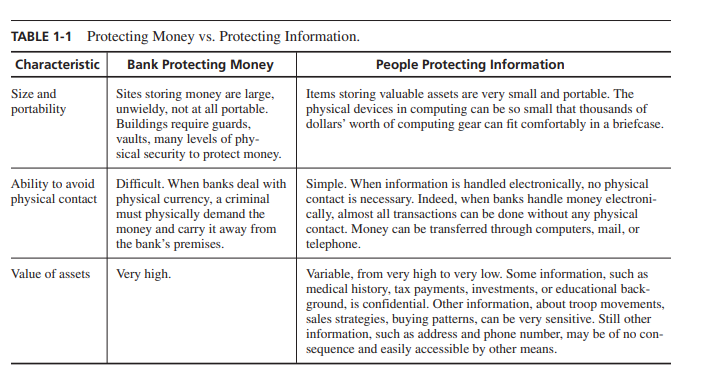
**Section 1.1**

**What Does Secure Mean?**



**Computing System:** A collection of hardware, software, storage media, data, and people that an organization uses to perform computing tasks.

Any system is most vulnerable at its **weakest point.** A robber intent on stealing something from your house will not attempt to penetrate a two-inch-thick metal door if a window gives easier access.

**Principle of Easiest Penetration**: An intruder must be expected to use any available means of penetration. The penetration may not necessarily be by the most obvious means, nor is it necessarily the one against which the most solid defense has been installed. And it certainly does not have to be the way we want the attacker to behave.

Computer security is a game with rules only for the defending team: The attackers can (and will) use any means they can.

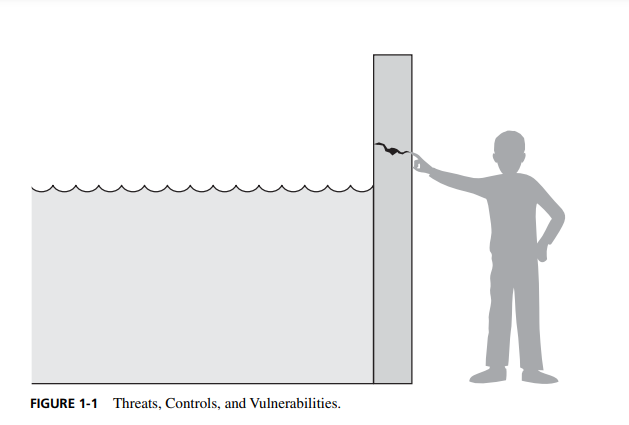
**Section 1.2**

**Attacks**

A computer-based system has three separate but valuable components: **hardware, software, and data**.

**Vulnerability:**  A weakness in the security system, for example, in procedures, design, or implementation, that might be exploited to cause loss or harm. For instance, a particular system may be vulnerable to unauthorized data manipulation because the system does not verify a user’s identity before allowing data access.

**Threat: i**s a set of circumstances that has the potential to cause loss or harm. To see the difference between a threat and a vulnerability, consider the illustration in Figure 1-1. Here, a wall is holding water back. The water to the left of the wall is a threat to the man on the right of the wall: The water could rise, overflowing onto the man, or it could stay beneath the height of the wall, causing the wall to collapse. So the threat of harm is the potential for the man to get wet, get hurt, or be drowned. For now, the wall is intact, so the threat to the man is unrealized. However, we can see a small crack in the wall—a vulnerability that threatens the man’s security. If the water rises to or beyond the level of the crack, it will exploit the vulnerability and harm the man



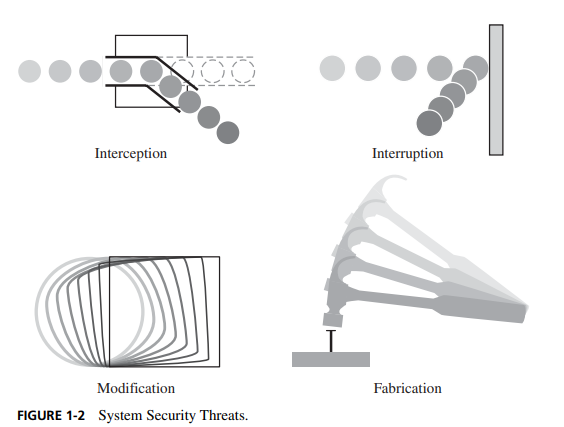
A human who exploits a vulnerability perpetrates an **attack** on the system. An attack can also be launched by another system, as when one system sends an overwhelming set of messages to another, virtually shutting down the second system’s ability to function. Unfortunately, we have seen this type of attack frequently, as denial-of-service attacks flood servers with more messages than they can handle.

How do we address these problems? We use a **control** as a protective measure. That is, a control is an action, device, procedure, or technique that removes or reduces a vulnerability. In Figure 1-1, the man is placing his finger in the hole, controlling the threat of water leaks until he finds a more permanent solution to the problem.

In general, we can describe the relationship among threats, controls, and vulnerabilities in this way:

A **threat** is blocked by **control** of **vulnerability.**

**Types of Threats**



**Interception**: means that some unauthorized party has gained access to an asset. The outside party can be a person, a program, or a computing system. Examples of this type of failure are illicit copying of program or data files, or wiretapping to obtain data in a network. Although a loss may be discovered fairly quickly, a silent interceptor may leave no traces by which the interception can be readily detected.

**Interruption**, an asset of the system becomes lost, unavailable, or unusable. An example is malicious destruction of a hardware device, erasure of a program or data file, or malfunction of an operating system file manager so that it cannot find a particular disk file.

**Modification:** If an unauthorized party not only accesses but tampers with an asset. . For example, someone might change the values in a database, alter a program so that it performs an additional computation, or modify data being transmitted electronically. It is even possible to modify hardware. Some cases of modification can be detected with simple measures, but other, more subtle, changes may be almost impossible to detect

**Fabrication**: An unauthorized party might create counterfeit objects on a computing system. The intruder may insert spurious transactions to a network communication system or add records to an existing database. Sometimes these additions can be detected as forgeries, but if skillfully done, they are virtually indistinguishable from the real thing.

**Method, Opportunity, and Motive (MOM)**

A malicious attacker must have these three things:

* Method: the skills, knowledge, tools, and other things to be able to pull off the attack
* Opportunity: The time and access to accomplish the attack
* Motive: A reason to want to perform the attack against this system

**Section 1.3**

**The Meaning of Computer Security**

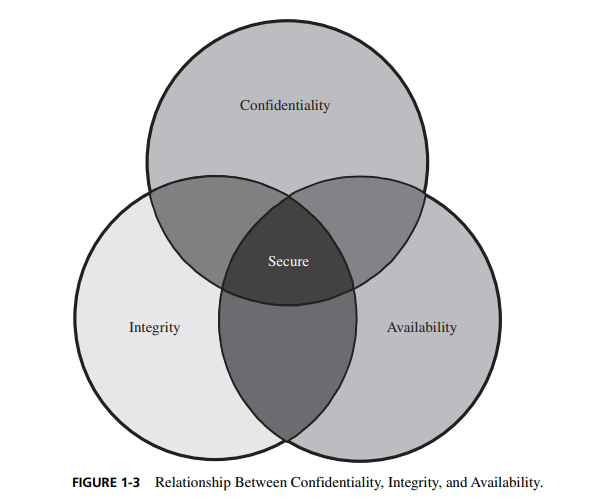
When we talk about computer security, we mean that we are addressing three important aspects of any computer-related system: **confidentiality, integrity, and availability.**

**Confidentiality:** ensures that computer-related assets are accessed only by authorized parties. That is, only those who should have access to something will actually get that access. By “access,” we mean not only reading but also viewing, printing, or simply knowing that a particular asset exists. Confidentiality is sometimes called ***secrecy*** or ***privacy***.

**Integrity:** means that assets can be modified only by authorized parties or only in authorized ways. In this context, modification includes writing, changing, changing status, deleting, and creating.

**Availability:** means that assets are accessible to authorized parties at appropriate times. In other words, if some person or system has legitimate access to a particular set of objects, that access should not be prevented. For this reason, availability is sometimes known by its opposite, ***denial of service***.

Security in computing addresses these three goals. One of the challenges in building a secure system is finding the right balance among the goals, which often conflict. For example, it is easy to preserve a particular object’s confidentiality in a secure system simply by preventing everyone from reading that object. However, this system is not secure, because it does not meet the requirement of availability for proper access. That is, there must be a balance between confidentiality and availability.



**Confidentiality**

You may find the notion of confidentiality to be straightforward: Only authorized people or systems can access protected data. However, as we see in later chapters, ensuring confidentiality can be difficult. For example, who determines which people or systems are authorized to access the current system? By “accessing” data, do we mean that an authorized party can access a single bit? the whole collection? pieces of data out of context? Can someone who is authorized disclose those data to other parties?

Confidentiality is the security property we understand best because its meaning is narrower than the other two. We also understand confidentiality well because we can relate computing examples to those of preserving confidentiality in the real world

**Integrity**

When we survey the way some people use the term, we find several different meanings. For example, if we say that we have preserved the integrity of an item, we may mean that the item is

• precise

• accurate

• unmodified

• modified only in acceptable ways

• modified only by authorized people

• modified only by authorized processes

• consistent

• internally consistent

• meaningful and usable

Integrity can also mean two or more of these properties. Welke and Mayfield recognize three particular aspects of integrity—authorized actions, separation and protection of resources, and error detection and correction. Integrity can be enforced in much the same way as can confidentiality: by rigorous control of who or what can access which resources in what ways. Some forms of integrity are well represented in the real world, and those precise representations can be implemented in a computerized environment. But not all interpretations of integrity are well reflected by computer implementations.

**Availability**

Availability applies both to data and to services (that is, to information and to information processing), and it is similarly complex. As with the notion of confidentiality, different people expect availability to mean different things. For example, an object or service is thought to be available if

• It is present in a usable form.

• It has capacity enough to meet the service’s needs.

• It is making clear progress, and, if in wait mode, it has a bounded waiting time.

• The service is completed in an acceptable period of time.

We can construct an overall description of availability by combining these goals. We say a data item, service, or system is available if

• There is a timely response to our request.

• Resources are allocated fairly so that some requesters are not favored over others.

• The service or system involved follows a philosophy of fault tolerance, whereby hardware or software faults lead to graceful cessation of service or to work-arounds rather than to crashes and abrupt loss of information.

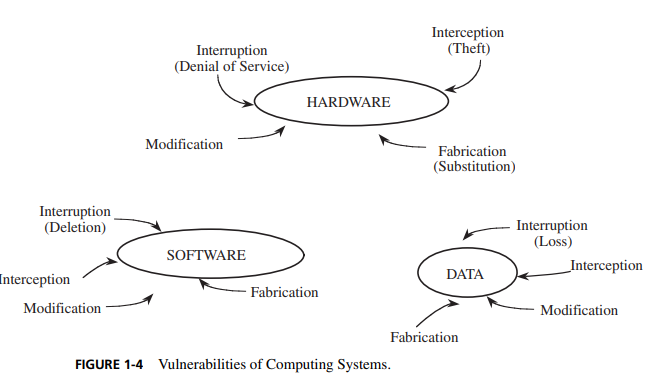
• The service or system can be used easily and in the way it was intended to be used.

• Concurrency is controlled; that is, simultaneous access, deadlock management, and exclusive access are supported as required.

As you can see, expectations of availability are far-reaching. Indeed, the security community is just beginning to understand what availability implies and how to ensure it. A small, centralized control of access is fundamental to preserving confidentiality and integrity, but it is not clear that a single access control point can enforce availability. Much of computer security’s past success has focused on confidentiality and integrity; full implementation of availability is security’s next great challenge.

**Vulnerabilities**

Rather than to start with the security goals themselves. Figure 1-4 shows the types of vulnerabilities we might find as they apply to the assets of hardware, software, and data. These three assets and the connections among them are all potential security weak points. Let us look in turn at the vulnerabilities of each asset.



**Hardware Vulnerabilities**

In particular, deliberate attacks on equipment, intending to limit availability, usually involve theft or destruction. Managers of major computing centers long ago recognized these vulnerabilities and installed physical security systems to protect their machines. However, the proliferation of PCs, especially laptops, as office equipment has resulted in several thousands of dollars’ worth of equipment sitting unattended on desks outside the carefully protected computer room.

**Software Vulnerabilities**

Software can be replaced, changed, or destroyed maliciously, or it can be modified, deleted, or misplaced accidentally. Whether intentional or not, these attacks exploit the software’s vulnerabilities. It is possible to change a program so that it does all it did before, and then some. That is, a malicious intruder can “enhance” the software to enable it to perform functions you may not find desirable. In this case, it may be very hard to detect that the software has been changed, let alone to determine the extent of the change.

*Software Deletion*

Because of software’s high value to a commercial computing center, access to software is usually carefully controlled through a process called **configuration management** so that software cannot be deleted, destroyed, or replaced accidentally.

*Software Modification*

Software is vulnerable to modifications that either cause it to fail or cause it to perform an unintended task. Indeed, because software is so susceptible to “off by one” errors, it is quite easy to modify. Changing a bit or two can convert a working program into a failing one. Depending on which bit was changed, the program may crash when it begins, or it may execute for some time before it falters.

The program may be maliciously modified to fail when certain conditions are met or when a certain date or time is reached. Because of this delayed effect, such a program is known as a **logic bomb**. For example, a disgruntled employee may modify a crucial program so that it accesses the system date and halts abruptly after July 1. The employee might quit on May l and plan to be at a new job mile away by July

Other categories of software modification include:

**Trojan horse**: a program that overtly does one thing while covertly doing another

**Virus**: a specific type of Trojan horse that can be used to spread its “infection” from one computer to another

**Trapdoor**: a program that has a secret entry point

**Information leaks in a program**: code that makes information accessible to unauthorized people or programs

*Software Theft*

This attack includes unauthorized copying of software. Software authors and distributors are entitled to fair compensation for use of their product, as are musicians and book authors. Unauthorized copying of software has not been stopped satisfactorily. As we see in Chapter 11, the legal system is still grappling with the difficulties of interpreting paper-based copyright laws for electronic media.

**Data Vulnerabilities**

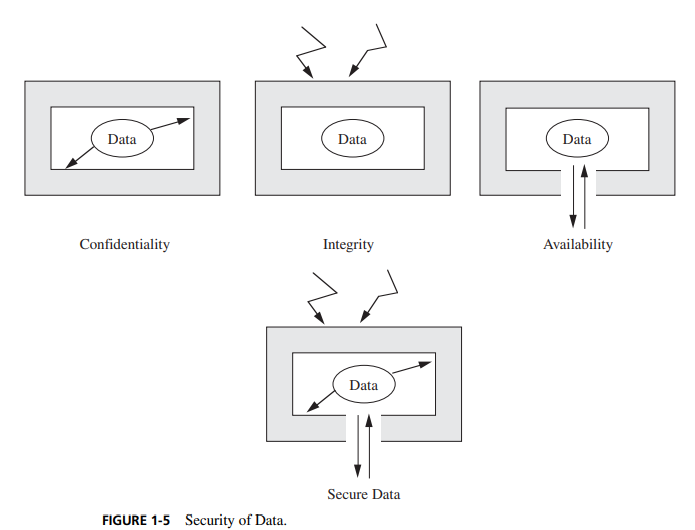
Printed data can be readily interpreted by the general public. Because of its visible nature, a data attack is a more widespread and serious problem than either a hardware or software attack. Thus, data items have greater public value than hardware and software because more people know how to use or interpret data.

Typically, both hardware and software have a relatively long life. No matter how they are valued initially, their value usually declines gradually over time. By contrast, the value of data over time is far less predictable or consistent. Initially, data may be valued highly. However, some data items are of interest for only a short period of time, after which their value declines precipitously.

For instance, suppose an analyst develops the data 24 hours before their release and then wishes to communicate the results to other analysts for independent verification before release. The data vulnerability here is clear, and, to the right people, the data are worth more before the scheduled release than afterward.

**Principle of Adequate Protection:** Computer items must be protected only until they lose their value. They must be protected to a degree consistent with their value.

Figure 1-5 illustrates how the three goals of security apply to data. In particular, confidentiality prevents unauthorized disclosure of a data item, integrity prevents unauthorized modification, and availability prevents denial of authorized access.



*Data Confidentiality*

Data can be gathered by many means, such as tapping wires, planting bugs in output devices, sifting through trash receptacles, monitoring electromagnetic radiation, bribing key employees, inferring one data point from other values, or simply requesting the data. Because data are often available in a form people can read, the confidentiality of data is a major concern in computer security.

Data are not just numbers on paper; computer data include digital recordings such as CDs and DVDs, digital signals such as network and telephone traffic, and broadband communications such as cable and satellite TV. Other forms of data are biometric identifiers embedded in passports, online activity preferences, and personal information such as financial records and votes. Protecting this range of data types requires many different approaches.

*Data Integrity*

Data are especially vulnerable to modification. Small and skillfully done modifications may not be detected in ordinary ways. For instance, we saw in our truncated interest example that a criminal can perform what is known as a **salami attack**: The crook shaves a little from many accounts and puts these shavings together to form a valuable result, like the meat scraps joined in a salami.

A more complicated process is trying to reprocess used data items. With the proliferation of telecommunications among banks, a fabricator might intercept a message ordering one bank to credit a given amount to a certain person’s account. The fabricator might try to **replay** that message, causing the receiving bank to credit the same account again. The fabricator might also try to modify the message slightly, changing the account to be credited or the amount, and then transmit this revised message.

**Other Exposed Assets**

*Networks*

Networks are specialized collections of hardware, software, and data. Each network node is itself a computing system; as such, it experiences all the normal security problems. In addition, a network must confront communication problems that involve the interaction of system components and outside resources. The problems may be introduced by a very exposed storage medium or access from distant and potentially untrustworthy computing systems.

Thus, networks can easily multiply the problems of computer security. The challenges are rooted in a network’s lack of physical proximity, use of insecure shared media, and the inability of a network to identify remote users positively.

*Access*

Access to computing equipment leads to three types of vulnerabilities. In the first, an intruder may steal computer time to do general-purpose computing that does not attack the integrity of the system itself. This theft of computer services is analogous to the stealing of electricity, gas, or water. However, the value of the stolen computing services may be substantially higher than the value of the stolen utility products or services. Moreover, the unpaid computing access spreads the true costs of maintaining the computing system to other legitimate users. In fact, the unauthorized access risks affecting legitimate computing, perhaps by changing data or programs.

A second vulnerability involves malicious access to a computing system, whereby an intruding person or system actually destroys software or data.

Finally, unauthorized access may deny service to a legitimate user. For example, a user who has a time-critical task to perform may depend on the availability of the computing system. For all three of these reasons, unauthorized access to a computing system must be prevented.

*Key People*

People can be crucial weak points in security. If only one person knows how to use or maintain a particular program, trouble can arise if that person is ill, suffers an accident, or leaves the organization (taking her knowledge with her). In particular, a disgruntled employee can cause serious damage by using inside knowledge of the system and the data that are manipulated. For this reason, trusted individuals, such as operators and systems programmers, are usually selected carefully because of their potential ability to affect all computer users.

**Section 1.4**

**Computer Criminals**

Some computer criminals are mean and sinister types. But many more wear business suits, have university degrees, and appear to be pillars of their communities. Some are high school or university students. Others are middle-aged business executives. Some are mentally deranged, overtly hostile, or extremely committed to a cause, and they attack computers as a symbol. Others are ordinary people tempted by personal profit, revenge, challenge, advancement, or job security. No single profile captures the characteristics of a “typical” computer criminal, and many who fit the profile are not criminals at all.

For the purposes of studying computer security, we say computer crime is any crime involving a computer or aided by the use of one. Although this definition is admittedly broad, it allows us to consider ways to protect ourselves, our businesses, and our communities against those who use computers maliciously.

One approach to prevention or moderation is to understand who commits these crimes and why. Many studies have attempted to determine the characteristics of computer criminals. By studying those who have already used computers to commit crimes, we may be able in the future to spot likely criminals and prevent the crimes from occurring. In this section, we examine some of these characteristics.

**Amateurs**

Amateurs have committed most of the computer crimes reported to date. Most embezzlers are not career criminals but rather are normal people who observe a weakness in a security system that allows them to access cash or other valuables. In the same sense, most computer criminals are ordinary computer professionals or users who, while doing their jobs, discover they have access to something valuable.

**Crackers or Malicious Hackers**

System crackers, often high school or university students, attempt to access computing facilities for which they have not been authorized. Cracking a computer’s defenses is seen as the ultimate victimless crime. The perception is that nobody is hurt or even endangered by a little stolen machine time. Crackers enjoy the simple challenge of trying to log in, just to see whether it can be done.

Most crackers can do their harm without confronting anybody, not even making a sound. In the absence of explicit warnings not to trespass in a system, crackers infer that access is permitted. An underground network of hackers helps pass along secrets of success; as with a jigsaw puzzle, a few isolated pieces joined together may produce a large effect. Others attack for curiosity, personal gain, or self-satisfaction. And still others enjoy causing chaos, loss, or harm. There is no common profile or motivation for these attackers.

**Career Criminals**

By contrast, the career computer criminal understands the targets of computer crime. Criminals seldom change fields from arson, murder, or auto theft to computing; more often, criminals begin as computer professionals who engage in computer crime, finding the prospects and payoff good. There is some evidence that organized crime and international groups are engaging in computer crime. Recently, electronic spies and information brokers have begun to recognize that trading in companies’ or individuals’ secrets can be lucrative.

**Terrorists**

The link between computers and terrorism is quite evident. We see terrorists using computers in three ways:

• targets of attack: denial-of-service attacks and web site defacements are popular for any political organization because they attract attention to the cause and bring undesired negative attention to the target of the attack.

• propaganda vehicles: web sites, web logs, and e-mail lists are effective, fast, and inexpensive ways to get a message to many people.

• methods of attack: to launch offensive attacks requires use of computers

We cannot accurately measure the amount of computer-based terrorism because our definitions and measurement tools are rather weak. Still, there is evidence that all three of these activities are increasing.

**Section 1.5**

**Methods of Defense**

Harm occurs when a threat is realized against a vulnerability. To protect against harm, then, we can neutralize the threat, close the vulnerability, or both. The possibility for harm to occur is called **risk**. We can deal with harm in several ways. We can seek to

• prevent it, by blocking the attack or closing the vulnerability

• deter it, by making the attack harder but not impossible

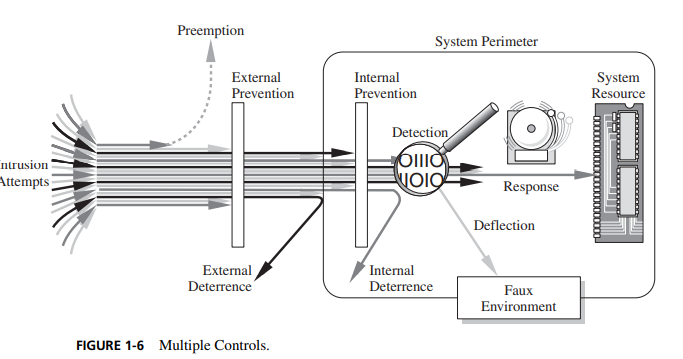
• deflect it, by making another target more attractive (or this one less so)

• detect it, either as it happens or some time after the fact

• recover from its effects

Of course, more than one of these can be done at once. So, for example, we might try to prevent intrusions. But in case we do not prevent them all, we might install a detection device to warn of an imminent attack. And we should have in place incident response procedures to help in the recovery in case an intrusion does succeed

**Controls**



To consider the controls or countermeasures that attempt to prevent exploiting a computing system’s vulnerabilities, we begin by thinking about traditional ways to enhance physical security. In the Middle Ages, castles and fortresses were built to protect the people and valuable property inside. The fortress might have had one or more security characteristics, including:

• a strong gate or door, to repel invaders

• heavy walls to withstand objects thrown or projected against them

• a surrounding moat, to control access

• arrow slits, to let archers shoot at approaching enemies

• crenellations to allow inhabitants to lean out from the roof and pour hot or vile liquids on attackers

• a drawbridge to limit access to authorized people

• gatekeepers to verify that only authorized people and goods could enter

Computer security has the same characteristics. We have many controls at our disposal. Some are easier than others to use or implement. Some are cheaper than others to use or implement. And some are more difficult than others for intruders to override. Figure 1-6 illustrates how we use a combination of controls to secure our valuable resources. We use one or more controls, according to what we are protecting, how the cost of protection compares with the risk of loss, and how hard we think intruders will work to get what they want.

**Encryption**

Encryption is the formal name for the scrambling process. We take data in their normal, unscrambled state, called **cleartext**, and transform them so that they are unintelligible to the outside observer; the transformed data are called **enciphered text** or **ciphertext.**

Encryption clearly addresses the need for confidentiality of data. Additionally, it can be used to ensure integrity; data that cannot be read generally cannot easily be changed in a meaningful manner. Furthermore, as we see throughout this book, encryption is the basis of protocols that enable us to provide security while accomplishing an important system or network task.

A **protocol** is an agreed-on sequence of actions that leads to a desired result. For example, some operating system protocols ensure availability of resources as different tasks and users request them. Thus, encryption can also be thought of as supporting availability. That is, encryption is at the heart of methods for ensuring all aspects of computer security

Although encryption is an important tool in any computer security tool kit, we should not overrate its importance. Encryption does not solve all computer security problems, and other tools must complement its use. Furthermore, if encryption is not used properly, it may have no effect on security or could even degrade the performance of the entire system. Weak encryption can actually be worse than no encryption at all, because it gives users an unwarranted sense of protection. Therefore, we must understand those situations in which encryption is most useful as well as ways to use it effectively.

**Software Controls**

If encryption is the primary way of protecting valuables, programs themselves are the second facet of computer security. Programs must be secure enough to prevent outside attack. They must also be developed and maintained so that we can be confident of the programs’ dependability. Program controls include the following:

• **internal program controls**: parts of the program that enforce security restrictions, such as access limitations in a database management program

• **operating system and network system controls**: limitations enforced by the operating system or network to protect each user from all other users

**• independent control programs**: application programs, such as password checkers, intrusion detection utilities, or virus scanners, that protect against certain types of vulnerabilities

• **development controls**: quality standards under which a program is designed, coded, tested, and maintained to prevent software faults from becoming exploitable vulnerabilities

We can implement software controls by using tools and techniques such as hardware components, encryption, or information gathering. Software controls frequently affect users directly, such as when the user is interrupted and asked for a password before being given access to a program or data.

**Hardware Controls**

Numerous hardware devices have been created to assist in providing computer security. These devices include a variety of means, such as:

• hardware or smart card implementations of encryption

• locks or cables limiting access or deterring theft

• devices to verify users’ identities

• firewalls

• intrusion detection systems

• circuit boards that control access to storage media

**Policies and Procedures**

Some of the simplest controls, such as frequent changes of passwords, can be achieved at essentially no cost but with tremendous effect. Training and administration follow immediately after establishment of policies, to reinforce the importance of security policy and to ensure their proper use.

We must not forget the value of community standards and expectations when we consider how to enforce security. There are many acts that most thoughtful people would consider harmful, and we can leverage this commonality of belief in our policies. For this reason, legal and ethical controls are an important part of computer security.

**Effectiveness of Controls**

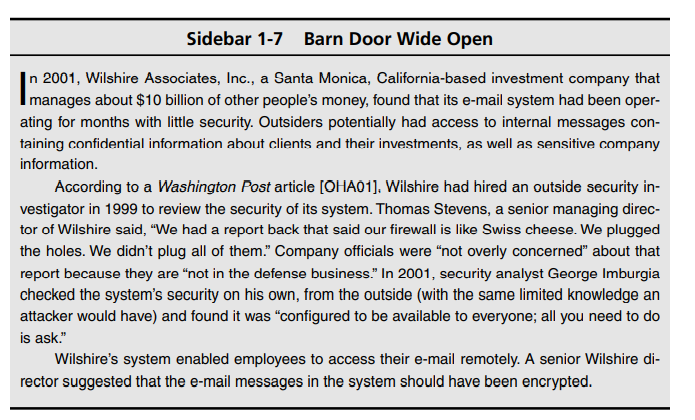
Merely having controls does no good unless they are used properly. Let us consider several aspects that can enhance the effectiveness of controls.

**Awareness of Problem**

People using controls must be convinced of the need for security. That is, people will willingly cooperate with security requirements only if they understand why security is appropriate in a given situation. However, many users are unaware of the need for security, especially in situations in which a group has recently undertaken a computing task that was previously performed with lax or no apparent security.

**Likelihood of Use**

Of course, no control is effective unless it is used. The lock on a computer room door does no good if people block the door open. As Sidebar 1-7 tells, some computer systems are seriously uncontrolled



**Principle of Effectiveness**: Controls must be used—and used properly—to be effective. They must be efficient, easy to use, and appropriate

This principle implies that computer security controls must be efficient enough, in terms of time, memory space, human activity, or other resources used, that using the control does not seriously affect the task being protected. Controls should be selective so that they do not exclude legitimate accesses.

**Overlapping Controls**

As we have seen with fortress or home security, several different controls may apply to address a single vulnerability. For example, we may choose to implement security for a microcomputer application by using a combination of controls on program access to the data, on physical access to the microcomputer and storage media, and even by file locking to control access to the processing programs.

**Periodic Review**

Seldom, if ever, are controls perfectly effective. Controls fail, controls are incomplete, or people circumvent or misuse controls, for example. For that reason, we use overlapping controls, sometimes called a layered defense, in the expectation that one control will compensate for a failure of another. In some cases, controls do nicely complement each other. But two controls are not always better than one and, in some cases, two can even be worse than one. This brings us to another security principle.

**Principle of Weakest Link**: Security can be no stronger than its weakest link. Whether it is the power supply that powers the firewall or the operating system under the security application or the human who plans, implements, and administers controls, a failure of any control can lead to a security failure.