UNIT-II

DATA BACKUP AND RECOVERY

You live in a world that is driven by the exchange of information. Ownership of information is one of the most highly valued assets of any business striving to compete in today' s global economy. Companies that can provide reliable and rapid access to their information are now the fastest growing organizations in the world. To remain competitive and succeed, they must protect their most valuable asset—data.

Backup Obstacles

The following are obstacles to backing up applications:
Backup window
Network bandwidth
System throughput
Lack of resources

backup window

The backup window is the period of time when backups can be run. The backup window is generally timed to occur during nonproduction periods when network bandwidth and CPU utilization are low. However, many organizations now conduct operations 7 days a week, 24 hours a day—effectively eliminating traditional backup windows altogether.

Network Bandwidth

Many companies now have more data to protect than can be transported across existing local area networks (LANs) and wide area networks (WANs). If a network cannot handle the impact of transporting hundreds of gigabytes of data over a short period of time, the organization' s centralized backup strategy is not viable.

System Throughput

Three I/O bottlenecks are commonly found in traditional backup schemes. These are

1. The ability of the system being backed up to push data to the backup server
2. The ability of the backup server to accept data from multiple systems simultaneously
3. The available throughput of the tape device(s) onto which the data is moved .

Lack of Resources

Many companies fail to make appropriate investments in data protection until it is too late. Often, information technology (IT) managers choose not to allocate funding for centralized data protection because of competing demands resulting from emerging issues such as e-commerce [2], Internet and intranet applications, and other new technologies.

The Future of Data Backup

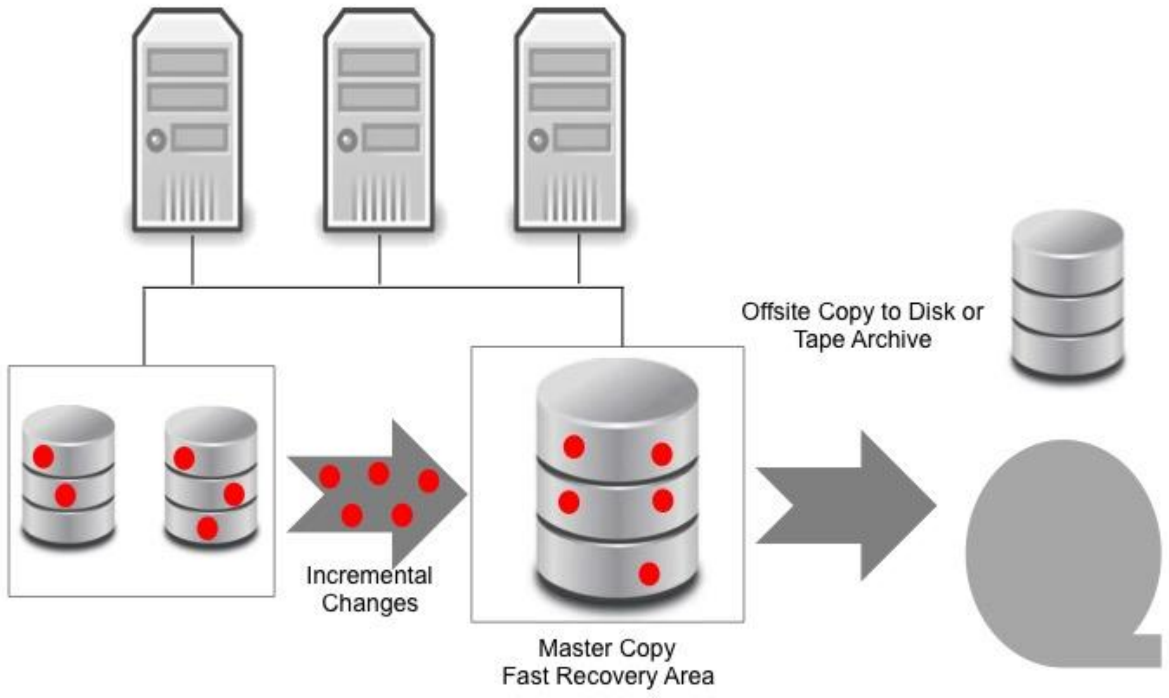
Successful data backup and recovery is composed of four key elements: the backup server, the network, the backup window, and the backup storage device (or devices). These components are highly dependent on one another, and the overall system can only operate as well as its weakest link. To help define how data backup is changing to accommodate the issues described earlier, let' s take a look at each element of a backup and recovery design and review the improvements being made.

The Backup Server

The backup server is responsible for managing the policies, schedules, media catalogs,and indexes associated with the systems it is configured to back up. The systems being backed up are called *clients*. Traditionally, all managed data that was being backed up had to be processed through the backup server. Conversely, all data that needed to be restored had to be accessed through the backup server as well. This meant that the overall performance of a backup or recovery was directly related to the ability of the backup server to handle the I/O load created by the backup process.

The Network Data Path

Centralization of a data-management process such as backup and recovery requires a robust and available network data path. The movement and management of hundreds or thousands of megabytes of data can put a strain on even the best-designed networks.


Offsite Copy to Disk or Tape Archive
Incremental Changes
Master Copy Fast Recovery Area

The Backup Window

Of all the parameters that drive the design of a backup application, one remains an absolute constant, and that is time. A backup window defines how much time is available to back up the network. Time plays an important role in choosing how much server, network, and resource support needs to be deployed. Today, most companies are managing too much data to complete backup during these ever shrinking backup windows.

In the past, companies pressured by inadequate backup windows were forced to add additional backup servers to the mix and divide the backup groups into smaller and smaller clusters of systems. However, the backup-software community has once again developed a way to overcome the element of time by using incremental backup, block-level backup, image backups, and data archiving.

Incremental Backup

Incremental backups only transfer data that has changed since the last backup. On average, no more than 5% of data in a file server changes daily. That means an incremental backup may only require 5% of the time it takes to back up the entire file system.

Block-Level Incremental Backup

Block-level incremental backups provide similar benefits as incremental backups, but with even more efficiency. Rather than backing up entire files that have been modified since the last backup, only the blocks that have changed since the last backup are marked for backup. This approach can reduce the amount of incremental data requiring backup nightly by orders of magnitude.

Image Backups

Image backups are quickly gaining favor among storage administrators. This type of backup creates copies, or *snapshots*, of a file system at a particular point in time. Image backups are much faster than incremental backups and provide the ability to easily perform a *bare bones* recovery of a server without loading the operating systems, applications, and the like. Image backups also provide specific point-in-time backups that can be done every hour rather than once a day.

Data Archiving

Removing infrequently accessed data from a disk drive can reduce the size of a scheduled backup by up to 80%. By moving static, infrequently accessed data to tape, backup applications are able to focus on backing up and recovering only the most current and critical data. Static data that has been archived is easily recalled when needed but does not add to the daily data backup requirements of the enterprise. This method also provides the additional benefit of freeing up existing disk space without adding required additional capacity.

THE ROLE OF BACKUP IN DATA RECOVERY

Many factors affect back-up:
Storage costs are decreasing.
Systems have to be online continuously.
The role of backup has changed.

Storage Costs Are Decreasing

The cost per megabyte of primary (online) storage has fallen dramatically over the past several years and continues to do so as disk drive technologies advance. This has a huge impact on backup. As users become accustomed to having immediate access to more and more information online, the time required to restore data from secondary media is found to be unacceptable.

Systems Have to Be Online Continuously

Seven/twenty-four (7 · 24) operations have become the norm in many of today' s businesses. The amount of data that has to be kept online and available (operationally ready data) is very large and constantly increasing. Higher and higher levels of fault tolerance for the primary data repository is a growing requirement. Because systems must be continuously online, the dilemma becomes that you can no longer take files offline long enough to perform backup.

The Role of Backup Has Changed

It' s no longer just about restoring data. Operationally, ready or *mirrored* data does not guard against data corruption and user error. The role of backup now includes the responsibility for recovering user error and ensuring that *good* data has been saved and can quickly be restored.

Issues with Today's Backup

Network backup creates network performance problems. Using the production network to carry backup data, as well as for normal user data access, can severely overburden today' s busy network resources. This problem can be minimized by installing
a separate network exclusively for backups, but even dedicated backup networks may become performance bottlenecks.
Offline backup affects data accessibility. Host processors must be quiescent during the backup. Backup is not host-independent, nor is it non disruptive to normal data access. Therefore, the time that the host is offline for data backup must be minimized. This requires extremely high-speed, continuous parallel backup of the raw image of the data. Even in doing this, you have only deferred the real problem, which is the time needed to restore the information. Restoration of data needs to occur at the file level, not the full raw image, so the most critical information can be brought back into operation first.
Live backups allow data access during the backup process but affect performance. Many database vendors offer *live* backup features. The downside to the live backup is that it puts a tremendous burden on the host. Redirection lives on the host, and journaling has to occur on the host. This requires consideration of local storage, host CPU cycles, and host operating system dependencies. Up to 50% of all host CPU cycles may be consumed during the backup process, severely impacting performance.
Mirroring doesn' t protect against user error and replication of *bad* data. Fully replicated online data sounds great, albeit at twice the cost per megabyte of a single copy of online data. However, synchronizing, breaking, and resynchronizing mirrors is not a trivial process and influences data access speeds while they are occurring. Also, duplicating data after a user has deleted a critical file or making

a mirrored copy of a file that has been corrupted by a host process doesn' t help. Mirroring has its place inbackup and recovery but cannot solve the problem by itself.

HIDING AND RECOVERING HIDDEN DATA

It is common knowledge that what is deleted from the computer can sometimes be brought back. Recent analysis of security implications of "alternative datastreams" on Windows NT has shown that Windows NTFS filesystem allows data hiding in alternative datastreams connected to files. These datastreams are not destroyed by many file wiping utilities that promise irrecoverable removal of information. Wiping the file means "securely" deleting it from disk (unlike the usual removal of file entries from directories), so that file restoration becomes extremely expensive or impossible .

For example, Linux has no alternative data streams, but files removed using /bin/rm still remain on the disk. Most Linux systems use the ext2 filesystem (or its journaling version, ext3 by Red Hat). A casual look at the design of the ext2 filesystem shows several places where data can be hidden [5].

Let' s start with the classic method to hide material on UNIX filesystems (not even ext2 specific). Run a process that keeps the file open and then remove the file. The file contents are still on disk and the space will not be reclaimed by other programs.

If an executable erases itself, its contents can be retrieved from /proc memory image: command "cp /proc/$PID/exe /tmp/file" creates a copy of a file in /tmp.

If the file is removed by /bin/rm, its content still remains on disk, unless overwritten by other files. Several Linux unerase utilities including e2undel attempt automated recovery of files. They are based on Linux Ext2fs Undeletion mini-HOWTO that provides a nice guide to file recovery from Linux partitions. Recovery can also be performed manually using debugfs Linux utility .

Overall, if recovery is attempted shortly after file removal and the partition is promptly unmounted, chances of complete recovery are high. If the system was heavily used, the probability of successful data undeletion significantly decreases. However, if you are to look at the problem from the forensics point of view, the chances of recovering something (such as a small part of the illegal image for the prosecution) is still very high. It was reported that sometimes parts of files from several years ago are found by forensic examiners .

Thus, files can be hidden in free space. If many copies of the same file are saved and then erased, the chance of getting the contents back becomes higher using the preceding recovery methods. However, due to the intricacies of ext2 filesystem, the process can only be reliably automated for small files .

WHY COLLECT EVIDENCE?

Electronic evidence can be very expensive to collect. The processes are strict and exhaustive, the systems affected may be unavailable for regular use for a long period oftime, and analysis of the data collected must be performed. So, why bother collecting the evidence in the first place? There are two simple reasons: future prevention and responsibility.

Future Prevention

Without knowing what happened, you have no hope of ever being able to stop someone else (or even the original attacker) from doing it again. It would be analogous to not fixing the lock on your door after someone broke in. Even though the cost of collection can be high, the cost of repeatedly recovering from compromises is much higher, both in monetary and corporate image terms.

Responsibility

There are two responsible parties after an attack: the attacker and the victim. The attacker is responsible for the damage done, and the only way to bring him to justice (and to seek recompense) is with adequate evidence to prove his actions.
The victim, on the other hand, has a responsibility to the community. Information gathered after a compromise can be examined and used by others to prevent further attacks. The victim may also have a legal obligation to perform an analysis of evidence collected, for instance if the attack on their system was part of a larger attack.

TYPES OF EVIDENCE

Before you start collecting evidence, it is important to know the different types of evidence categories. Without taking these into consideration, you may find that the evidence you've spent several weeks and quite a bit of money collecting is useless.
Real evidence is any evidence that speaks for itself without relying on anything else. In electronic terms, this can be a log produced by an audit function—provided that the log can be shown to be free from contamination.

Testimonial Evidence

Testimonial evidence is any evidence supplied by a witness. This type of evidence is subject to the perceived reliability of the witness, but as long as the witness can be considered reliable, testimonial evidence can be almost as powerful as real evidence. Word processor documents written by a witness may be considered testimonial—as long as the author is willing to state that he wrote it.

Hearsay

Hearsay is any evidence presented by a person who was not a direct witness. Word processor documents written by someone without direct knowledge of the incident are hearsay. Hearsay is generally inadmissible in court and should be avoided.

THE RULES OF EVIDENCE

There are five rules of collecting electronic evidence. These relate to five properties
that evidence must have to be useful.

1. Admissible
2. Authentic
3. Complete
4. Reliable
5. Believable

Admissible

Admissible is the most basic rule. The evidence must be able to be used in court or otherwise. Failure to comply with this rule is equivalent to not collecting the evidence in the first place, except the cost is higher.

Authentic

If you can't tie the evidence positively to the incident, you can't use it to prove anything. You must be able to show that the evidence relates to the incident in a relevant way.

Complete

It's not enough to collect evidence that just shows one perspective of the incident. You collect not only evidence that can prove the attacker's actions, but also evidence that could prove their innocence. For instance, if you can show the attacker was logged in at the time of the incident, you also need to show who else was logged in and why you think they didn't do it. This is called *exculpatory evidence* and is an important part of proving a case.

Reliable

The evidence you collect must be reliable. Your evidence collection and analysis procedures must not cast doubt on the evidence's authenticity and veracity.

Believable

The evidence you present should be clearly understandable and believable to a jury. There's no point presenting a binary dump of process memory if the jury has no idea what it all means. Similarly, if you present them with a formatted, human understandable version, you must be able to show the relationship to the original binary, otherwise there's no way for the jury to know whether you've faked it. Using the preceding rules, you can derive some basic do's and don'ts:

1) Minimize handling and corruption of original data.
2) Account for any changes and keep detailed logs of your actions.
3) Comply with the five rules of evidence.
4) Do not exceed your knowledge.
5) Follow your local security policy.
6) Capture as accurate an image of the system as possible.
7) Be prepared to testify.
8) Work fast.
9) Proceed from volatile to persistent evidence.
10) Don't shutdown before collecting evidence.
11) Don't run any programs on the affected system.

COLLECTION STEPS

You now have enough information to build a step-by-step guide for the collection of the evidence. Once again, this is only a guide. You should customize it to your specific situation. You should perform the following collection steps:

1. Find the evidence.
2. Find the relevant data.
3. Create an order of volatility.

4. Remove external avenues of change.
5. Collect the evidence.
6. Document everything.

Find the Evidence

Determine where the evidence you are looking for is stored. Use a checklist. Not only does it help you to collect evidence, but it also can be used to double-check that everything you are looking for is there.

Find the Relevant Data

Once you've found the evidence, you must figure out what part of it is relevant to the case. In general, you should err on the side of over-collection, but you must remember that you have to work fast. Don't spend hours collecting information that is obviously useless.

Create an Order of Volatility

Now that you know exactly what to gather, work out the best order in which to gather it. The order of volatility for your system is a good guide and ensures that you minimize loss of uncorrupted evidence.

Remove External Avenues of Change

It is essential that you avoid alterations to the original data, and prevention is always better than a cure. Preventing anyone from tampering with the evidence helps you create as exact an image as possible. However, you have to be careful. The attacker may have been smart and left a dead-man switch. In the end, you should try to do as much as possible to prevent changes.

Collect the Evidence

You can now start to collect the evidence using the appropriate tools for the job. As you go, reevaluate the evidence you've already collected. You may find that you missed something important. Now is the time to make sure you get it.

Document Everything

Your collection procedures may be questioned later, so it is important that you document everything you do. Timestamps, digital signatures, and signed statements are all important. Don't leave anything out.

SPECIAL NEEDS OF EVIDENTIAL AUTHENTICATION

A wealth of mathematical algorithms deal with secure encryption, verification, and authentication of computer-based material. These display varying degrees of security and complexity, but all of them rely on a *second channel* of information, whereby certain elements of the encryption/decryption/authentication processes are kept secret. This is characterized most plainly in the systems of public and private key encryption but is also apparent in other protocols.

Consider the investigative process where computers are concerned. During an investigation, it is decided that evidence may reside on a computer system. It may be possible to seize or impound the computer system, but this risks violating the basic principle of *innocent until proven guilty*, by depriving an innocent party of the use of his or her system. It should be perfectly possible to copy all the information from the computer system in a manner that leaves the original system untouched and yet makes all contents available for forensic analysis.

When this is done, the courts may rightly insist that the copied evidence is protected from either accidental or deliberate modification and that the investigating authority should prove that this has been done. Thus, it is not the content that needs protection, but its integrity.
This protection takes two forms: a secure method of determining that the data has not been altered by even a single bit since the copy was taken and a secure method of determining that the copy is genuinely the one taken at the time and on the computerin question. For the purpose of this chapter, these elements are collectively referred to here as the digital image verification and authentication protocol.
It is argued that when considering forensic copies of computer contents, encryption of data is not the point at issue. Neither are the provisions of the many digital signature protocols appropriate to the requirements of evidential authentication(see sidebar, “Digital IDs and Authentication Technology”).

HOW AUTHENTICODE WORKS WITH VERISIGN DIGITAL IDS

Authenticode relies on industry-standard cryptography techniques such as X.509 v3 or higher certificates and PKCS #7 and #10 signature standards. These are well-proven cryptography protocols, which ensure a robust implementation of code-signing technology. Developers can use the WinVerifyTrust API, on which Authenticode is based, to verify signed code in their own Win32 applications.
Authenticode uses digital signature technology to assure users of the origin and integrity of software. In digital signatures, the private key generates the signature,
and the corresponding public key validates it. To save time, the Authenticode protocols use a cryptographic digest, which is a one-way hash of the document. The process is outlined below and shown in following fig .

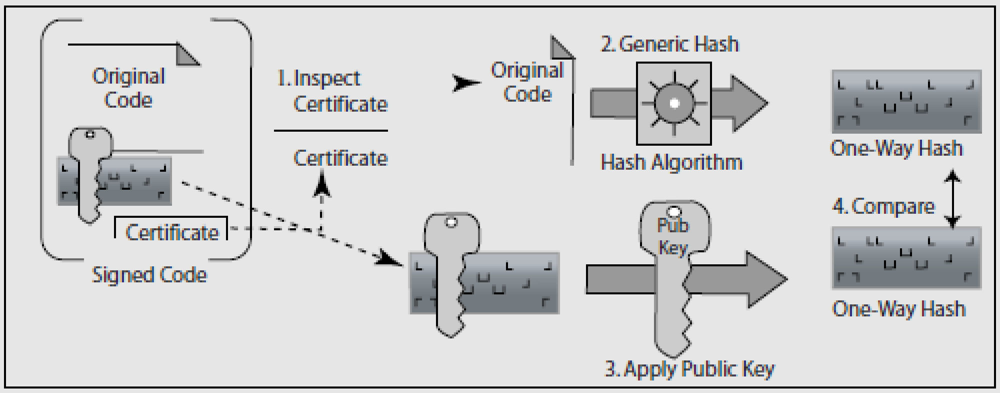


FIGURE 8.4 Authenticode: VeriSign Digital ID process.

1. Publisher obtains a software developer digital ID from VeriSign.
2. Publisher creates code
3. Using the SIGNCODE.EXE utility, the publisher
a. Creates a hash of the code, using an algorithm such as MD5 or SHA
b. Encrypts the hash using his private key

c. Creates a package containing the code, the encrypted hash, and the publisher’ s certificate
4. The end user encounters the package.
5. The end user’ s browser examines the publisher’ s digital ID. Using the VeriSignRroot public key, which is already embedded in Authenticode-enabled applications, the end user’ s browser verifies the authenticity of the software developer digital ID (which is itself signed by the VeriSign root Private Key).
6. Using the publisher’ s public key contained within the publisher’ s digital ID, the end user’ s browser decrypts the signed hash.
7. The end user’ s browser runs the code through the same hashing algorithm as the publisher, creating a new hash.
8. The end user’ s browser compares the two hashes. If they are identical, the browser messages that the content has been verified by VeriSign, and the end user has confidence that the code was signed by the publisher identified in the digital ID and that the code hasn’ t been altered since it was signed.

TIMEKEEPING

It seems that, although every computer has a clock, none of them appear to be synchronized—unless the computer in question is running the Network Time Protocol (NTP). With NTP, you can synchronize against truly accurate time sources such as the atomic clocks run by the National Institute of Standards and Technology (NIST), the U.S. Naval Observatory, or counterparts in other countries around the world.

NTP is a protocol built on top of transmission control protocol/Internet protocol (TCP/IP) that ensures accurate local timekeeping with reference to radio, atomic, or other clocks located on the Internet. This protocol is capable of synchronizing distributed clocks within milliseconds over long periods of time.

What does accurate timekeeping have to do with computer forensics? Keeping a consistent sense of time is critical for many computer-forensic-related activities. Financial organizations rely on accurate timekeeping for their transactions. Many authentication systems, Kerberos being the most prominent example, use dated tickets to control access to systems and resources. Investigating incidents that involve multiple computers is much easier when the timestamps on files and in logs are in sync.

Time Matters

Why bother having accurate clocks? Isn’ t the one that comes in your desktop PC or your enterprise server adequate? The answer is that accurate timekeeping is an advanced science, an avocation practiced by hundreds of scientists around the world, and the paltry clock chip you have in your PC or expensive server winds up being a bit less accurate than your SwatchR watch for several reasons.
Computer clocks, like most electronic clocks, detect the oscillations of a quartz crystal and calculate the passing time based on these oscillations. Not all quartz crystals are the same to begin with, but put one inside a nice, hot computer that’ s cool whenever it’ s turned off, and the crystal’ s frequency tends to wander. Also,

Unix systems base their notion of time on interrupts generated by the hardware clock. Delays in processing these interrupts cause Unix system clocks to lose time—slowly, but erratically. These small changes in timekeeping are what time scientists call *jitter*. Over time, scientists and programmers have developed different techniques for synchronizing clocks over TCP/IP or other network protocols. The time protocol provides a server' s notion of time in a machine-readable format, and there' s also an Internet Control Message Protocol (ICMP) timestamp message. Though these remain available Internet standards, neither is currently sufficient for accurate timekeeping, and, hence, both are considered out-of-date. The Unix r commands include rdate, which permits setting a local clock based on a remote server. There are modem-based programs that contact NIST timeservers and fetch a time message (along with an estimate of round-trip time to account for latency), which you can still use today.

Clock Filters

Automatically accepting another system' s statement about the current time can be harmful: suppose the timekeeping system has been taken over by an attacker who needs to *turn back the clock* so that a replay attack can function. NTP guards against this in several ways. First, NTP assumes that time moves forward, not backward, although small backward changes are acceptable. Also, if a system has been using NTP, the NTP software assumes that changes in a local clock will be small, generally less than a second. This makes controlling a local clock or making large changes literally a time-consuming process—even a one-second change is a big deal.

HOW TO BECOME A DIGITAL DETECTIVE

Recovering electronic data is only the beginning. Once you recover it, you need to determine how to use it in your case. In other words, how do you reconstruct past events to ensure that your findings will be admissible as evidence in your case? What follows are some recommendations for accomplishing that goal.

If You Need Help, Get Help

When you receive the package of evidence containing a Zip disk and cover letter stating, "Enclosed and produced upon your request, please find ...," you may not know what to do with the disk. If you don' t know, get help.

Help may be just down the hall. If you have an information services department, consider going there. They might not understand what you mean by a discovery request, but they may be able to help you convert the contents of the disk to a form you can look at. If you have a litigation support group, consider contacting them. They may have the tools you need to look at and start working with the data you just received. Even if there is no formal entity within your office dedicated to dealing with technological issues, there may be informal resources.

Convert Digital Evidence

Before you can reconstruct past events and present the data, you need it on a medium and in a format you can work with. In other words, you need to get the

data onto a medium you can use, if it is not already on one. Today, data can come on a variety of media, such as holograms, video, data tapes, Zip disks, CD-ROM disks, and even 3.5-inch floppy disks.
For example, you could use Zip disks. Zip disks are simpler. The cost of Iomega Zip drives (*http://www.iomega.com/global/index.jsp*) is so low that you can keep one on hand just to copy data from Zip disks you receive (and to copy data to Zip disks when others request data from you on that medium). CDs are even simpler, as CD drives have become commonplace on PCs. Similarly, even 3.5-inch disks generally pose no problem.

Put the Evidence in a Useable Format

Having data on a useable medium is useless unless it also is in a useable format. At times this is not an issue. If the data comes in a format that you already use, then you can begin to work with it as soon as you get it off the media. The formats most likely to be useable without conversion are word processing files (principally Word-Perfect and Word files), spreadsheet files (principally Excel and Lotus), and presentation files (principally PowerPoint files).

USEABLE FILE FORMATS

Even if the data is in a format that appears to be one you already use, conversion still may be necessary. The format may be too new. The problem is a basic one. In a similar vein, you may have to get the data converted if it comes to you in a format thatis too old or runs on a different operating system. Although simple files created withone company's software generally can be opened without a problem using a competitor's comparable product, this often does not hold true for more complex files.

UNUSABLE FILE FORMATS

You may get electronic data in a format that you cannot use "out of the box." When that happens, you have to convert the files to a format you can use—or find someone to do the conversion for you. You may have already encountered these issues with a variety of files including email files, database files from mainframe systems, and ".txt" files containing data dumped from database files. Anyone who has undertaken this task can attest that it is potentially a difficult and painstaking process.
For example, if you receive a ".txt" file that appears to contain information from a database file, try to find out, among other things, the make and model of the computer the file came from; the name and version of the operating system the computer ran; the name and version of the database program used; the name of the database file; a list of all fields in the database; and descriptions of each field with the descriptions including the type, length, and other characteristics of the field.

CONVERTING FILES

If you are going to attempt converting the data yourself, you may be fortunate enough to have received electronic data that you can covert directly into programs such as Access or Excel using the wizards built into those programs. This can be the

case with “.txt” files. Sometimes the first line in a file you are converting may even contain the names of the fields that need to be created, further simplifying your task. If that information is not in the file, then try to get the field names and descriptions from the producing party. Should you fail at that, you may have an exceedingly difficult time carrying out a meaningful conversion. Sometimes data will not be in a format amenable to immediate conversion.
Email files are a common example.

Get the Right Software, Hardware, and Personnel

Concomitant with getting the data into a useable format is getting the right software, hardware, and personnel to work with the format you choose. For software, you may have already found that Access, Excel.