CHAPTER

16

Auditing IT Controls Part II: Security and Access

Learning Objectives

After studying this chapter, you should:

- Be able to identify the principal threats to the operating system and the control techniques used to minimize the possibility of actual exposures.
- Be familiar with the principal risks associated with electronic commerce conducted over intranets and the Internet and understand the control techniques used to reduce these risks.
- Be familiar with the risks to database integrity and the controls used to mitigate them.
- Recognize the unique exposures that arise in connection with electronic data interchange (EDI) and understand how these exposures can be reduced.

This chapter continues the treatment of general IT controls as the COSO control framework describes. The focus of the chapter is on Sarbanes-Oxley (SOX) compliance regarding the security and control of operating systems, database management systems, and communication networks. This chapter examines the risks, controls, audit objectives, and tests of controls that may be performed to satisfy either compliance or attest responsibilities.

Controlling the Operating System

The **operating system** is the computer's control program. It allows users and their applications to share and access common computer resources, such as processors, main memory, databases, and printers. If operating system integrity is compromised, controls within individual accounting applications may also be circumvented or neutralized. Because the operating system is common to all users, the larger the computer facility, the greater the scale of potential damage. Thus, with an ever-expanding user community sharing more and more computer resources, operating system security becomes an important control issue.

OPERATING SYSTEM OBJECTIVES

The operating system performs three main tasks. First, it translates high-level languages, such as COBOL, C++, BASIC, and SQL, into the machine-level language that the computer can execute. The language translator modules of the operating system are called **compilers** and **interpreters**. The control implications of language translators are examined in Chapter 17.

Second, the operating system allocates computer resources to users, workgroups, and applications. This includes assigning memory work space (partitions) to applications and authorizing access to terminals, telecommunications links, databases, and printers.

Third, the operating system manages the tasks of job scheduling and multiprogramming. At any point, numerous user applications (jobs) are seeking access to the computer resources under the control of the operating system. Jobs are submitted to the system in three ways: (1) directly by the system operator, (2) from various batch-job queues, and (3) through telecommunications links from remote workstations. To achieve efficient and effective use of finite computer resources, the operating system must schedule job processing according to established priorities and balance the use of resources among the competing applications.

To perform these tasks consistently and reliably, the operating system must achieve five fundamental control objectives.[1]

1. The operating system must protect itself from users. User applications must not be able to gain control of, or damage in any way, the operating system, thus causing it to cease running or to destroy data.
2. The operating system must protect users from each other. One user must not be able to access, destroy, or corrupt the data or programs of another user.
3. The operating system must protect users from themselves. A user's application may consist of several modules stored in separate memory locations, each with its own data. One module must not be allowed to destroy or corrupt another module.
4. The operating system must be protected from itself. The operating system is also made up of individual modules. No module should be allowed to destroy or corrupt another module.
5. The operating system must be protected from its environment. In the event of a power failure or other disaster, the operating system should be able to achieve a controlled termination of activities from which it can later recover.

OPERATING SYSTEM SECURITY

Operating system security involves policies, procedures, and controls that determine who can access the operating system, which resources (files, programs, printers) they can access, and what actions they can take. The following security components are found in secure operating systems: log-on procedure, access token, access control list, and discretionary access privileges.

Log-On Procedure

A formal **log-on procedure** is the operating system's first line of defense against unauthorized access. When the user initiates the process, he or she is presented with a dialog box requesting the user's ID and password. The system compares the ID and password to a database of valid users. If the system finds a match, then the log-on attempt is authenticated. If, however, the password or ID is entered incorrectly, the log-on attempt fails, and a message is returned to the user. The message should not reveal whether the password or the ID caused the failure. The system should allow the user to reenter the log-on information. After a specified number of attempts (usually no more than five), the system should lock out the user from the system.

Access Token

If the log-on attempt is successful, the operating system creates an **access token** that contains key information about the user, including user ID, password, user group, and privileges granted to the user. The information in the access token is used to approve all actions the user attempts during the session.

1 F. M. Stepczyk, "Requirements for Secure Operating Systems," Data Security and Data Processing, vol. 5; Study Results: TRW Systems, Inc. (New York: IBM Corporation, 1974), 25–73.

Access Control List

An **access control list** assigned to each resource controls access to system resources, such as directories, files, programs, and printers. These lists contain information that defines the access privileges for all valid users of the resource. When a user attempts to access a resource, the system compares his or her ID and privileges contained in the access token with those contained in the access control list. If there is a match, the user is granted access.

Discretionary Access Privileges

The central system administrator usually determines who is granted access to specific resources and maintains the access control list. In distributed systems, however, end users may control (own) resources. Resource owners in this setting may be granted **discretionary access privileges**, which allow them to grant access privileges to other users. For example, the controller, who is the owner of the general ledger, may grant read-only privileges to a manager in the budgeting department. The accounts payable manager, however, may be granted both read and write permissions to the ledger. Any attempt the budgeting manager makes to add, delete, or change the general ledger will be denied. The use of discretionary access control needs to be closely supervised to prevent security breaches because of its liberal use.

THREATS TO OPERATING SYSTEM INTEGRITY

Operating system control objectives may not be achieved because of flaws in the operating system that are exploited either accidentally or intentionally. Accidental threats include hardware failures that cause the operating system to crash. Errors in user application programs, which the operating system cannot interpret, also cause operating system failures. Accidental system failures may cause whole segments of memory to be dumped to disks and printers, resulting in the unintentional disclosure of confidential information.

Intentional threats to the operating system are most commonly attempts to illegally access data or violate user privacy for financial gain. However, a growing threat is destructive programs from which there is no apparent gain. These exposures come from three sources:

1. Privileged personnel who abuse their authority. Systems administrators and systems programmers require unlimited access to the operating system to perform maintenance and to recover from system failures. Such individuals may use this authority to access users' programs and data files.
2. Individuals, both internal and external to the organization, who browse the operating system to identify and exploit security flaws.
3. Individuals who intentionally (or accidentally) insert computer viruses or other forms of destructive programs into the operating system.

OPERATING SYSTEM CONTROLS AND TESTS OF CONTROLS

This section describes a variety of control techniques for preserving operating system integrity. If operating system integrity is compromised, controls within individual accounting applications that impact financial reporting may also be compromised. For this reason, the design and assessment of these controls are SOX compliance issues. In this section, controls and associated tests over the following areas are examined: access privileges, password control, virus control, and audit trail control.

Controlling Access Privileges

User access privileges are assigned to individuals and to entire workgroups authorized to use the system. Privileges determine which directories, files, applications, and other resources an individual or group may access. They also determine the types of actions that can be taken. Recall that the systems administrator or the owner of the resource may assign privileges. Management should

be concerned that individuals are not granted privileges that are incompatible with their assigned duties. Consider, for example, a cash receipts clerk who is granted the right to access and make changes to the accounts receivable file.

Overall, the way access privileges are assigned influences system security. Privileges should, therefore, be carefully administered and closely monitored for compliance with organizational policy and principles of internal control.

Audit Objectives Relating to Access Privileges

The objective of the auditor is to verify that access privileges are granted in a manner that is consistent with the need to separate incompatible functions and is in accordance with the organization's policy.

Audit Procedures Relating to Access Privileges

- Review the organization's policies for separating incompatible functions and ensure that they promote reasonable security.
- Review the privileges of a selection of user groups and individuals to determine if their access rights are appropriate for their job descriptions and positions. The auditor should verify that individuals are granted access to data and programs based on their need to know.
- Review personnel records to determine whether privileged employees undergo an adequately intensive security clearance check in compliance with company policy.
- Review employee records to determine whether users have formally acknowledged their responsibility to maintain the confidentiality of company data.
- Review the users' permitted log-on times. Permission should be commensurate with the tasks being performed.

Password Control

A **password** is a secret code the user enters to gain access to systems, applications, data files, or a network server. If the user cannot provide the correct password, the operating system should deny access. Although passwords can provide a degree of security, when imposed on nonsecurity-minded users, password procedures can result in end-user behavior that actually circumvents security. The most common forms of contra-security behavior include:

- Forgetting passwords and being locked out of the system.
- Failing to change passwords on a frequent basis.
- The Post-it syndrome, whereby passwords are written down and displayed for others to see.
- Simplistic passwords that a computer criminal easily anticipates.

REUSABLE PASSWORDS. The most common method of password control is the **reusable password**. The user defines the password to the system once and then reuses it to gain future access. The quality of the security a reusable password provides depends on the quality of the password itself. If the password pertains to something personal about the user, such as a child's name, pet's name, birth date, or hair color, a computer criminal can often deduce it. Even if the password is derived from nonpersonal data, it may be weak. For example, a string of keystrokes (such as A-S-D-F) or the same letter used multiple times can easily be cracked. Passwords that contain random letters and digits are more difficult to crack but are also more difficult for the user to remember.

To improve access control, management should require that passwords be changed regularly and disallow weak passwords. Software is available that automatically scans password files and notifies users that their passwords have expired and need to be changed. These systems also use extensive databases of known weak passwords to validate the new password and disallow weak ones. An alternative to the standard reusable password is the one-time password.

ONE-TIME PASSWORDS. The **one-time password** was designed to overcome the aforementioned problems. Under this approach, the user's password changes continuously. This technology employs a credit card-sized smart card that contains a microprocessor programmed with an algorithm that generates, and electronically displays, a new and unique password every 60 seconds. The card works in conjunction with special authentication software located on a mainframe or network server computer. Each user's card is synchronized to the authentication software so that at any point in time both the smart card and the network software are generating the same password for the same user.

To access the network, the user enters the PIN followed by the current password displayed on the card. The password can be used one time only. If, for example, a computer hacker intercepts the password and PIN during transmission and attempts to use them within the one-minute time frame, access will be denied. Also, if the smart card should fall into the hands of a computer criminal, access cannot be achieved without the PIN.

Another one-time password technique uses a challenge/response approach to achieve the same end. When the user attempts to log on, the network authentication software issues a six-character code (the challenge) that the card can either scan optically, or the code can be entered into the card via its built-in keypad. The card's internal algorithm then generates a one-time password (the response) that the user enters through the keyboard of the remote terminal. If the firewall recognizes the current password, access is permitted.

Audit Objectives Relating to Passwords

The auditor's objective here is to ensure that the organization has an adequate and effective password policy for controlling access to the operating system.

Audit Procedures Relating to Passwords

The auditor may achieve this objective by performing the following tests:

- Verify that all users are required to have passwords.
- Verify that new users are instructed in the use of passwords and the importance of password control.
- Review password control procedures to ensure that passwords are changed regularly.
- Review the password file to determine that weak passwords are identified and disallowed. This may involve using software to scan password files for known weak passwords.
- Verify that the password file is encrypted and that the encryption key is properly secured.
- Assess the adequacy of password standards, such as length and expiration interval. Review the account lockout policy and procedures. Most operating systems allow the system administrator to define the action to be taken after a certain number of failed log-on attempts. The auditor should determine how many failed log-on attempts are allowed before the account is locked. The duration of the lockout also needs to be determined. This could range from a few minutes to a permanent lockout that requires formal reactivation of the account.

Controlling against Malicious and Destructive Programs

Malicious and destructive programs are responsible for millions of dollars of corporate losses annually. The losses are measured in terms of data corruption and destruction, degraded computer performance, hardware destruction, violations of privacy, and the personnel time devoted to repairing the damage. This class of programs includes viruses, worms, logic bombs, back doors, and Trojan horses. Because these have become popular press terms in recent years, we will not devote space at this point to define them. The appendix to this chapter, however, contains a detailed discussion of this material. Threats from destructive programs can be substantially reduced through a combination of technology controls and administrative procedures. The following examples are relevant to most operating systems.

- Purchase software only from reputable vendors, and accept only those products that are in their original, factory-sealed packages.
- Issue an entity-wide policy pertaining to the use of unauthorized software or illegal (bootleg) copies of copyrighted software.
- Examine all upgrades to vendor software for viruses before they are implemented.
- Inspect all public-domain software for virus infection before using.
- Establish entity-wide procedures for making changes to production programs.
- Establish an educational program to raise user awareness regarding threats from viruses and malicious programs.
- Install all new applications on a stand-alone computer, and thoroughly test them with antiviral software prior to implementing them on the mainframe or local area network (LAN) server.
- Routinely make backup copies of key files stored on mainframes, servers, and workstations.
- Wherever possible, limit users to read and execute rights only. This allows users to extract data and run authorized applications, but it denies them the ability to write directly to mainframe and server directories.
- Require protocols that explicitly invoke the operating system's log-on procedures to bypass Trojan horses. A typical scenario is one in which a user sits down at a terminal that is already displaying the log-on screen and proceeds to enter his or her ID and password. This, however, may be a Trojan horse rather than the legitimate procedure. Some operating systems allow the user to directly invoke the operating system log-on procedure by entering a key sequence, such as CTRL + ALT + DEL. The user then knows that the log-on procedure on the screen is legitimate.
- Use antiviral software (also called vaccines) to examine application and operating system programs for the presence of a virus and remove it from the affected program. Antiviral programs are used to safeguard mainframes, network servers, and personal computers. Most antiviral programs run in the background on the host computer and automatically test all files that are uploaded to the host. The software, however, works only on known viruses. If a virus has been modified slightly (mutated), there is no guarantee that the vaccine will work. Therefore, maintaining a current version of the vaccine is critical.

Audit Objective Relating to Viruses and Other Destructive Programs

The key to computer virus control is prevention through strict adherence to organizational policies and procedures that guard against virus infection. The auditor's objective is to verify that effective management policies and procedures are in place to prevent the introduction and spread of destructive programs, including viruses, worms, back doors, logic bombs, and Trojan horses.

Audit Procedures Relating to Viruses and Other Destructive Programs

- Through interviews, determine that operations personnel have been educated about computer viruses and are aware of the risky computing practices that can introduce and spread viruses and other malicious programs.
- Verify that new software is tested on stand-alone workstations prior to being implemented on the host or network server.
- Verify that the current version of antiviral software is installed on the server and that upgrades are regularly downloaded to workstations.

System Audit Trail Controls

System audit trails are logs that record activity at the system, application, and user level. Operating systems allow management to select the level of auditing to be recorded in the log. They need

to decide on their threshold between information and irrelevant facts. An effective audit policy will capture all significant events without cluttering the log with trivial activity. Audit trails typically consist of two types of audit logs: (1) detailed logs of individual keystrokes and (2) event-oriented logs.

KEYSTROKE MONITORING. Keystroke monitoring involves recording both the user's keystrokes and the system's responses. This form of log may be used after the fact to reconstruct the details of an event or as a real-time control to prevent unauthorized intrusion. Keystroke monitoring is the computer equivalent of a telephone wiretap. While some situations may justify this level of surveillance, keystroke monitoring may also be regarded as a violation of privacy. Before implementing this type of control, management and auditors should consider the possible legal, ethical, and behavioral implications.

EVENT MONITORING. Event monitoring summarizes key activities related to system resources. Event logs typically record the IDs of all users accessing the system; the time and duration of a user's session; programs that were executed during a session; and the files, databases, printers, and other resources accessed.

Setting Audit Trail Objectives

Audit trails can be used to support security objectives in three ways: (1) detecting unauthorized access to the system, (2) facilitating the reconstruction of events, and (3) promoting personal accountability.

DETECTING UNAUTHORIZED ACCESS. Detecting unauthorized access can occur in real time or after the fact. The primary objective of real-time detection is to protect the system from outsiders attempting to breach system controls. A real-time audit trail can also be used to report changes in system performance that may indicate infestation by a virus or worm. Depending on how much activity is being logged for review, real-time detection can add significantly to operational overhead and degrade performance. After-the-fact detection logs can be stored electronically and reviewed periodically or as needed. When properly designed, they can be used to determine if unauthorized access was accomplished, or attempted and failed.

RECONSTRUCTING EVENTS. Audit trail analysis can be used to reconstruct the steps that led to events such as system failures or security violations by individuals. Knowledge of the conditions that existed at the time of a system failure can be used to assign responsibility and to avoid similar situations in the future.

PERSONAL ACCOUNTABILITY. Audit trails can be used to monitor user activity at the lowest level of detail. This capability is a preventive control that can influence behavior. Individuals are less likely to violate an organization's security policy when they know that their actions are recorded in an audit log. A system audit log can also serve as a detective control to assign personal accountability for actions taken, such as abuse of authority. For example, consider an accounts receivable clerk with authority to access customer records. The audit log may disclose that the clerk has been printing an inordinate number of records, which may indicate that the clerk is selling customer information in violation of the company's privacy policy.

Implementing a System Audit Trail

The information contained in audit logs is useful to accountants in measuring the potential damage and financial loss associated with application errors, abuse of authority, or unauthorized access by outside intruders. Audit logs, however, can generate data in overwhelming detail. Important information can easily get lost among the superfluous details of daily operation. Thus, poorly designed logs can actually be dysfunctional. Protecting exposures with the potential for material financial loss should drive management's decision as to which users, applications, or

operations to monitor, and how much detail to log. As with all controls, the benefits of audit logs must be balanced against the costs of implementing them.

Audit Objectives Relating to System Audit Trails

The auditor's objective is to ensure that the established system audit trail is adequate for preventing and detecting abuses, reconstructing key events that precede systems failures, and planning resource allocation.

Audit Procedures Relating to System Audit Trails

- Most operating systems provide some form of audit manager function to specify the events that are to be audited. The auditor should verify that the audit trail has been activated according to organization policy.
- Many operating systems provide an audit log viewer that allows the auditor to scan the log for unusual activity. These can be reviewed on screen or by archiving the file for subsequent review. The auditor can use general-purpose data extraction tools for accessing archived log files to search for defined conditions such as:
  - Unauthorized or terminated user
  - Periods of inactivity
  - Activity by user, workgroup, or department
  - Log-on and log-off times
  - Failed log-on attempts
  - Access to specific files or applications
- The organization's security group has responsibility for monitoring and reporting security violations. The auditor should select a sample of security violation cases and evaluate their disposition to assess the effectiveness of the security group.

Controlling Database Management Systems

Controls over database management fall into two general categories: access controls and backup controls. **Access controls** are designed to prevent unauthorized individuals from viewing, retrieving, corrupting, or destroying the entity's data. **Backup controls** ensure that in the event of data loss due to unauthorized access, equipment failure, or physical disaster, the organization can recover its files and databases.

ACCESS CONTROLS

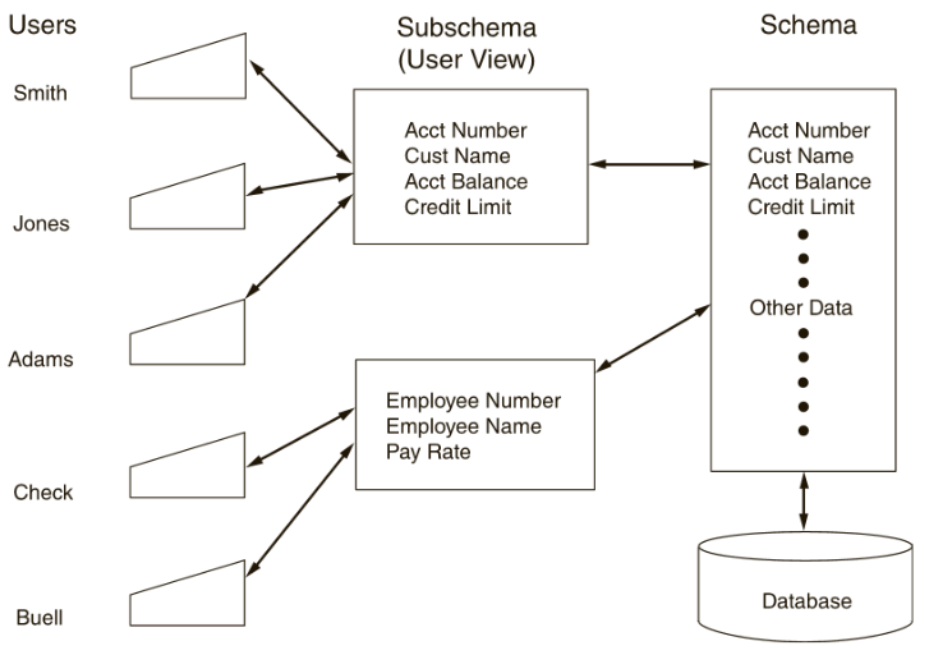
Risks to corporate databases include corruption, theft, misuse, and destruction of data. These threats originate from both unauthorized intruders and authorized users who exceed their access privileges. Several database control features that reduce these risks are reviewed in the following sections.

User Views

The **user view** or subschema is a subset of the total database that defines the user's data domain and restricts his or her access to the database accordingly. Figure 16-1 illustrates the role of the user view. The database administrator typically is responsible for defining user views. The auditor is concerned that such access privileges are commensurate with the users' legitimate needs.

Although user views can restrict user access to a limited set of data, they do not define task privileges, such as read, delete, or write. Often, several users may share a single user view but have different authority levels. For example, users Smith, Jones, and Adams in Figure 16-1 all may have access to the same set of data: account number, customer name, account balance, and credit

FIGURE 16-1 SUBSCHEMA RESTRICTING ACCESS TO DATABASE



limit. Let's assume that all have read authority, but only Jones has authority to modify and delete the data. To achieve this level of restriction requires additional security measures, discussed next.

Database Authorization Table

The **database authorization table** contains rules that limit the actions a user can take. Each user is granted certain privileges that are coded in the authority table, which is used to verify the user's action requests. For example, the authorization table in Figure 16-2 shows that of the three users, only Jones has the authority to modify and delete the data.

FIGURE 16-2 DATABASE AUTHORIZATION TABLE

Dept	Accounts Rec			Billings	
User	Jones	Smith	Adams	Check	Buell
Password	Bugs	Dog	Katie	Lucky	Star
Authority:					
Read	Y	Y	Y	Y	Y
Insert	Y	N	Y	Y	N
Modify	Y	N	N	Y	N
Delete	Y	N	N	N	N

User-Defined Procedures

A **user-defined procedure** allows the user to create a personal security program or routine to provide more positive user identification than a password can. For example, in addition to a password, the security procedure asks a series of personal questions (such as the user's mother's maiden name), which only the legitimate user is likely to know.

Data Encryption

Many database systems use encryption procedures to protect highly sensitive data, such as product formulas, personnel pay rates, password files, and certain financial data. **Data encryption** uses an algorithm to scramble selected data, thus making it unreadable to an intruder browsing the database. In addition to protecting stored data, encryption is used for protecting data that are transmitted across networks. We discuss various encryption techniques later in this chapter.

Biometric Devices

The ultimate in user authentication procedures is the use of **biometric devices**, which measure various personal characteristics, such as fingerprints, voiceprints, retina prints, or signature characteristics. These user characteristics are digitized and stored permanently in a database security file or on an identification card that the user carries. When an individual attempts to access the database, a special scanning device captures his or her biometric characteristics, which it compares with the profile data stored internally or on the ID card. If the data do not match, access is denied.

Audit Objectives Relating to Database Access

The auditor's objectives are to verify (1) that individuals who are authorized to use the database are limited to accessing only the data needed to perform their duties, and (2) that unauthorized individuals are denied access to the database.

Audit Procedures for Testing Access Controls

RESPONSIBILITY FOR AUTHORITY TABLES AND SUBSCHEMAS. The auditor should verify that database administration personnel retain sole responsibility for creating authority tables and designing user views. Evidence of compliance can come from three sources: (1) by reviewing company policy and job descriptions, which specify these technical responsibilities; (2) by examining programmer authority tables for access privileges to data definition language **(DDL)** commands; and (3) through personal interviews with programmers and database administration personnel.

APPROPRIATE ACCESS AUTHORITY. The auditor can select a sample of users and verify that their access privileges stored in the authority table are consistent with their organizational functions.

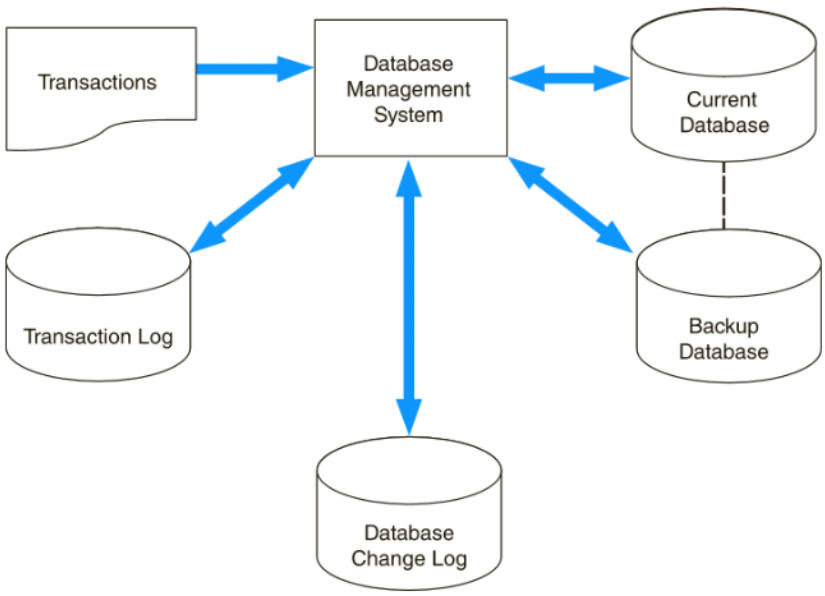
BIOMETRIC CONTROLS. The auditor should evaluate the costs and benefits of biometric controls. Generally, these would be most appropriate when a very limited number of users may access highly sensitive data.

ENCRYPTION CONTROLS. The auditor should verify that sensitive data, such as passwords, are properly encrypted. This can be done by printing the file contents to hard copy.

BACKUP CONTROLS

Malicious acts from external hackers, disgruntled employees, disk failure, program errors, fires, floods, and earthquakes can corrupt and destroy data. To recover from such disasters, the organization needs to reconstruct the database to prefailure status. This can be done only if the database was properly backed up in the first place. Organizations must, therefore, implement policies,

FIGURE 16-3 DATABASE BACKUP AND RECOVERY



procedures, and techniques that systematically and routinely provide backup copies of critical data. This section examines backup controls used in the database environment. Legacy systems that use a flat-file structure present a different set of issues that were discussed in Chapter 3.

Large-scale database management systems employ a backup and recovery procedure similar to that illustrated in Figure 16-3. This system provides four backup and recovery features: database backup, a transaction log, checkpoints, and a recovery module. Each of these is described in the following section.

DATABASE BACKUP. The backup feature makes a periodic backup of the entire database. This is an automatic procedure that should be performed at least once a day. The backup copy should then be stored in a secure remote area.

TRANSACTION LOG (JOURNAL). The **transaction log** feature provides an audit trail of all processed transactions. It lists transactions in a transaction log file and records the resulting changes to the database in a separate database change log.

CHECKPOINT FEATURE. The **checkpoint feature** suspends all data processing while the system reconciles the transaction log and the database change log against the database. At this point, the system is in a quiet state. Checkpoints occur automatically several times an hour. If a failure occurs, it is usually possible to restart the processing from the last checkpoint. Thus, only a few minutes of transaction processing must be repeated.

RECOVERY MODULE. The **recovery module** uses the logs and backup files to restart the system after a failure.

Audit Objectives Relating to Database Backup

The auditor's objective is to verify that database backup controls are adequate to facilitate the recovery of lost, destroyed, or corrupted data.

Audit Procedures for Testing Backup Controls

Database backup should be a routine activity.

- The auditor should verify from system documentation that production databases are copied at regular intervals (perhaps several times an hour).
- The auditor should verify through documentation and observation that backup copies of the database are stored off-site to support disaster recovery procedures.

Controlling Networks

Chapter 12 examined the operational characteristics of several network topologies used in Internet and intranet communications. Network topologies consist of various configurations of (1) communications lines (twisted-pair wires, coaxial cable, microwaves, and fiber optics), (2) hardware components (modems, multiplexers, servers, and front-end processors), and (3) software (protocols and network control systems). The technology of network communications are subject to two general forms of risk:

1. Risks from subversive threats. These include, but are not limited to, a computer criminal intercepting a message transmitted between the sender and the receiver, a computer hacker gaining unauthorized access to the organization's network, and a denial of service attack from a remote location of the Internet.
2. Risks from equipment failure. For example, equipment failures in the communications system can disrupt, destroy, or corrupt transmissions between senders and receivers. Equipment failure can also result in the loss of databases and programs stored on network servers.

CONTROLLING RISKS FROM SUBVERSIVE THREATS

Firewalls

Organizations connected to the Internet or other public networks often implement an electronic firewall to insulate their intranet from outside intruders. A **firewall** is a system that enforces access control between two networks. To accomplish this:

- All traffic between the outside network and the organization's intranet must pass through the firewall.
- Only authorized traffic between the organization and the outside, as formal security policy specifies, is allowed to pass through the firewall.
- The firewall must be immune to penetration from both outside and inside the organization.

Firewalls can be used to authenticate an outside user of the network, verify his or her level of access authority, and then direct the user to the program, data, or service requested. In addition to insulating the organization's network from external networks, firewalls can also be used to insulate portions of the organization's intranet from internal access. For example, a LAN controlling access to financial data can be insulated from other internal LANs. Some commercially available firewalls provide a high level of security, whereas others are less secure but more efficient. Firewalls may be grouped into two general types: network-level firewalls and application-level firewalls.

Network-level firewalls provide efficient but low security access control. This type of firewall consists of a **screening router** that examines the source and destination addresses that are attached to incoming message packets. The firewall accepts or denies access requests based on filtering rules that have been programmed into it. The firewall directs incoming calls to the correct internal receiving node. Network-level firewalls are insecure because they are designed to facilitate the free flow of information rather than restrict it. This method does not explicitly authenticate outside users.

Application-level firewalls provide a higher level of customizable network security, but they add overhead to connectivity. These systems are configured to run security applications called proxies that permit routine services, such as e-mail, to pass through the firewall, but they can perform sophisticated functions, such as user authentication for specific tasks. Application-level firewalls also provide comprehensive transmission logging and auditing tools for reporting unauthorized activity.

A high level of firewall security is possible using a dual-homed system. This approach, illustrated in Figure 16-4, has two firewall interfaces. One screens incoming requests from the Internet; the other provides access to the organization's intranet. Direct communication to the Internet is disabled, and the two networks are fully isolated. Proxy applications that impose separate log-on procedures perform all access.

Choosing the right firewall involves a trade-off between convenience and security. Ultimately, organization management, in collaboration with internal audit and network professionals, must come to grips with what constitutes acceptable risk. The more security the firewall provides, however, the less convenient it is for authorized users to pass through it to conduct business.

Controlling Denial of Service Attacks

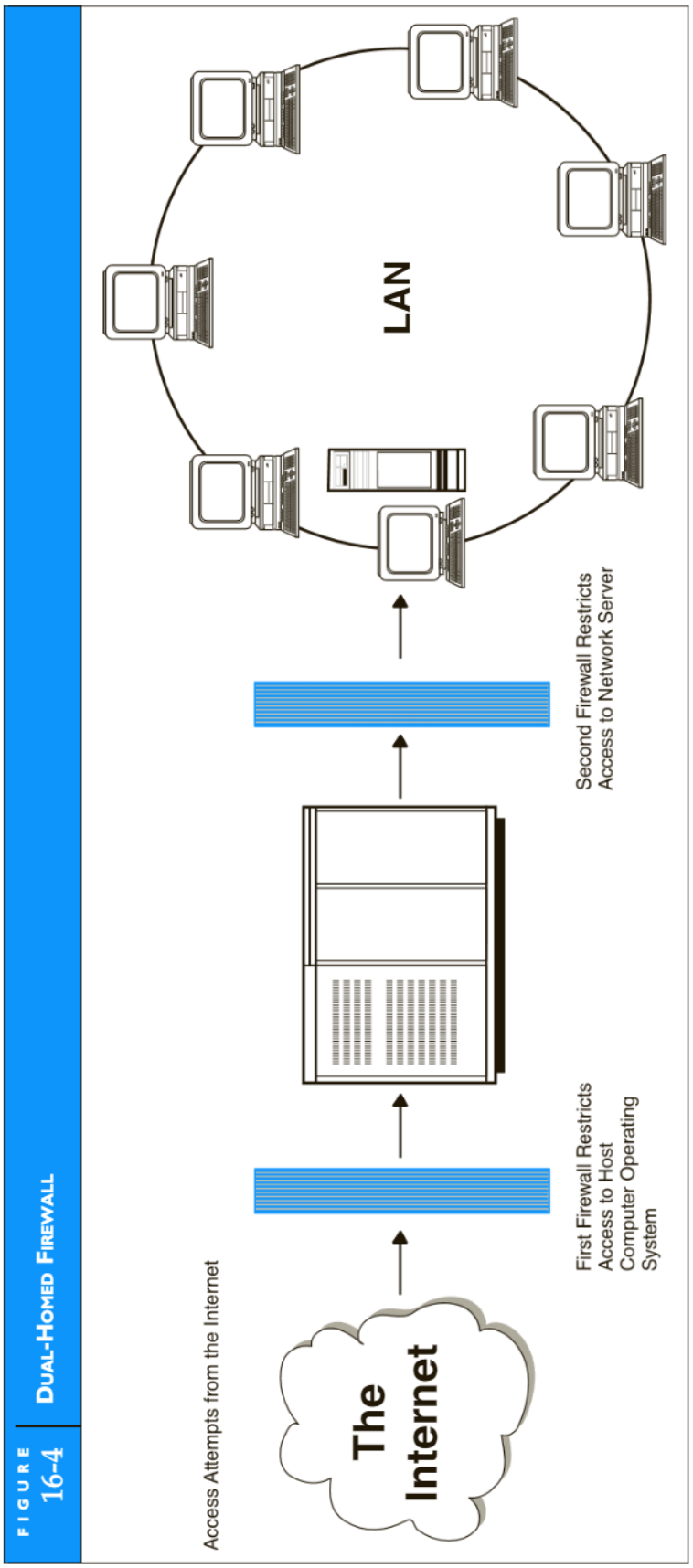
Chapter 12 described three common forms of denial of service attacks: SYN flood attacks, smurf attacks, and distributed denial of service (DDoS) attacks. Each of these techniques has a similar effect on the victim. By clogging the Internet ports of the victim's server with fraudulently generated messages, the targeted firm is rendered incapable of processing legitimate transactions and can be completely isolated from the Internet for the duration of the attack.

In the case of a smurf attack, the targeted organization can program its firewall to ignore all communication from the attacking site, once the attacker's IP address is determined. SYN flood attacks that use IP spoofing to disguise the source, however, are a more serious problem. Although the attack may actually be coming from a single disguised site, the victim's host computer views these transmissions as coming from all over the Internet. IT and network management can take two actions to defeat this sort of attack. First, Internet hosts must embrace a policy of social responsibility by programming their firewalls to block outbound message packets that contain invalid internal IP addresses. This would prevent attackers from hiding their locations from the targeted site and would assure the management of potential intermediary hosts that no undetected attacks could be launched from their sites. This strategy will not, however, prevent attacks from Internet sites that refuse to screen outgoing transmissions. Second, security software is available for the targeted sites that scan for half-open connections. The software looks for SYN packets that have not been followed by an ACK packet. The clogged ports can then be restored to allow legitimate connections to use them.

DDoS attacks are the most difficult of the three to counter. The victim's site becomes inundated with messages from thousands of zombie sites that are distributed across the Internet. The company is rendered helpless because it cannot effectively block transmissions from so many different locations.

As a countermeasure to DDoS attacks, many organizations have invested in **Intrusion Prevention Systems (IPS)** that employ **deep packet inspection (DPI)** to determine when an attack is in progress. DPI uses a variety of analytical and statistical techniques to evaluate the contents of message packets. It searches the individual packets for protocol noncompliance and employs predefined criteria to decide if a packet can proceed to its destination. This is in contrast to the normal packet inspection that simply checks the header portion of a packet to determine its destination. By going deeper and examining the payload or body of the packet, DPI can identify and classify malicious packets based on a database of known attack signatures. Once classified as malicious, the packet can then be blocked and redirected to a security team and/or network reporting agent.

IPS works in-line with a firewall at the perimeter of the network to act as a filter that removes malicious packets from the flow before they can affect servers and networks. IPS may also be used behind the firewall to protect specific network segments and servers. This provides additional



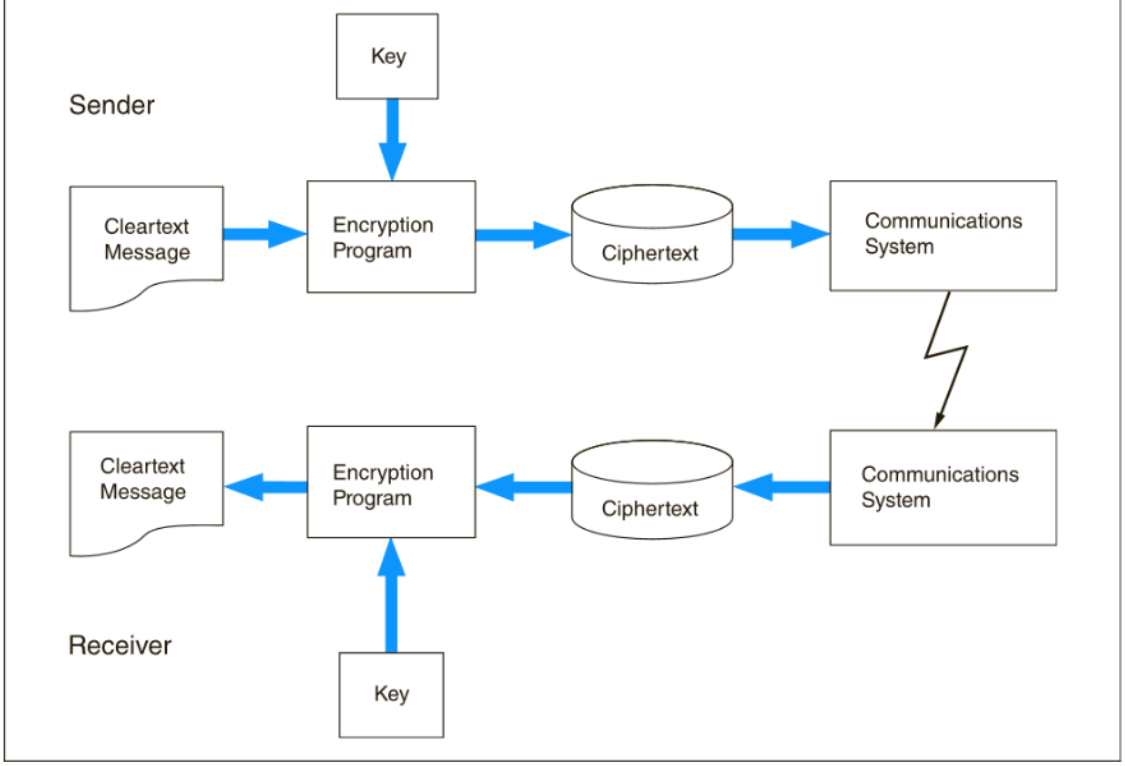
protection against careless laptop users who have been unknowingly infected with a Trojan horse or worm while working outside the protected network environment. IPS techniques can also be employed to protect an organization from becoming part of a botnet by inspecting outbound packets and blocking malicious traffic before it reaches the Internet.

Encryption

Encryption is the conversion of data into a secret code for storage in databases and transmission over networks. The discussion here pertains to transmitted data, but these basic principles apply also to stored data. The sender uses an encryption algorithm to convert the original message, called cleartext, into a coded equivalent, called ciphertext. At the receiving end, the ciphertext is decoded (decrypted) back into cleartext. The encryption algorithm uses a key, which is a binary number that typically is from 56 to 128 bits in length. The more bits in the key, the stronger the encryption method. Today, nothing less than 128-bit algorithms are considered truly secure. Two general approaches to encryption are **private key** and **public key encryption**.

PRIVATE KEY ENCRYPTION. Advanced encryption standard (AES) is a 128-bit encryption technique that has become a US government standard for private key encryption. The AES algorithm uses a single key known to both the sender and the receiver of the message. To encode a message, the sender provides the encryption algorithm with the key, which is used to produce a ciphertext message. The message enters the communication channel and is transmitted to the receiver's location, where it is stored. The receiver decodes the message with a decryption program that uses the same key the sender employs. Figure 16-5 illustrates this technique.

FIGURE 16-5 THE ADVANCED ENCRYPTION STANDARD TECHNIQUE



Triple-DES encryption is an enhancement to an older encryption technique called the Data Encryption Standard (DES). Triple DES provides considerably improved security over most single encryption techniques. Two forms of triple-DES encryption are EEE3 and EDE3. **EEE3** uses three different keys to encrypt the message three times. **EDE3** uses one key to encrypt the message. A second key is used to decode it. The resulting message is garbled because the key used for decoding is different from the one that encrypted it. Finally, a third key is used to encrypt the garbled message. The use of multiple keys greatly reduces the chances of breaking the cipher. Triple-DES encryption is thought to be very secure, and major banks use it to transmit transactions. Unfortunately, it is also very slow. The EEE3 and EDE3 techniques are illustrated in Figure 16-6.

All private key techniques have a common problem: The more individuals who need to know the key, the greater the probability of it falling into the wrong hands. If a perpetrator discovers the key, he or she can intercept and decipher coded messages. Therefore, encrypting data that are to be transmitted among large numbers of relative strangers (such as Internet transactions between businesses and customers) require a different approach. The solution to this problem is public key encryption.

PUBLIC KEY ENCRYPTION. Public key encryption uses two different keys: one for encoding messages and the other for decoding them. Each recipient has a private key that is kept secret and a public key that is published. The sender of a message uses the receiver's public key to encrypt the message. The receiver then uses his or her private key to decode the message. Users never need to share their private keys to decrypt messages, thus reducing the likelihood that they fall into the hands of a criminal.

RSA (Rivest-Shamir-Adleman) is a highly secure public key cryptography method. This method is, however, computationally intensive and much slower than standard DES encryption. Sometimes, both DES and RSA are used together in what is called a **digital envelope**. The actual message is encrypted using DES to provide the fastest decoding. The DES private key needed to decrypt the message is encrypted using RSA and transmitted along with the message. The receiver first decodes the DES key, which is then used to decode the message.

Digital Signatures

A **digital signature** is electronic authentication that cannot be forged. It ensures that the message or document the sender transmitted was not tampered with after the signature was applied. Figure 16-7 illustrates this process. The sender uses a one-way hashing algorithm to calculate a **digest** of the text message. The digest is a mathematical value calculated from the text content of the message. The digest is then encrypted using the sender's private key to produce the digital signature. Next, the digital signature and the text message are encrypted using the receiver's public key and transmitted to the receiver. At the receiving end, the message is decrypted using the receiver's private key to produce the digital signature (encrypted digest) and the cleartext version of the message. The receiver then uses the sender's public key to decrypt the digital signal to produce the digest. Finally, the receiver recalculates the digest from the cleartext using the original hashing algorithm and compares this to the decoded digest. If the message is authentic, the two digest values will match. If even a single character of the message was changed in transmission, the digest figures will not be equal.

Digital Certificate

The aforementioned process proves that the message received was not tampered with during transmission. It does not prove, however, that the sender is who he or she claims to be. The sender could be an impersonator. Verifying the sender's identity requires a **digital certificate**, which a trusted third party issues, called a **certification authority (CA)**. A digital certificate is used in conjunction with a public key encryption system to authenticate the sender of a message. The process for certification varies depending on the level of certification desired. It involves establishing one's identity with formal documents, such as a driver's license, notarization, and fingerprints, and

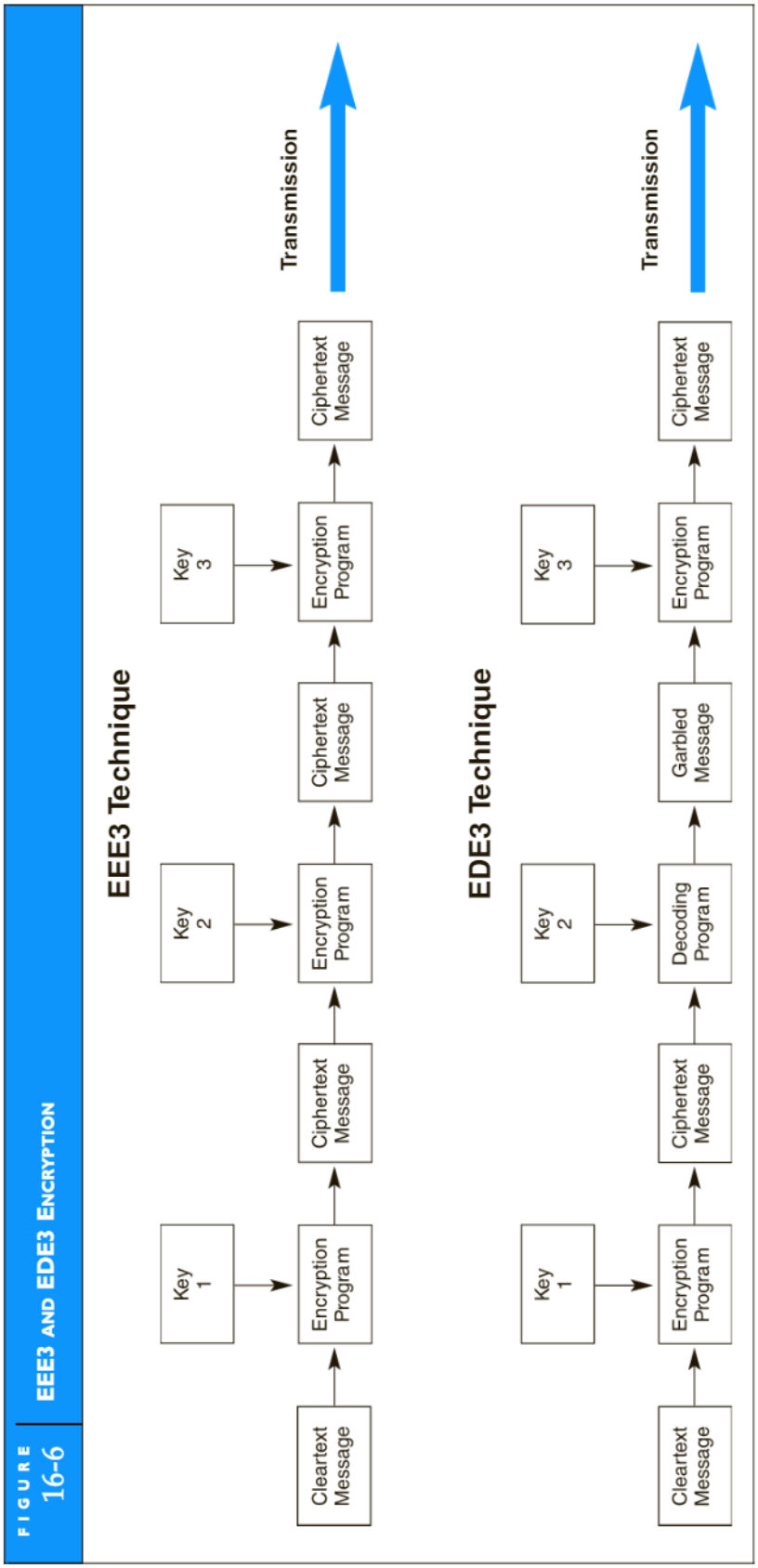


FIGURE 16-6 EEE3 AND EDE3 ENCRYPTION

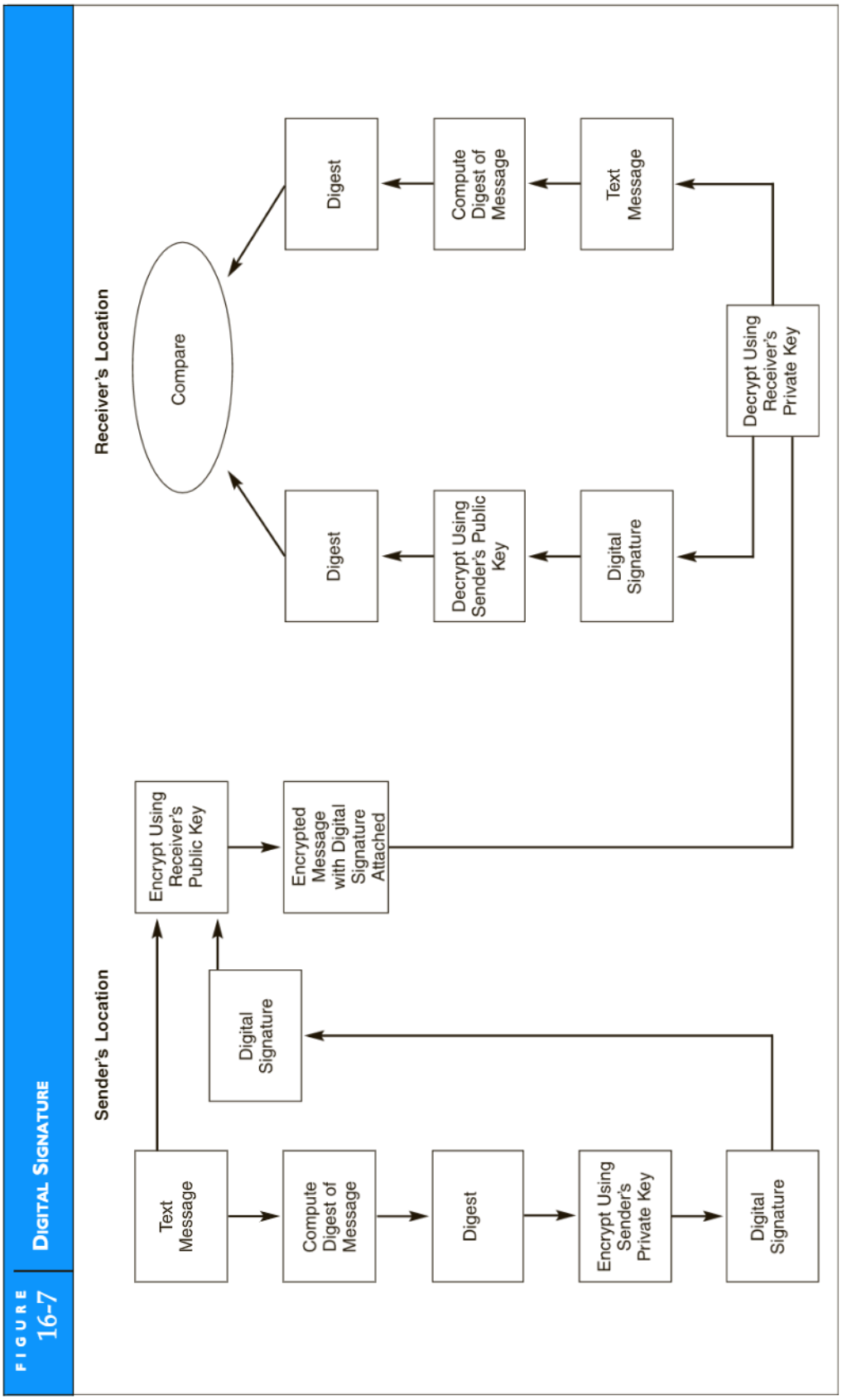


FIGURE 16-7 DIGITAL SIGNATURE

proving one's ownership of the public key. After verifying the owner's identity, the CA creates the certification, which is the owner's public key, and other data the CA has digitally signed.

The digital certificate is transmitted with the encrypted message to authenticate the sender. The receiver uses the CA's public key, which is widely publicized, to decrypt the sender's public key attached to the message. The sender's public key is then used to decrypt the message.

Message Sequence Numbering

An intruder in the communications channel may attempt to delete a message from a stream of messages, change the order of messages received, or duplicate a message. Through **message sequence numbering**, a sequence number is inserted in each message, and any such attempt will become apparent at the receiving end.

Message Transaction Log

An intruder may successfully penetrate the system by trying different password and user ID combinations. Therefore, all incoming and outgoing messages, as well as attempted (failed) access, should be recorded in a **message transaction log**. The log should record the user ID, the time of the access, and the terminal location or telephone number from which the access originated.

Request-Response Technique

An intruder may attempt to prevent or delay the receipt of a message from the sender. When senders and receivers are not in constant contact, the receiver may not know whether the communications channel has been interrupted and messages have been diverted. Using the **request-response technique**, a control message from the sender and a response from the receiver are sent at periodic, synchronized intervals. The timing of the messages should follow a random pattern that will be difficult for the intruder to determine and circumvent.

Call-Back Devices

As we have seen, networks can be equipped with security features, such as passwords, authentication devices, and encryption. The common weakness to all of these technologies is that they apply the security measure after the criminal has connected to the network server. Many believe that the key to security is to keep the intruder off the network to begin with.

A **call-back device** requires the dial-in user to enter a password and be identified. The system then breaks the connection to perform user authentication. If the caller is authorized, the call-back device dials the caller's number to establish a new connection. This restricts access to only authorized terminals or telephone numbers and prevents an intruder masquerading as a legitimate user.

Audit Objectives Relating to Subversive Threats

The auditor's objective is to verify the security and integrity of financial transactions by determining that network controls (1) can prevent and detect illegal access both internally and from the Internet, (2) will render useless any data that a perpetrator successfully captures, and (3) are sufficient to preserve the integrity and physical security of data connected to the network.

Audit Procedures Relating to Subversive Threats

To achieve these control objectives, the auditor may perform the following tests of controls:

1. Review the adequacy of the firewall in achieving the proper balance between control and convenience based on the organization's business objectives and potential risks. Criteria for assessing the firewall effectiveness include:
   - *Flexibility.* The firewall should be flexible enough to accommodate new services as the security needs of the organization change.

- *Proxy services.* Adequate proxy applications should be in place to provide explicit user authentication to sensitive services, applications, and data.
- *Filtering.* Strong filtering techniques should be designed to deny all services that are not explicitly permitted. In other words, the firewall should specify only those services the user is permitted to access, rather than specifying the services that are denied.
- *Segregation of systems.* Systems that do not require public access should be segregated from the Internet.
- *Audit tools.* The firewall should provide a thorough set of audit and logging tools that identify and record suspicious activity.
- *Probe for weaknesses.* To validate security, the auditor (or a professional security analyst) should periodically probe the firewall for weaknesses, just as a computer Internet hacker would do. A number of software products are currently available for identifying security weaknesses.

2. Verify that an Intrusion Prevention Systems (IPS) with deep packet inspection (DPI) is in place for organizations that are vulnerable to DDoS attacks, such as financial institutions.
3. Review security procedures governing the administration of data encryption keys.
4. Verify the encryption process by transmitting a test message and examining the contents at various points along the channel between the sending and receiving locations.
5. Review the message transaction logs to verify that all messages were received in their proper sequence.
6. Test the operation of the call-back feature by placing an unauthorized call from outside the installation.

CONTROLLING RISKS FROM EQUIPMENT FAILURE

Line Errors

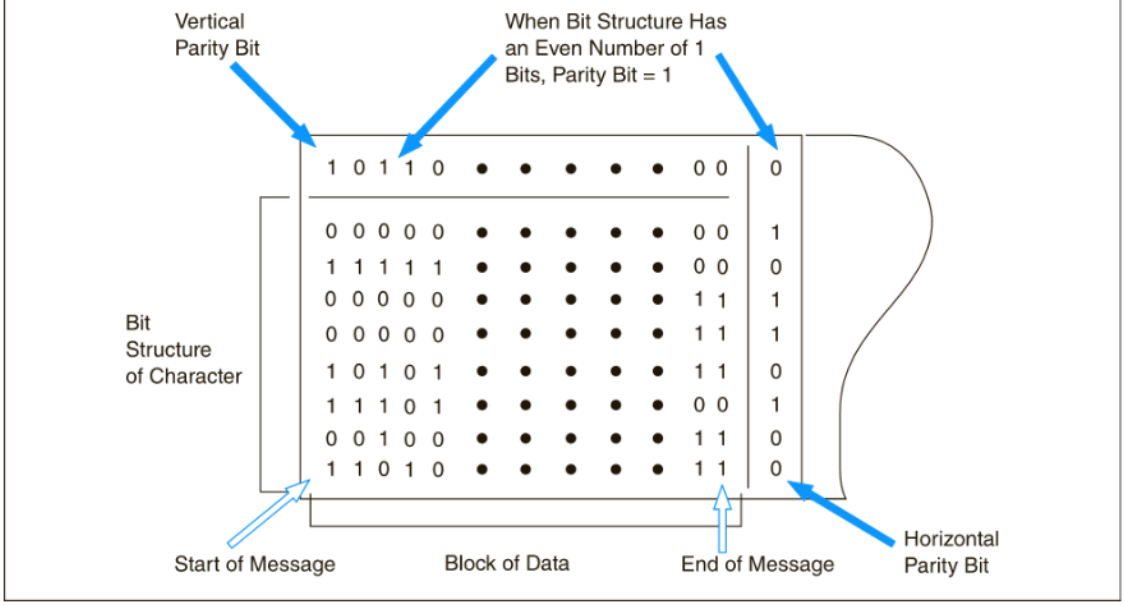
The most common problem in data communications is data loss due to **line error**. The bit structure of the message can be corrupted through noise on the communications lines. Noise is made up of random signals that can interfere with the message signal when they reach a certain level. Electric motors, atmospheric conditions, faulty wiring, defective components in equipment, or noise spilling over from an adjacent communications channel may cause these random signals. If not detected, bit structure changes to transmitted data can be catastrophic to the firm. For example, in the case of a database update program, the presence of line errors can result in incorrect transaction values being posted to the accounts. The following two techniques are commonly used to detect and correct such data errors before they are processed.

ECHO CHECK. The **echo check** involves the receiver of the message returning the message to the sender. The sender compares the returned message with a stored copy of the original. If there is a discrepancy between the returned message and the original, suggesting a transmission error, the message is retransmitted. This technique reduces, by one-half, throughput over communications channels. Using full-duplex channels, which allow both parties to transmit and receive simultaneously, can increase throughput.

PARITY CHECK. The **parity check** incorporates an extra bit (the parity bit) into the structure of a bit string when it is created or transmitted. Parity can be both vertical and horizontal (longitudinal). Figure 16-8 illustrates both types of parities. Vertical parity adds the parity bit to each character in the message when the characters are originally coded and stored in magnetic form. For example, the number of 1 bits in the bit structure of each character is counted. If the number is even (for instance, there are four 1 bits in a given eight-bit character), the system assigns the parity bit a value of one. If the number of 1 bits is odd, a 0 parity bit is added to the bit structure.

The concern is that during transmission, a 1 bit will be converted to a 0 bit or vice versa, thus destroying the bit structure integrity of the character. In other words, the original character is incorrectly presented as a different yet valid character. Errors of this sort, if undetected, could alter

FIGURE 16-8 VERTICAL AND HORIZONTAL PARITY USING ODD PARITY



financial numbers. A parity check can detect errors at the receiving end. The system again counts the 1 bits, which should always equal an odd number. If a 1 bit is added to or removed from the bit structure during transmission, the number of 1 bits for the character will be even, which would signal an error.

The problem with using vertical parity alone is the possibility that an error will change two bits in the structure simultaneously, thus retaining the parity of the character. In fact, some estimates indicate a 40 to 50 percent chance that line noise will corrupt more than one bit within a character. Using horizontal parity in conjunction with vertical parity reduces this problem. In Figure 16-8, notice the parity bit following each block of characters. The combination of vertical and horizontal parity provides a higher degree of protection from line errors.

Audit Objectives Relating to Equipment Failure

The auditor's objective is to verify the integrity of the electronic commerce transactions by determining that controls are in place to detect and correct message loss due to equipment failure.

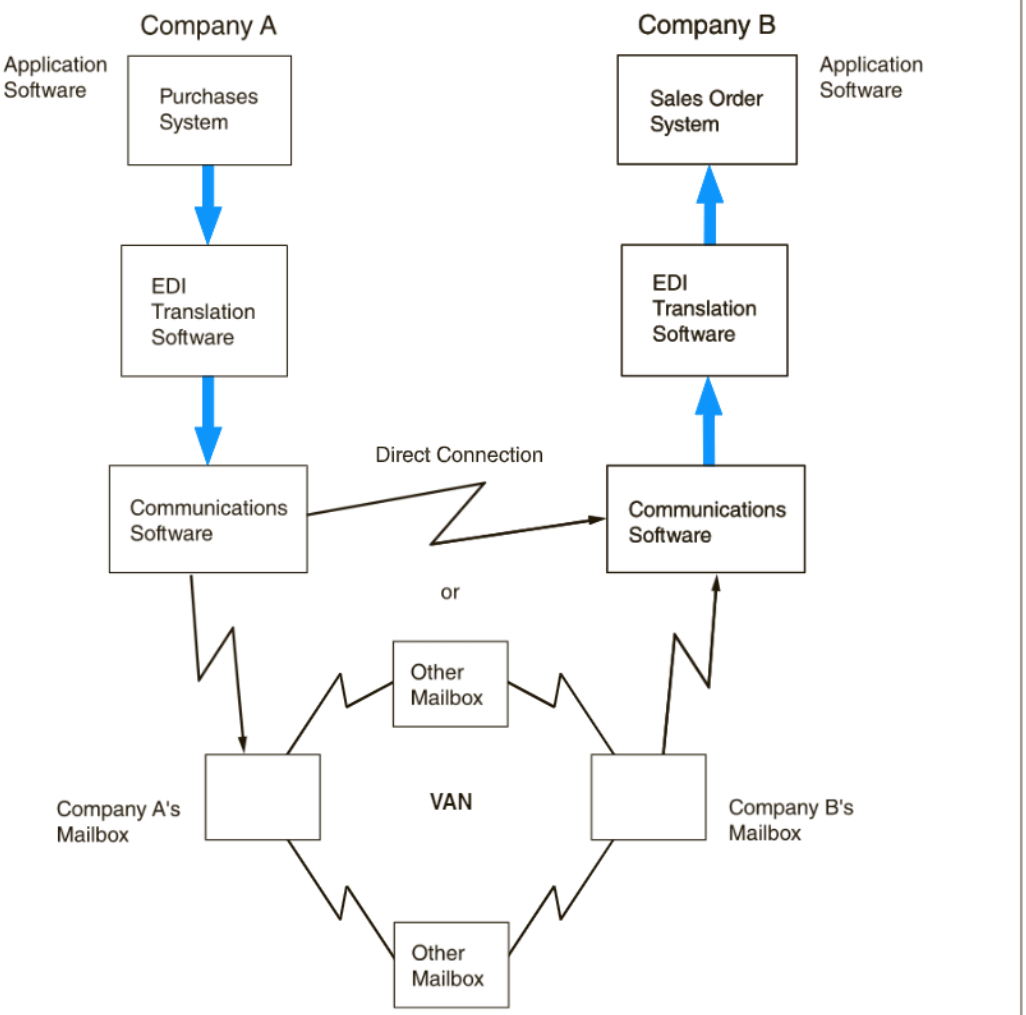
Audit Procedures Relating to Equipment Failure

To achieve this control objective, the auditor can select a sample of messages from the transaction log and examine them for garbled contents that line noise causes. The auditor should verify that all corrupted messages were successfully retransmitted.

Electronic Data Interchange (EDI) Controls

EDI substantially changes the way companies do business and creates unique control issues that accountants need to recognize. Before examining these issues, let's first review the EDI concept. Figure 16-9 illustrates the data flow through the basic elements of an EDI system that links two trading partners—the customer (Company A) and the vendor (Company B). When Company A wishes to place an order with Company B, Company A's purchases system automatically creates and sends an electronic purchase order to its EDI translation software. The translation software converts the purchase order from Company A's internal format to a standard format, such as

FIGURE 16-9 EDI SYSTEM



ANSI X.12. Next, the communications software adds the protocols to the message to prepare it for transmission over the communication channel. The transmission may be either a direct connection between the trading partners or an indirect connection through a value-added network (VAN). At Company B, the process is reversed, yielding a sales order in Company B's internal format, which its sales order system processes automatically.

The absence of human intervention in this process presents a unique twist to traditional control problems, including ensuring that transactions are authorized and valid, preventing unauthorized access to data files, and maintaining an audit trail of transactions. The following techniques are used in dealing with these issues.

TRANSACTION AUTHORIZATION AND VALIDATION

Both the customer and the supplier must establish that the transaction being processed is to (or from) a valid trading partner and is authorized. This can be accomplished at three points in the process.

1. Some VANs have the capability of validating passwords and user ID codes for the vendor by matching these against a valid customer file. The VAN rejects any unauthorized trading partner transactions before they reach the vendor's system.

2. Before being converted, the translation software can validate the trading partner's ID and password against a validation file in the firm's database.
3. Before processing, the trading partner's application software references the valid customer and vendor files to validate the transaction.

ACCESS CONTROL

To function smoothly, EDI trading partners must permit a degree of access to private data files that would be forbidden in a traditional environment. The trading partner agreement will determine the degree of access control in place. For example, it may permit the customer's system to access the vendor's inventory files to determine if inventories are available. Also, trading partners may agree that the prices on the purchase order will be binding on both parties. The customer must, therefore, periodically access the vendor's price list file to keep pricing information current. Alternatively, the vendor may need access to the customer's price list to update prices.

To guard against unauthorized access, each company must establish valid vendor and customer files. Inquiries against databases can thus be validated, and unauthorized attempts at access can be rejected. User authority tables can also be established, which specify the degree of access a trading partner is allowed. For example, the partner may be authorized to read inventory or pricing data but not change values.

EDI AUDIT TRAIL

The absence of source documents in EDI transactions eliminates the traditional audit trail and restricts the ability of accountants to verify the validity, completeness, timing, and accuracy of transactions. One technique for restoring the audit trail is to maintain a control log, which records the transaction's flow through each phase of the EDI system. Figure 16-10 illustrates how this approach may be employed.

As the transaction is received at each stage in the process, an entry is made in the log. In the customer's system, the transaction log can be reconciled to ensure that all transactions the purchases system initiated were correctly translated and communicated. Likewise, in the vendor's system, the control log will establish that the sales order system correctly translated and processed all messages that the communications software received.

Audit Objectives Relating to EDI

The auditor's objectives are to determine that (1) all EDI transactions are authorized, validated, and in compliance with the trading partner agreement; (2) no unauthorized organizations gain access to database records; (3) authorized trading partners have access only to approved data; and (4) adequate controls are in place to ensure a complete audit trail of all EDI transactions.

Audit Procedures Relating to EDI

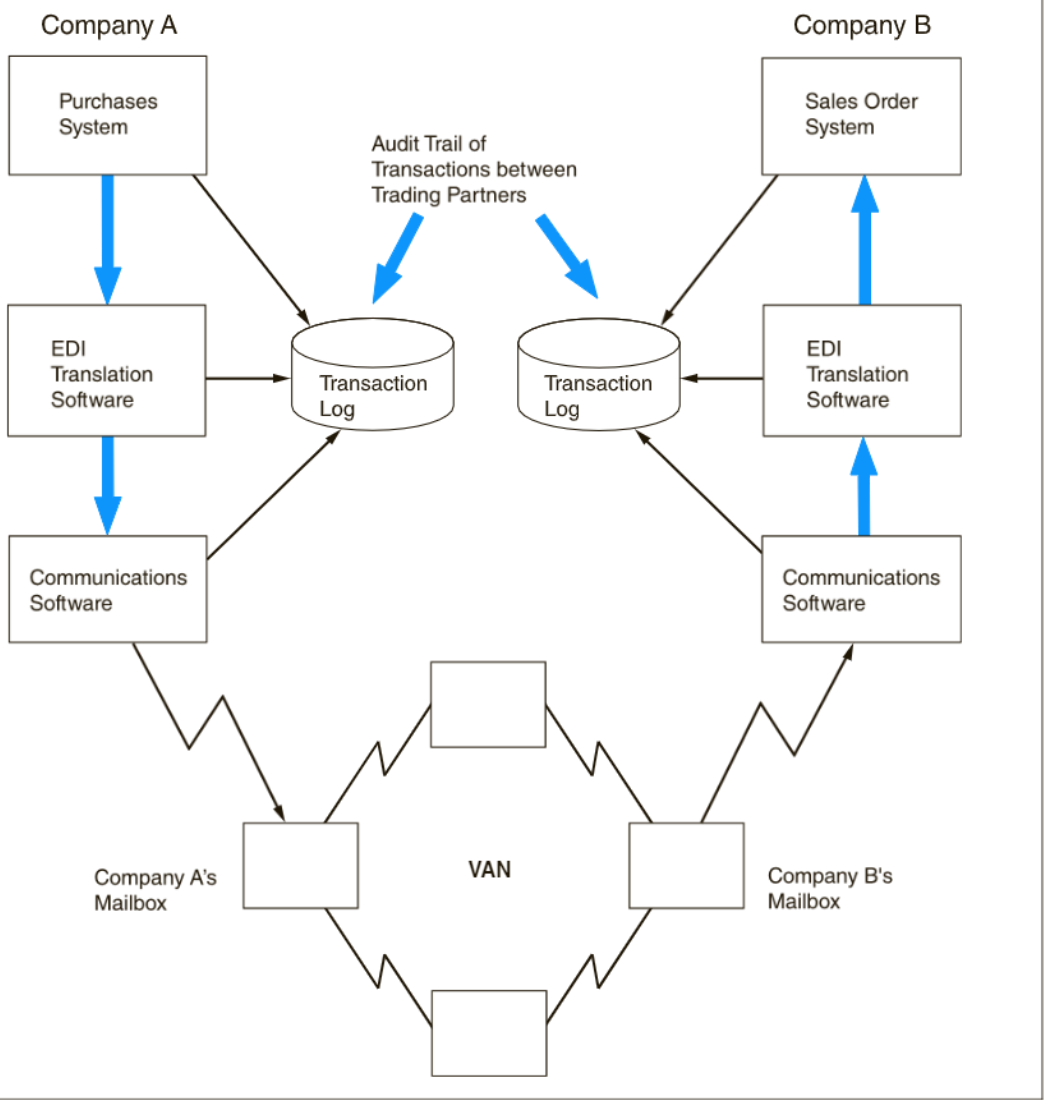
To achieve these control objectives, the auditor may perform the following tests of controls.

TESTS OF AUTHORIZATION AND VALIDATION CONTROLS. The auditor should establish that trading partner identification codes are verified before transactions are processed. To accomplish this, the auditor should (1) review agreements with the VAN facility to validate transactions and ensure that information regarding valid trading partners is complete and correct, and (2) examine the organization's valid trading partner file for accuracy and completeness.

TESTS OF ACCESS CONTROLS. Security over the valid trading partner file and databases is central to the EDI control framework. The auditor can verify control adequacy in the following ways:

1. The auditor should determine that access to the valid vendor or customer file is limited to authorized employees only. The auditor should verify that passwords and authority tables control access to this file and that the data are encrypted.

FIGURE 16-10 EDI System Using Transaction Control Log for Audit Trail



2. The trading agreement will determine the degree of access a trading partner should have to the firm's database records (such as inventory levels and price lists). The auditor should reconcile the terms of the trading agreement against the trading partner's access privileges stated in the database authority table.
3. The auditor should simulate access by a sample of trading partners and attempt to violate access privileges.

TESTS OF AUDIT TRAIL CONTROLS. The auditor should verify that the EDI system produces a transaction log that tracks transactions through all stages of processing. By selecting a sample of transactions and tracing these through the process, the auditor can verify that key data values were recorded correctly at each point.

Summary

This chapter continued the discussion of IT general controls and audit tests begun in Chapter 15. It examined the risks and controls over operating systems, database management systems, networks, and EDI systems. The principal threats to the operating system are (l) unauthorized access, (2) intentional or unintentional insertion of viruses, and (3) loss of data due to system malfunctions.

Unauthorized access to the database can be effectively controlled through the use of well-designed user views, authorization rules, user-defined procedures, and data encryption. Backup and recovery techniques can be used to safeguard data against system malfunctions. Networks and communication links are susceptible to exposures from both criminal subversion and equipment failure. Subversive threats can be minimized through a variety of security and access control measures, including firewalls, IPS, DPI, data encryption, and call-back devices. Equipment failure usually takes the form of line errors, which noise in communication lines causes. These can be effectively reduced through echo checks and parity checks. The discussion then turned to EDI, in which firms are faced with a variety of exposures that arise in connection with an environment void of human intermediaries to authorize or review transactions. Controls in an EDI environment are achieved primarily through programmed procedures to authorize transactions, limit access to data files, and ensure that transactions the system processes are valid.

Appendix

Malicious and Destructive Programs

Virus

A virus is a program (usually destructive) that attaches itself to a legitimate program to penetrate the operating system and destroy application programs, data files, and the operating system itself. An insidious aspect of a virus is its ability to spread throughout the host system and to other systems before perpetrating its destructive acts. Often, a virus will have a built-in counter that will trigger its destructive role only after it has copied itself a specified number of times to other programs and systems. The virus thus grows geometrically, which makes tracing its origin extremely difficult.

Personal computers are a major source of virus penetration. When connected in a network or a mainframe, an infected PC can upload the virus to the host system. Once in the host, the virus can spread throughout the operating system and to other users. Virus programs usually attach themselves to the following types of files:

1. An .EXE or .COM program file
2. An .OVL (overlay) program file
3. The boot sector of a disk
4. A device driver program

Mechanisms for spreading viruses include e-mail attachments, downloading of public-domain programs from the Internet, and using illegal bootleg software. Because of the general lack of control in PC operating systems, microcomputers connected to mainframes pose a serious threat to the mainframe environment as well.

Worm

The term *worm* is used interchangeably with *virus.* A worm is a software program that virtually burrows into the computer's memory and replicates itself into areas of idle memory. The worm systematically occupies idle memory until the memory is exhausted and the system fails. Technically, worms differ from viruses in that the replicated worm modules remain in contact with the original worm that controls their growth, whereas the replicated virus modules grow independently.

Logic Bomb

A logic bomb is a destructive program, such as a virus, that some predetermined event triggers. Often a date (such as Friday the 13th, April Fool's Day, or the 4th of July) will be the logic bomb's trigger. Events of less public prominence, such as the dismissal of an employee, have also triggered these bombs. For example, during the customary two-week severance period, a terminated programmer may embed a logic bomb in the system that will get activated six months after his or her departure from the firm.

Back Door

A back door (also called a trap door) is a software program that allows unauthorized access to a system without going through the normal (front door) log-on procedure. Programmers who want to provide themselves with unrestricted access to a system that they are developing for users may create a log-on procedure that will accept either the user's private password or their own secret password, thus creating a back door to the system. The purpose of the back door may be to provide easy access to perform program maintenance, or it may be to perpetrate a fraud or insert a virus into the system.

Trojan Horse

A Trojan horse is a program whose purpose is to capture IDs and passwords from unsuspecting users. These programs are designed to mimic the normal log-on procedures of the operating system. When the user enters his or her ID and password, the Trojan horse stores a copy of them in a secret file. At some later date, the author of the Trojan horse uses these IDs and passwords to access the system and masquerade as an authorized user.

Key Terms

access control list (683)
access controls (688)
access token (682)
advanced encryption standard (AES) (695)
application-level firewalls (693)
backup controls (688)
biometric devices (690)
call-back device (699)
certification authority (CA) (696)
checkpoint feature (691)
compilers (681)
data encryption (690)
database authorization table (689)
deep packet inspection (DPI) (693)
digest (696)
digital certificate (696)
digital envelope (696)
digital signature (696)
discretionary access privileges (683)
echo check (700)
EDE3 (696)
EEE3 (696)
encryption (695)
event monitoring (687)