Technology Primer: Backups in Civil Discovery

Craig Ball

Technology Primer: Backups in Civil Discovery
By Craig Ball

E-discovery lawyers think they know all they need to know about backup: "It's tapes, right? You send 'em to a vendor, you look at what they send back, you bill the client. Simple."

Backup is the Rodney Dangerfield of the e-discovery world. It don't get no respect. Or, maybe it's more like Milton, the sad sack with the red stapler from the movie, *Office Space*. Backup is pretty much ignored...until headquarters burns to the ground or it turns out the old tapes in the basement hold the only copy of the all-important TPS reports demanded in discovery.

Would you be surprised to learn that backup is the hottest, fastest moving area of information technology? Consider the:

- Migration of data to the "cloud" *(Minsk! Why's our data in Minsk?);*
- Explosive growth in hard drive capacities *(Two terabytes! On a desktop?);*
- Ascendency of virtual machines *(Isn't that the title of the next Terminator movie?);* and
- Increased reliance on replication *(D2D2T? That's the cute Star Wars droid, right?).*

If you don't fully understand how backup systems work, you can't reliably assess whether discoverable data exists or how much it will cost in terms of sweat and coin to access, search and recover that data.

The Good and Bad of Backups

Ideally, the contents of a backup system would be entirely cumulative of the active "online" data on the servers, workstations and laptops that make up a network. But because businesses entrust the power to destroy data to every computer user--including those motivated to make evidence disappear—and because companies configure systems to purge electronically stored information as part of records retention programs, backup tapes may be the only evidence containers beyond the reach of those who've failed to preserve evidence and those with an incentive to destroy or fabricate it. Going back as far as Col. Oliver North's deletion of e-mail subject to subpoena in the Iran-Contra affair, it's long been backup systems that ride to truth's rescue with "smoking gun" evidence.

Backup tapes can also be fodder for pointless fishing expeditions mounted without regard for the cost and burden of turning to backup media, or targeted prematurely in discovery, before more accessible data sources have been exhausted.

Grappling with Backup Tapes

Backup tapes are made for ***disaster recovery***, i.e., picking up the pieces of a damaged or corrupted data storage system. Some call backups "snapshots" of data, and like a photo, backup tapes capture only what's in focus. To save time and space, backups typically ignore commercial software programs that can be reinstalled in the event of disaster, so ***full backups*** typically focus on all *user created* data. ***Incremental backups*** grab just what's been created or changed since the last full or incremental backup. Together, they put Humpty-Dumpty back together again in a process called ***tape restoration*.**

Jargon Watch

disaster recovery
full backup
incremental backup
tape restoration
tape rotation
legacy tapes
replication
drive imaging
backup set
backup catalog
tape log
linear serpentine
helical recording
virtual tape library
D2D2T
RAID
striping
parity
hash value
single-instance storage
non-native restoration

Tape is cheap, durable and portable, the last important because backups need to be stored away from the systems at risk. Tape is also slow and cumbersome, downsides discounted because it's so rarely needed for restoration.

Because backup systems have but one legitimate purpose--being the retention of data required to get a business information system "back up" on its feet after disaster--a business only needs recovery data covering a brief interval. No business wants to replicate its systems as they existed six months or even six weeks before a crash. Thus, *in theory*, older tapes are supposed to be recycled by overwriting them in a practice called ***tape rotation*.**

But, as theory and practice are rarely on speaking terms, companies may keep backup tapes long past (sometimes *years* past) their usefulness for disaster recovery and often beyond the companies' ability to access tapes created with obsolete software or hardware. These ***legacy tapes*** are business records—sometimes the last surviving copy—but are afforded little in the way of *records management*. Even businesses that overwrite tapes every two weeks replace their tape sets from time to time as faster, bigger options hit the market. The old tapes are frequently set aside and forgotten in offsite storage or a box in the corner of the computer room.

Like the Delorean in "Back to the Future," legacy tapes allow you to travel back in time. It doesn't take 1.2 million gigawatts of electricity, just lots of cabbage.

Duplication, Replication and Backup

We save data from loss or corruption via one of three broad measures: duplication, replication and backup.

Duplication is the most familiar--protecting the contents of a file by making a copy of the file to another location. If the copy is made to another location on the same medium (e.g., another folder on the hard drive), the risk of corruption or overwriting is reduced. If the copy is made to another medium (another hard drive), the risk of loss due to media failure is reduced. If the copy is made to a distant physical location, the risk of loss due to physical catastrophe is reduced.

You may be saying, "Wait a second. Isn't backup just a form of duplication?" To some extent, it is, and certainly, it's the most common "backup" method used on a standalone machine. But true enterprise backup injects other distinctive elements, the foremost being that backups are not user-initiated but occur systematically, untied to the whims and preferences of individual users.

Replication is duplication without discretion. That is, the contents of one storage medium are periodically or continuously mirrored to another storage medium. Replication may be as simple as RAID 1 mirroring of two local hard drives or as elaborate as employing a distant data recovery center ready to roll in the event of a catastrophe.

Unlike duplication and replication, backup involves (reversible) alteration of the data and logging and cataloging of the stored data. Typically, backup entails the use of software or hardware that compresses and encrypts data. Further, backup systems are designed to support iteration, e.g., they manage the scheduling and scope of backup, track the content and timing of backup "sets" and record the allocation of backup volumes across multiple devices or media.

Major Elements of Backup Systems

Understanding backups requires an appreciation of the three major elements of a backup system: the source data, the target data ("backup set") and the catalog.

1. Source Data (Logical or Physical) Though users tend to think of the source data as a collection of files, backup may instead be drawn from the broader, logical divisions of a storage medium—"partitions," "volumes" and "folders" in the parlance of hard drive organization. ***Drive imaging,*** a specialized form of backup employed by IT specialists and computer forensic examiners, may draw from below the logical hierarchy of a drive, collecting a "bitstream" of the drive's contents reflecting the contents of the medium at the physical level. The bitstream of the medium may be stored in a single large file, but more often is broken into manageable, like-sized "chunks" of data to facilitate more flexible storage.

2. Backup Set (Physical or Logical, Full or Changed-File) A ***backup set*** may refer to a *physical* collection of *media* housing backed up data, i.e., the collective group of magnetic tape cartridges required to hold the data, or the "set" may reference the *logical* grouping of *files* (and associated catalog) which collectively comprise the backed up data.

Backup sets further divide between what can be termed “full backups” and “changed-file backups.” As you might expect, full backups tend to copy everything present on the source (or at least “everything” as defined in the full backup set) where changed-file backups duplicate items that have been added or altered since a full backup. The changed-file components further subdivide into incremental backups, differential backups and delta block backups. The first two identify changed files based on either the status of a file’s archive bit or a file’s created and modified date values. The delta block method examines the contents of a file and stores only the *difference*s between the version of the file contained in the full backup and the modified version. This approach is trickier, but it permits the creation of more compact backup sets and accelerates backup and restoration.

3. Backup Catalog vs. Tape Log Unlike duplication and replication, where generally no record is kept of the files moved or their characteristics, the creation and maintenance of a catalog is a key element of backup. The ***backup catalog*** tracks*, inter alia,* the source and metadata of each file or component of the backup set as well as the location of the element within the set. The catalog delineates the quantity of target media and identifies and sequences each tape or disk required for restoration. Without a catalog setting out the logical organization of the data as stored, it would be impossible to distinguish between files from different sources having the same names or to extract selected files without restoration of all of the backed up data.

Equally important is the catalog’s role in facilitating single instance backup of identical files. Multiple computers—especially those within the same company—store many files with identical names, content and metadata. It’s a waste of time and resources to backup multiple iterations of identical data, so the backup catalog makes it possible to store just a single instance of such files and employ placeholder “stubs” or pointers to track all locations to which the file should be restored.

Obviously, *lose* the catalog, and it’s tough to put Humpty Dumpty back together again.

It's important to distinguish the catalog--a detailed digital record that, if printed, would run to hundreds of pages or more--from the ***tape log,*** which is typically a simple listing of backup events and dates, machines and tape identifier. *See*, *e.g.,* the sample page of a tape log attached as Appendix A.

Backup Media: Tape and Disk-to-Disk

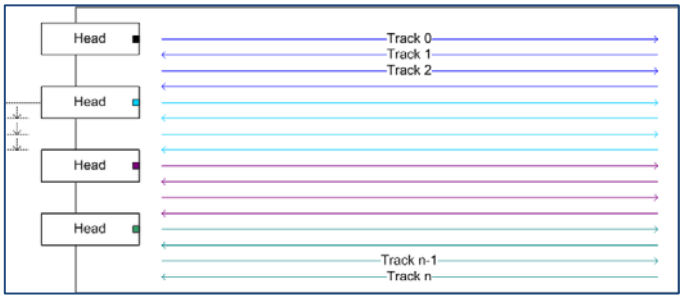
Tape Backup

Though backup tape seems almost antique, tape technology has adapted well to modern computing environments. The IBM 3420 reel-to-reel backup tapes that were a computer room staple in the 1970s and ‘80s employed 240 feet of half-inch tape on 10.5-inch reels. These tapes were divided into 9 tracks of data and held a then-impressive 100 megabytes of

information traveling at 1.2 megabytes per second. Today's common LTO-4 tapes are housed in a 4-inch square LTO cartridge less than an inch thick and feature 2600 feet of half-inch tape divided into 896 tracks holding 800 gigabytes of information traveling at 120 megabytes per second.

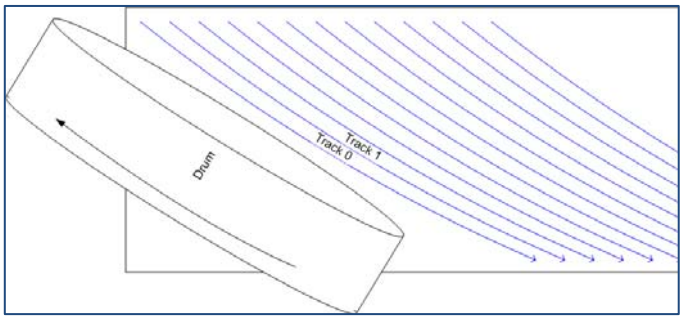
That's 100 times as many tracks, 100 times faster data transfer and *8,000 times greater* data storage capability.

Some readers may recall "auto-reverse" tape transport mechanisms, which eliminated the need to eject and turn over an audiocassette to play the other side. Many modern backup tapes use a scaled-up version of that back-and-forth or ***linear serpentine*** recording scheme. "Linear" because it stores data in parallel tracks running the length of the tape, and "serpentine" because its path snakes back-and-forth like a mountain road.[1] Sixteen of the LTO-4 cartridge's 896 tracks are read or written as the tape moves past the heads, so it takes *56 back-and-forth passes* or "wraps" to read or write the full contents of a single LTO-4 cartridge.



That's about *28 miles* of tape passing the heads!

An alternate recording scheme employed by SAIT-2 tape systems employs a ***helical recording*** system that writes data in parallel tracks running diagonally across the tape, much like a household VCR. Despite a slower transfer rate, helical recording also achieves 800GB of storage capacity on 755 feet of 8mm tape housed i n a compact cartridge like that used in handheld video cameras.



Why is Tape So Slow?

Clearly, tape is a pretty remarkable technology that's seen great leaps in speed and capacity.

Still, there are those pesky laws of physics.

[1] Or, if you prefer, "Serpentine!" like the evasive action to avoid gunfire Peter Falk urges on Alan Arkin in the 1979 screwball comedy, "The In-Laws."

All that serpentine shuttling back and forth over 28 miles of tape is a mechanical process. It occurs at a glacial pace relative to the speed with which computer circuits or even hard drives move data.

Further, backup restoration is often an incremental process. Reconstructing reliable data sets may require data from multiple tapes to be combined. Add to the mix the fact that as hard drive capacities have exploded, tape must store more and more information to keep pace. Gains in performance are offset by growth in volume.

How Long to Restore?

The big Atlanta-based tape house, eMag Solutions, LLC, recently weighed in concerning the difference between the time it *should* take to restore a backup tape considering just its capacity and data transfer rate versus the time it *really* takes considering the following factors that impact restoration:

- Tape format;
- Device interface, i.e., SCSI or fiber channel;
- Compression;
- Device firmware;
- The number of devices sharing the bus;
- The operating system driver for the tape unit;
- Data block size (large blocks fast, small blocks slow);
- File size (with millions of small files, each must be cataloged);
- Processor power and adapter card bus speed;
- Tape condition (retries eat up time);
- Data structure (e.g., big database vs. brick level mailbox accounts);
- Backup methodology (striped data? multi server?).

The following table reflects eMag's reported experience:

Drive Type	Native cartridge capacity	Drive Native Data Transfer Speed[2]	Theoretical Minimum Data Transfer Time	Typical Real World Data Transfer Time
DLT7000	35GB	3MB/sec	3.25 Hrs	6.5 Hrs
DLT8000	40GB	3MB/sec	3.7 Hrs	7.4 Hrs
LTO1	100GB	15MB/sec	1.85 Hrs	4.0 Hrs
LTO2	200GB	35MB/sec	1.6 Hrs	6.0 Hrs
SDLT 220	110GB	11MB/sec	2.8 Hrs	6.0 Hrs
SDLT 320	160GB	16MB/sec	2.8 Hrs	6.0 Hrs

The upshot is that it takes *about twice as long* to restore a tape under real world conditions than the media's stated capacity and transfer rate alone would suggest. Just

[2] " *How Long Does it Take to Restore a Tape*," eMag blog, 7/17/2009 at http://tinyurl.com/tapetime, Some of these transfer rate values are at variance with manufacturer's stated values, but they are reported here as published by eMag.

to generate a catalog for a tape, the tape must be read in its entirety. Consequently, it's not feasible to deliver 3,000 tapes to a vendor on Friday and expect a catalog to be generated by Monday. The *price* to do the work has dropped dramatically, but the *time* to do the work has not.

Common Tape Formats

Here at the close of 2009, the LTO tape format is the clear winner of the tape format wars, having eclipsed all contenders save the disk storage options that now threaten to (finally) extinguish tape as the leading backup medium. The LTO-5 format coming early in 2010 will natively hold 1.5 terabytes of data at a transfer rate of 140 megabytes per second.

But the dusty catacombs beneath Iron Mountain still brim with all manner of legacy tape formats that will be drawn into e-discovery fights for years to come. Here are some of the more common formats seen in the last 25 years and their characteristics:

Name	Format	A/K/A	Length	Width	Capacity (GB)	Transfer Rate (MB/sec)
DLT 2000	DLT3	DLT	1200 ft	1/2”	10	1.25
DLT 2000 XT	DLT3XT	DLT	1828 ft	1/2”	15	1.25
DLT 4000	DLT 4	DLT	1828 ft	1/2”	20	1.5
DLT 7000	DLT 4	DLT	1828 ft	1/2”	35	5
DLT VS-80	DLT 4	TK-88	1828 ft	1/2”	40	3
DLT 8000	DLT 4	DLT	1828 ft	1/2”	40	6
DLT-1	DLT 4	TK-88	1828 ft	1/2”	40	3
DLT VS-160	DLT 4	TK-88	1828 ft	1/2”	80	8
SDLT-220	SDLT 1		1828 ft	1/2”	110	10
DLT V4	DLT 4	TK-88	1828 ft	1/2”	160	10
SDLT-320	SDLT 1		1828 ft	1/2”	160	16
SDLT 600	SDLT 2		2066 ft	1/2”	300	36
DLT-S4	DLT-S4	DLT Sage	2100 ft	1/2”	800	60
DDS-1	DDS-1	DAT	60M	4mm	1.3	.18
DDS-1	DDS-1	DAT	90M	4mm	2.0	.18
DDS-2	DDS-2	DAT	120M	4mm	4	.60
DDS-3	DDS-3	DAT	125M	4mm	12	1.1
DDS-4	DDS-4	DAT	150M	4mm	20	3
DDS-5	DAT72	DAT	170M	4mm	36	3
DDS-6	DAT160	DAT	150M	4mm	80	6.9
M1	AME	Mammoth	22M	8mm	2.5	3
M1	AME	Mammoth	125M	8mm	14	3
M1	AME	Mammoth	170M	8mm	20	3
M2	AME	Mammoth 2	75M	8mm	20	12
M2	AME	Mammoth 2	150M	8mm	40	12

Name	Format	A/K/A	Length	Width	Capacity (GB)	Transfer Rate (MB/sec)
M2	AME	Mammoth 2	225M	8mm	60	12
Redwood	SD3	Redwood	1200 ft	1/2”	10/25/50	11
TR-1		Travan	750 ft	8mm	.40	.25
TR-3		Travan	750 ft	8mm	1.6	.50
TR-4		Travan	740 ft	8mm	4	1.2
TR-5		Travan	740 ft	8mm	10	2.0
TR-7		Travan	750 ft	8mm	20	4.0
AIT 1	AIT		170M	8mm	25	3
AIT 1	AIT		230M	8mm	35	4
AIT 2	AIT		170M	8mm	36	6
AIT 2	AIT		230M	8mm	50	6
AIT 3	AIT		230M	8mm	100	12
AIT 4	AIT		246M	8mm	200	24
AIT 5	AIT		246M	8mm	400	24
Super AIT 1	AIT	SAIT-1	600M	8mm	500	30
Super AIT 2	AIT	SAIT-2	640M	8mm	800	45
3570 B	3570b	IBM Magstar MP		8mm	5	2.2
3570 C	3570c	IBM Magstar MP		8mm	5	7
3570 C	3570c XL	IBM Magstar MP		8mm	7	7
IBM3592	3592	3592	609m	1/2”	300	40
T9840A	Eagle		886 ft	1/2”	20	10
T9840B	Eagle		886 ft	1/2”	20	20
T9840C	Eagle		886 ft	1/2”	40	30
T9940A			2300 ft	1/2”	60	10
T9940B			2300 ft	1/2”	200	30
T10000	T10000	STK Titanium		1/2”	500	120
T10000B	T10000B			1/2”	1000	120
Ultrium	Ultrium	LTO 1	609M	1/2”	100	15
Ultrium	Ultrium	LTO 2	609M	1/2”	200	40
Ultrium	Ultrium	LTO 3	680M	1/2”	400	80
Ultrium	Ultrium	LTO 4	820M	1/2”	800	120

Disk-to-Disk Backup

Tapes are stable, cheap and portable—a natural media for moving data in volumes too great to transmit by wire without consuming excessive bandwidth and disrupting network traffic. But strides in deduplication and compression technologies, joined by

drops in hard drive costs and leaps in hard drive capacities, have eroded the advantages of tape-based transfer and storage.

When data sets are deduplicated to unique content and further trimmed by compression, much more data resides in much less drive space. With cheaper, bigger drives flooding the market, hard drive storage capacity has grown to the point that disk backup intervals are on par with the routine rotation intervals of tape systems (e.g., 8-16 weeks), Consequently, disk-to-disk backup options once considered too expensive or disruptive are feasible.

Hard disk arrays can now hold months of disaster recovery data at a cost that competes favorably with tape, Thus, tape is ceasing to be a disaster recovery medium and is instead being used solely for long-term data storage; that is, as a place to migrate disk backups for purposes *other than* disaster recovery, i.e., archival..

Of course, the demise of tape backup has been confidently predicted for years, even while the demand for tape continued to grow. But for the first time, the demand curve for tape has begun to head south.

D2D (for Disk-to-Disk) backup made its appearance wearing the sheep's clothing of tape. In order to offer a simple segue from the 50-year dominance of tape, the first disk arrays were designed to emulate tape drives so that existing software and programmed backup routines needn't change. These are ***virtual tape libraries*** or *VTL*s.

As D2D supplants tape for backup, the need remains for a stable, cheap and portable medium for long-term retention of archival data--the stuff too old to be of value for disaster recovery but comprising the digital annals of the enterprise. This need continues to be met by tape, a practice that has given rise to a new acronym: ***D2D2T*****,** for Disk-to-Disk-to-Tape. By design, tape now holds the company's archives, which ensures the continued relevance of tape backup systems to e-discovery.

You can't talk about D2D without mentioning the primary enabling technology that made it possible for hard drive arrays to challenge and best tape on the fields of cost and reliability: RAID.

RAID Technology Enables D2D Backup

The lowest echelon of backup--geared to avoiding failures leading to data loss--is ***fault tolerance,*** typically achieved through redundancy. The most frequently encountered form of redundancy in computer systems, particularly servers, is the use of multiple hard

drives configured to work together in a RAID, an acronym for **R**edundant **A**rray of **I**ndependent **D**isks.[3]

Understanding RAID is helpful in selecting cost-effective preservation protocols in e-discovery and when estimating the potential for and cost of computer forensics. For example, knowing that a RAID 1 disk array creates a mirrored duplicate of all data on two separate, identical hard drives might enable you to save a client time, money and business disruption. Instead of hiring an expert to forensically image drives, an in-house IT person might achieve the same end by simply swapping out one of the two drives in the array.

Similarly, it's important to understand the redundancy and performance aspects of RAID in order to judge the potential for forensic examination of the server media. Although, at first blush, this information seems beyond the pale for legal counsel, it has a decisive impact on costly, consequential decisions made by the legal team.

RAIDs serve two ends: redundancy and performance. The redundancy aspect is obvious—two drives holding identical data safeguard against data loss due to mechanical failure of either drive—but how do multiple drives improve **performance**? The answer lies in splitting the data across more than one drive using a technique called **striping**.

Imagine you stored data on pieces of paper in your pants pocket. Since only one hand can go into the pocket at a time, the rate at which you can retrieve data is limited. But what if you could *divide* the data up between *two* pockets? Since you can now reach into both a left- and right-hand pocket at the same time, the rate at which you can retrieve data doubles. If you were an octopus and had eight hands and pockets…well, you get the idea.

A RAID improves performance by dividing data across more than one physical drive. The data stored on a RAID drive before a same-sized block is stored on the next drive is called the "stripe." By striping data across drives, each drive can deliver data ("reach into a pocket") at the same time, increasing the amount of information handed off to the processor.

But, when you divide information across two or more drives, the failure of any drive creates gaps--so many gaps, in fact, that all of the information may be lost forever. You gain performance, but lose redundancy.

The type of RAID just described is called a **RAID 0** configuration. It's popular among gamers and others trying to wring maximum performance from their systems; but it's so risky, you're unlikely to see it in a business setting.

---

[3] RAID originally meant **R**edundant **A**rray of ***I****nexpensive* **D**isks, but as RAIDs were often constructed of the most expensive, high-performance SCSI drives on the market, "inexpensive" didn't make much sense.

If RAID 0 is for gamblers, **RAID 1** is ideal for the risk averse. As noted, a RAID 1 completely duplicates everything on one drive to another, so that a failure of one drive won't lead to data loss by mechanical failure. Because a RAID 1 duplicates *everything*, it may duplicate a virus or data corruption as well. Thus, it only protects against drive failure, not bad behavior or user error. Two other downsides of RAID 1 are, it doesn't improve performance and it's expensive to dedicate two hard drives to storing the same information.

So, how do we secure the *performance* of RAID 0 and the *protection* of RAID 1?

You could create what's called a "RAID 0+1" and mirror the two striped drives to two more drives, but then you'd need four hard drives and end up with access to only half of their total storage capacity, Safe and fast, but not cost-efficient. The solution lies in a concept called ***parity,*** key to a range of other sequentially numbered RAID configurations. Of those other configurations, the one you most need to understand is called **RAID 5.**

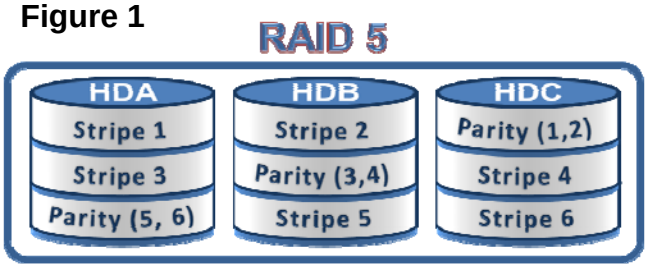
Parity

Consider the simple equation 5 + 2 = 7. If you didn't know one of the three values in this equation, you could easily solve for the missing value, i.e., presented with "5 + __ = 7," you can reliably calculate the missing value is 2. In this example, "7" is the ***parity value*** or checksum for "5" and "2."

The same process is used in many RAID configurations to gain increased performance by striping data across multiple drives while, at the same time, using parity values to permit the calculation of any missing values lost to drive failure. Any one of the three drives can fail, and we can use the remaining two to recreate the third.

Looking at Figure 1, data is striped across three hard drives, A, B and C. Hard Drive C holds the parity values for data stripe 1 on hard drive A and stripe 2 on hard drive B. It's shown as "Parity (1, 2)" in Figure 1. The parity values for the other stripes are distributed on the other drives. Again, any one of the three drives can fail and 100% of the data can be recovered. This configuration is called RAID 5 and, though it requires a minimum of three drives, it can be expanded to dozens of disks.

Figure 1



Essential Technologies: Compression and Deduplication

Along with big, cheap hard drives and RAID redundancy, compression and deduplication have made cost-effective disk-to-disk backup possible. But compression and deduplication are important for tape, too, and bear further mention.

Compression

The design of backup systems is driven by considerations of speed and cost. Perhaps surprisingly, the speed and expense with which an essential system can be brought back online after failure is less critical than the speed and cost of each backup. The reason for this is that (hopefully) failure is a rare occurrence whereas backup is (or should be) frequent and routine. Certainly, no one would seriously contend that restoring a failed system from a morass of magnetic tape is the fastest, cheapest way to rebuild a failed system. No, the advantage of tape is its relatively low cost per gigabyte to store data, not to restore it.

Electrons move much faster than machines. The slowest parts of any backup systems are the mechanical components: the spinning reels, moving heads and the human beings loading and unloading tape transports. One way to maximize the cost advantage and efficiency of tape is to increase the density of data that can be stored per inch of tape. The more you can store per inch, the fewer tapes to be purchased and loaded and the fewer miles of tape to pass by the read-write heads.

Because electrons move speed-of-light faster than mechanical parts of backup systems, a lot of computing power can be devoted to restructuring data in ways that it fits more efficiently on tape or disk. For example, if a horizontal line on a page were composed of one hundred dashes, it takes up less space to describe the line as “100 dashes” or 100(-) than to actually type out 100 dashes. Of course, it would take some time to count the dashes, determine there were precisely 100 of them and ensure the shorthand reference “100 dashes” doesn’t conflict with some other part of the text; but, these tasks can be accomplished by digital processors in infinitely less time than that required to spin a reel of tape to store the difference between the data and its shorthand reference.

This is the logic behind data compression; that is, the use of computing power to re-express information in more compact ways to achieve higher transfer rates and consume less storage space. Compression is an essential, ubiquitous technology. Without it, there would be no iPods, Tivos, YouTube, music CDs, DVD movies, digital cameras, Internet radio or pretty web pages.

And without compression, you’d need a whole lot more time, tape and money to backup a computer system.

While compression schemes for files tend to comprise a fairly small number of published protocols (e.g., Zip, LZH), compression algorithms for backup have tended to be proprietary to the backup software or hardware implementing them and to change from version-to-version. Because of this, undertaking the restoration of legacy backup tapes entails more than simply finding a compatible tape drive and determining the order and contents of the tapes. You may also need particular software to decompress the data.

Deduplication

Companies that archive backup tapes may retain years of tapes, numbering in the hundreds or thousands. Because each full backup is a snapshot of a computer system at the time it's created, there is a substantial overlap between backups. An e-mail in a user's Sent Items mailbox may be there for months or years, so every backup replicates that e-mail, and restoration of every backup adds an identical copy to the material to be reviewed. Restoration of a year of monthly backups would generate 12 copies of the same message, thereby wasting reviewers' time, increasing cost and posing a risk of inconsistent treatment of identical evidence (as occurs when one reviewer flags a message as privileged but another decides it's not). The level of duplication between ne backup to the next is often as high as 90%.

Consider, too, how many messages and attachments are dispatched to all employees or members of a product team. Across an enterprise, there's a staggering level of repetition.

Accordingly, an essential element of backup tape restoration is deduplication; that is, using computers to identify and cull identical electronically stored information before review. Deduplicating within a single custodian's mailboxes and documents is called ***vertical deduplication***, and it's a straightforward process. However, corporate backup tapes aren't geared to single users. Instead, business backup tapes hold messages and documents for multiple custodians storing identical messages and documents. Restoration of backup tapes generates duplicates within individual accounts (vertically) and across multiple users (horizontally). Deduplication of messages and documents across multiple custodians is called (not surprisingly) ***horizontal deduplication*****.**

Horizontal deduplication significantly reduces the volume of information to be reviewed and minimizes the potential for inconsistent characterization of identical items; however, it can make it impossible to get an accurate picture of an individual custodian's data collection because many constituent items may be absent, eliminated after being identified as identical to another user's items.

Consequently, deduplication plays two crucial roles when backup sets are used as a data source in e-discovery. First, deduplication must be deployed to eliminate the substantial identicality from one backup iteration to the next; that is, to eliminate that 90% overlap mentioned above. Second, deduplication is useful in reducing the cost and burden of review by eliminating vertical and horizontal repetition within and across custodians.

Modern backup systems are designed to deduplicate ESI *before* it's stored; that is, to eliminate all but a single instance of recurring content, hence the name, *single-instance storage*. Using a method called *in-line deduplication,* a unique digital fingerprint or *hash value* is calculated for each file or data block as it's stored and that hash value is added to a list of stored files. Before being stored, each subsequent file or data block has its hash value checked against the list of stored files. If an identical file has already been stored, the duplicate is not added to the backup media but, instead, a pointer or stub to

the duplicate is created. An alternate approach, called *post-process deduplication*, works in a similarly, except that all files are first stored on the backup medium, then analyzed and selectively culled to eliminate duplicates.

Data Restoration

Clearly, data in a backup set is a bit like the furniture at Ikea: It's been taken apart and packed tight for transport and storage. But, when that data is needed for e-discovery--it must be reconstituted and reassembled. It starts to take up a lot of space again. That restored data has to go *somewhere,* usually t*o* a native computing environment just like the one from which it came.

But the system where it came from may be at capacity with new data or not in service anymore. Historically, small and mid-size companies lacked the idle computing capacity to effect restoration without a significant investment in equipment and storage. Larger enterprises devote more stand-by resources to recovery for disaster recovery and may have had alternate environments ready to receive restored data, but those resources had to be at the ready in the event of emergency. It was often unacceptably risky to dedicate them, even briefly, to electronic discovery.

The burden and cost of recreating a restoration platform for backup data was a major reason why backup media came to be emblematic of ESI deemed "not reasonably accessible." But while the inaccessibility presumption endures, newer technology has largely eliminated the need to recreate a native computing environment in order to restore backup tapes. Today, when a lawyer or judge opines that "backups are not reasonably accessible, *per se*," you can be sure they haven't looked at the options in several years.

Non-Native Restoration

A key enabler of low cost access to tapes and other backup media has been the development of software tools and computing environments that support ***non-native restoration***. Non-native restoration dispenses with the need to locate copies of particular backup software or to recreate the native computing environment from which the backup was obtained. It eliminates the time, cost and aggravation associated with trying to reconstruct a sometimes decades-old system. All major vendors of tape restoration services offer non-native restoration options, and it's even possible to purchase software facilitating in-house restoration of tape backups to non-native environments.

Perhaps the most important progress has been made in the ability of vendors both to generate comprehensive indices of tape contents and extract specific files or file types from backup sets. Consequently, it's often feasible for a vendor to, e.g., acquire just certain types of documents for particular custodians without the need to restore all data in a backup. In some situations, backups are simply not that much harder or costlier to

deal with in e-discovery than active data, and they're occasionally the smarter *first* resort in e-discovery.

Going to the Tape *First*?

Perhaps due to the *Zubulake*[4] opinion or the commentary to the 2006 amendments to the Federal Rules of Civil Procedure,[5] e-discovery dogma is that backup tapes are the costly, burdensome recourse of last resort for ESI.

Pity. Sometimes backup tapes are the *easiest, most cost-effective* source of ESI.

For example, if the issue in the case turns on e-mail communications between Don and Elizabeth during the last week of June of 2007, but Don's no longer employed and Elizabeth doesn't keep all her messages, what are you going to do? If these were messages that should have been preserved, you could pursue a forensic examination of Elizabeth's computer (cost: $5,000-$10,000) or collect and search the server accounts and local mail stores of 50 other employees who might have been copied on the missing messages (cost: $25,000-$50,000).

Or, you could go to the backup set for the company's e-mail server from July 1 and recover just Don's or Elizabeth's mail stores (cost: $1,000-$2,500).

The conventional wisdom would be to fight any effort to go to the tapes, but the numbers show that, on the right facts, it's both faster and cheaper to do so.

Sampling

Sampling backup tapes entails selecting parts of the tape collection deemed most likely to yield responsive information and restoring and searching only those selections before deciding whether to restore more tapes. Sampling backup tapes is like drilling for oil: You identify the best prospects and drill exploratory wells. If you hit dry holes, you pack up and move on. But if a well starts producing, you keep on developing the field.

The size and distribution of the sample hinges on many variables, among them the breadth and organization of the tape collection, relevant dates, fact issues, business units and custodians, resources of the parties and the amount in controversy. Ideally, the parties can agree on a sample size or they can be encouraged to arrive at an agreement through a mediated process.

Because a single backup may span multiple tapes, and because recreation of a full backup may require the contents of one or more incremental or differential backup tapes, sampling of backup tapes should be thought of as the selection of data snapshots at intervals rather than the selection of tapes. Sensible sampling necessitates access to and an understanding of the tape catalog. Understanding the catalog likely requires explanation of both the business system hardware (e.g., What is

[4] Zubulake v. UBS Warburg, 217 F.R.D. 309 (S.D.N.Y. 2003

[5] Fed R. Civ. P. 26(b)(2)(B).

the SQL Server's purpose?) and the logical arrangement of data on the source machines (e.g., What's stored in the Exchange Data folder?). Parties should take pains to insure that each sample is complete for a selected date or interval; that is, the number of tapes shouldn't be arbitrary but should fairly account for the totality of information captured in a single relevant backup event.

Welcome to the Future

Harvard Law professor Lawrence Lessig recently observed, "We are not going back to the twentieth century. In a decade, a majority of Americans will not even remember what that century was like."[6] Yet, much of what even tech-savvy lawyers understand about enterprise backup systems harkens back to a century ten years gone.

Backup is unlikely to play a large role in e-discovery in the twenty-first century, if only because the offline backup we knew--dedicated to disaster recovery and accreted grandfather-father-son[7]--is fast giving way to data repositories nearly as accessible as our own laptops. The distinction between inaccessible backups and accessible active data stores will soon be just a historical curiosity, like pet rocks or Sarah Palin. Instead, we will turn our attentions to a panoply of electronic archives encompassing tape, disk and "cloud" components. The information we now pull from storage and extract tape-by-tape will simply be available to us--all the time--until someone jumps through hoops to make it go away.

Our challenge won't be in restoring information, but in making sense of it.

[6] Lawrence Lessig, *Against Transparency*, The New Republic, October 9, 2009.

[7] Grandfather-father-son describes the most common rotation scheme for backup media. The last daily "son" backup graduates to "father" status at the end of each week. Weekly "father" backups graduate to "grandfather" status at the end of each month. Grandfather backups are often stored offsite long past their utility for disaster recovery.

Appendix 1: Exemplar Backup Tape Log

Tape No.	Sess. ID	Host Name	Backup Date/Time	Size in Bytes	Session Type
ABC 001	37	EX1	8/1/2007 6:15	50,675,122,176	Exchange 200x
ABC 001	38	EX1	8/1/2007 8:28	337,707,008	System state
ABC 001	39	MGT1	8/1/2007 8:29	6,214,713,344	files incremental or differential
ABC 001	40	MGT1	8/1/2007 8:45	5,576,392,704	SQL Database Backup
ABC 001	41	SQL1	8/1/2007 8:58	10,004,201,472	files incremental or differential
ABC 001	42	SQL1	8/1/2007 9:30	8,268,939,264	SQL Database Backup
ABC 001	43	SQL1	8/1/2007 9:52	272,826,368	System state
ABC 005	2	EX1	8/14/2007 18:30	51,735,363,584	Exchange 200x
ABC 005	3	EX1	8/14/2007 20:35	338,427,904	System state
ABC 005	4	MGT1	8/14/2007 20:38	6,215,368,704	files incremental or differential
ABC 005	5	MGT1	8/14/2007 20:53	5,677,776,896	SQL Database Backup
ABC 005	6	SQL1	8/14/2007 21:06	10,499,260,416	files incremental or differential
ABC 005	7	SQL1	8/14/2007 21:38	8,322,023,424	SQL Database Backup
ABC 005	8	SQL1	8/14/2007 21:57	273,022,976	System state
ABC 002	207	NT1	8/15/2007 20:19	31,051,481,088	loose files
ABC 002	18	NT1	8/16/2007 8:06	47,087,616,000	loose files
ABC 014	9	EX1	8/17/2007 6:45	52,449,443,840	Exchange 200x
ABC 014	10	EX1	8/17/2007 8:53	337,969,152	System state
ABC 014	11	MGT1	8/17/2007 8:54	6,215,368,704	files incremental or differential
ABC 014	12	MGT1	8/17/2007 9:09	5,698,748,416	SQL Database Backup
ABC 014	13	SQL1	8/17/2007 9:22	10,537,009,152	files incremental or differential
ABC 014	14	SQL1	8/17/2007 9:47	8,300,986,368	SQL Database Backup
ABC 014	15	SQL1	8/17/2007 10:08	272,629,760	System state
ABC 003	16	NT1	8/18/2007 6:15	46,850,179,072	loose files
ABC 003	17	NT1	8/18/2007 9:26	44,976,308,224	loose files
ABC 004	19	NT1	8/21/2007 6:16	46,901,690,368	loose files
ABC 004	20	NT1	8/21/2007 9:30	44,742,868,992	loose files
ABC 009	30	EX1	8/22/2007 8:52	53,680,603,136	Exchange 200x
ABC 009	31	EX1	8/22/2007 11:01	348,782,592	System state
ABC 009	32	MGT1	8/22/2007 11:03	6,215,434,240	files incremental or differential
ABC 009	33	MGT1	8/22/2007 11:18	5,715,722,240	SQL Database Backup
ABC 009	34	SQL1	8/22/2007 11:31	10,732,371,968	files incremental or differential
ABC 009	35	SQL1	8/23/2007 4:08	8,362,000,384	SQL Database Backup
ABC 009	36	SQL1	8/23/2007 4:33	272,629,760	System state
ABC 011	44	NT1	8/23/2007 6:16	46,938,193,920	loose files
ABC 011	45	NT1	8/23/2007 9:32	44,611,403,776	loose files