Chapter 1. Is There a Security Problem in Computing?

In this chapter:

- The risks involved in computing
- The goals of secure computing: confidentiality, integrity, availability
- The threats to security in computing: interception, interruption, modification, fabrication
- Controls available to address these threats: encryption, programming controls, operating systems, network controls, administrative controls, law, and ethics

1.1 What Does "Secure" Mean?

How do we protect our most valuable assets? One option is to place them in a safe place, like a bank. We seldom hear of a bank robbery these days, even though it was once a fairly lucrative undertaking. In the American Wild West, banks kept large amounts of cash on hand, as well as gold and silver, which could not be traced easily. In those days, cash was much more commonly used than checks. Communications and transportation were primitive enough that it might have been hours before the legal authorities were informed of a robbery and days before they could actually arrive at the scene of the crime, by which time the robbers were long gone. To control the situation, a single guard for the night was only marginally effective. Should you have wanted to commit a robbery, you might have needed only a little common sense and perhaps several days to analyze the situation; you certainly did not require much sophisticated training. Indeed, you usually learned on the job, assisting other robbers in a form of apprenticeship. On balance, all these factors tipped very much in the favor of the criminal, so that bank robbery was, for a time, considered to be a profitable business. Protecting assets was difficult and not always effective.

Today, however, asset protection is easier, with many factors working against the potential criminal. Very sophisticated alarm systems silently protect secure places like banks whether people are around or not. The techniques of criminal investigation have become very effective, so that a person can be identified by genetic material (DNA), fingerprints, retinal patterns, voice, a composite sketch, ballistics evidence, or other hard-to-mask characteristics. The assets are stored in a safer form. For instance, many bank branches now contain less cash than some large retail stores because much of a bank's business is conducted with checks, electronic transfers, credit cards, or debit cards. Sites that must store large amounts of cash or currency are protected with many levels of security: several layers of physical systems, complex locks, multiple-party systems requiring the agreement of several people to allow access, and other schemes. Significant improvements in transportation and communication mean that police can be at the scene of a crime in minutes; dispatchers can alert other officers in seconds about the suspects to watch for. From the criminal's point of view, the risk and required sophistication are so high that there are usually easier ways than bank robbery to make money.

Protecting Valuables

This book is about protecting our computer-related assets, not about protecting our money and gold bullion. That is, we plan to discuss security for computing systems, not banks. But we can learn from our analysis of banks because they tell us some general principles about protection. In other words, when we think about protecting valuable information, we can learn a lot from the way we have protected other valuables in the past. For example, Table 1-1 presents the differences between how people protect computing systems and how banks protect money. The table reinforces the notion that we have many challenges to address when protecting computers and data, but the nature of the challenges may mean that we need different and more effective approaches than we have used in the past.

Table 1-1. Protecting Money vs. Protecting Information.

Characteristic	Bank Protecting Money	People Protecting Information
Size and portability	Sites storing money are large, unwieldy, not at all portable. Buildings require guards, vaults, many levels of physical security to protect money.	Items storing valuable assets are very small and portable. The physical devices in computing can be so small that thousands of dollars' worth of computing gear can fit comfortably in a briefcase.
Ability to avoid physical contact	Difficult. When banks deal with physical currency, a criminal must physically demand the money and carry it away from the bank's premises.	Simple. When information is handled electronically, no physical contact is necessary. Indeed, when banks handle money electronically, almost all transactions can be done without any physical contact. Money can be transferred through computers, mail, or telephone.
Value of assets	Very high.	Variable, from very high to very low. Some information, such as medical history, tax payments, investments, or educational background, is confidential. Other information, about troop movements, sales strategies, buying patterns, can be very sensitive. Still other information, such as address and phone number, may be of no consequence and easily accessible by other means.

Protecting our valuables, whether they are expressed as information or in some other way, ranges from quite unsophisticated to very sophisticated. We can think of the Wild West days as an example of the "unsophisticated" end of the security spectrum. And even today, when we have more sophisticated means of protection than ever before, we still see a wide range in how people and businesses actually use the protections available to them.

In fact, we can find far too many examples of computer security that seem to be back in the Wild West days. Although some organizations recognize computers and their data as valuable and vulnerable resources and have applied appropriate protection, others are dangerously deficient in their security measures. In some cases, the situation is even worse than that in the Wild West; as Sidebar 1-1 illustrates, some enterprises do not even recognize that their resources should be controlled and protected. And as software consumers, we find the lack of protection is all the more dangerous when we are not even aware that we are susceptible to software piracy or corruption.

Sidebar 1-1 Protecting Software in Automobile Control Systems

The amount of software installed in an automobile grows larger from year to year. Most cars, especially more expensive ones, use dozens of microcontrollers to provide a variety of features to entice buyers. There is enough variation in microcontroller range and function that the Society of Automotive Engineers (Warrendale, Pennsylvania) has set standards for the U.S. automotive industry's software. Software in the microcontrollers ranges through three classes:

- low speed (class A—less than 10 kb per second) for convenience features, such as radios.
- medium speed (class B—10 to 125 kb per second) for the general transfer of information, such as that related to emissions, speed, or instrumentation.
- high speed (class C—more than 125 kb per second) for real-time control, such as the power train or a brake-by-wire system.

These digital cars use software to control individual subsystems, and then more software to connect the systems in a network. [WHI01]

However, the engineers designing and implementing this software see no reason to protect it from hackers.

Whitehorn-Umphres reports that, from the engineers' point of view, the software is too complicated to be understood by a hacker. "And even if they could [understand it], they wouldn't want to."

Whitehorn-Umphres points out a major difference in thinking between hardware designers and software designers. "As hardware engineers, they assumed that, perhaps aside from bolt-on aftermarket parts, everything else is and should be a black box." But software folks have a different take: "As a software designer, I assume that all digital technologies are fair game for being played with....it takes a special kind of personality to look at a software-enabled device and see the potential for manipulation and change—a hacker personality."

He points out that hot-rodders and auto enthusiasts have a long history of tinkering and tailoring to make specialized changes to mass-produced cars. And the unprotected software beckons them to continue the tradition. For instance, there are reports of recalibrating the speedometer of two types of Japanese motorcycles, to fool the bike about how fast it is really going (and thereby enable faster-than-legal speeds). Whitehorn-Umphres speculates that soon you will be able to "download new ignition mappings from your PC. The next step will be to port the PC software to handheld computers so as to make on-the-road modifications that much easier."

The possibility of crime is bad enough. But worse yet, in the event of a crime, some organizations neither investigate nor prosecute for fear that the revelation will damage their public image. For example, would you feel safe depositing your money in a bank that had just suffered a several million-dollar loss through computer-related embezzlement? In fact, the breach of security makes that bank painfully aware of all its security weaknesses. Once bitten, twice shy; after the loss, the bank will probably enhance its security substantially, quickly becoming safer than a bank that had not been recently victimized.

Even when organizations want to take action against criminal activity, criminal investigation and prosecution can be hindered by statutes that do not recognize electromagnetic signals as property. The news media sometimes portray computer intrusion by teenagers as a prank no more serious than tipping over an outhouse. But, as we shall see in later chapters, computer intrusion can hurt businesses and even take lives. The legal systems around the world are coming to grips with the nature of electronic property as intellectual property critical to organizational or mission success; laws are being implemented and court decisions declared that acknowledge the value of information stored or transmitted via computers. But this area is still new to many courts, and few precedents have been established.

Throughout this book, we look at examples of how computer security affects our lives—directly and indirectly. And we examine techniques to prevent security breaches or at least to mitigate their effects. We address the security concerns of software practitioners as well as those professionals, managers, and users whose products, services, and well being depend on the proper functioning of computer systems. By studying this book, you will develop an understanding of the basic problems underlying computer security and the methods available to deal with them.

In particular, we do the following:

- Examine the *risks* of security in computing
- Consider available *countermeasures* or *controls*
- Stimulate thought about *uncovered vulnerabilities*
- Identify areas where *more work* is needed

In this chapter, we begin by examining *what* kinds of vulnerabilities computing systems are prone to. We then consider *why* these vulnerabilities are exploited: the different kinds of attacks that are possible. This chapter's third focus is on *who* is involved: the kinds of people who contribute to the security problem. Finally, we introduce *how* to prevent possible attacks on systems.

Characteristics of Computer Intrusion

Any part of a computing system can be the target of a crime. When we refer to a **computing system**,[1] we mean a collection of hardware, software, storage media, data, and people that an organization uses to perform computing tasks. Sometimes, we assume that parts of a computing system are not valuable to an outsider, but often we are mistaken. For instance, we tend to think that the most valuable property in a bank is the cash, gold, or silver in the vault. But in fact the customer information in the bank's computer may be far more valuable. Stored on paper, recorded on a storage medium, resident in memory, or transmitted over telephone lines or satellite links,

this information can be used in myriad ways to make money illicitly. A competing bank can use this information to steal clients or even to disrupt service and discredit the bank. An unscrupulous individual could move money from one account to another without the owner's permission. A group of con artists could contact large depositors and convince them to invest in fraudulent schemes. The variety of targets and attacks makes computer security very difficult.

[1] In this book, **boldface** identifies new terms being introduced.

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. Similarly, a sophisticated perimeter physical security system does not compensate for unguarded access by means of a simple telephone line and a modem. We can codify this idea as one of the principles of computer security:

> **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.

This principle implies that computer security specialists must consider all possible means of penetration. Moreover, the penetration analysis must be done repeatedly, and especially whenever the system and its security change. Strengthening one aspect of a system may simply make another means of penetration more appealing to intruders. For this reason, let us look at the various ways that a system can be breached.

1.2 Attacks

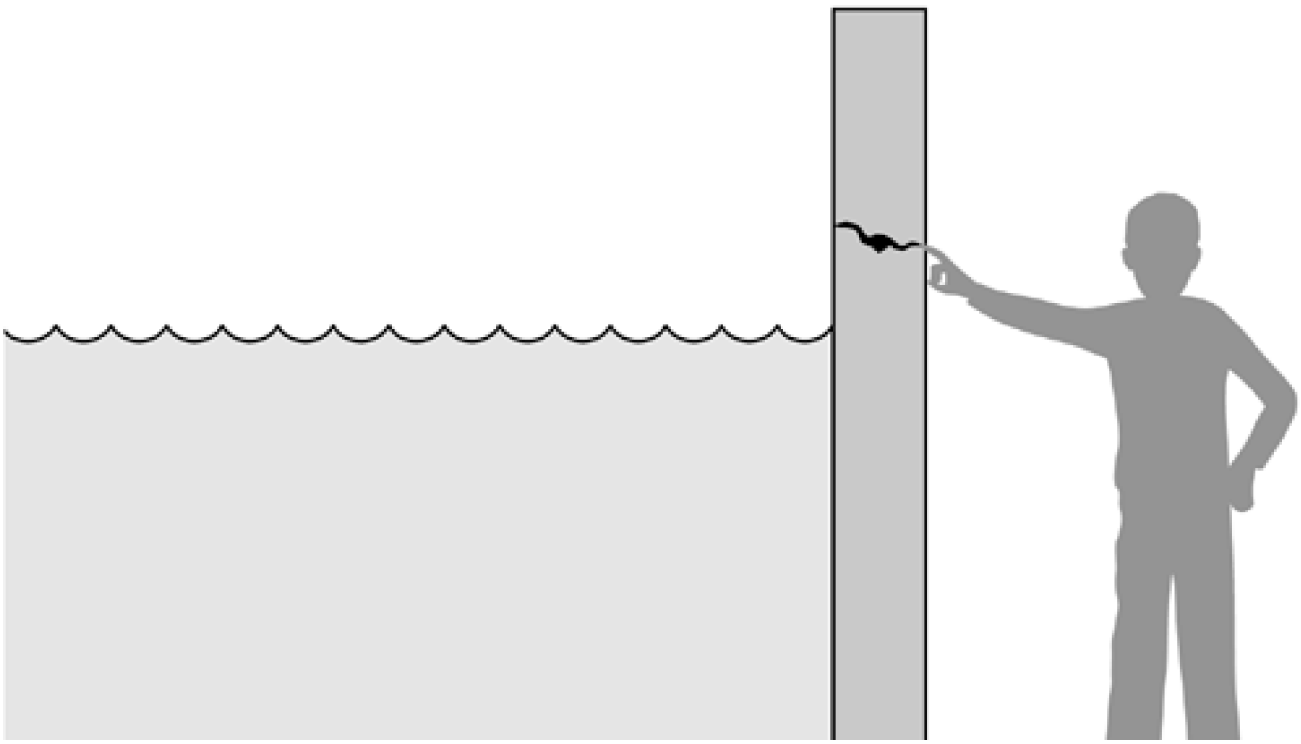
When you test any computer system, one of your jobs is to imagine how the system could malfunction. Then, you improve the system's design so that the system can withstand any of the problems you have identified. In the same way, we analyze a system from a security perspective, thinking about ways in which the system's security can malfunction and diminish the value of its assets.

Threats, Vulnerabilities, and Controls

A computer-based system has three separate but valuable components: **hardware, software,** and **data.** Each of these assets offers value to different members of the community affected by the system. To analyze security, we can brainstorm about the ways in which the system or its information can experience some kind of loss or harm. For example, we can identify data whose format or contents should be protected in some way. We want our security system to make sure that no data are disclosed to unauthorized parties. Neither do we want the data to be modified in illegitimate ways. At the same time, we want to ensure that legitimate users have access to the data. In this way, we can identify weaknesses in the system. A **vulnerability** is 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.

A **threat** to a computing system is 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 it to collapse. So the threat of harm is the potential for the man to get wet, get hurt, or drown. For now, the wall is intact, so the threat to the man is unrealized.

Figure 1-1. Threats, Controls, and Vulnerabilities.



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.

There are many threats to a computer system, including human-initiated and computer-initiated ones. We have all experienced the results of inadvertent human errors, hardware design flaws, and software failures. But natural disasters are threats, too; they can bring a system down when the computer room is flooded or the data center collapses from an earthquake, for example.

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. (We take a closer look at denial of service in Chapter 7.)

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:

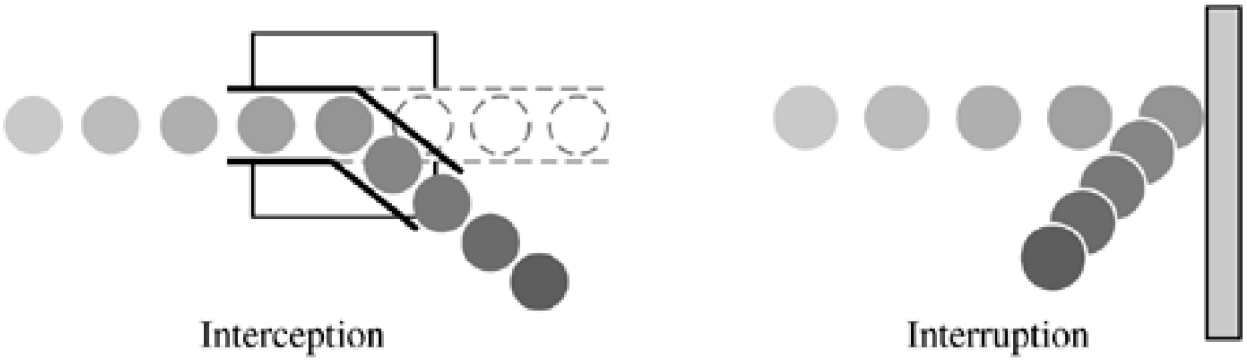
> A *threat* is blocked by *control of* a *vulnerability*.

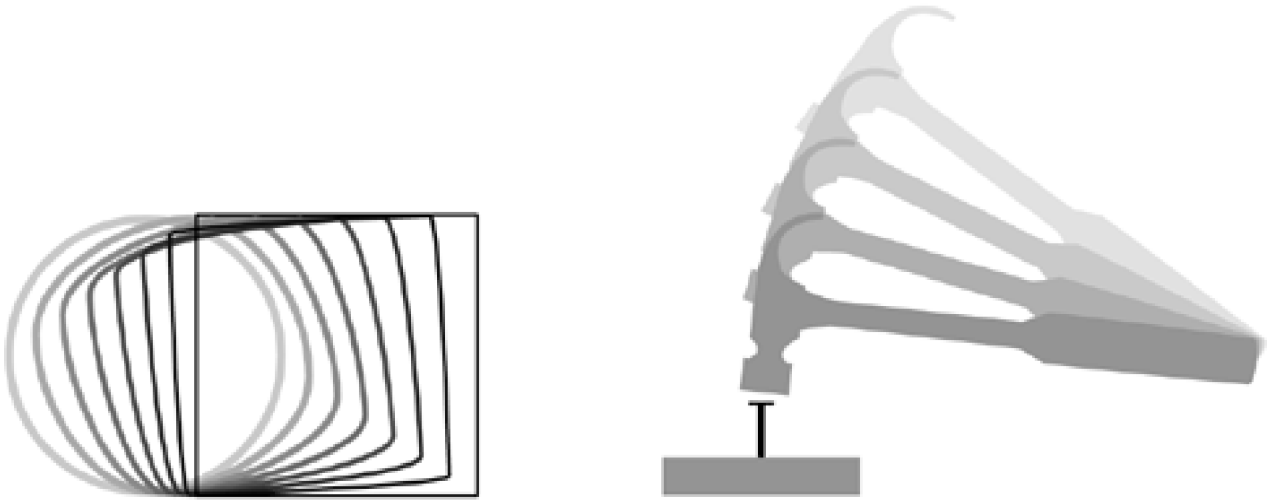
Much of the rest of this book is devoted to describing a variety of controls and understanding the degree to which they enhance a system's security.

To devise controls, we must know as much about threats as possible. We can view any threat as being one of four kinds: interception, interruption, modification, and fabrication. Each threat exploits vulnerabilities of the assets in computing systems; the threats are illustrated in Figure 1-2.

- An **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.
- In an **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.
- If an unauthorized party not only accesses but tampers with an asset, the threat is a **modification**. 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.
- Finally, an unauthorized party might create a **fabrication** of 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.

Figure 1-2. System Security Threats.





These four classes of threats—interception, interruption, modification, and fabrication—describe the kinds of problems we might encounter. In the next section, we look more closely at a system's vulnerabilities and how we can use them to set security goals.

Method, Opportunity, and Motive

A malicious attacker must have three things:

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

(Think of the acronym "MOM.") Deny any of those three things and the attack will not occur. However, it is not easy to cut these off.

Knowledge of systems is widely available. Mass market systems (such as the Microsoft or Apple or Unix operating systems) are readily available, as are common products, such as word processors or database management systems. Sometimes the manufacturers release detailed specifications on how the system was designed or operates, as guides for users and integrators who want to implement other complementary products. But even without documentation, attackers can purchase and experiment with many systems. Often, only time and inclination limit an attacker.

Many systems are readily available. Systems available to the public are, by definition, accessible; often their owners take special care to make them fully available, so that if one hardware component fails, the owner has spares instantly ready to be pressed into service.

Finally, it is difficult to determine motive for an attack. Some places are what are called "attractive targets," meaning they are very appealing to attackers. Popular targets include law enforcement and defense department computers, perhaps because they are presumed to be well protected against attack (so that a successful attack shows the attacker's prowess). Other systems are attacked because they are easy. (See Sidebar 1-2 on universities as targets.) And other systems are attacked simply because they are there: random, unassuming victims.

Sidebar 1-2 Why Universities Are Prime Targets

Universities make very good targets for attack, according to an Associated Press story from June 2001 [HOP01]. Richard Power, editorial director for the Computer Security Institute has reported that universities often run systems with vulnerabilities and little monitoring or management. Consider that the typical university research or teaching lab is managed by a faculty member who has many other responsibilities or by a student manager who may have had little training. Universities are havens for free exchange of ideas. Thus, their access controls typically are configured to promote sharing and wide access to a population that changes significantly every semester.

A worse problem is that universities are really loose federations of departments and research groups. The administrator for one group's computers may not even know other administrators, let alone share intelligence or tools. Often, computers are bought for a teaching or research project, but there is not funding for ongoing maintenance, either buying upgrades or installing patches. Steve Hare, managing director of the computer security research group at Purdue University, noted that groups are usually strapped for resources.

David Dittrich, a security engineer at the University of Washington, said he is certain that cracker(s) who attacked the eBay and CNN.com web sites in 2000 first practiced on university computers. The large and frequently changing university student body gives the attacker great opportunity to maintain anonymity while developing an attack.

Protecting against attacks can be difficult. Anyone can be a victim of an attack perpetrated by an unhurried, knowledgeable attacker. In the remainder of this book we discuss the nature of attacks and how to protect against them.

1.3 The Meaning of Computer Security

We have seen that any computer-related system has both theoretical and real weaknesses. The purpose of computer security is to devise ways to prevent the weaknesses from being exploited. To understand what preventive measures make the most sense, we consider what we mean when we say that a system is "secure."

Security Goals

We use the term "security" in many ways in our daily lives. A "security system" protects our house, warning the neighbors or the police if an unauthorized intruder tries to get in. "Financial security" involves a set of investments that are adequately funded; we hope the investments will grow in value over time, so that we have enough money to survive later in life. And we speak of a child's "physical security," hoping he or she is safe from any potential harm. Just as each of these terms has a very specific meaning in the context of its use, so too does the phrase "computer security."

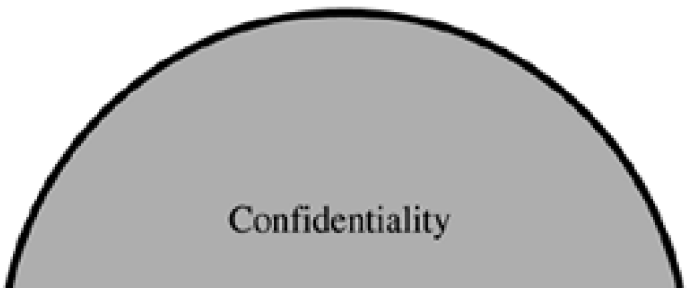
When we talk about "computer security," we mean that we are addressing three very important aspects of any computer-related system: **confidentiality, integrity,** and **availability.**

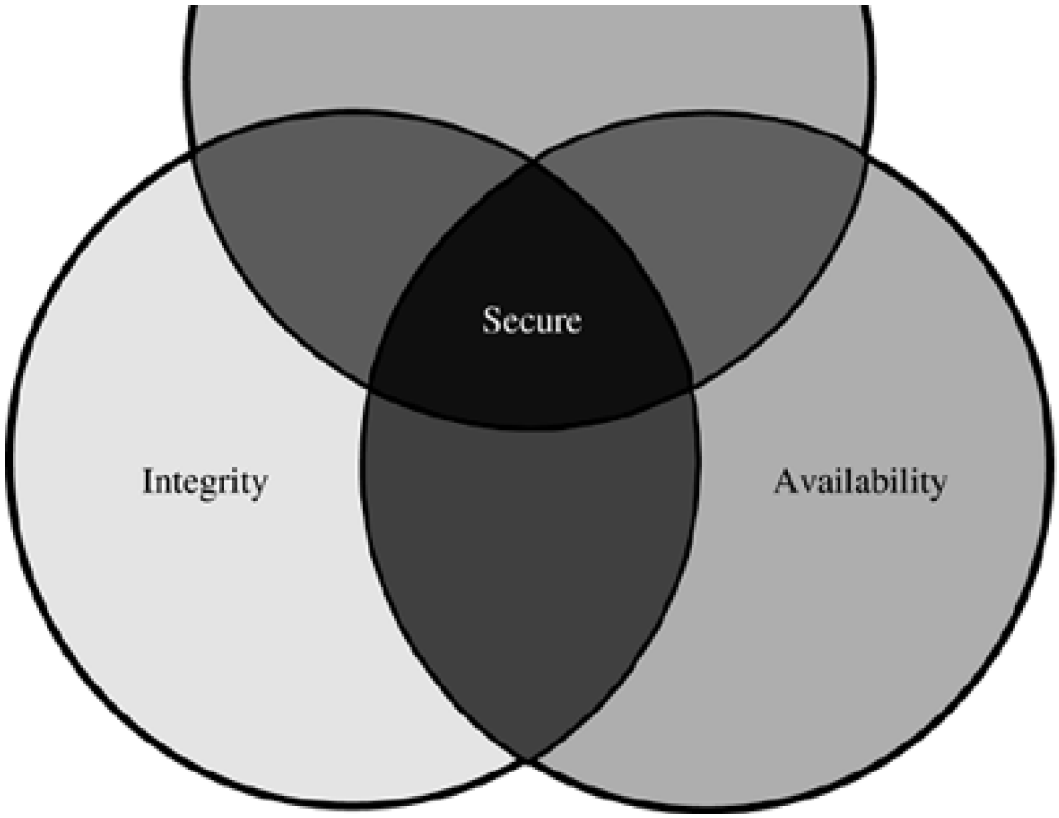
- *Confidentiality* ensures that computer-related assets are accessed only by authorized parties. That is, only those who should have access to something will actually get that access. By "access," we mean not only reading but also viewing, printing, or simply knowing that a particular asset exists.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

But balance is not all. In fact, these three characteristics can be independent, can overlap (as shown in Figure 1-3), and can even be mutually exclusive. For example, we have seen that strong protection of confidentiality can severely restrict availability. Let us examine each of the three qualities in depth.

Figure 1-3. Relationship Between Confidentiality, Integrity, 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? 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

Integrity is much harder to pin down. As Welke and Mayfield [WEL90, MAY91, NCS91b] point out, *integrity* means different things in different contexts. 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

The Trusted Network Interpretation [NCS87] clarifies by saying that integrity ensures that computerized data are the same as those in source documents; they have not been exposed to accidental or malicious alteration or destruction. 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.
- There is a fair allocation of resources, 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.
- There is controlled concurrency; that is, there is support for simultaneous access, deadlock management, and exclusive access, 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.

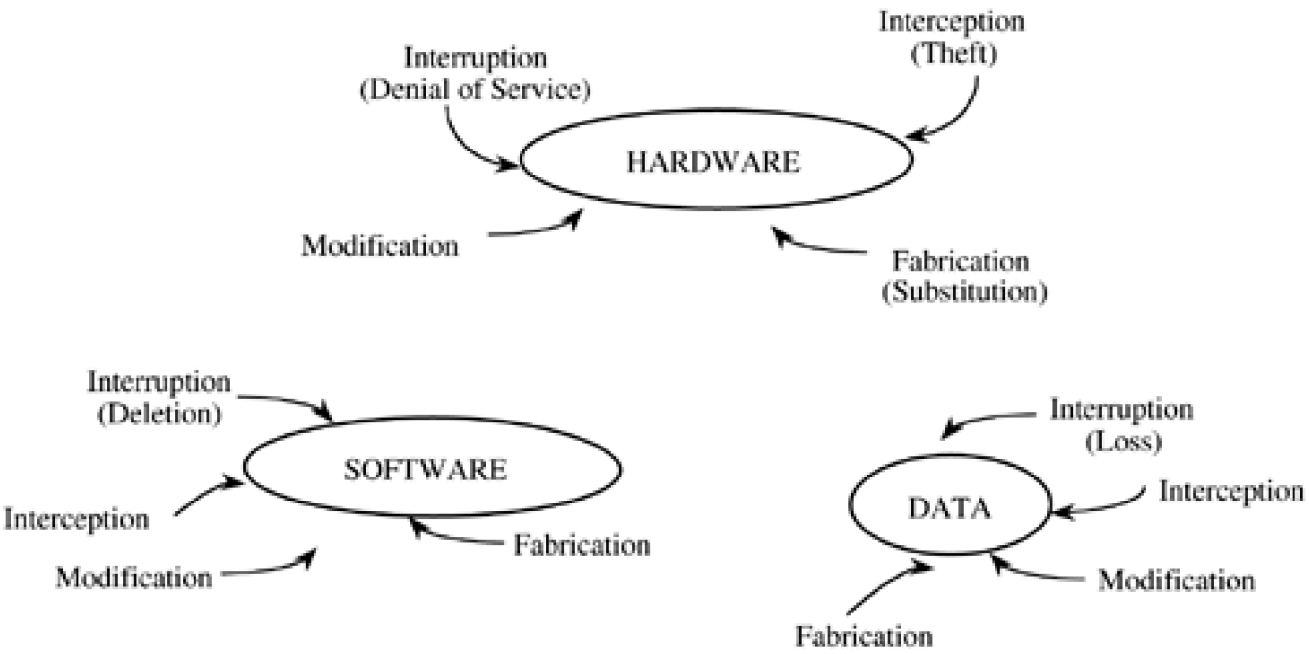
Vulnerabilities

When we prepare to test a system, we usually try to imagine how the system can fail; then, we look for ways in which the requirements, design, or code can enable such failures. In the same way, when we prepare to specify, design, code, or test a secure system, we try to imagine the vulnerabilities that would prevent us from reaching one or more of our three security goals.

It is sometimes easier to consider vulnerabilities as they apply to all three broad categories of system resources (hardware, software, and

data), 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.

Figure 1-4. Vulnerabilities of Computing Systems.



Hardware Vulnerabilities

Hardware is more visible than software, largely because it is composed of physical objects. Because we can see what devices are hooked to the system, it is rather simple to attack by adding devices, changing them, removing them, intercepting the traffic to them, or flooding them with traffic until they can no longer function. However, designers can usually put safeguards in place.

But there are other ways that computer hardware can be attacked physically. Computers have been drenched with water, burned, frozen, gassed, and electrocuted with power surges. People have spilled soft drinks, corn chips, ketchup, beer, and many other kinds of food on computing devices. Mice have chewed through cables. Particles of dust, and especially ash from cigarette smoke, have threatened precisely engineered moving parts. Computers have been kicked, slapped, bumped, jarred, and punched. Although such attacks might be intentional, most are not; this abuse might be considered "involuntary machine slaughter": accidental acts not intended to do serious damage to the hardware involved.

A more serious attack, "voluntary machine slaughter" or "machinicide," usually involves someone who actually wishes to harm the computer hardware or software. Machines have been shot with guns and stabbed with knives. Bombs, fires, and collisions have destroyed computer rooms. Ordinary keys, pens, and screwdrivers have been used to short-out circuit boards and other components. Devices and whole systems have been carried off by thieves. The list of kinds of human attacks perpetrated on computers is almost endless.

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 microcomputers in office equipment has resulted in several thousands of dollars' worth of equipment sitting unattended on desks outside the carefully protected computer room. (Curiously, the supply cabinet, containing only a few hundred dollars' worth of pens, stationery, and paper clips, is often locked.) Sometimes the security of hardware components can be enhanced greatly by simple physical measures such as locks and guards.

Software Vulnerabilities

Computing equipment is of little use without the software (operating system, controllers, utility programs, and application programs) that users expect. 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.

Sometimes, the attacks are obvious, as when the software no longer runs. More subtle are attacks in which the software has been altered but seems to run normally. Whereas physical equipment usually shows some mark of inflicted injury when its boundary has been breached, the loss of a line of source or object code may not leave an obvious mark in a program. Furthermore, 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.

A classic example of exploiting software vulnerability is the case in which a bank worker realized that software truncates the fractional interest on each account. In other words, if the monthly interest on an account is calculated to be $14.5467, the software credits only $14.54 and ignores the $.0067. He amended the software so that the throw-away interest (the $.0067) was placed into his own account. Since the accounting practices ensured only that all accounts balanced, he built up a large amount of money from the thousands of account throw-aways, without detection. It was only when he bragged to a colleague of his cleverness that the scheme was discovered.

Software Deletion

Software is surprisingly easy to delete. Each of us has, at some point in our careers, accidentally erased a file or saved a bad copy of a program, destroying a good previous copy. 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 is not deleted, destroyed, or replaced accidentally. Configuration management uses several techniques to ensure that each version or release retains its integrity. When configuration management is used, an old version or release can be replaced with a newer version only when it has been thoroughly tested to verify that the improvements work correctly without degrading the functionality and performance of other functions and services.

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.

With a little more work, the change can be much more subtle, so that the program works well most of the time but fails in specialized circumstances. For instance, 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 miles away by July.

Another type of change can extend the functioning of a program so that an innocuous program has a hidden side effect. For example, a program that ostensibly structures a listing of files belonging to a user may also modify the protection of all those files to permit access by another user.

Other categories of software modification include

- a **Trojan horse**: a program that overtly does one thing while covertly doing another
- a **virus**: a specific type of Trojan horse that can be used to spread its "infection" from one computer to another
- a **trapdoor**: a program that has a secret entry point

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

More details on these and other software modifications are provided in Chapter 3.

Of course, it is possible to invent a completely new program and install it on a computing system. Inadequate control over the programs that are installed and run on a computing system permits this kind of software security breach.

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 9, the legal system is still grappling with the difficulties of interpreting paper-based copyright laws for electronic media.

Data Vulnerabilities

Hardware security is usually the concern of a relatively small staff of computing center professionals. Software security is a larger problem, extending to all programmers and analysts who create or modify programs. Computer programs are written in a dialect intelligible primarily to computer professionals, so a "leaked" source listing of a program might very well be meaningless to the general public.

Printed data, however, 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.

By themselves, out of context, pieces of data have essentially no intrinsic value. For example, if you are shown the value "42," it has no meaning for you unless you know what the number represents. Likewise, "326 Old Norwalk Road" is of little use unless you know the city, state, and country for the address. For this reason, it is hard to measure the value of a given data item.

On the other hand, data items in context do relate to cost, perhaps measurable by the cost to reconstruct or redevelop damaged or lost data. For example, confidential data leaked to a competitor may narrow a competitive edge. Data incorrectly modified can cost human lives. To see how, consider the flight coordinate data used by an airplane that is guided partly or fully by software, as many now are. Finally, inadequate security may lead to financial liability if certain personal data are made public. Thus, data have a definite value, even though that value is often difficult to measure.

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.

To see why, consider the following example. In many countries, government analysts periodically generate data to describe the state of the national economy. The results are scheduled to be released to the public at a predetermined time and date. Before that time, access to the data could allow someone to profit from advance knowledge of the probable effect of the data on the stock market. 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. However, there are simple ways to protect the data and control the threat. For example, we could devise a scheme that would take an outsider more than 24 hours to break; even though the scheme may be eminently breakable (that is, an intruder could eventually reveal the data), it is adequate for those data because there is no need for confidentiality beyond the 24-hour period.

Data security suggests the second principle of computer security:

> **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.

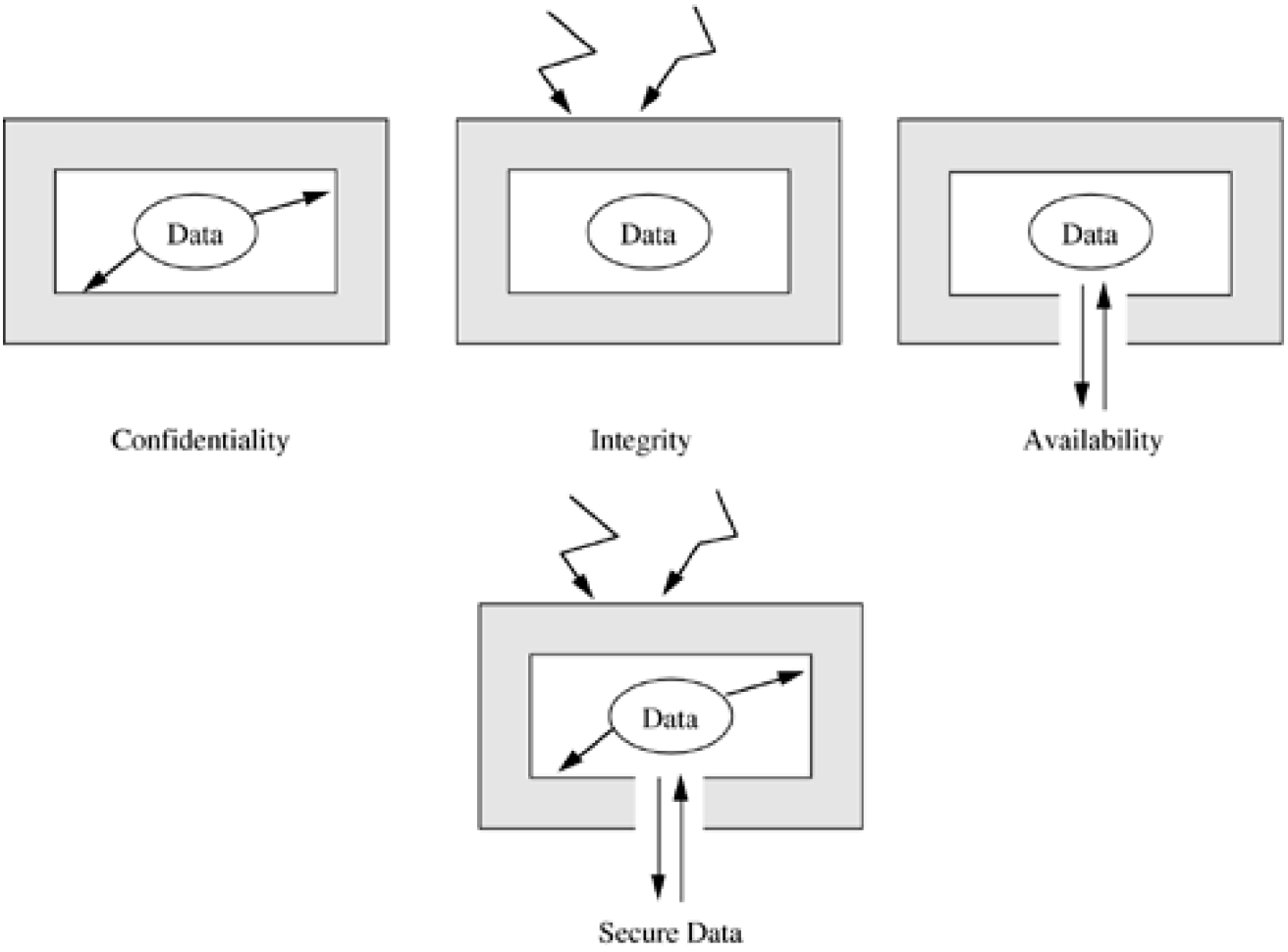
This principle says that things with a short life can be protected by security measures that are effective only for that short time. The notion

of a small protection window applies primarily to data, but it can in some cases be relevant for software and hardware, too.

Sidebar 1-3 confirms that intruders take advantage of vulnerabilities to break in by whatever means they can.

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.

Figure 1-5. Security of Data.



Sidebar 1-3 Top Methods of Attack

In 2001, *Information Week* magazine commissioned a global information survey of 4,500 security professionals. As part of the survey, the respondents were asked to name the primary methods of attack used by intruders against their organizations. (Multiple responses were allowed.)

The top method was exploiting known operating system vulnerabilities; almost one-third of the respondents had experienced this kind of attack. The next most popular method was exploiting an unknown application (27 percent). Other commonly used attacks were guessing passwords (22 percent), abusing valid user accounts or permissions (17 percent), and using an internal denial of service (12 percent).

Common wisdom had always been that four out of five attacks on corporate networks or computers were perpetrated by malevolent insiders who could take advantage of their understanding of the system. The survey sought to determine if this "rule of thumb" were true. In fact, with the growing use of Internet applications, outsiders are now considered the greater threat. Hulme [HUL01b] points out that "Many companies suspect hackers and terrorists (46 percent) and even customers (14 percent) of trying to breach their systems." This suspicion is supported by another survey, conducted by the Computer

Security Institute and the U.S. Federal Bureau of Investigation [CSI02]. The second survey notes that almost three in four businesses cite the Internet as a point of attack, whereas only one in three cites internal systems.

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 Integrity

Stealing, buying, finding, or hearing data requires no computer sophistication, whereas modifying or making new data requires some understanding of the technology by which the data are transmitted or stored, as well as the format in which the data are maintained. Thus, a higher level of sophistication is needed to modify existing data or to fabricate new data than to intercept existing data. The most common sources of this kind of problem are malicious programs, errant file system utilities, and flawed communication facilities.

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 together 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

We have noted that the major points of weakness in a computing system are hardware, software, and data. However, other components of the system may also be possible targets. In this section, we identify some of these other points of attack.

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.

We have described common assets at risk. In fact, there are valuable assets in almost any computer system. (See Sidebar 1-4 for an example of exposed assets in ordinary business dealings.)

Next, we turn to the people who design, build, and interact with computer systems, to see who can breach the systems' confidentiality, integrity, and availability.

1.4 Computer Criminals

In television and film westerns, the bad guys always wore shabby clothes, had mean and sinister looks, and lived in gangs somewhere out of town. By contrast, the sheriff dressed well, stood proud and tall, was known and respected by everyone in town, and struck fear in the hearts of most criminals.

To be sure, 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.

Sidebar 1-4 Hollywood at Risk

Do you think only banks, government sites, and universities are targets? Consider Hollywood.

In 2001 Hollywood—specifically the motion picture industry—was hit with a series of attacks. Crackers entered computers and were able to obtain access to scripts for new projects, and digital versions of films in production, including *Ocean's 11* at Warner Brothers and *The One* at Columbia Pictures. The attackers also retrieved and made public executives' e-mail messages.

But, as is true of many computer security incidents, at least one attacker was an insider. Global Network Security Services, a security consulting firm hired by several Hollywood companies to test the security of their networks, found that an employee was copying the day's (digital) film, taking it home, and allowing his roommate to post it to an Internet site.

Whatever their characteristics and motivations, computer criminals have access to enormous amounts of hardware, software, and data; they have the potential to cripple much of effective business and government throughout the world. In a sense, then, the purpose of computer security is to prevent these criminals from doing damage.

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 pretty broad, it allows us to consider ways to protect ourselves, our businesses, and our communities against those who use computers maliciously.

The U.S. Federal Bureau of Investigation regularly reports uniform crime statistics. The data do not separate computer crime from crime of other sorts. Moreover, many companies do not report computer crime at all, perhaps because they fear damage to their reputation, they are ashamed to have allowed their systems to be compromised, or they have agreed not to prosecute if the criminal will "go away." These conditions make it difficult for us to estimate the economic losses we suffer as a result of computer crime; our dollar estimates are really only vague suspicions. Still, the estimates, ranging from $300 million to $500 billion per year, tell us that it is important for us to pay attention to computer crime and to try to prevent it or at least to moderate its effects.

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 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 doing their jobs, when they discover they have access to something

valuable.

When no one objects, the amateur may start using the computer at work to write letters, maintain soccer league team standings, or do accounting. This apparently innocent time-stealing may expand until the employee is pursuing a business in accounting, stock portfolio management, or desktop publishing on the side, using the employer's computing facilities. Alternatively, amateurs may become disgruntled over some negative work situation (such as a reprimand or denial of promotion) and vow to "get even" with management by wreaking havoc on a computing installation.

Crackers

System **crackers**,[2] 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.

[2] The security community distinguishes between a "hacker," someone who (nonmaliciously) programs, manages, or uses computing systems, and a "cracker," someone who attempts access to computing systems for malicious purposes. Crackers are the "evildoers." Now, hacker has come to be used outside security to mean both benign and malicious users.

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.

As mentioned earlier, some companies are reticent to prosecute computer criminals. In fact, after having discovered a computer crime, the companies are often thankful if the criminal quietly resigns. In other cases, the company is (understandably) more concerned about protecting its assets and so it closes down an attacked system rather than gathering evidence that could lead to identification and conviction of the criminal. The criminal is then free to continue the same illegal pattern with another company.

1.5 Methods of Defense

In Chapter 9, we investigate the legal and ethical restrictions on computer-based crime. But unfortunately computer crime is certain to continue for the foreseeable future. For this reason, we must look carefully at controls for preserving confidentiality, integrity, and availability. Sometimes these controls can prevent or mitigate attacks; other, less powerful methods can only inform us that security has been compromised, by detecting a breach as it happens or after it occurs.

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 valuable people and property inside. The fortress might have had one or more security characteristics, including

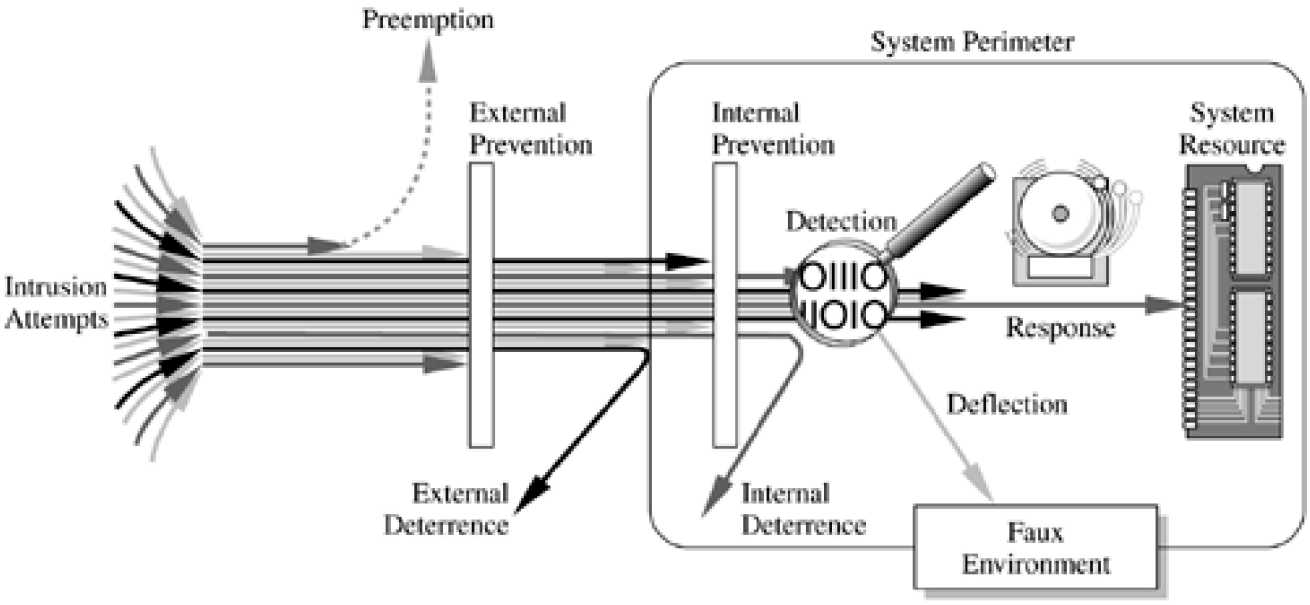
- a strong gate or door to repel invaders
- heavy walls to withstand objects thrown or projected against them
- a surrounding moat, to control access
- arrow slits, to let archers shoot at approaching enemies
- crenellations to allow inhabitants to lean out from the roof and pour hot or vile liquids on attackers
- a drawbridge to limit access to authorized people
- gatekeepers to verify that only authorized people and goods could enter

Similarly, today we use a multipronged approach to protect our homes and offices. We may combine strong locks on the doors with a burglar alarm, reinforced windows, and even a nosy neighbor to keep an eye on our valuables. In each case, we select one or more ways to deter an intruder or attacker, and we base our selection not only on the value of what we protect but also on the effort we think an attacker or intruder will expend to get inside.

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.

Figure 1-6. Multiple Controls.



In this section, we present an overview of the controls available to us. In later chapters, we examine each control in much more detail.

Encryption

We noted earlier that we seek to protect hardware, software, and data. We can make it particularly hard for an intruder to find data useful if we somehow scramble the data so that interpretation is meaningless without the intruder's knowing how the scrambling was done. Indeed, the most powerful tool in providing computer security is this scrambling or encoding.

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**. Using encryption, security professionals can virtually nullify the value of an interception and the possibility of effective modification or fabrication. In Chapters 2 and 10 we study many ways of devising and applying these transformations.

Encryption clearly addresses the need for confidentiality of data. Additionally, it can be used to ensure integrity; data that cannot be read generally also cannot easily be changed in a meaningful manner. Furthermore, as we will 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-upon 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. For this reason, we often think of software controls when we think of how systems have been made secure in the past. Because they influence the way users interact with a computing system, software controls must be carefully designed. Ease of use and potency are often competing goals in the design of a collection of software controls.

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

Sometimes, we can rely on agreed-upon procedures or policies among users, rather than enforcing security through hardware or software means. In fact, 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. However, the law is slow to evolve, and the technology involving computers has emerged relatively suddenly. Although legal protection is necessary and desirable, it may not be as dependable in this area

as it would be when applied to more well understood and long-standing crimes.

Society in general and the computing community have not adopted formal standards of ethical behavior. As we see in Chapter 9, some organizations have devised codes of ethics for computer professionals. However, before codes of ethics can become widely accepted and effective, the computing community and the general public must discuss and make clear what kinds of behavior are inappropriate and why.

Physical Controls

Some of the easiest, most effective, and least expensive controls are physical controls. Physical controls include locks on doors, guards at entry points, backup copies of important software and data, and physical site planning that reduces the risk of natural disasters. Often the simple physical controls are overlooked while we seek more sophisticated approaches.

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 1-5 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.

Sidebar 1-5 Barn Door Wide Open

In 2001, Wilshire Associates, Inc., a Santa Monica, California-based investment company that manages about $10 billion of other people's money, found that its e-mail system had been operating for months with little security. Outsiders potentially had access to internal messages containing confidential information about clients and their investments, as well as sensitive company information.

According to a *Washington Post* article [OHA01], Wilshire had hired an outside security investigator in 1999 to review the security of its system. Thomas Stevens, a senior managing director of Wilshire said "We had a report back that said our firewall is like Swiss cheese. We plugged the holes. We didn't plug all of them." Company officials were "not overly concerned" about that report because they are "not in the defense business." In 2001, security analyst George Imburgia checked the system's security on his own, from the outside (with the same limited knowledge an attacker would have) and found it was "configured to be available to everyone; all you need to do is ask."

Wilshire's system enabled employees to access their e-mail remotely. A senior Wilshire director suggested that the e-mail messages in the system should have been encrypted.

Sidebar 1-6 U.S. Government's Computer Security Report Card

The U.S. Congress requires government agencies to supply annual reports to the Office of Management and Budget (OMB) on the state of computer security in the agencies. In November 2001, two-thirds of the government agencies received a grade of F (the lowest possible) on the computer security report card based on the OMB data.

The agencies must report efforts to protect their computer networks against crackers, terrorists, and other attackers.

Among the failing agencies were Defense, Justice, and Treasury. The State Department received a D+. The highest grade was a B+ to the National Science Foundation. (*Washington Post*, 10 November 2001.)

Periodic Review

Few controls are permanently effective. Just when the security specialist finds a way to secure assets against certain kinds of attacks, the opposition doubles its efforts in an attempt to defeat the security mechanisms. Thus, judging the effectiveness of a control is an ongoing task. (Sidebar 1-6 reports on periodic reviews of computer security.)

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.

1.6 What's Next

This book describes all aspects of security in computing. By studying it, you will become acquainted with computer security's major problem areas, the controls that are effective against them, and how current research is addressing the open problems.

To present security in a comprehensive way, this book is organized in four parts. The first part introduces encryption, an important tool on which many controls are based. That introduction presents encryption's goals, terminology, and use. You will be able to understand the role of encryption in addressing security needs without having to learn the intricate details of particular encryption methods. Then, the second part contains material on the hardware and software components of computing systems. We describe the types of problems to which each is subject and the kinds of protection that can be implemented for each component. The third part of the book discusses factors outside the system's hardware, software, and data that can influence the system's security. In particular, this part contains a study of physical factors in security, as well as characteristics of the people who use the system. The book's final section is a more detailed study of encryption, for those readers who are interested in understanding the intricacies of encryption techniques and evaluating their effectiveness.

The remainder of this section presents the contents of these parts in more depth.

Encryption Overview

Chapter 2 presents the goals and terminology of encryption so that you will understand not only why data are scrambled but also the role of the scrambling in the larger context of protecting assets. This chapter provides you with knowledge of encryption sufficient for us to study its use as part of other security tools and techniques.

Hardware and Software Security

Chapters 3 through 7 address the role of security in general programs, operating systems, database management systems, and networks. In particular, the security problems and features of programs are introduced in Chapter 3. Here, we look at viruses and other malicious code and ways to devise controls against them.

Operating systems are considered separately, in Chapter 4, because they play a major role in security and are fundamental to proper computer usage. While providing security features to protect one user from another, operating systems can at the same time introduce security vulnerabilities themselves. Chapter 5 focuses on a special type of operating system, called a trusted operating system, to study how to make certain data and functions accessible only to those who have need or permission to view or handle them. This chapter is especially important for those developers who plan to design their own operating systems or modify functions in an existing operating system.

Database management systems are also specialized programs; they permit many users to share access to one common set of data. Because these systems are partially responsible for the confidentiality, integrity, and availability of the shared data, we look at database security in Chapter 6.

Chapter 7 contains material on security problems and solutions particular to computer networks and the communications media by which networked computers are connected. Network security has become very significant because of the rapid growth in use of networks, especially the Internet.

Human Controls in Security

The first two parts of this book form a progression from simple security applications and tools to complex security technology in multiuser, multicomputer systems. These technology-based security methods are rather sophisticated, and researchers continue to look to technology to assist in security assurance. However, most computer-based security breaches are caused by either human or environmental factors. Thus, Chapters 8 and 9 suggest an alternative or supplemental approach to computer security: treat the causes (people and the environment) rather than the symptoms (attacks and vulnerabilities). We examine procedures that can be implemented in spite of, or in addition to, any controls built into hardware and software.

Chapter 8 addresses the administration of security. It begins with security planning and the particularly important role played by risk analysis. The chapter also explains physical security mechanisms that can be used to protect computing systems against human attacks or natural disasters. It explains why security policy is essential to security planning, illustrating the concepts with several examples from actual organizational policy documents. The chapter concludes with a discussion of disaster recovery: how to deal with the failure of other controls.

Chapter 9 considers the use of law and ethics to control malicious behavior. Although computer law is a relatively new field, it is evolving rapidly and is an important tool in the defense of computing systems. We look at how ethical systems may address some situations where the law is ineffective, inappropriate, or inadequately defined. Chapter 9 also addresses privacy, a computer security issue that has both technical and ethical aspects.

Encryption In-Depth

Chapter 10 builds on the simple encryption methods and terminology presented in Chapter 2. It progresses from theoretical encryption algorithms to current standard practices in the field. We study what makes a cryptosystem secure enough for commercial use, for protecting government data, or for securing your own private, personal information.

Throughout the book, we raise issues related to the important problems in computer security today. When the solution is known, we describe it or at least give you pointers to a fuller description of the solution. At the same time, we discuss work in progress so that you can watch the media and the literature for significant achievements in improving computer security.

It is important to remember that computer security is a relatively new field that is gaining prominence as computing itself becomes pervasive. The speed of new development in computing far outpaces capabilities in computer security. It sometimes seems as if each advance in computing brings with it new security problems. In a sense, this is true. However, there is reason to be optimistic. The fundamental work in security provides tools (such as encryption and operating system features) that form the basis of controls for these new problems as the problems arise. Part of the excitement of computer security is that there are always new challenges to address.

1.7 Summary

Computer security attempts to ensure the confidentiality, integrity, and availability of computing systems' components. Three principal pieces of a computing system are subject to attacks: hardware, software, and data. These three, and the communications among them, constitute the basis of computer security vulnerabilities. In turn, those people and systems interested in compromising a system can devise attacks that exploit the vulnerabilities. This chapter has identified four kinds of attacks on computing systems: interception, interruption, modification, and fabrication.

Four principles affect the direction of work in computer security. By the principle of easiest penetration, a computing system penetrator will use whatever means of attack is the easiest; therefore, all aspects of computing system security must be considered at once. By the principle of timeliness, a system must be protected against penetration only so long as the penetration has value to the penetrator. The principle of effectiveness states that controls must be usable and used in order to serve their purpose. And the weakest link principle states that security is no stronger than its weakest point.

Controls can be applied at the levels of the data, the programs, the system, the physical devices, the communications links, the environment, and the personnel. Sometimes several controls are needed to cover a single vulnerability, and sometimes one control addresses many problems at once.

1.8 Terms and Concepts

Virus, Trojan horse, worm, rabbit, salami, firewall, spray paint, mental poker, orange book, war dialer. The vocabulary of computer security is rich with terms that capture your attention. Also, the field is filled with acronyms: DES, AES, RSA, TCSEC, CTCPEC, ITSEC, PEM, PGP, and SSE CMM, to list a few. All of these are explained in this book. Each chapter ends with a list of terms and concepts, in order of their occurrence, as a way to review and see whether you have learned the important points of the chapter.

The list for this chapter includes some terms that may be new, as well as the major concepts introduced in this chapter. Although these terms are elaborated in future chapters, it is good to begin now to learn the terms and the underlying concepts.

computing system, 4

principle of easiest penetration, 5

hardware, 5

software, 5

data, 5

vulnerability, 6

attack, 6

threat, 6

control, 7

interruption, 7

interception, 7

modification, 8

fabrication, 8

method, 8

opportunity, 8

motive, 8

prevention, 22

deterrence, 22

deflection, 22

detection, 22

recovery, 22

encryption, 23

protocol, 24

security, secure, 9

confidentiality, 10

integrity, 10

availability, 10

secrecy, 10

privacy, 10

configuration management, 14

logic bomb, 15

Trojan horse, 15

virus, 15

trapdoor, 15

information leak, 15

principle of adequate protection, 16

salami attack, 17

replay, 18

cracker, 21

policy, 25

procedure, 25

physical control, 25

principle of effectiveness, 26

overlapping control, 26

layered defense, 27

principle of weakest link, 27

1.9 Where the Field Is Headed

We conclude most chapters with a paragraph or two highlighting some interesting work being done. For students interested in pursuing a career in security, these sections may identify an area of interest.

The number of computer security professionals is growing rapidly but so, too, is the number of attackers. The U.S. CERT and its counterpart organizations around the world do an exceptional job of tracking serious system vulnerabilities and countermeasures. Several efforts are underway to categorize and catalog computer security incidents and vulnerabilities (for example, Landwehr et al. [LAN94]). Being able to sort and correlate incident information is critical to successful forensic analysis of large incidents.

The severity of the computer security problem is causing many companies, schools and universities, government bodies, and individuals to address their security needs. Looking at these groups separately can be daunting and also risks your missing the ones who do it really well. Several groups have promulgated codes of security best practices. The Information Security Forum [ISF00] and the Internet Security Alliance [ISA02] have published codes of best security practices, which are recommendations for secure computing. Governments and regulatory bodies are beginning to enforce standards.

Obviously, the popular attack point today is computer networks, and specifically the Internet. Do not be misled, however, into thinking that all computer security is network security. As you will see throughout the remainder of this book, network security problems are often just the latest instantiation of computer security problems that predate the rise of the Internet—problems such as identification and authentication, limited privilege, and designing for security. So, although the problems of networks are pressing, they are long-standing, open problems.

1.10 To Learn More

Today's bookshelves are full of books about computer security: its meaning, its impact, and the people involved in preventing malicious behavior. However, two key works form the foundation for much of subsequent work in computer security: the exploration of vulnerabilities and controls by Ware [WAR79] and the security technology planning study by Anderson [AND72]. The concepts and ideas put forth are still relevant, even though the papers are several decades old.

Two very good surveys of the field of computer security are Denning's classic textbook [DEN82], much of which is still valid, and Gollmann's textbook [GOL99]. Also, Schneier's book [SCH00] is an enjoyable overview.

Some books focus on a particular aspect of security. Confidentiality is explored by the Dennings [DEN79a], and integrity is studied carefully by Welke and Mayfield [WEL90, MAY91, NCS91b]. Availability considerations are documented by Pfleeger and Mayfield [PFL92] and by Millen [MIL92].

Since 1991, the National Research Council of the National Academy of Science has published seven reports on the state of aspects of computer security. The first volume [NRC91] lays out the significant risk of the then current state of computing. Frighteningly, the latest report [NRC02] concludes: "not much has changed with respect to security as it is practiced." These volumes are worth reading for their realistic assessment of today's threats and preparedness.

For further study of threats affecting computer systems, see [DEN99]. For examples of how computer system vulnerabilities are exploited, you may want to read [STO89, SHI96].

1.11 Exercises

1: Distinguish among vulnerability, threat, and control.

2: Theft usually results in some kind of harm. For example, if someone steals your car, you may suffer financial loss, inconvenience (by losing your mode of transportation), and emotional upset (because of invasion of your personal property and space). List three kinds of harm a company might experience from theft of computer equipment.

3: List at least three kinds of harm a company could experience from electronic espionage or unauthorized viewing of confidential company materials.

4: List at least three kinds of damage a company could suffer when the integrity of a program or company data is compromised.

5: Describe two examples of vulnerabilities in automobiles for which auto manufacturers have instituted controls. Tell whether you think these controls are effective, somewhat effective, or ineffective.

6: One control against accidental software deletion is to save all old versions of a program. Of course, this control is prohibitively expensive in terms of cost of storage. Suggest a less costly control against accidental software deletion. Is your control effective against all possible causes of software deletion? If not, what threats does it not cover?

7: On a typical multiuser computing system (such as a shared Unix system at a university or an industry), who *can* modify the code (software) of the operating system? Of a major application program such as a payroll program or a statistical analysis package? Of a program developed and run by a single user? Who *should be* permitted to modify each of these examples of code?

8: Suppose a program to print paychecks secretly leaks a list of names of employees earning more than a certain amount each month. What controls could be instituted to limit the vulnerability of this leakage?

9: Some terms have been introduced intentionally without definition in this chapter. You should be able to deduce their meanings. What is an electronic spy? What is an information broker?

10: Preserving confidentiality, integrity, and availability of data is a restatement of the concern over interruption, interception, modification, and fabrication. How do the first three concepts relate to the last four? That is, is any of the four equivalent to one or more of the three? Is one of the three encompassed by one or more of the four?

11: Do you think attempting to break in to (that is, obtain access to or use of) a computing system without authorization should be illegal? Why or why not?

12: Describe an example (other than the one mentioned in this chapter) of data whose confidentiality has a short timeliness, say, a day or less. Describe an example of data whose confidentiality has a timeliness of more than a year.

13: Do you currently use any computer security control measures? If so, what? Against what attacks are you trying to protect?

14: Describe an example in which absolute denial of service to a user (that is, the user gets no response from the computer) is a serious problem to that user. Describe another example where 10 percent denial of service to a user

(that is, the user's computation progresses, but at a rate 10 percent slower than normal) is a serious problem to that user. Could access by unauthorized people to a computing system result in a 10 percent denial of service to the legitimate users? How?

15: When you say that software is of high quality, what do you mean? How does security fit in your definition of quality? For example, can an application be insecure and still be "good"?

16: Developers often think of software quality in terms of faults and failures. Faults are problems, such as loops that never terminate or misplaced commas in statements, that developers can see by looking at the code. Failures are problems, such as a system crash or the invocation of the wrong function, that are visible to the user. Thus, faults can exist in programs but never become failures, because the conditions under which a fault becomes a failure are never reached. How do software vulnerabilities fit into this scheme of faults and failures? Is every fault a vulnerability? Is every vulnerability a fault?

17: Consider a program to display on your web site your city's current time and temperature. Who might want to attack your program? What types of harm might they want to cause? What kinds of vulnerabilities might they exploit to cause harm?

18: Consider a program that allows consumers to order products from the web. Who might want to attack the program? What types of harm might they want to cause? What kinds of vulnerabilities might they exploit to cause harm?

19: Consider a program to accept and tabulate votes in an election. Who might want to attack the program? What types of harm might they want to cause? What kinds of vulnerabilities might they exploit to cause harm?

20: Consider a program that allows a surgeon in one city to assist in an operation on a patient in another city via an Internet connection. Who might want to attack the program? What types of harm might they want to cause? What kinds of vulnerabilities might they exploit to cause harm?

21: Computer security failures appear frequently in the daily news. Cite a reported failure that exemplifies one (or more) of the principles listed in this chapter: easiest penetration, adequate protection, effectiveness, weakest link.