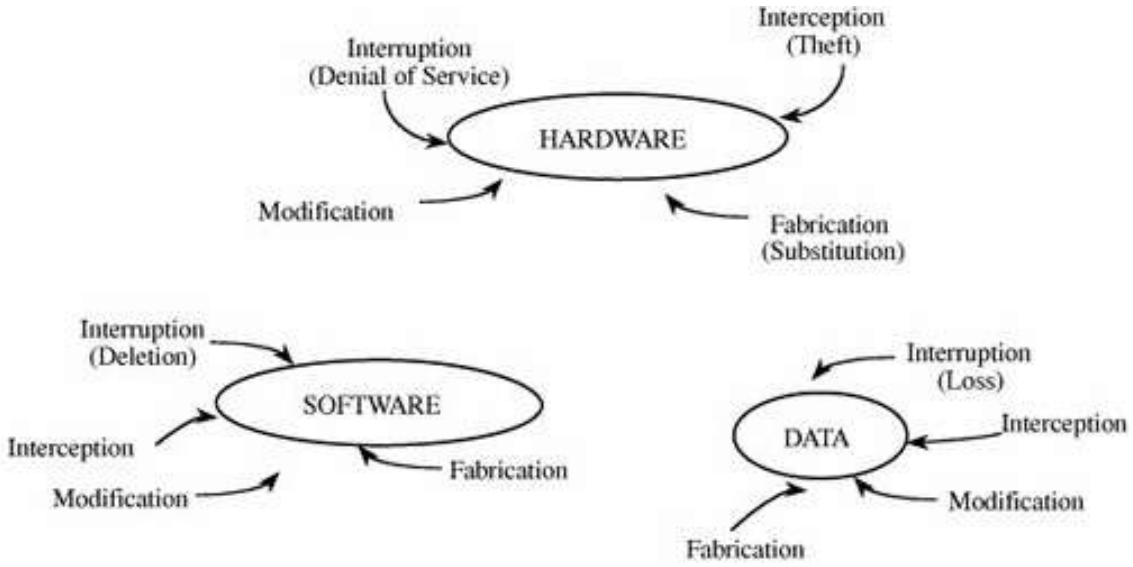
Security - Chapter 1- part 2

[6[Vulnerability: *is a* ***weakness*** *in the security system* *نقطة الضعف اللي في النظام*
(in ***design****,* ***implementation****..etc. any point used to cause* ***loss*** *or* ***harm*** *)*

vulnerabilities prevent three security goals.



1. Hardware Vulnerabilities
Hardware is more visible than software, because it is physical objects.

Accidents that happen and not intended to do serious damage to the hardware is considered an attack:

- **people have spilled soft drinks, ketchup, واحد يكب خمرة أو كاشطب**
- **mice have chewed through cables. فار يقرقض الكابلات**
- **Particles of dust, حبيبات التراب , ash in cigarette, سجاير, fire نار**
- **computers have been kicked, حد يعطي الكنبيوتر شلوط**

A more serious attacks

- Machines have been **shot with guns**, stabbed with knives, and smashed
- **Bombs**, fires, and collisions have destroyed computer rooms.
- Ordinary keys, pens, and screwdrivers have been used to **short-out circuit**
- Devices and whole systems have been carried off by **thieves**.

Laptop are especially vulnerable because they are designed to easy carry.

2. 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. **The worker amended the software so that the throw-away interest (the $.0067) was placed into his own account.**

2.1 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 cannot be 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.

2.2 Software Modification

Software is vulnerable to modifications that either cause it to fail or cause it to perform an unintended task.

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: 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.

Other categories of software modification include

- **Trojan horse:** a program that overtly does one thing while covertly doing another
- **virus:** a specific type of Trojan horse that can be used to spread its "infection" from one computer to another
- **trapdoor:** a program that has a secret entry point
- **information leaks** in a program: code that makes information accessible to unauthorized people or programs

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.

2.3 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.

3. Data Vulnerabilities

Hardware security is usually of less concern. Software security is a larger problem, extending to all programmers and analysts who create or modify programs.

Printed data, however, can be readily interpreted by the general public. Thus, data items have greater public value than hardware and software because more people know how to use or interpret data

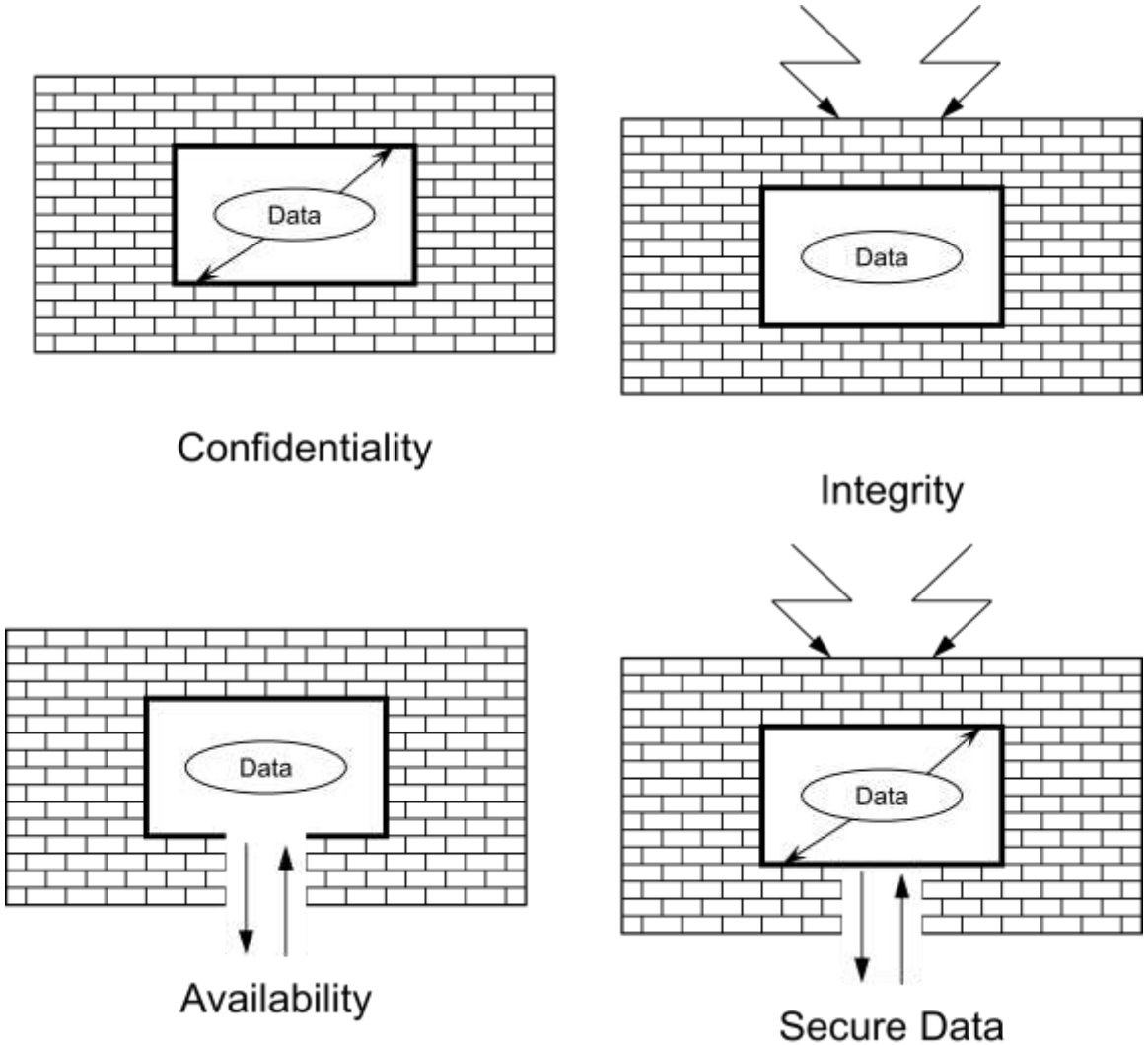
Data incorrectly modified can cost human lives.
data items in context do relate to cost, perhaps measurable by the cost to reconstruct or redevelop damaged or lost data.

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.

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.

Figure 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.

Security of data



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.

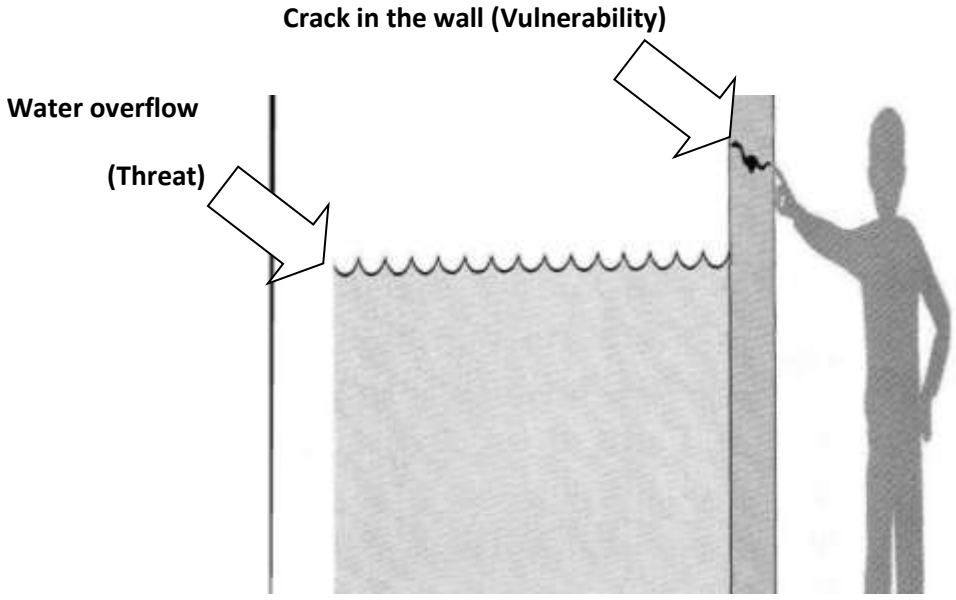
Data Integrity
Stealing, buying, finding, or hearing data requires no computer sophistication, whereas modifying or fabricating new data requires some understanding of the technology by which the data are transmitted or stored, as well as the format in which the data are maintained. Thus, a higher level 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 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.

[7] 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.

Vulnerabilities, Threats, Attacks, 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

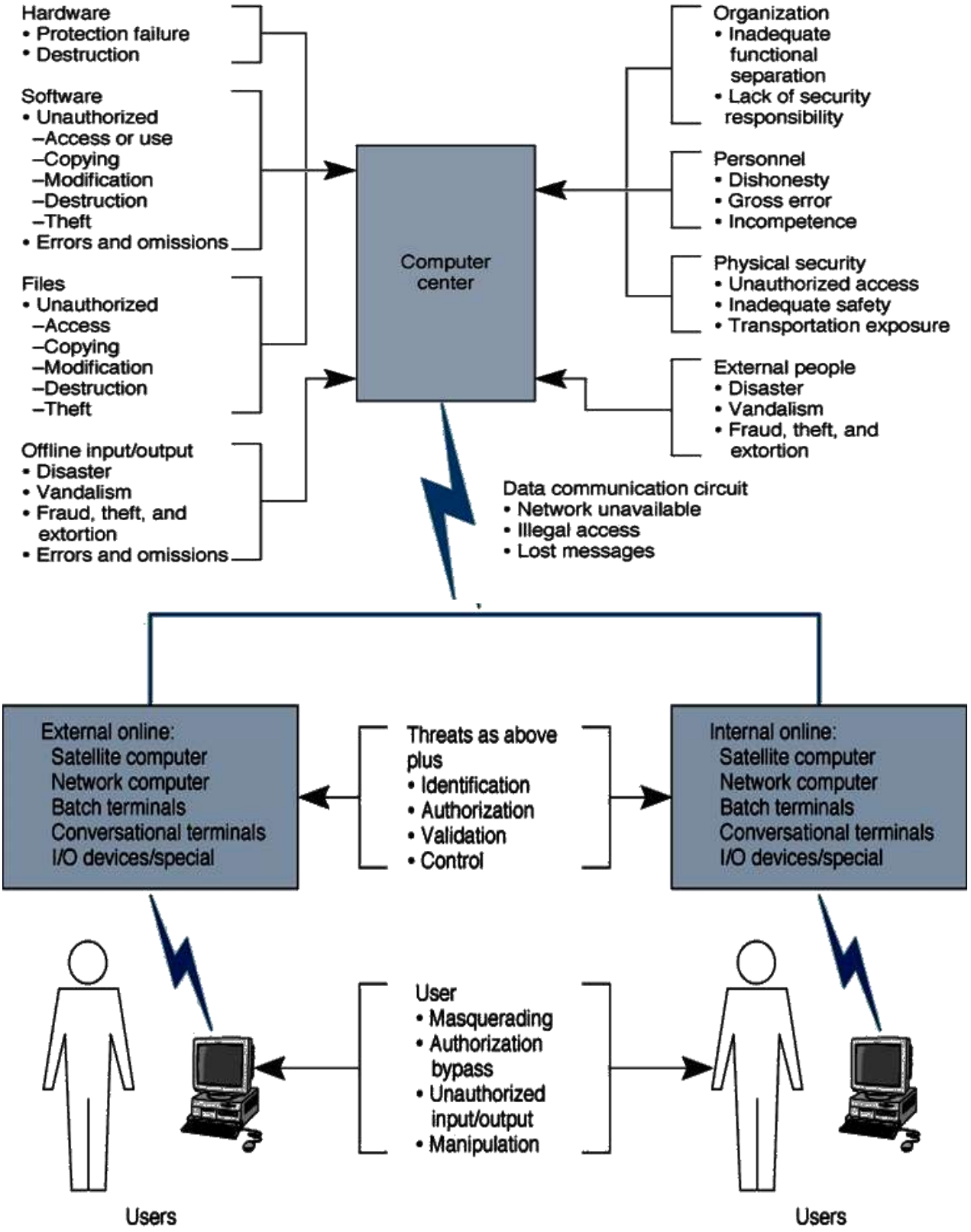
or harm. For example, we can identify data whose format or contents should be protected in some way. We want our security system to make sure that no data are disclosed to unauthorized parties. Neither do we want the data to be modified in illegitimate ways. At the same time, we must ensure that legitimate 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 the wall to collapse. So the threat of harm is the potential for the man to get wet, get hurt, or be drowned. For now, the wall is intact, so the threat to the man is unrealized.



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.



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.

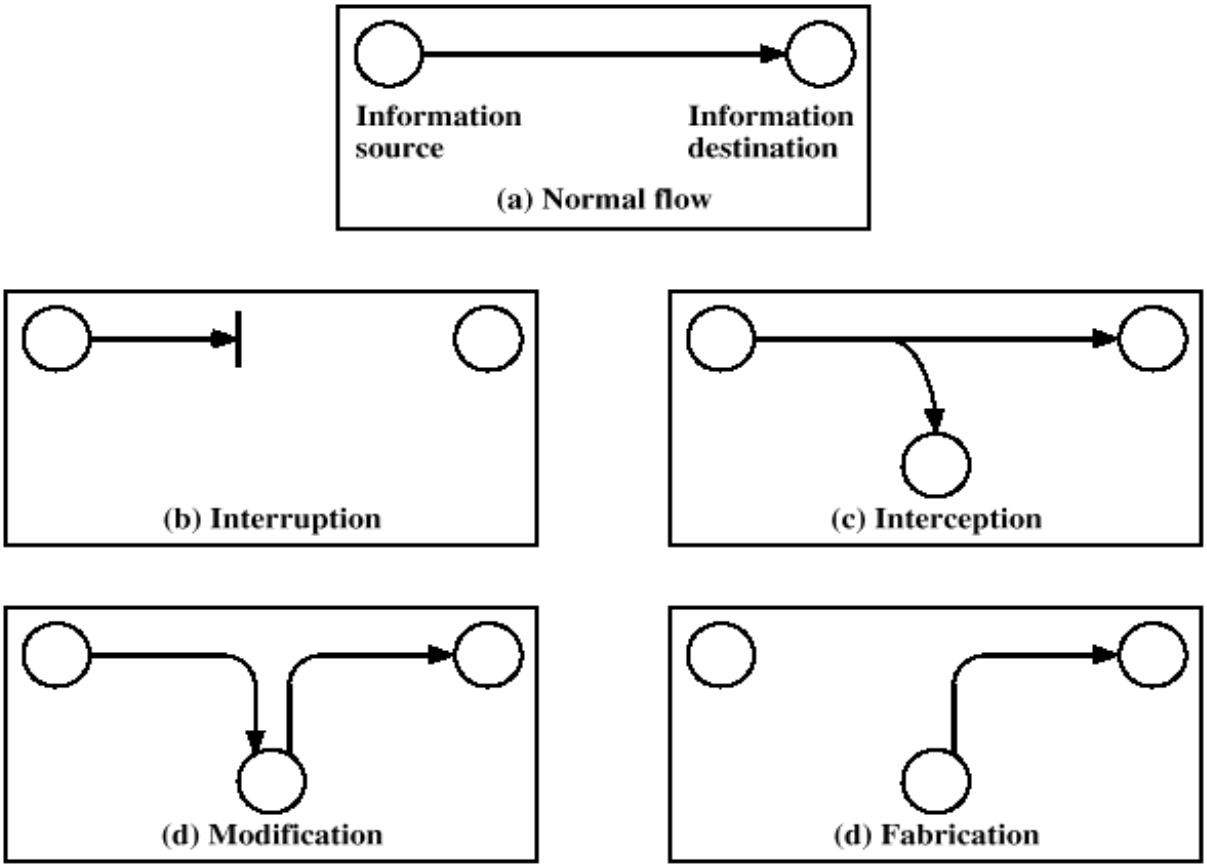
a **control** is an action, device, procedure, or technique that removes or reduces a vulnerability.

the relationship among threats, controls, and vulnerabilities in this way:

A threat is blocked by control of a vulnerability.

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;

System Security Threats.



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.

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.

A malicious attacker must have three things (MOM):

- **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

Deny any of those three things and the attack will not occur. However, it is not easy to cut these off.

What Makes a Network Vulnerable?

1- Anonymity (no personal identification): An attacker can mount an attack from thousands of miles away and never come into direct contact with the system.
2- Many points of attack both targets and origins: A large network offers many points of vulnerability.
3- Sharing. Because networks enable resource and workload sharing, more users have the potential to access networked systems than on single computers.

4-Complexity of system: A network combines two or more possibly dissimilar operating systems. Therefore, a network operating/control system is likely to be more complex than an operating system for a single computing system.
5-Unknown network boundary: A network's expandability implies uncertainty about the network boundary. One host may be a node on two different networks, so resources on one network are accessible to the users of the other network as well. Although wide accessibility is an advantage, this unknown or uncontrolled group of possibly malicious users is a security disadvantage.
6-Unknown path. there may be many paths from one host to another. The message might be routed through different hosts before arriving at destination host. Network users seldom have control over the routing of their messages.

Classes of Security Risks

- *Breaching secret data – Confidential data is often stored and transmitted in encrypted form weak encryption or lack of protection lead to data breaching.*
- *Unauthorized logons results from misuse of stolen and guessed passwords, lack of authentication*
- *Unauthorized denial of service the hacker is interested in shutting down the computer system, degrading its performance, consuming its resources i.e affect the availability and is usually done by injecting malicious software.*
- *Network Spoofing occurs when one host on the network is used to impersonate another host*

Network security goals

- ***Identification:*** user ID for each application.
- ***Authentication:*** verification of the identity of user (password one-way authentication, application to user is two-way authentication)
- ***Authorization:*** is the process of assigning access to each user ID (access rights include read, write, update)
- ***Access Control:*** the process of enforcing access rights for network access
- ***Confidentiality:*** the process used to protect secret information from unauthorized disclosure.
- ***Data integrity:*** allows detection of unauthorized modification of data (even through transmission)
- ***Non repudiation:*** *is the capability to provide proof of the origin of data or proof of the delivery of data*
- ***Denial of Services:*** *when attacker consumes a resource so that no one else can use it.*