

## Module 1

### Introduction to the Concepts of Security

We are living in the information age. We need to keep information about every aspect of our lives. Information is an asset that has a value like any other asset. As an asset, information needs to be secured from attacks.

To be secured, information needs to be hidden from **unauthorized access** (confidentiality), protected from **unauthorized change** (integrity) and **available to an authorized entity** when it is needed (availability).

A few decades ago, the information collected by an organization was stored on **physical files**. The confidentiality of the files was achieved by restricting the access to a few authorized and trusted people in the organization. In the same way, only a few authorized people were allowed to change the contents of the files. Availability was achieved by designating at least one person who would have access to the files at all times.

With the advent of computers, **information storage became electronic**. Instead of being stored on physical media, it was stored in computers. The three security requirements, however, did not change. The files stored in computers require confidentiality, integrity, and availability. The implementation of these requirements is different and more challenging.

During the last two decades, **computer networks** created a revolution in the use of information. Information is now distributed. Authorized people can send and retrieve information from a distance using computer networks. Although the three above-mentioned requirements confidentiality, integrity, and availability have not changed, they now have some new - dimensions. Not only should information be confidential when it is stored in a computer, there should also be a way to maintain its confidentiality when it is transmitted from one computer to another.

### Need for Security

Two typical examples of security mechanisms were as follows:

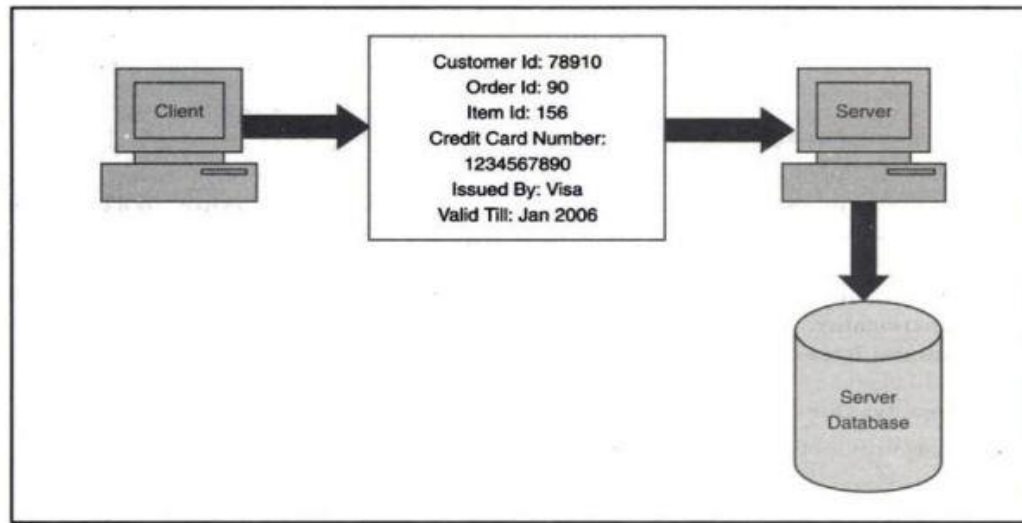
- Provide a **user id and password to every user**, and use that information to authenticate a user
- **Encode information stored in the databases** in some fashion, so that it is **not visible to users** who do not have the right permissions.

Organizations employed their own mechanisms in order to provide for these kinds of basic security mechanisms.

Furthermore, the Internet took the world by storm, and there were many examples of what could happen if there was insufficient security built in applications developed for the Internet.

Figure 1.I 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 details such as user id, order details such

as order id and item id, and payment details such as credit card information travel across the Internet to the merchant's server i.e. to the merchant's computer). The merchant's server stores these details in its database.



**Fig. 1.1** Example of information traveling from a client to a server over the Internet

There are various **security holes** here.

First of all, 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 so as to process the order and later obtain payments, the merchant stores the credit card details into its database. Now, an attacker can simply succeed in accessing this **database, and therefore, gain access to all the credit card numbers stored** therein!

### 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 also help us in thinking about the possible solutions to tackle them. We shall take an example to understand these concepts.

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 will think of, in such a case? A will write the check for \$100, put it inside an envelope, and send it to B.

- A will like to ensure that no one except B gets the envelope, and even if someone else gets it, she does not come to know about the details of the check. This is the principle of **confidentiality**.
- A and B will further like to make sure that no one can tamper with the contents of the check (such as its amount, date, signature, name of the payee, etc.). 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 in that case). This is the principle of **authentication**.

- What will happen tomorrow if B deposits the check in her account, the money is transferred from A's account to B's account, and then A refuses having 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 a particular message, but are linked to the overall system as a whole.

## 1. Confidentiality

The principle of **confidentiality specifies that only the sender and the intended recipient(s) should be able to access the contents of a message**. Confidentiality gets compromised if an unauthorized person is able to access a message. Example of compromising the confidentiality of a message is shown in Fig. 1.2. Here, the user of computer A sends a message to the user of computer B. (The term A to mean the user A, B to mean user B etc., although we shall just show the computers of user A, B, etc.). Another user C gets access to this message, which is not desired, and therefore, defeats the purpose of confidentiality. Example of this could be a confidential email message sent by A to B, which is accessed by C without the permission or knowledge of A and B. **This type of attack is called as interception.**

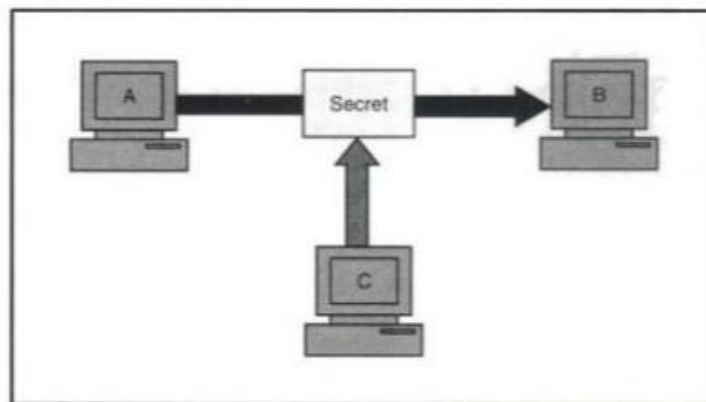
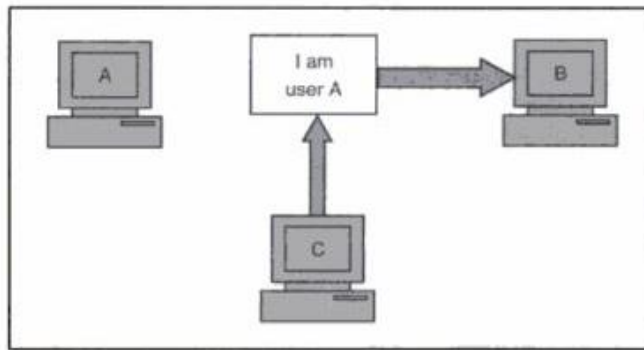


Fig. 1.2 Loss of confidentiality

Note: **Interception causes loss of message confidentiality.**

## 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 had posed as user A when she sent this document to user B. How would user B know that the message has come from user C, who is posing as user A? A real life example of this could 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 might happily transfer the funds from A's account to C's account-after all, it would think that user A has requested for the funds transfer! This concept is shown in Fig. 1.3. **This type of attack is called as fabrication.**

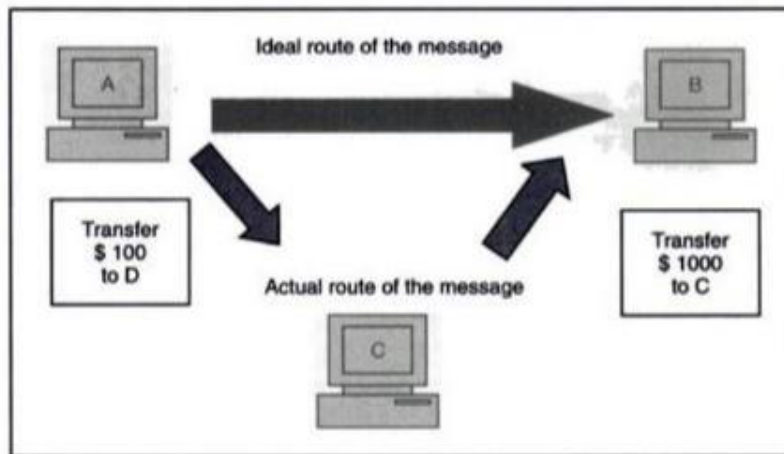


**Fig. 1.3** Absence of authentication

Note: Fabrication is possible in the absence of proper authentication mechanisms.

### 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 the goods bought from the US. However, when you see your next account statement, you are startled to see that the cheque resulted in a payment of \$1000! This is the case for loss of message integrity. Conceptually, this is shown in Fig. 1.4. Here, user C tampers with a message originally sent by user A, which is 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 had sent it. User A also does not know about this change. This type of attack is called as modification.



**Fig. 1.4** Loss of integrity

Note: Modification causes loss of message integrity.

### 4. Non-repudiation

There are situations where a user sends a message, and later on refuses that she had 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 such possibilities of denying something, having done it.**

*Note:* Non-repudiation does not allow the sender of a message to refute the claim of not sending that message.

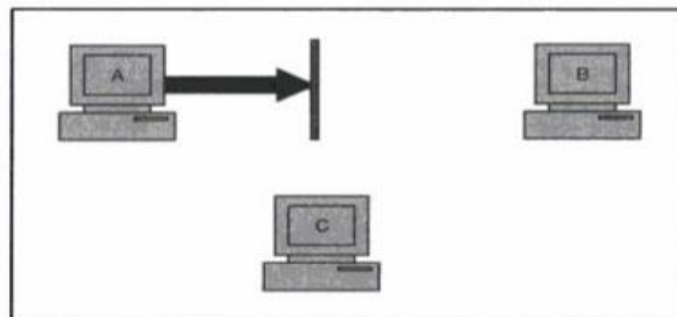
## 5. Access Control

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 mechanism can be set up to ensure this. 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 resources side (**which resource is accessible, and under what circumstances**). Based on the decisions taken here, an **access control matrix** is prepared, which lists the users against a list of items 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.

*Note:* Access control specifies and controls who can access what.

## 6. Availability

The principle of availability states **that resources (i.e. information) should be available to authorized parties at all times**. For example, due to the intentional actions of another unauthorized user C, an authorized user A may not be able to contact a server computer B, as shown in Fig. 1.5. This would defeat the principle of availability. Such an **attack is called as interruption**.



**Fig. 1.5** Attack on availability

*Note:* Interruption puts the availability of resources in danger.

### Types of Attacks

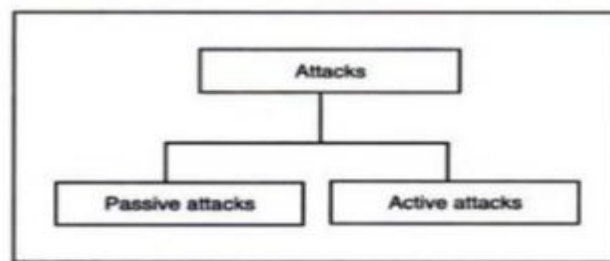
We can classify the types of attacks on computers and network systems into two categories for better understanding:

- (a) Theoretical concepts behind these attacks
- (b) Practical approaches used by the attackers.

**Theoretical Concepts** As we have discussed earlier, the principles of security face threat from various attacks. These attacks are generally classified into four categories, as mentioned earlier. They are:

- **Interception**-Discussed in the context of confidentiality, earlier. It means that an unauthorized party has gained access to a resource. The party can be a person, program or computer-based system. Examples of interception are **copying of data or programs** and **listening to network traffic**.
- **Fabrication**-Discussed in the context of authentication, earlier. This involves creation of illegal objects on a computer system. Example: **the attacker may add fake records to a database**.
- **Modification**-Discussed in the context of integrity, earlier. Example: the **attacker may modify the values in a database**.
- **Interruption**-Discussed in the context of availability, earlier. Here the resource becomes unavailable, lost or unusable. Examples of interruption are **causing problems to a hardware device, erasing program, data**.

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



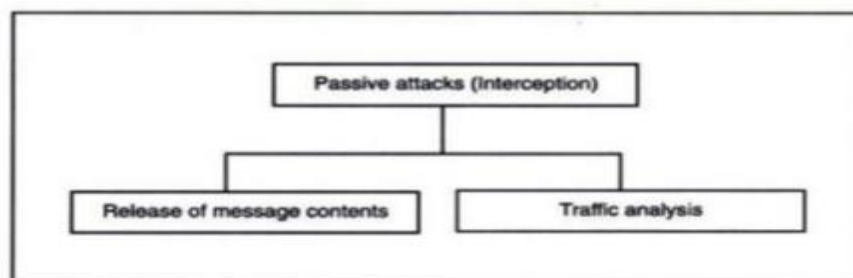
**Fig. 1.6** Types of attacks

## 1. Passive attacks

Passive attacks are those, wherein the attacker indulges in eavesdropping or monitoring of data transmission. In other words, **the attacker aims 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**.

**Note:** Passive attacks do not involve any modifications to the contents of an original message.

Figure 1.7 shows further classification of passive attacks into two sub-categories. These categories are **release of message contents** and **traffic analysis**.



**Fig. 1.7** Passive attacks

**Release of message** contents is quite simple to understand. When we send a confidential email message to our friend, we desire that only she be able to access it. Otherwise, the contents of the message are released against our wishes to someone else. Using certain security mechanisms, we can prevent release of 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 some sort of pattern that provides her some clues regarding the communication that is taking place. Such attempts of analyzing (encoded) messages to come up with likely patterns are the work of the **traffic analysis attack**.

## 2. Active attacks

Unlike passive attacks, the active attacks are based on **modification of the original message in some manner, or on 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**.

Note: In active attacks, the contents of the original message are modified in some way.

- Trying to pose as another entity involves **masquerade** attacks.
- Modification attacks can be classified further into **replay attacks** and **alteration of messages**.
- Fabrication sometimes causes **Denial Of Service (DOS)** attacks.

This classification is shown in Fig. 1.8.

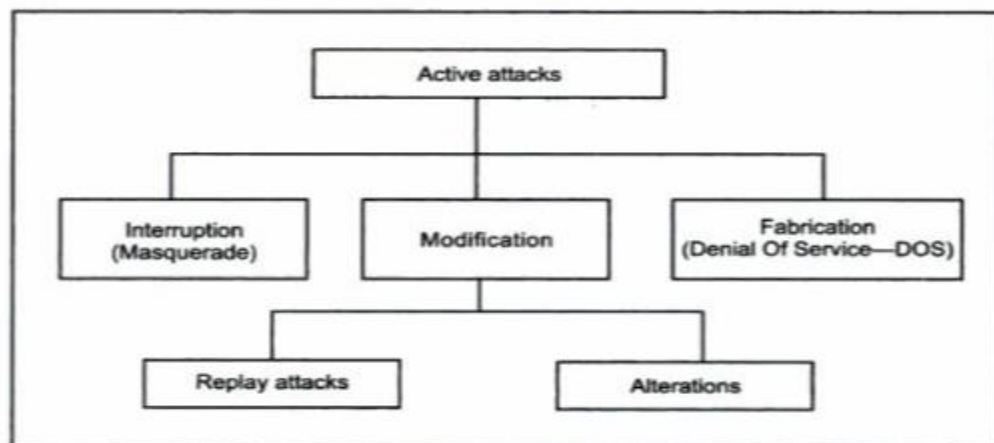


Fig. 1.8 Active attacks

**Masquerade** is caused when **an unauthorized entity pretends to be another entity**. As we have seen, 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

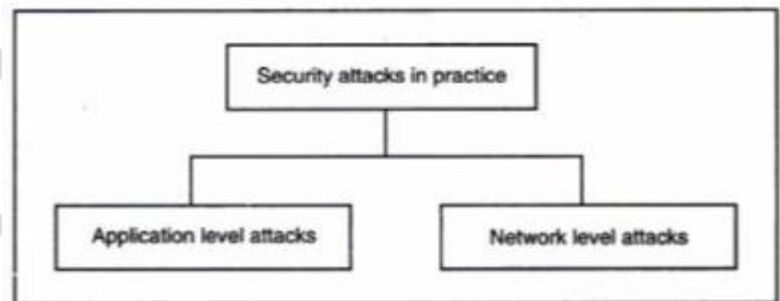


have accounts with bank B. User A might send an electronic message to bank B, requesting for the funds transfer. User C could capture this message, and send a second copy of the same to bank B. Bank B would have no idea that this is an unauthorized message, and would treat this as a second, and different,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

**Alteration of messages** involves **some change to the original message**. For instance, suppose user A sends an electronic message Transfer \$1000 to D's account to bank B. User C might capture this, and change it to Transfer \$10000 to C's account. Note that both the beneficiary and the amount have been changed- instead, only one of these could have also caused alteration of the message.

**Denial of Service (DOS)** attacks make an attempt to prevent legitimate users from accessing some services, which they are eligible for. For instance, an unauthorized user might send too many login requests to a server using random user ids one after the other in quick succession, so as to flood the network and deny other legitimate users an access to the network.

**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 Fig. 1.9.



**Fig. 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 card information on the Internet, or changing the contents of a message to change the amount in a transaction**, etc.
- **Network level attacks:** These attacks generally aim at **reducing the capabilities of a network** by a number of possible means. These attacks generally **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, usually she is able to access/modify at least some sensitive information.

## Programs that attack

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.

It can then infect other programs in that computer, or programs that are in other computers but on the same network. This is shown in Fig.

1.10. In this example, after deleting all the files from the current user's computer, the virus self-propagates by sending its code to all users whose email addresses are stored in the current user's address book.

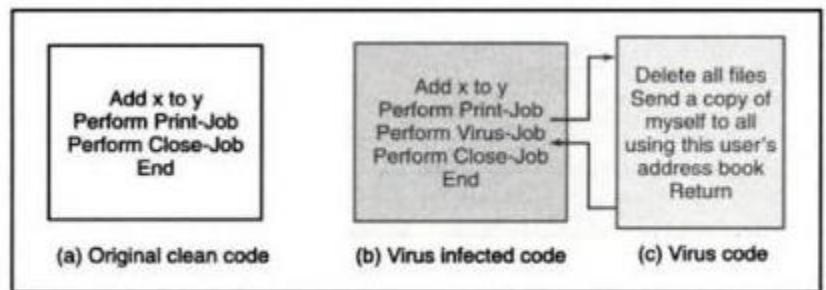


Fig. 1.10 Virus

Viruses can also be triggered by specific events (e.g. a virus could automatically execute at 12 PM every day). Usually viruses cause damage to computer and network systems to the extent that it can be repaired, assuming that the organization deploys good backup and recovery procedures.

During its lifetime a **virus** goes through **four phases**:

1. **Dormant phase**: Here the virus is idle. It gets activated based on certain action or event (example the user typing a certain key, or certain date or time is reached etc). This is an optional phase.
2. **Propagation phase**: In this face a virus copies itself and each copy starts creating more copies of self, thus propagating the virus.
3. **Triggering phase**: A dormant virus moves into this phase when the action or event for which it was waiting is initiated.
4. **Execution phase**: This is the actual work of the virus which could be harmless (display some message on the screen) or destructive (delete a file on the disk).

**Virus** can be **classified** into the following categories:

1. Parasitic virus
2. Memory resident virus
3. Boot sector virus
4. Stealth virus
5. Polymorphic virus
6. Metamorphic virus
7. Macro virus

2. **Worm**: 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). A worm, however, does not modify a program. Instead, it **replicates itself again and again**. This is shown in Fig. 1.11. The replication grows so much that ultimately the computer or the network on which the worm resides, becomes very slow, finally

coming to a halt. Thus, the basic purpose of a worm attack is different from that of a virus. A worm attack attempts to make the computer or the network under attack unusable by eating all its resources.

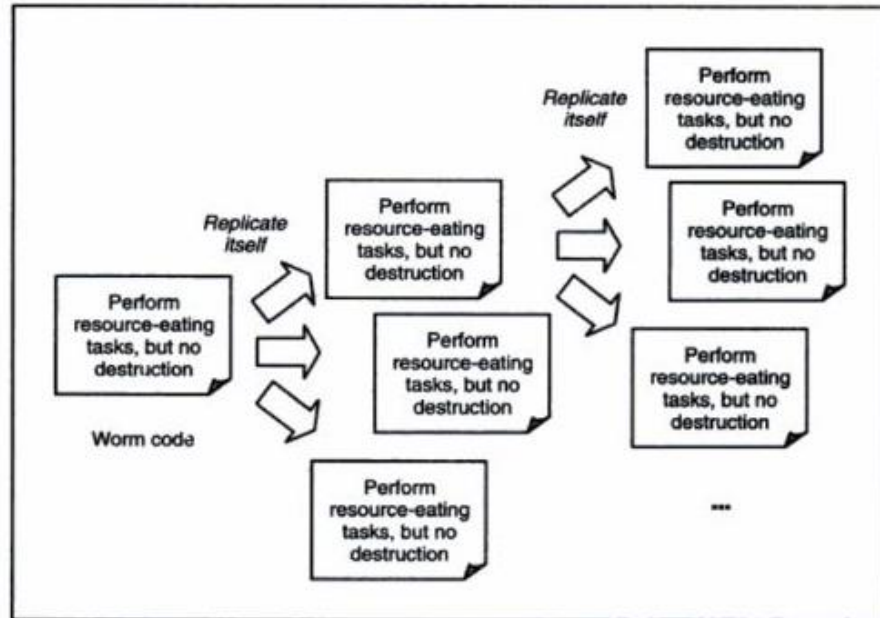


Fig. 1.11 Worm

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 some sort of modifications to the target computer or network, whereas a Trojan horse attempts to reveal confidential information to an attacker. The name (Trojan horse) is due to the Greek soldiers, who hid inside a large hollow horse, which was pulled by Troy citizens, unaware of its contents. Once the Greek soldiers entered the city of Troy, they opened the gates for the rest of Greek soldiers.

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 had entered the id and password. The attacker can then merrily use the user id and password to gain access to the system. This is shown in Fig. 1.12.

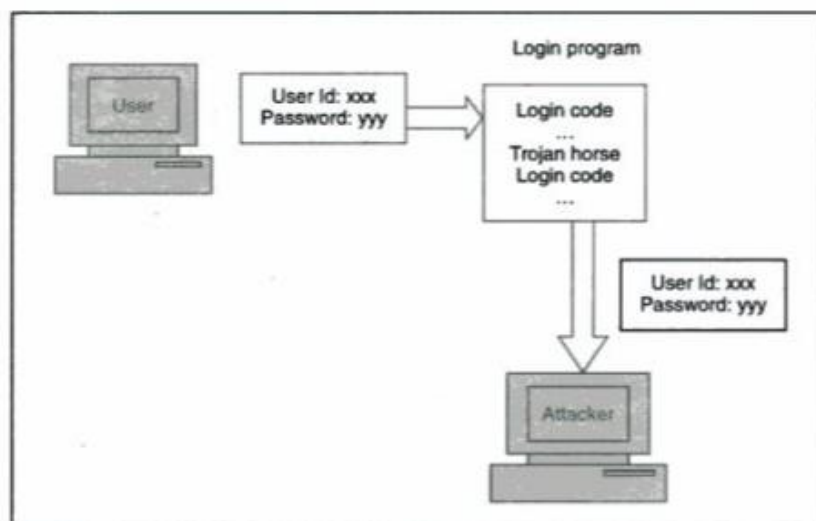


Fig. 1.12 Trojan horse

## Services and mechanism

### **International telecommunication Union-Telecommunication Standardization Sector(ITU-T)**

provides some security services and some mechanism to implement those services. Security services and mechanism are closely related because mechanism or combination of mechanism are used to provide a service. Also a mechanism can be used in one or more services.

### **Security Services**

ITU-T has defined **five services** related to the security goals.

- Data confidentiality
- Data integrity
- Authentication
- Non-repudiation
- Access control

### **Security mechanism**

ITU-T also recommend some security mechanism to provide the security services.

**Encipherment:** It means **hiding or covering** data can provide confidentiality. It can also be used to complement other mechanism to provide other services. Two techniques: **Cryptography and Stenography** are used for enciphering.

**Data integrity:** This mechanism appends to the data as short **checkvalue** that has been created by a specific process from the data itself. The receiver receives the data and the checkvalue. He create a new checkvalue from the received data and the compare the newly created checkvalue with the one received. If the two checkvalues are the same, then integrity of data have been preserved.

**Digital signature:** Digital signature is a means by which the sender can electronically sign the data and the receiver can electronically verify the signature. The sender uses a process that involves showing that she owns a private key related to the public key that she has announced publicly. The receiver uses sender's public key to prove that the message is indeed signed by the sender who claims to have sent the message.

**Authentication exchange:** In authentication exchange two entities exchange some messages to prove their identity to each other before start the original communication.

**Traffic padding:** It means inserting some bogus data into the data traffic thwart adversary's attempt to use the traffic analysis.

**Routing Control:** It means selecting and continuously changing different available routes between the sender and the receiver to prevent the opponent from eavesdropping on a particular route.

**Notarization:** It means selecting a **third trusted party** to control the communication between two entities. This can be done, for example, to prevent repudiation. The receiver can involve a trusted party to store the sender request in order to prevent the sender from later denying that she has made such a request.

**Access Control:** It uses method to prove that a user has access right to the data or resources owned by a system. Example of proofs are password and PIN.

### Relationship between services and Mechanisms

| Service              | Mechanism  |
|----------------------|--|
| Data confidentiality | Encipherment, Routing Control, Traffic padding           |
| Data integrity       | Encipherment, Digital signature, Data integrity          |
| Authentication       | Encipherment, Digital signature, Authentication exchange |
| Non-repudiation      | Digital signature, Data integrity, Notarization          |
| Access control       | Access Control   |

### Introduction to Cryptography

Some security mechanisms listed in the previous section can be implemented using cryptography. Cryptography, a word with **Greek** origins, means "**secret writing**." We use the term to refer **the science and art of transforming messages to make them secure and immune to attacks**. Although in the past cryptography referred only to the encryption and decryption of messages using secret keys, today it is defined as involving **three distinct mechanisms**:

- **Symmetric-key encipherment**
- **Asymmetric-key encipherment**
- **Hashing**

### **Symmetric-Key Encipherment:**

In symmetric-key encipherment (sometimes called secret-key encipherment or secret key cryptography), an entity, say Alice, can send a message to another entity, say Bob, over an insecure channel with the assumption that an adversary, say Eve, cannot understand the contents of the message by simply eavesdropping over the channel. Alice encrypts the message using an encryption algorithm: Bob decrypts the message using a decryption algorithm. **Symmetric-key encipherment uses a single secret key for both encryption and decryption**. Encryption/decryption can be thought of as electronic locking. In symmetric key enciphering, Alice puts the message in a box and locks the box using the shared secret key, Bob unlocks the box with the same key and takes out the message.

### **Asymmetric-Key Encipherment:**

In asymmetric-key encipherment (sometimes called public-key encipherment or public-key cryptography), we have the same situation as the symmetric-key encipherment, with a few exceptions. First, **there are two keys** instead of one: **one public key and one private key**. To send a secured message to Bob, Alice first encrypts the message using Bob's public key. To decrypt the message, Bob uses his own private key.

### **Hashing:**

In hashing, a **fixed-length message digest is created out of a variable-length message**. The digest is normally much smaller than the message. To be useful, both the message and the digest must be sent to Bob. Hashing is used to provide checkvalues, it is used to provide data integrity.

### Steganography

Another technique that was used for secret communication in the past is being revived at the present time: Steganography. The word Steganography, with **origin in Greek**, means **"covered writing."** in contrast with cryptography, which means "secret writing." Cryptography means concealing the contents of a message by enciphering, Steganography means **concealing the message itself by covering it** with something else.

### **Historical Use**

History is full of facts and myths about the use of Steganography. In China, war messages were written **on thin pieces of silk and rolled into a small ball and swallowed by the messenger**. In Rome and Greece, **messages were carved on pieces of wood, that were later dipped into wax to cover the writing**. Invisible inks (such as onion juice or ammonia salts) were also used to write a secret message between the lines of the covering message or on the back of the paper, the secret message was exposed when the paper was heated or treated with another substance.

In recent times other methods have been devised. Some letters in an innocuous message might be overwritten in a pencil lead that is visible only when exposed to light at an angle. Null ciphers were used to hide a secret message inside an innocuous simple message. For example, the first or second letter of each word in the covering message might compose a secret message. Microdots were also used for this purpose. Secret messages were photographed and reduced to a size of a dot (period) and inserted into simple cover messages in place of regular periods at the end of sentences

### **Modern Use**

Today, any form of data, such as text, image, audio, or video, can be digitized, and it is possible to insert secret binary information into the data during digitization process. Such hidden information is not necessarily used for secrecy, it can also be used to protect copyright, prevent tampering, or add extra information

## Text Cover

The **cover of secret data can be text**. There are several ways to insert binary data into an innocuous text. For example, we can use **single space between words to represent the binary digit 0** and **double space to represent binary digit 1**. The following short message hides the 8-bit binary representation of the letter A in ASCII code (01000001).

**This book is mostly about cryptography, not in steganography.**

**0 1 0 0 0 0 1**

In the above message there are two spaces between the "book" and "is" and between the "in" and "steganography". Of course, sophisticated software can insert spaces that differ only slightly to hide the code from immediate recognition

Another, more efficient method, is to **use a dictionary of words organized according to their grammatical usages**. We can have a dictionary containing **2 articles, 8 verbs, 32 nouns, and 4 prepositions**. Then we agree to use cover text that always use sentences with the pattern **article-non-verb-article-noun**. The secret binary data can be divided into **16-bit chunks**. The **first bit** of binary data can be represented by an article (for example, for a and 1 for the). The **next five bits** can be represented by a noun (subject of the sentence), the **next four** bits can be represented by a verb, the **next bit** by the second article, and the last **five bits** by another noun (object). For example, the secret data "HI", which is 01001000 01001001 in ASCII could be a sentence like the following:

A friend called a doctor.

0 10010 0001 0 01001

This is a very trivial example. The actual approach uses more sophisticated design and a variety of patterns.

## Image Cover

Secret data can also be covered under a color image. Digitized images are made of pixels (picture elements), in which normally each pixel uses 24 bits (three bytes). Each byte represents one of the primary colors (red, green, or blue). We can therefore have 2 different shades of each color. In a method called **LSB (least significant bit)**, the **least significant bit of each byte is set to zero**. This may make the image a little bit lighter in some areas, but this is not normally noticed. Now we can hide a binary data in the image by keeping or changing the least significant bit. If our binary digit is 0, we keep the bit: if it is 1, we change the bit to 1. In this way, we can hide a character (eight ASCII bits) in three pixels. For example, suppose the first eight pixels of the original image have the following values:

01010010 10111100 01010110

01110000 11000110 01010000

01001000 01000010

After hiding the letter M(01001101), the pixels become

|                  |                  |                  |
|------------------|------------------|------------------|
| 0101001 <b>0</b> | 1011110 <b>1</b> | 0101011 <b>0</b> |
| 0111000 <b>0</b> | 1100011 <b>1</b> | 0101000 <b>1</b> |
| 0100100 <b>0</b> | 0100001 <b>1</b> |                  |

Of course, more sophisticated approaches are used these days.

#### Other Covers

Other covers are also possible. The secret message, for example, can be covered under audio (sound and music) and video. Both audio and video are compressed today, the secret data can be embedded during or before the compression.

#### Secret Sharing Scheme

In cryptography, secret sharing refers to any method for distributing a secret among a group of participants, each of which allocates a share of the secret. The secret can only be reconstructed when the shares are combined together; individual shares are of no use on their own.

The secret is opened only when specific conditions are fulfilled. Each of  $n$  participants is given a number of share, and any group of  $t$  (threshold) or more shares together can open the secret but no group of less than  $t$  shares can.

A secure secret sharing scheme distributes shares so that anyone with fewer than  $t$  shares has no more information about the secret than someone with 0 shares.

Consider the secret sharing scheme in which the secret phrase "password" is divided into the shares "pa---", "---ss---", "----wo--", and "-----rd,". A person with 0 shares knows only that the password consists of eight letters. He would have to guess the password from  $26^8 = 208$  billion possible combinations. A person with one share, however, would have to guess only the six letters from  $26^6 = 308$  million combinations. This system is not a secure secret sharing scheme, because a player with less than  $t$  shares gains significant information about the content of the secret. In a secure scheme, even a player missing only one share should still face  $26^8 = 208$  billion combinations.

Secret sharing was invented by both Adi Shamir and George Blakley independently in 1979.

#### Examples

- Imagine that the Board of Directors of Coca-Cola would like to protect Coke's secret formula. The president of the company should be able to access the formula when needed, but in an emergency, any 3 of the 12 board members would be able to unlock the secret formula together. This can be accomplished by a secret sharing scheme with  $t = 3$  and  $n = 15$ , where 3 shares are given to the president, and 1 share is given to each board member.

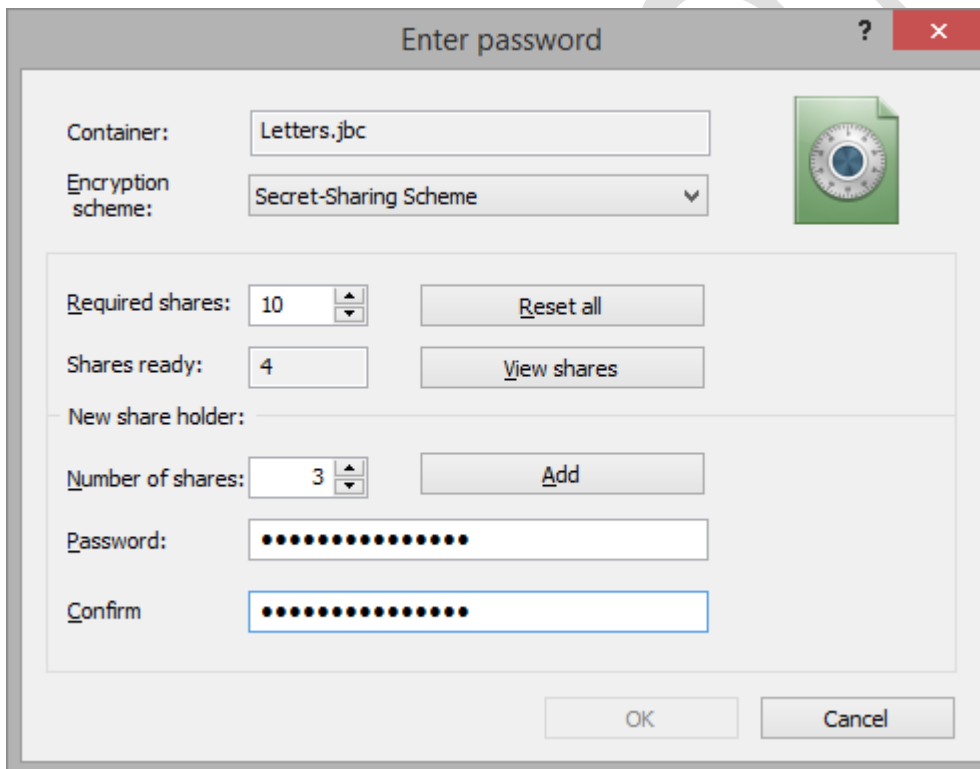


- Let's imagine that there is an organization where all members are equal. Say, an alliance between 5 states. When the organization holds a conference, quorum must be reached. Let's assume that according to the rules, the quorum is 4. If the quorum is reached, then the secret documents can be opened and the conference will start. Such scheme is accomplished with  $t = 4$  and  $n = 5$ , where all members are equal and have 1 share.

Implementation(Example: BestCrypt)

How to use Secret Sharing Scheme([How to Use Secret Sharing Scheme | Jetico](#))

To create a new container with a secret sharing scheme, open New Container dialog and make all usual settings like name, size and location. When the Enter password dialog appears, click Advanced and choose Secret-sharing scheme in Key Block Type the edit box. The following dialog window will appear:



The screenshot shows the 'Enter password' dialog box. The 'Container' field is 'Letters.jbc' and the 'Encryption scheme' is 'Secret-Sharing Scheme'. The 'Required shares' is set to 10, and 'Shares ready' is 4. The 'New share holder' section shows 'Number of shares' as 3. The 'Password' and 'Confirm' fields are both masked with dots. The 'OK' and 'Cancel' buttons are at the bottom.

The first step is making an agreement between all the participants. You should come together and define appropriate ways of getting access to the container. You should design your secret sharing scheme based upon your needs. In terms of the scheme, you will have to define the threshold value - **the number of shares required for opening the container** (the value is called "Required shares" in the dialog) and number of shares for each member.

Then, each participant of the scheme will enter his/her own password and his/her own Number of shares, according to the agreement. After performing these actions and clicking the Add button, the password

will be added to the scheme and Shares ready counter will be increased by the corresponding number of shares.

View shares allows you to see how many passwords have already been entered and how the shares are distributed.

When the counter reaches the threshold value, OK will become available and creation process can be finished. But it is possible to continue the creation process, until all participants enter their passwords.

When all the participants finish entering passwords, click OK to continue the process of creating a new container file.

DRAFT