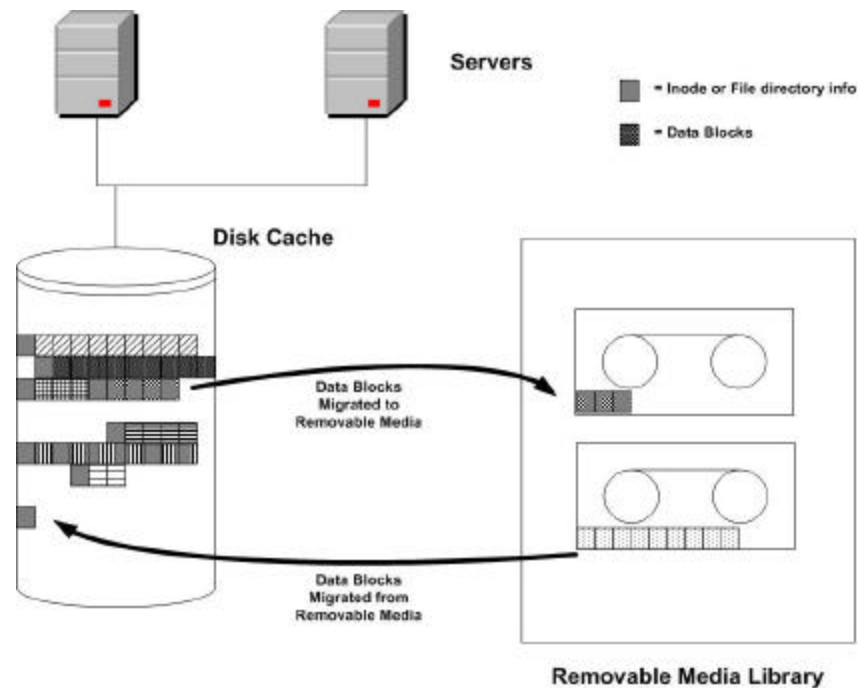


Figure 6

The biggest issue is the fact that the disk extender treats the removable media robot as a giant disk. This means that no media can be removed from the library and that files cannot span from one media to the next. The latter becomes more important in the realm of high bit rate video. If a video file is larger than a single piece of media, it cannot span to a second piece of media and, therefore, cannot be archived. The second big issue is latency on restoring video clips. When the material has migrated from the disk cache to removable media, the restore path is back to the disk cache instead of back to the target requested by the application. This means that the video file must be restored to the disk cache before being transferred back to the local storage on the application server (on-air or editing server). Migrated material with HSM or data mover applications are restored directly back to the target.

With that said, disk extenders do have a place in the broadcast plant for streaming media applications. In this case, the disk extender can be implemented on the native storage of the streaming media server. The requested stream can begin playing while the disk extender backfills the missing data blocks. For other applications, the overall architecture needs to be evaluated carefully before committing to a disk extender product.

Data Movers

Data movers operate as explicit migration tools rather than implicit migration tools as described above. This means that data movers wait to move any files from one level to another until they are told to do so by a controlling application. Controlling applications

include automation, editing, and asset management. Figure 7 shows a typical data mover configuration. A vast majority of the archive applications implemented in broadcast today are data mover applications from providers like Avalon, Front Porch, Masstech and SGL.

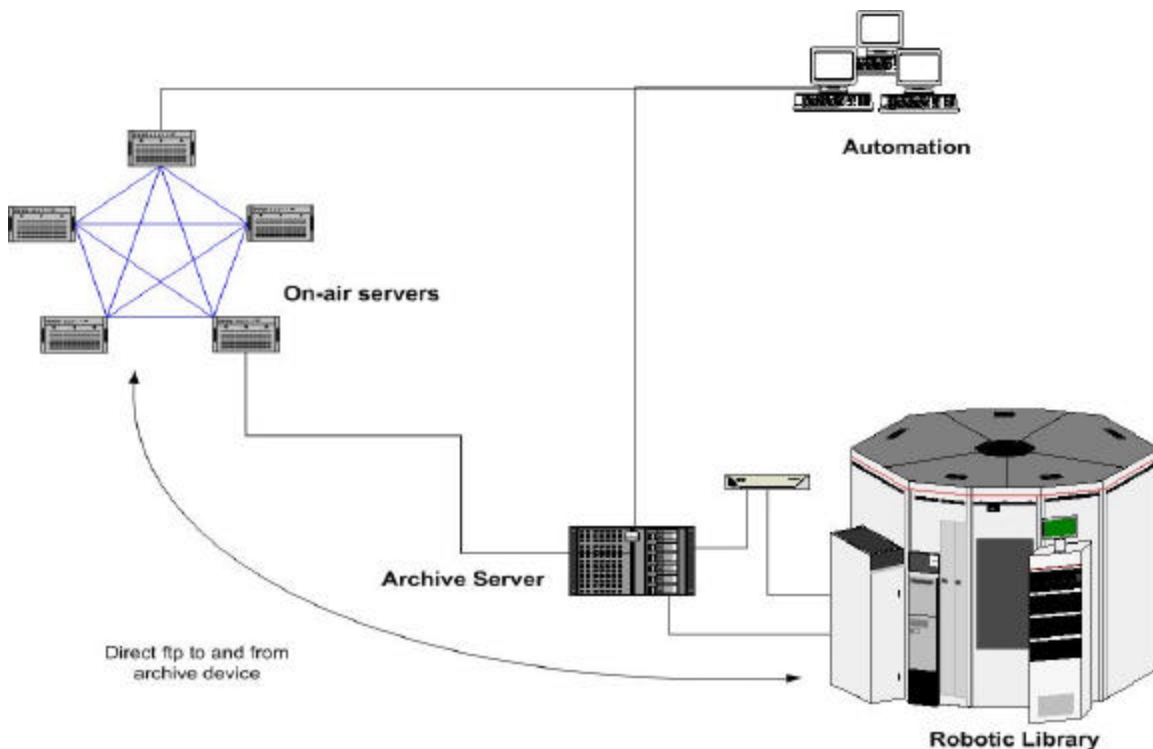


Figure 7

Explicit migration introduces a high level of determinism into the movement of video files. This is in line with common operations within a broadcast plant that tend to be very synchronous. Whether the controlling application is automation or news editing, the application knows what material is needed where and when and can control where the file needs to be. A typical operation on the part of the controlling application would be to archive video clip 1 from video server A to archive B through the data mover API. The movement of files can be to different levels of storage or multiple storage locations of the same type.

The advantage to data movers is also its downside. The movement of data is explicit and data movers have no inherent intelligence in moving data around to the correct location. For that reason, data mover providers are incorporating more HSM type functionality into their products. Specifically, the ability to offer multiple levels of storage as a single entity to the application with rules-based migration is becoming a standard feature of data movers. The rules in the case of broadcast may be different than just least used, though. It can include migration based on not only time but type of file and targets to include not only levels of storage but other equipment like transcoders. In the end, the standard architecture will be a data mover/HSM hybrid to bring in the best elements of both approaches.

One other point to note about data movers is that they are very specific to their industries, in this case, broadcast video. Data movers work very well with the automation, video server and editing systems in the market which have very specific interfaces. Broadcasters need the storage software to work in their environment and also have companies which are responsive to their needs. This means that the install base is of the hundreds rather than the thousands. The problem for the broadcaster is that this makes the data mover application more expensive because the development and marketing costs are spread over a smaller customer base.

File Backup Systems

File backup is the process of making sure there are multiple copies of important material in physically disparate locations. This often involves a combination of the three subcategories of mirroring, simple backup and disaster recovery. The reason why this often gets confused with HSM discussions is that the removable media library is viewed as a huge bit bucket that everything gets thrown into. A broadcaster may buy a storage management system assuming that material is all being migrated into the removable media library but this is not necessarily the case. File backup is a completely separate discipline from storage management and needs to be treated with a separate strategy.

Mirroring

This approach is already the predominant philosophy in the video server implementations. Even when servers each had their own storage as opposed to shared storage, servers were kept in lockstep by automation. With shared storage, mirroring is still a standard practice. Conceptually, mirroring is pretty simple although implementation can be complex. As the name implies, material in location A is copied in location B whether that is two separate servers with their own storage or shared storage between servers. It applies to both on-air and production systems.

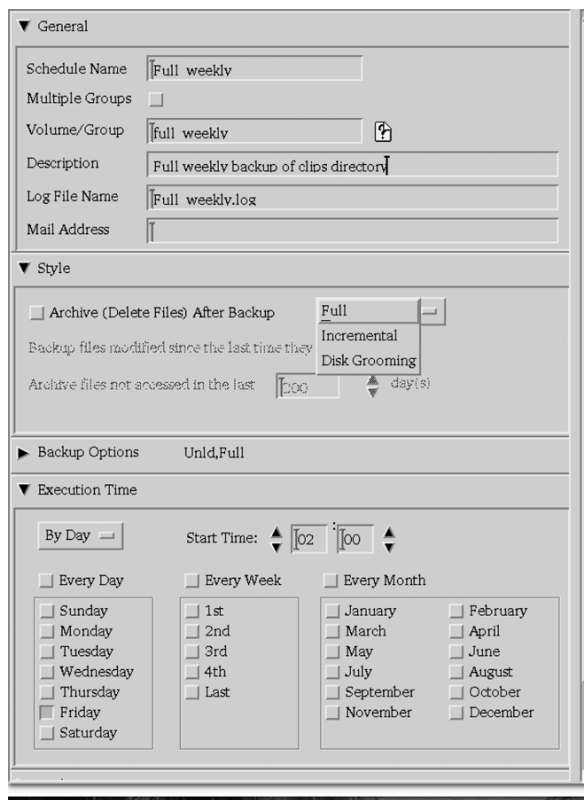
With RAID systems applied to storage and mirroring to store multiple copies, most broadcasters feel that their video files are adequately protected. This is a false sense of security for a couple of reasons. First, the larger, cheaper, faster disk drives which seem to double in size every year are a double edged sword. While overall, this is a good thing, the increased storage density also dictates that the time to rebuild a disk in a RAID system is longer. A RAID system is protected if there is not a second disk failure during the rebuild process. The longer rebuild time increases the amount of exposure for a possible second disk failure, which would cause the entire array to be corrupt.

Extending this to mirrored systems, the fact that files are mirrored does not ensure correct data. If a video file is corrupt on section A, it will also be corrupt on section B. This is particularly important to remember in the Windows-based, virus-laden world that we live in. This is particularly pronounced in production environments where producers and editors use web based music and graphics services. Even if the mirrored system is on an isolated in-house network, a producer can always bring an infected file into the system via floppy, memory stick, or CD.

For valuable material, the only way to ensure data integrity is to store the material electronically disconnected from the outside world. This means removable media whether it be DVD or data tape. This in turn dictates that simple file backup or disaster recovery must be implemented to make sure important material is secure. While this sounds straightforward, the data structures which make up a video file makes this more complicated than one might imagine.

Simple File Backup

File backup packages have been around almost as long as computers and are often included as part of the operating system purchase. These packages offer fundamentally the same functionality in that they run at a regularly scheduled time set by a system



administrator and back up the files on a computer system or network of computers to removable media in a standalone drive or robotic system.

Backup types include full and incremental backups. A full backup is creating a copy on removable media of everything on a computer, shared storage or individual folders on either. An incremental backup is a backup of only the files that have changed since the last backup. Common practice is to do a full backup at first and incremental backups thereafter. These backups are scheduled and unattended. They are usually run in off hours so as to not impact operations. Schedules can also be set to not run on certain days like holidays so as to not waste storage media if no one is using a system. Figure 8 shows a common scheduler implementation along with scheduled backup types.

Figure 8

Database backup is a completely different implementation. Because databases are commonly in use 24 hours a day, the data contained in the database is constantly changing and the database cannot be shut down for backup purposes. Until recently, a common practice was to shut down access to portions of the database while backup was performed. Now most relational databases have cell locking mechanisms to allow backup to be performed while the database is active. This is called hot database backup. This is important to broadcasters as more and more broadcast applications have a relational database embedded.

With backup being a well-known discipline and inexpensive (\$500 and less) computer backup packages on the market, why is broadcast backup so complicated and expensive? As discussed earlier with HSM systems, a video "file" is actually a collection

of files (see figure 4). The files can be backed up but cannot be reassembled into a meaningful format. For this single reason, most video systems do not have the ability to perform scheduled backups. The databases with these applications are backed up but the media itself is not. Backup of media files must be done explicitly in one of three ways. First, most broadcasters still have source material on videotape. Rather than backing up, the source material can be re-ingested if needed. This works fine but is labor intensive and may cost editing time in re-working the source. Also, direct download to servers eliminates videotape source being in the broadcast plant which in turn means the broadcaster would have to go to the material source to get material if lost.

Second, an administrator can explicitly decide which material on storage is important and needs to be backed up. This can be done with the automation or editing tools embedded with most systems but does not happen automatically. A recommended practice is to have a staff engineer responsible for this task as producers, editors, and operations personnel may not have the discipline to do backups in a timely manner.

Third, material can be consolidated into a flattened file and placed in a folder which a standard scheduled backup product can monitor. This also calls for some manual intervention in that a person has to decide which material to consolidate and move to the backup folder.

Overall, the industry is lacking in automated backup tools but it is expected that this problem will be addressed as the broadcast industry becomes totally file based.

Disaster Recovery

This is very similar to simple backup from the standpoint that the goal is to make a copy of the system in the event of some sort of catastrophic failure. The difference between this and simple file backup is that disaster recovery (DR) encompasses both the data files and the applications themselves. This involves backing up the data files, application files and registries on the computer to be backed up. In effect, it makes an exact copy of the whole machine such that it can be reloaded in the event of a failure.

The reason for doing this is more for time savings than any other reason. If a simple file backup is performed instead of DR, all the applications for all affected machines would have to be found and reloaded. In the event of multiple machines failures or virus issues, this could be very time consuming. For the same reasons, discussed earlier in file backup no real tools exist to do this automatically. Currently, the only approach is to "ghost" or make a copy of the machine as it exists at a point in time and make incremental copies of the data files themselves.

Conclusion

We are not there yet. The data mover providers have made significant progress in providing the migration tools needed to effectively manage storage. In parallel, common file formats will help make shared storage for all departments easier to implement as opposed to the isolated departmental storage available today. More disturbingly, backup

strategies are not even considered from a data management standpoint within the broadcast community by either the vendors or broadcasters themselves. The healthy paranoia of broadcasters in duplicating almost everything has been not been applied to media files themselves. Inevitably, it will take someone losing everything before the industry as a whole recognizes this as an issue and demands better tools from the vendor community.