CHAPTER 1

SECURITY CONCEPTS

Chapter Goals

- The principles of any security mechanism
- The need for the security
- Security models
- Denial-of-Service (DoS) and other types of active attacks
- Viruses, Worms, Trojan horses, and Java Applets
- Java Security
- Applet and ActiveX controls

1.1 SECURITY INTRODUCTION

This is a book on network and Internet security. As such, before we embark on our journey of understanding the various concepts and technical issues related to security, it is essential to know what we are trying to protect. What are the dangers of using computers, computer networks, and the biggest network of them all, the Internet? What are the likely pitfalls? What happens if we do not implement the right security policies, frameworks, and technology? This chapter attempts to provide answers to these basic questions.

We start with a discussion of the fundamental point: Why is security required in the first place? People sometimes say that security is like statistics: what it reveals is trivial, what it conceals is vital. In other words, the right security infrastructure opens up just enough doors. We will discuss a few real-life incidents that should prove beyond any doubt that security is important. Now

that critical business and other types of transactions are being conducted over the Internet to such a large extent, inadequate or improper security mechanisms can destroy a business or play havoc with people's lives.

We will also discuss the key principles of security. These principles will help us identify the various areas that are crucial while determining the security threats and possible solutions. Electronic documents and messages are now considered the equivalent to paper documents in terms of their legal validity, and we will examine the implications of this new view of information.

1.2 THE NEED FOR SECURITY

Most of the first computer applications had no or at best, very little, security. This lack of security continued for a number of years until the importance of data was truly realized. Until then, computer data was considered useful, but not something that needed to be protected. When computer applications were developed to handle financial and personal data, the real need for security was felt like never before. People realized that the data on computers is an extremely important aspect of modern life, and various areas in security began to gain importance. Two typical examples of security mechanisms are as follows:

- Provide a user id and password to every user, and use that information to authenticate a user
- Encode information stored in the databases in some fashion so that it is not visible to users who do not have the right permissions.

Organizations employed their own mechanisms to provide security. As technology improved, the communication infrastructure became extremely mature, and newer applications were developed to meet various user demands and needs. Soon, people realized that the basic security measures were not enough.

The Internet is used globally, and there were many examples of what could happen if there was insufficient security built into the applications developed for the Internet. Figure 1.1 shows such an example of what can happen when you use your credit card for making purchases over the Internet. From the user's computer, the user's details, such as the user id, order details, such as the order id and item id, and payment details, such as the credit card information, travel across the Internet to the merchant's server. The merchant's server stores these details in its database.

There are various security holes in this process. First, an intruder can capture the credit card details as they travel from the client to the server. If we somehow protect this transit from an intruder's attack, it still does not

solve our problem. Once the merchant receives the credit card details and validates them to process the order and obtain payments, the merchant stores the credit card details in its database. An attacker can simply access this database and gain access to all the credit a card numbers stored therein! One Russian attacker (called Maxim) managed to hack a merchant's Internet site and obtain 300,000 credit card numbers from its database. He then attempted extortion by demanding protection money ($100,000) from the merchant. The merchant refused to oblige. Following this, the attacker published about 25,000 of the credit card numbers on the Internet. Some banks reissued all the credit cards at a cost of $20 per card, and others warned their customers about unusual entries in their statements.

Such attacks can obviously lead to great losses, both in terms of finances and goodwill. Generally, it takes about $20 to replace a credit card. Therefore, if a bank has to replace 300,000 such cards, the total cost of such an attack is about $6 million. Had the merchant in the example employed proper security measures, he would have saved money and bother.

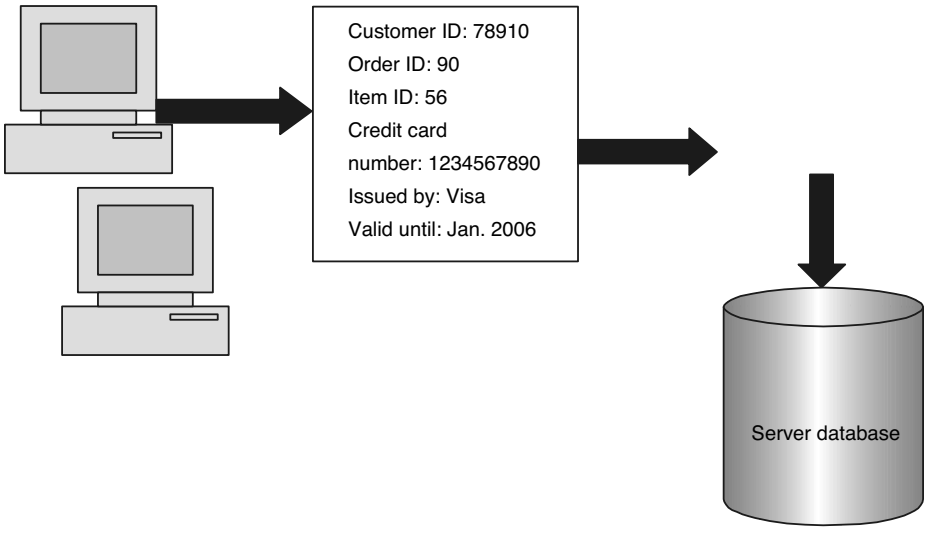

FIGURE 1.1 Example of information traveling from a client to a server over the Internet

Of course, this is just one example. More cases have been reported, and the need for proper security is increasing with every attack. In another example, in 1999, a Swedish hacker broke into Microsoft's Hotmail Website and created a mirror site. This site allowed anyone to enter any Hotmail user's email id and read the user's emails.

Also in 1999, two independent surveys were conducted to invite people's opinions about the losses that occur due to successful attacks on security. One survey pegged the losses at an average of $256,296 per incident, and another found the average was $759,380 per incident. In the following year, this figure rose to $972,857.

1.3 SECURITY APPROACHES

The last twenty years have witnessed a major development in a formal methods used to improve security protocols. The design of such protocols has so far been largely an empirical and ad-hoc procedure, giving rise to various approaches. These methods are now being systematically applied to develop a system that is both efficient and can comply with strong security requirements.

1.3.1 Security Models

An organization can take several approaches to implement its security model:

- **No security:** In this simplest case, the approach could be a decision to implement no security at all.
- **Security through obscurity:** In this model, a system is secure simply because nobody knows about its existence and content.
- **Host security:** In this scheme, the security for each host is enforced individually. This is a very safe approach, but the trouble is that it cannot scale well. The complexity and the diversity of modern sites/organizations make the task even harder.
- **Network security:** Host security is difficult to achieve as organizations grow and become more diverse. In this technique, the focus is to control network access to various hosts and their services, rather than the individual host's security. This is an efficient and scalable model.

1.3.2 Security Management Practices

Good security management practices always include a security policy. Putting a security policy in place is actually quite difficult. A good security policy and its proper implementation go a long way in ensuring adequate security management practices. A good security policy generally takes care of four key aspects.

- **Affordability:** How much money and effort does this security implementation cost?
- **Functionality:** What is the mechanism of providing security?
- **Cultural Issues:** Does the policy take into consideration people's expectations, working style, and beliefs?
- **Legality:** Does the policy meet the legal requirements?

Once a security policy is in place, the following points should be ensured:

a. Include an explanation of the policy to all concerned.

b. Outline everybody's responsibilities.

c. Use simple language in all communications.

d. Accountability should be established.

e. Provide for exceptions and periodic reviews.

1.4 PRINCIPLES OF SECURITY

Having discussed some of the attacks that have occurred in real life, let us now classify the principles related to security. This will help us understand the attacks better and think about the possible solutions.

Let us assume that a person, A, wants to send a check worth $100 to another person, B. Normally, what are the factors that A and B think of in such a case? A will write the check for $100, put it inside an envelope, and send it to B.

- A wants to ensure that no one expects B will get the envelope, and even if someone else gets it, she does not want anyone to know about the details of the check. This is the principle of *confidentially*.
- A and B would like to make sure that no one can tamper with the contents of the check (such as its amount, date, signature, or name of the payee). This is the principle of *integrity*.
- B would like to be assured that the check has indeed come from A, and not from someone else posing as A (as it could be a fake check). This is the principle of *authentication.*
- What will happen tomorrow if B deposits the check into her account, the money is transferred from A's account, and then A claims to have not written/sent the check? The court of law will use A's signature to disallow A to refute this claim and settle the dispute. This is the principle of *non-repudiation*.

These are the four chief principles of security. There are two more, *access control* and *availability*, which are not related to the particular message, but are linked to the overall system as a whole.

1.4.1 Confidentiality

The principle of *confidentiality* specifies that only the sender and the intended recipient(s) should be able to access the content of a message. Confidentiality gets compromised if an unauthorized person is unable to access a message. An example of compromising the confidentiality of a message is shown in Figure 1.2. The user of a computer A sends the message to the user of a computer B. (From here onwards, we use "A" to mean the user A, and "B" to mean user B, although we just show the computers of the users A and B). Another user, C, gets access to this message, which is not desired, and therefore, defeats the purpose of confidentiality. An example of this could be a confidential email message sent by A and B. This type of attack is called *interception*.

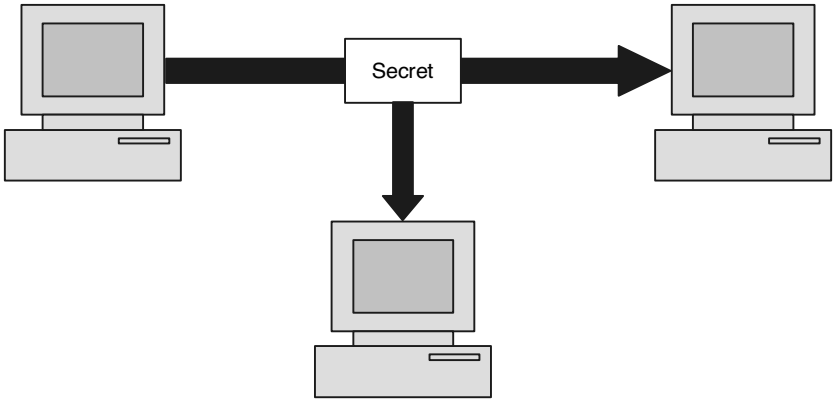

FIGURE 1.2 The loss of confidentiality: Interception causes the loss of the message's confidentiality.

1.4.2 Authentication

Authentication mechanisms help establish proof of identities. The authentication process ensures that the origin of an electronic message or document is correctly identified. For instance, suppose that user C sends an electronic document over the Internet to user B. However, the trouble is that user C is posing as user A when she sent this document to user B. However, would user B know that the message has come from user C, who is posing as user A? A real-life example of this would be the case of a user C, posing as user A, sending a funds transfer request (from A's account to C's account) to bank B.

The bank will happily transfer the funds from A's account to C's account, and it would think that user A has requested the funds transfer. This concept is shown in Figure 1.3. This type of attack is called *fabrication*.

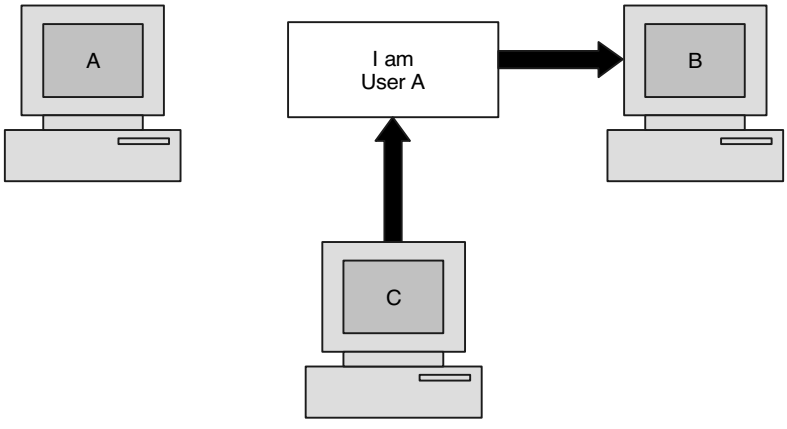

FIGURE 1.3 The absence of authentication

1.4.3 Integrity

When the contents of a message are changed after the sender sends it, but before it reaches the intended recipient, we say that the integrity of the message is lost. For example, suppose you write a cheque for $100 to pay for some goods. However, when you see your next account statement, you are startled to see that the cheque resulted in a payment of $1,000. This is type of case is about the loss of the message's integrity. Conceptually, this is shown in Figure 1.4. User C tampers with a message originally sent by user A, which was actually destined for user B. User C somehow manages to access it, change its contents, and send the changed message to user B. User B has no way of knowing that the contents of the message were changed after user A sent it. User A also does not know about this change. This type of attack is called *modification*.

1.4.4 Non-Repudiation

There are situations where a user sends a message, and then says that she never sent that message. For instance, user A could send a funds transfer request to bank B over the Internet. After the bank performs the funds transfer as per A's instructions, A could claim that she never sent the funds transfer instruction to the bank. Thus, A repudiates, or denies, her funds transfer

instruction. The principle of *non-repudiation* defeats the possibility of denying something after having done it.

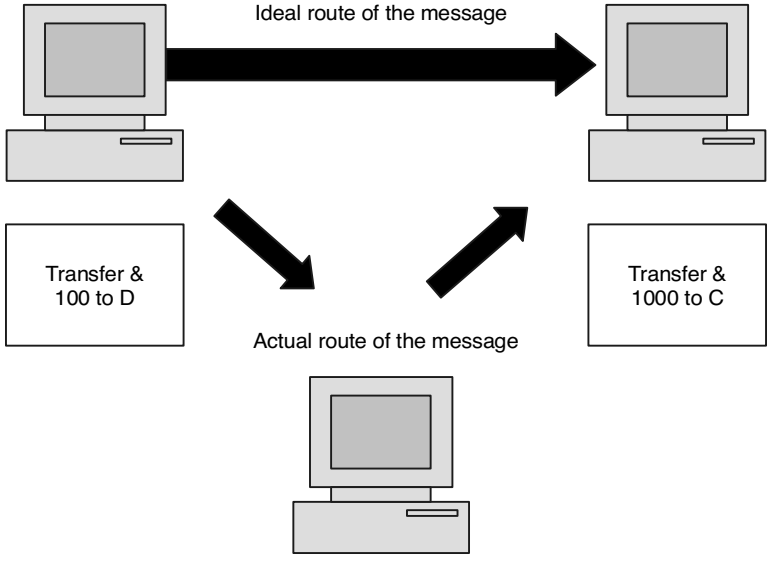

FIGURE 1.4 Loss of integrity

1.4.5 Access Control

The principle of *access control* determines *who* should be able to access what. For instance, we should be able to specify that user A can view the records in a database, but cannot update them. However, user B might be allowed to make updates, as well. An access control is broadly related to two areas: *role management* and *rule management.* Role management concentrates on the user side (which user can do what), whereas rule management focuses on the decisions taken, and so an access control matrix is prepared. A list of items is generated, including what they can access (*e.g.*, it can say that user A can write to file X, but can only update files Y and Z). An *Access Control List (ACL)* is a subset of an access control matrix.

1.4.6 Availability

The principle of *availability* states that resources should be available to authorized parties at all times. For example, due to the intentional actions of another unauthorized user C, the authorized user A may not be able to

contact a server computer B, as shown in Figure 1.5. This would defeat the principle of availability. Such an attack is called *interruption*.

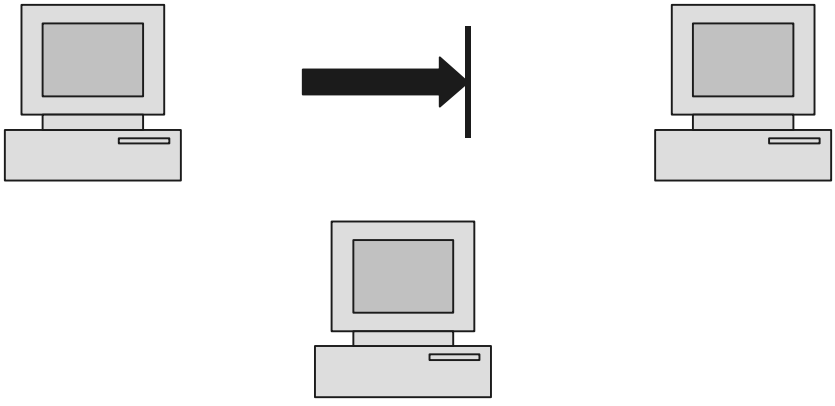

FIGURE 1.5 Attack on availability

Having discussed the various principles of security, let us now discuss the different types of attacks that are possible from a technical perspective.

1.5 TYPES OF ATTACKS

We can classify the types of attacks on computers and network systems into two categories for a better understanding: (*a*) the theoretical concepts behind these attacks and (*b*) practical approaches used by the attackers.

1.5.1 Theoretical Concepts

The principle of security faces threats from various attacks. These attacks are generally classified into four categories. They are

- **Interception:** Discussed in the context of *confidentiality* earlier.
- **Fabrication:** Discussed in the context of *authentication* earlier.
- **Modification:** Discussed in the context of *integrity* earlier.
- **Interruption:** Discussed in the context of *availability* earlier.

These attacks are further grouped into two types: passive attacks and active attacks, as shown in Figure 1.6.

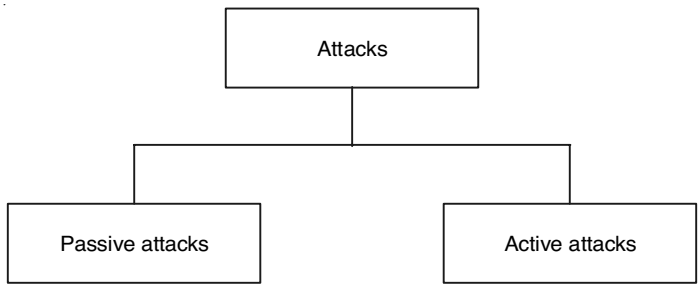

FIGURE 1.6 Types of attacks

1.5.1.1 Passive Attacks

Passive attacks are those wherein the attacker indulges in eavesdropping or the monitoring of data transmissions. The attacker attempts to obtain information that is in transit. The term passive indicates that the attacker does not attempt to perform any modifications to the data. In fact, this is also why passive attacks are harder to detect, thus, the general approach to deal with passive attacks is to think about prevention, rather than detection or corrective actions.

Figure 1.7 shows a further classification of passive attacks into two subcategories. These categories are the release of the message contents and traffic analysis.

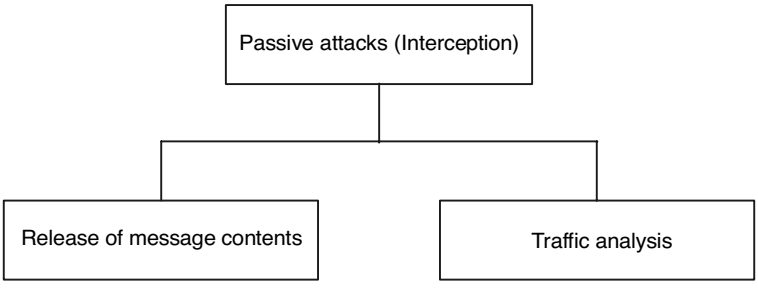

FIGURE 1.7 Passive attacks

The *release of message contents* is quite simple to understand. When we send a confidential email message to our friend, we only want her to access it. Otherwise, the contents of the message are released against our wishes to someone else. Using certain security mechanisms, we can prevent the release of the message contents. For example, we can encode messages using a code language, so that only the desired parties understand the contents of a message because only they know the code language. However, if many such messages are passing through, a passive attacker could try to figure out the

similarities between them to come up with a pattern that provides her some clues regarding the communication that is taking place. Such attempts at analyzing (encoded) messages to come up with likely patterns are the work of *traffic analysis* attacks.

1.5.1.2 Active Attacks

Unlike passive attacks, *active attacks* are based on the modification of the original messages in some manner or on the creation of a false message. These attacks cannot be prevented easily. However, they can be detected with some effort, and attempts can be made to recover from them. These attacks can be in the form of interruption, modification, and fabrication.

- Interruption attacks are called masquerade attacks.
- Modification attacks can be classified further into the replay attacks and alteration of messages.
- Fabrication causes Denial of Service (DoS) attacks.

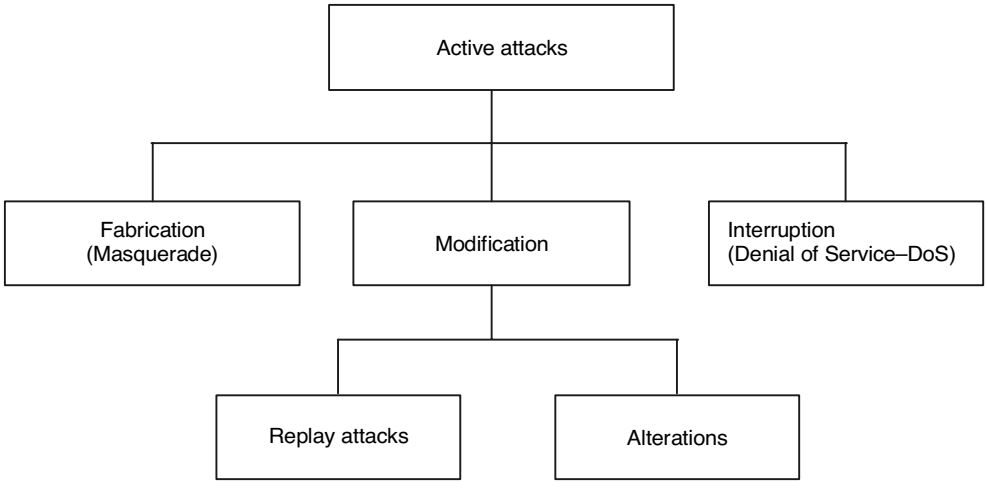

FIGURE 1.8 Active attacks

A *masquerade* is caused when an unauthorized entity pretends to be another entity. User C might pose as user A and send a message to user B. User B might be led to believe that the message indeed came from user A.

In a replay attack, a user captures a sequence of events, or some data units, and resends them. For instance, suppose user A wants to transfer some amount to user C's bank account. Both users A and C have accounts with bank B. User A might send an electronic message to bank B, requesting the

funds transfer. User C could capture this message and send a second copy of the same to bank B. Bank B would have no idea that it is an unauthorized message and would treat this as a second, and different, funds transfer request from user A. Therefore, user C would get the benefit of the funds transfer twice: once actually authorized and once through a replay attack.

The alteration of messages involves some change to the original message. For instance, suppose user A sends an electronic message to transfer $10,000 to C's account. The beneficiary captures this and changes it to transfer $100,000 to B's account. Note that both the beneficiary and the amount have been changed: only one of these could have caused the alteration of the message.

Denial-of-Service (DoS) attacks make an attempt to prevent legitimate users from accessing some services that they are eligible for. For instance, an unauthorized user might send too many emails so as to flood the network and deny other legitimate users access to the network.

1.5.2 The Practical Side of Attacks

The attacks discussed earlier can come in a number of forms in real life. They can be classified into two broad categories: application-level attacks and network-level attacks, as shown in Figure 1.9.

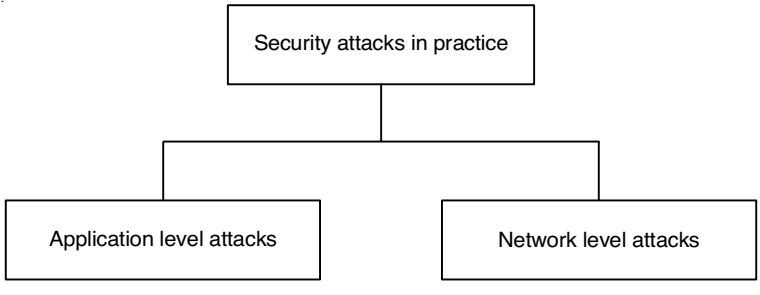

FIGURE 1.9 Practical side of attacks

- *Application level attacks*: These attacks happen at an application level in the sense that the attacker attempts to access, modify, or prevent access to information of a particular application or the application itself. Examples of this are trying to obtain someone's credit information on the Internet or changing the contents of message to change the amount in a transaction.
- *Network level attacks*: These attacks are generally aimed at reducing the capabilities of network. These attacks make an attempt to either slow

down, or completely bring to halt, a computer network. Note that this automatically can lead to application level attacks, because once someone is able to gain access to a network, she is able to access/modify at least some sensitive information, causing havoc.

These two types of attacks can be attempted by using varies mechanisms. These attacks are not encompassed in the above two categories, since they can span across application as well as network levels.

1.5.2.1 Virus

One can launch an application-level attack or a network level attack using a virus. A *virus* is a piece of program code that attaches itself to legitimate program code and runs when the legitimate program runs.

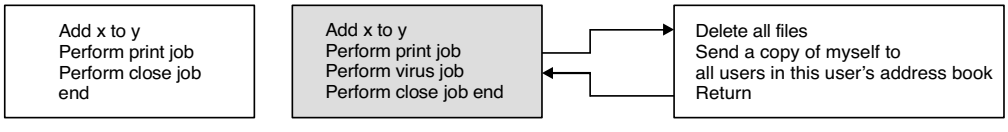

FIGURE 1.10 Virus

It can then infect other programs in that computer or programs that are on other computers but on the same network. This is shown in Figure 1.10. After deleting all the files from the current user's computer, the virus self-propagates by sending its code to all users whose e-mail addresses are stored in the current user's address book.

Viruses can also be triggered by specific events (*e.g.,* a virus could automatically execute at 12 p.m. every day). Viruses cause damage to computers and network systems, but this damage can be repaired, assuming that the organization deploys good backup and recovery producers.

1.5.2.2 Worms

Similar in concept to a virus, a worm is actually different in implementation. A virus modifies a program (*i.e.,* it attaches itself to the program under attack). This is shown in Figure 1.11. The replication grows so much that ultimately the computer or the network, on which the worm resides, become very slow, finally coming to a halt. Thus, the basic purpose of a worm attack is different from that of a virus. A *worm* attempts to make the computer or the network under attack unusable by consuming all its resources.

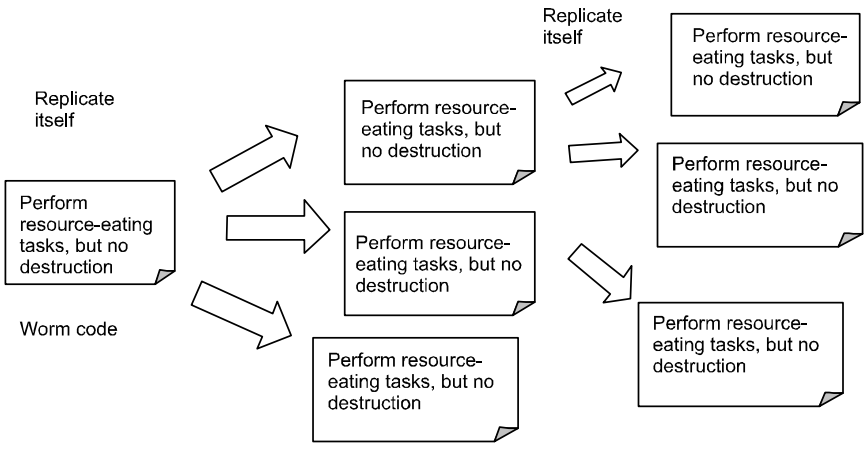

FIGURE 1.11 Worm

A worm does not perform any destructive actions, but instead only consumes system resources to bring it down.

1.5.2.3 Trojan Horse

A Trojan horse is a hidden piece of code, like a virus. However, the purpose of a Trojan horse is different. The main purpose of a virus is to make modifications to the target computer or network, whereas a *Trojan horse* attempts to reveal confidential information to an attacker. The name (Trojan horse) is taken from the secret attack executed by Greek soldiers, who hid inside a large hollow horse that was pulled into the city of Troy by its citizens, unaware of its contents. Once the Greek soldiers entered the city of Troy, they opened the gates for the rest of their army.

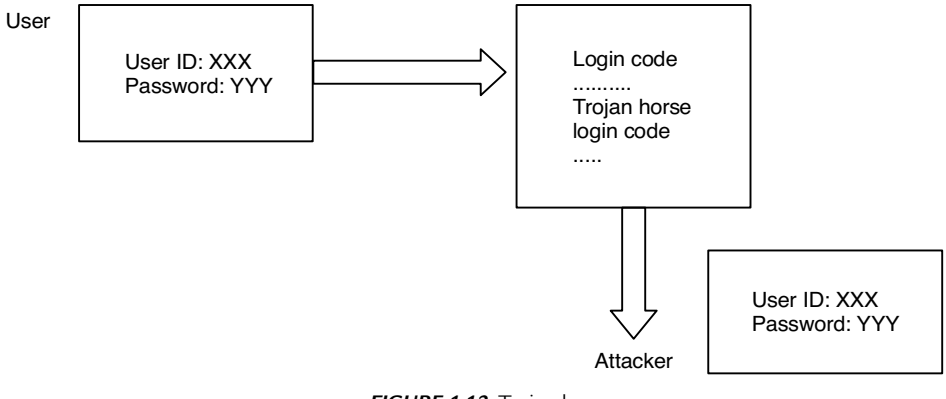

FIGURE 1.12 Trojan horse

In a similar fashion, a Trojan horse could silently sit in the code for a login screen by attaching itself to it. When the user enters the user ID and password, the Trojan horse captures these details and sends this information to the attacker without the knowledge of the user who entered the ID and password. The attacker can then use the user ID and password to gain access to the system.

1.5.2.4 Applets and ActiveX Controls

Applets and ActiveX controls were born through the technological development of the World Wide Web (WWW) applications. In its simplest form, the Web consists of the communication between client and server computers using a communications protocol called the Hyper Text Transfer Protocol (HTTP). The client uses software called a Web browser. The server runs a program called a Web server. In its simplest form, a browser sends an HTTP request for a Web page to a Web server. The Web server locates this Web page (actually a computer file) and sends it back to the browser, again using HTTP. The Web browser interprets the contents of that file and shows the results on the screen to the user. This is shown in Figure 1.13. Here, the client sends a request for the web page *www.yahoo.com/info*, which the server sends back to the client.

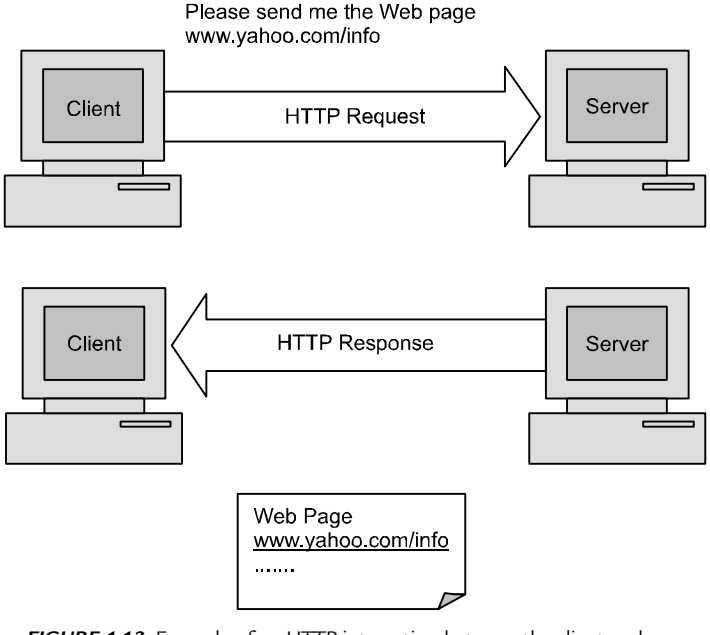

FIGURE 1.13 Example of an HTTP interaction between the client and server