Computer security

Second semester –Fourth stage

Assist. Prof. Dr. Nada Hussein M. Ali

Lecture 1

Introduction to Computer Security

1- Important Terminology

Computer Security: The protection afforded to an automated information system in order to attain the applicable objectives of preserving the **integrity, availability** and **confidentiality** of information system resources (includes hardware, software, firmware, information/data, and telecommunications).

This definition introduces three key objectives that are at the heart of computer security:

- ***Confidentiality*:** This term covers two related concepts:

  Data confidentiality*:* Assures that private or confidential information is not made available or disclosed to unauthorized individuals.

  Privacy: Assures that individuals control or influence what information related to them may be collected and stored and by whom and to whom that information may be disclosed.

- ***Integrity*:** This term covers two related concepts:

  Data integrity: Assures that information and programs are changed only in a specified and authorized manner.

  System integrity: Assures that a system performs its intended function in an unimpaired manner, free from deliberate or inadvertent unauthorized manipulation of the system.

- ***Availability*:** Assures that systems work promptly and service is not denied to authorized users.

These three concepts form what is often referred to as the **CIA triad** (Figure1). The three concepts embody the fundamental security objectives for both data and for information and computing services.

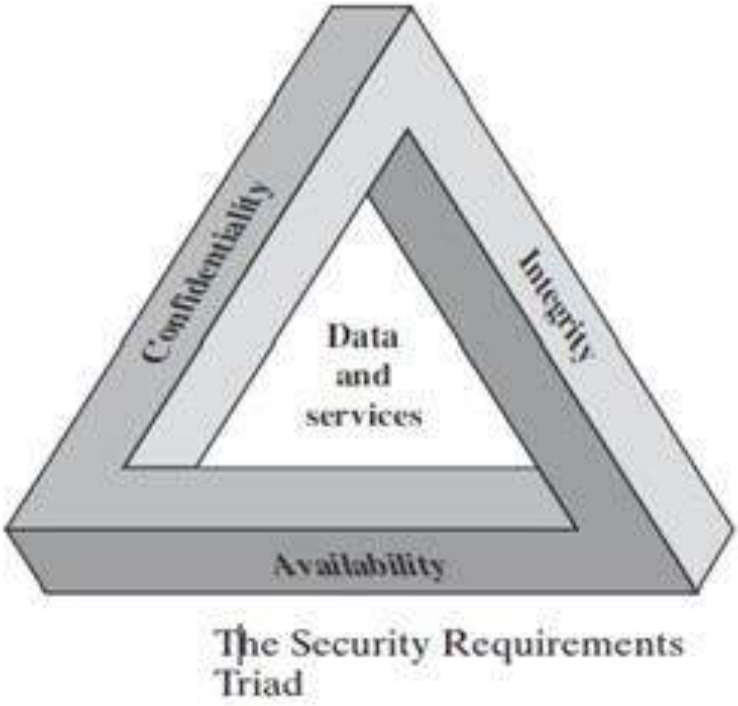


Figure (1): CIA concepts

Another way of looking at security in computer systems is that we attempt to protect the services and data it offers against **security threats**. There are four types of security threats to consider:

1. **Interception**:

Refers to the situation that an unauthorized party has gained access to a service or data. A typical example of interception is where communication between two parties has been overheard by someone else. Interception also happens when data are illegally copied

2. Interruption:

Refers to the situation in which services or data become unavailable, unusable, destroyed, and so on. In this sense, denial of service attacks by which someone maliciously attempts to make a service inaccessible to other parties

3. **Modification:**
Involve unauthorized changing of data or tampering with a service so that it no longer adheres to its original specifications. Examples of modifications include intercepting and subsequently changing transmitted data, tampering with database entries, and changing a program so that it secretly logs the activities of its user.

4. **Fabrication:**
Refers to the situation in which additional data or activity are generated that would normally not exist. For example, an intruder may attempt to add an entry into a password file or database. Likewise, it is sometimes possible to break into a system by replaying previously sent messages

2- Security policy

A set of rules and practices that specify or regulate how a system or organization provides security services to protect sensitive and critical system resources. Important security mechanisms are:

1. Encryption
Encryption transforms data into something an attacker cannot understand. In other words, encryption provides a means to implement confidentiality. In addition, encryption allows us to check whether data have been modified. It thus also provides support for integrity checks.

2. Authentication
Authentication is used to verify the claimed identity of a user, client, server, and so on.

3. Authorization
After a client has been authenticated, it is necessary to check whether that client is authorized to perform the action requested.

4. Auditing
Auditing tools are used to trace which clients accessed what, and which way. Although auditing does not really provide any protection against security threats, audit logs can be extremely useful for the analysis of a security breach, and subsequently taking measures against intruders.

3- Security attack:

A useful means of classifying security attacks:

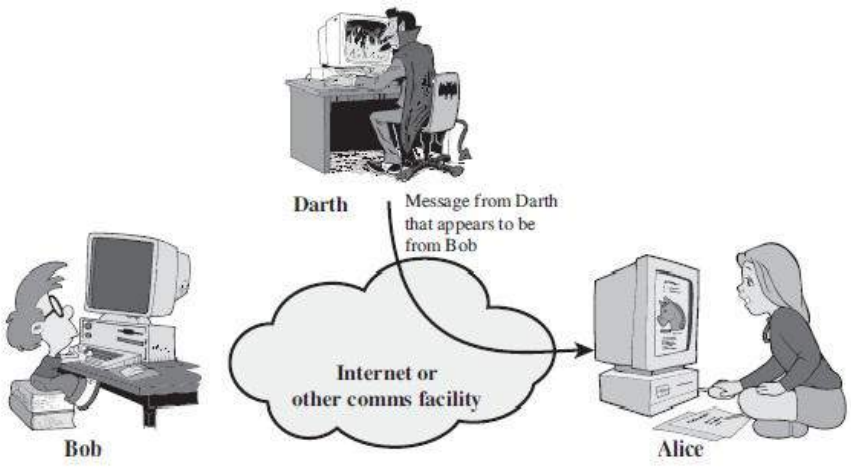
1. ***passive attacks***

Passive attacks are in the nature of eavesdropping on, or monitoring of, transmissions. The goal of the opponent is to obtain information that is being transmitted. Passive attacks are very difficult to detect, because they do not involve any alteration of the data. Typically, the message traffic is sent and received in an apparently normal fashion, and neither the sender nor receiver is aware that a third party has read the messages or observed the traffic pattern. However, it is feasible to prevent the success of these attacks, usually by means of encryption. Thus, the emphasis in dealing with passive attacks is on prevention rather than detection.

2- ***Active attacks***

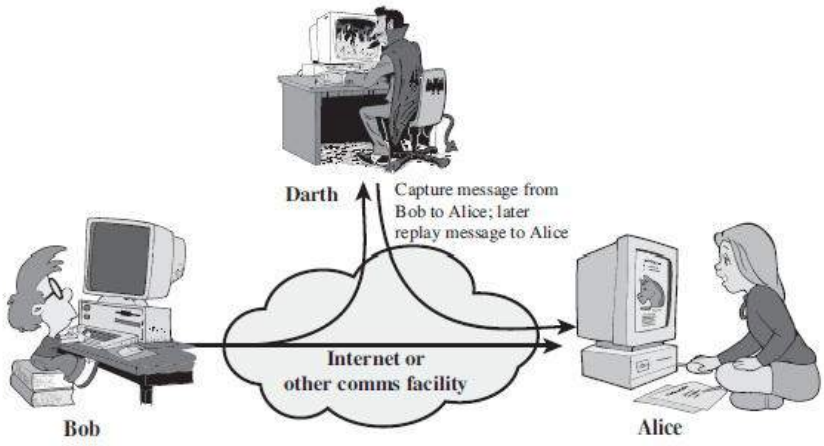
Active attacks involve some modification of the data stream or the creation of a false stream and can be subdivided into four categories: masquerade, replay, modification of messages, and denial of service.

- A **masquerade** takes place when one entity pretends to be a different entity (Figure a). A masquerade attack usually includes one of the other forms of active attack.



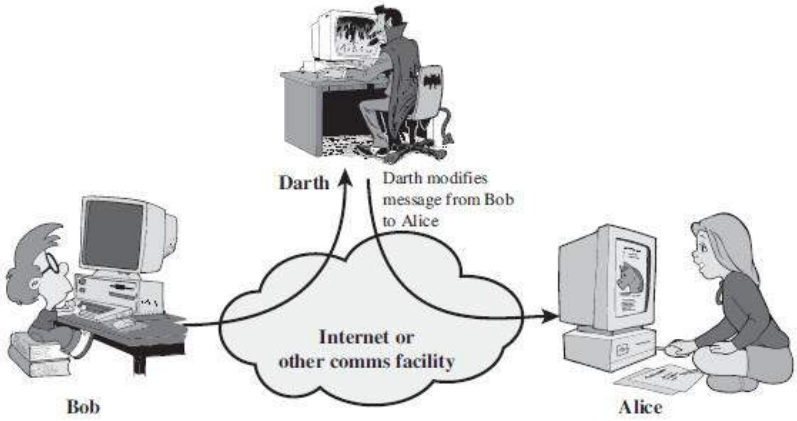
(a) Masquerade

- **Replay** involves the passive capture of a data unit and its subsequent retransmission to produce an unauthorized effect (Figure b).



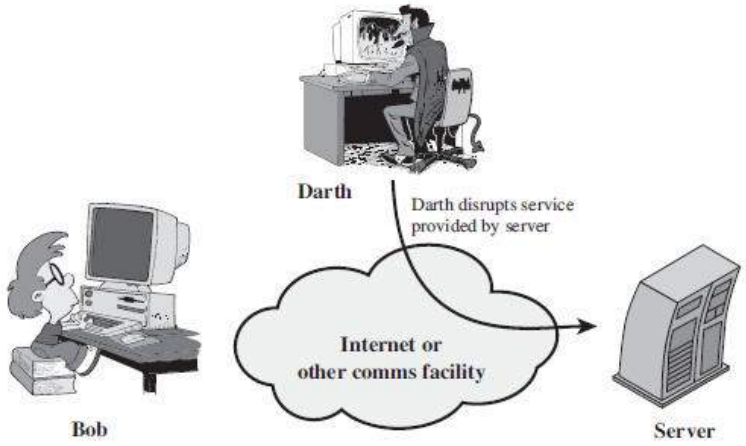
(b) Replay

❖ **Modification of messages** simply means that some portion of a legitimate message is altered, or that messages are delayed or reordered, to produce an unauthorized effect (Figure c).



(c) Modification of messages

❖ The **denial of service** prevents or inhibits the normal use or management of communications facilities (Figure d). This attack may have a specific target; for example, an entity may suppress all messages directed to a particular destination.



(d) Denial of service

It is quite difficult to prevent active attacks absolutely because of the wide variety of potential physical, software, and network vulnerabilities. Instead, the goal is to detect active attacks and to recover from any disruption or delays caused by them. If the detection has a deterrent effect, it may also contribute to prevention.

4- Security service

A security service as a service that is provided by a protocol layer of communicating open systems and that ensures adequate security of the systems or of data transfers. These services divided into five categories

- **Authentication**

The authentication service is concerned with assuring that a communication is authentic. In the case of a single message, such as a warning or alarm signal, the function of the authentication service is to assure the recipient that the message is from the source that it claims to be from.

In the case of an ongoing interaction, such as the connection of a terminal to a host, two aspects are involved:

- **First**, at the time of connection initiation, the service assures that the two entities are authentic, that is, that each is the entity that it claims to be.
- **Second**, the service must assure that the connection is not interfered with in such a way that a third party can masquerade as one of the two legitimate parties for the purposes of unauthorized transmission or reception.

❖ Access Control

In the context of network security, access control is the ability to limit and control the access to host systems and applications via communications links. To achieve this, each entity trying to gain access must first be identified, or authenticated, so that access rights can be tailored to the individual.

❖ Data Confidentiality

Confidentiality is the protection of transmitted data from passive attacks. With respect to the content of a data transmission, several levels of protection can be identified. The broadest service protects all user data transmitted between two users over a period of time.

❖ Data Integrity

A connection-oriented integrity service, one that deals with a stream of messages, assures that messages are received as sent with no duplication, insertion, modification, reordering, or replays. The destruction of data is also covered under this service. Thus, the connection-oriented integrity service addresses both message stream modification and denial of service. On the other hand, a connectionless integrity service, one that deals with individual messages without regard to any larger context, generally provides protection against message modification only.

❖ Nonrepudiation

It prevents either sender or receiver from denying a transmitted message. Thus, when a message is sent, the receiver can prove that the alleged sender in fact sent the message. Similarly, when a message is received, the sender can prove that the alleged receiver in fact received the message.

5- Security mechanisms

Security mechanism (security control) is a component, technique, or method to achieve or enforce security policy.
Examples:

1. password
2. locked cabinet for server
3. encryption

Security mechanisms are typically one of the following forms:

1. Prevention: keep security policy from being violated. Examples (password, encryption, memory bounds check)
2. Detection: detect when policy is violated. Examples (virus scanner)

Computer Security

Second semester –Fourth stage

Assist. Prof. Dr. Nada Hussein M. Ali

Lecture 2

Access Control

Access is the ability to do something with a computer resource (e.g., use, change, or view). *Access controls* ensures that all direct access to objects are authorized. By regulating the reading, changing, and deletion of data and programs, access control protect against accidental and malicious threats to secrecy, authenticity, and system availability.

Access control Examples:

- *Social Networks*: In most social networks, such as Facebook, some of your personal information can only be accessed by yourself, some can be accessed by your friends, and some can be accessed by everybody. The part of system that implements such kind of control is doing access control.

- *Operating Systems*: In an operating system, one user cannot arbitrarily access another user's files; a normal user cannot kill another user's processes. These are done by operating system access control.

- *Firewalls*: Firewalls inspect every incoming (sometimes outgoing) packet, if a Packet not matches with certain conditions; it will be dropped by the firewalls, preventing it from accessing the protected networks. This is also access control.

Subjects and Objects

A computer system can be seen to include two types of entities. Users and processes acting on the users' behalf are called ***subjects*** of the computer system. Resources that may need to be protected are called ***objects*** of the computer system.

The objects may be files, directories, peripheral devices or even processes. The subjects perform actions on the objects. These objects or resources must be protected from disclosure, unauthorized change, or exclusive reservation such that legitimate users cannot access the objects when needed (i.e. denial of service).

Policies, Models, and Mechanisms

When planning an access control system, three abstractions of controls should be considered:

- **Access control policies:** are high-level requirements that specify how access is managed and who, under what circumstances, may access what information.
- **Access Control Models** provide a formal representation of the access control security policy and its working.
- **Access Control Mechanisms** define the low level (software and hardware) functions that implement the controls imposed by the policy and formally stated in the model.

Access control policy

There are several well-known access control policies, which can be categorized as discretionary or non-discretionary.

1. Discretionary Access Control (DAC)

DAC leaves a certain amount of access control to the discretion of the object's owner or anyone else who is authorized to control the object's access. For example, it is generally used to limit a user's access to a file; it is the owner of the file who controls other users' accesses to the file. Only those users specified by the owner may have some combination of read, write, execute, and other permissions to the file.

- ❖ DAC policy tends to be ***very flexible*** and is widely used in the commercial and government sectors. However, DAC is known to be ***inherently weak*** for two reasons: First, granting read access is transitive; for example, when Ann grants Bob read access to a file, nothing stops Bob from copying the contents of Ann's file to an object that Bob controls. Bob may now grant any other user access to the copy of Ann's file without Ann's knowledge.

- ❖ Second, DAC policy is ***vulnerable*** to Trojan horse attacks. Because programs inherit the identity of the invoking user, Bob may, for example, write a program for Ann that, on the surface, performs some useful function, while at the same time destroys the contents of Ann's files. When investigating the problem, the audit files would indicate that Ann destroyed her own files.

Thus, formally, the ***drawbacks*** of DAC are as follows:

- ❖ Information can be copied from one object to another; therefore, there is no real assurance on the flow of information in a system.
- ❖ No restrictions apply to the usage of information when the user has received it.
- ❖ The privileges for accessing objects are decided by the owner of the object, rather than through a system-wide policy that reflects the organization's security requirements.

2. Mandatory access control (MAC)

Mandatory access control (MAC) policy means that access control policy decisions are made by a central authority, not by the individual owner of an object, and the owner cannot change access rights. The need for a MAC mechanism arises when the security policy of a system dictates that:

1. Protection decisions must not be decided by the object owner.
2. The system must enforce the protection decisions (i.e., the system enforces the security policy over the wishes or intentions of the object owner).

Security classifications

In multilevel mandatory policies, an access class is assigned to each object and subject. Most commonly an access class is defined as consisting of two components: a ***security level*** and a ***set of categories***. The security level is an element of a hierarchically ordered set, such as Top Secret (TS), Secret (S), Confidential (C), and Unclassified (U), where TS > S > C > U. The set of categories is a subset of an unordered set, whose elements reflect functional, or competence, areas (e.g., NATO, Nuclear, and Army, for military systems).

The dominance relationship ≥ is defined as follows: an access class c1 dominates (≥) an access class c2 iff the security level of c1 is greater than or equal to that of c2 and the categories of c1 include those of c2.

Secrecy-based mandatory policies

The security level of the access class associated with a user, also called *clearance*, reflects the user's trustworthiness not to disclose sensitive information to users not cleared to see it.

Categories define the area of competence of users and data and are used to provide finer grained security classifications of subjects and objects than classifications provided by security levels alone.

Requests by a subject to access an object are controlled with respect to the access class of the subject and the object and granted only if some relationship, depending on the requested access, is satisfied. In particular, two principles, first formulated by **Bell and LaPadula** as shown in Figure 1, must be satisfied to protect information confidentiality:

- **No-read-up:** A subject is allowed a read access to an object only if the access class of the subject dominates the access class of the object.
- **No-write-down:** A subject is allowed a write access to an object only if the access class of the subject is dominated by the access class of the object.

Satisfaction of these two principles prevents information to flow from high level subjects/objects to subjects/objects at lower (or incomparable) levels, thereby ensuring the satisfaction of the protection requirements (i.e., no process will be able to make sensitive information available to users not cleared for it).

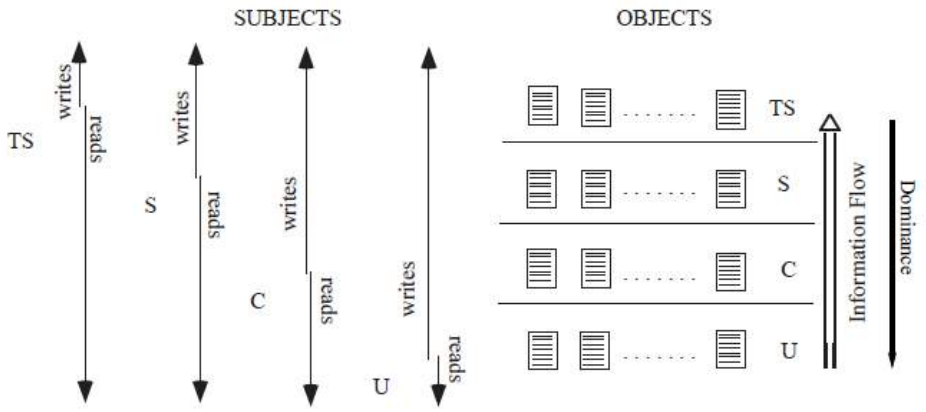


Figure (1): Bell and LaPadula structure

Access Control Mechanisms

In general, access control mechanisms require that security attributes be kept for users and resources. User security attributes can consist of categories

- user identifiers
- groups

Resource attributes can take on a wide variety of forms:

- They can consist of sensitivity labels
- types
- access control lists. In determining a user's ability to perform operations on a resource, access control mechanisms compare the user's security attributes to those of the resource.

Access control checks can be determined (evaluated) based on a previously determined set of rules. For example, the security label of the user must be greater than or equal to the security label of the resource for the user to read the contents of the resource.

1. Access control Matrix

The Access Control Matrix is a table in which each row represents a subject, each column represents an object, and each entry is the set of access rights for that subject to that object. A representation of an access control matrix is shown in the following figure. The figure demonstrates the User A may read and write File1, read and execute File3, and read File4. User B may read, write and execute File2 and read and execute File3. User T may read File1. User T may write File1, execute File2, read and write File4.

	File 1	File 2	File 3	File 4
User A	rw	-	rx	r
User B	-	rwx	rx	-
User S	r	-	-	-
User T	rw	x	-	rw

Figure(2) Access control matrix

In practice, the access control matrix is ***rarely*** done because it is ***large and sparse***. Most subjects have no access at all to most objects, so storing a very large, mostly empty matrix is a waste of disk space.

2. Access Control List (ACL)

There is one list for each object, and the list shows all subjects who should have access to the object and what their access is. In other word, decomposition of the matrix by columns.

Figure (3) illustrates the ACL. Here we have three users A, B, C and three files F1, F2, and F3. Each File has an ACL associated with it. File 1 has two entries in its ACL. The first entry says that User A may read and write the file. The second entry says that User B may read the file. All other access by these users and all other access by other users are forbidden.

File F2 has three entries in its ACL: Users A, B, and C can all read the file, and in addition B can also writ it. No other accesses are allowed. File F3 is apparently an executable program, since B and C can both read and execute it. B can also write it.

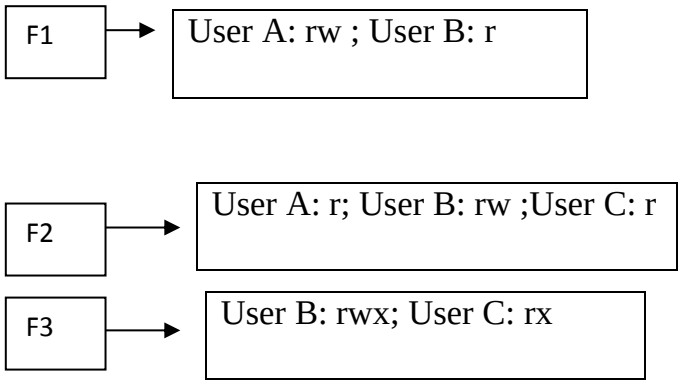


Figure (3) Access Control List

Many Systems support the concept of *group* of users. Groups have names and can be included in ACLs. In some system each user has ID (UID: ungroup ID) and group ID (GID: Group ID). In such systems, an ACL entry contain entries of the form:

UID1,GID1:rights1; UID2,GID2:rights2; ….

Under these conditions, when a request is made to access an object, a check is made using caller's UID and GID. If they are present in the ACL, the rights listed are available. If the (UID,GID) combination is not in the list, the access is not permitted.

Using groups this way effectively introduces the concept of *role.* Consider an installation in which *Tana* is system administrator, and thus in group *sysadm*. However, suppose that the company also has some clubs of employees and *Tana* is a member of the pigeon fanciers club. Club members belong to the group *pigfan* and have access to the company's computer for managing their pigeon database. A portion of the ACL might be shown in figure (4).

File	ACL
password	Tana, sysadm: rw
Pigeon_data	Bill; pigfan: rw ; tana,pigfan:rw

Figure (4) Two access control lists

If Tana tries to access one of these files, the result depends on which group she is currently logged in as. When she logs in, the system may asked her to choose which of here groups she is currently using, or there might even be different login names and/or password to keep them separate. The point of this scheme is to prevent Tana from accessing the password file when she currently has her pigeon fancier's hat on. She can only do that when logged in as the system administrator.

In some cases, a user may have access to certain files independent of which group she is currently logged in as. That case can be handled by introducing **wildcards**, which mean everyone. For example, the entry

Tana, *: rw

For the password file would give Tana access no matter which group she was currently in.

Yet another possibility is that if a user belongs to any of the groups that have certain access rights, the access is permitted. In this case, a user belonging to multiple groups does not have to specify which group to use at login time. All of them count all of the time. A disadvantage of this approach is that it provides less encapsulation: Tana can edit the password file during a pigeon club meeting.

In some times occurs that a user or a group has certain permissions with respect to a file that the file owner latter wishes to revoke. With access control list, it is relatively straightforward to revoke a previously granted access. All that has to be done is edit the ACL to make the change.

3. Capability List

The other way of slicing up the access control matrix is by rows. When this method is used, associated with each process is a list of objects that may be accessed, along with an indication of which operations are permitted on each. This list is called a **capability list** or **C-list with** each entry being a **Capability.** A set of three users and their capability list is depicted in figure below the user A can read F1 and F2, for example.

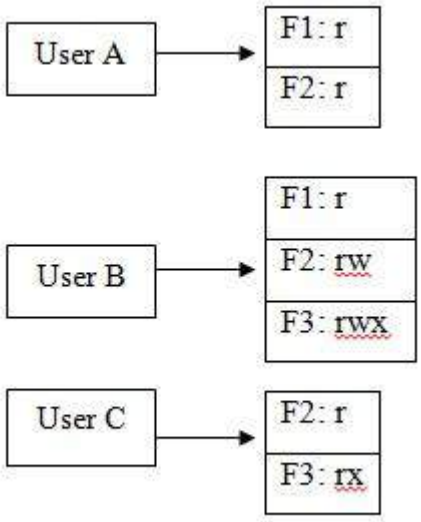


Figure (5) Capability list

Breifly summerized, ACLs and capability have somewhat complementry properties. Capabilities are very efficient because if a user says "open the file pointed by capbility 3" no machecking is needed. With ACLs, search of ACl may be needed.ACL allow selective revocation of rights, which capability do not. Finally, if an object is removed and capability are not or the capabilities are removed and an object is not, problems arise. ACLs do not suffer frim this problem.

3. Protection Bits

Protection Bits are attached to each file but instead of providing a complete list of all users and their allowed access modes, they specify permissions for specific classes of users. Figure (6) shows how Protection Bits can be used on a system to implement an **Owner**/ **Group**/ **World** scheme for our **Read**/ **Write**/ **Execute**/ **Delete**/ **List** access mode. Instead of a large list being attached to each object, the Protection Bits use only fifteen bits of memory. The fifteen bits are broken into three sets of five. The first set of five bits represents the permissions for the Owner of this file. The second set is used to delineate the access permission given to users in the same group as the owner of the file. The final set of bits describes the permissions for all other users on the system. The first bit in each set of five is used to grant Read access mode permission. If the bit is set to one, Read access is granted. If the bit is set to zero Read access is denied. The second bit of each group is used for Write access, the third for execute, and so forth.

Owner					Group					World				
R	W	E	D	L	R	W	E	D	L	R	W	E	D	L

Figure (6) Protection Bits for owner/ group/ world scheme

The most common example of the use of Protection Bits is the UNIX operating system. UNIX systems employ an Owner/ Group/ World for their file access with three access modes specified: Read, Write, and Execute. Associated with each file on a UNIX system is the set of nine required access bits to implement this scheme. Each file also has an owner able to be part of many groups and freely change between these groups during a session by executing a command to change the current group. This enables the user to limit the access granted to any specific file to a designated number of individuals in a particular group. The result of this is to allow the user to actually limit access to a particular file to only a very few people by defining a special group for these select few to part of. While extremely flexible, this method is somewhat cumbersome as users have to be aware of which group they are currently executing within.

Computer Security

Second semester –Fourth stage

Assist. Prof. Dr. Nada Hussein M. Ali

Lecture 3

Identification and Authentication

For most systems, identification and authentication (I&A) are the first line of defense. I&A is a technical measure that prevents unauthorized people (or unauthorized processes) from entering a computer system.

Identification is the means by which a user provides a claimed identity to the system.

Authentication is the means of establishing the validity of this claim.

There are three means of authenticating a user's identity which can be used alone or in combination:

- Something the individual knows (a secret e.g., a password, Personal Identification Number (PIN), or cryptographic key);
- Something the individual possesses (a token e.g., an ATM card or a smart card);
- Something the individual is (a biometric e.g., such characteristics as a voice pattern, handwriting dynamics, or a fingerprint).

Problems associated with each of the above means are:

1. If people wanted to pretend to be someone else on a computer system, they can guess or learn that individual's password; they can also steal or fabricate tokens.
2. For legitimate users and system administrators: users forget passwords and may lose tokens, and administrative overhead of keeping track of I&A data and tokens can be substantial.
3. Biometric systems have significant technical, user acceptance, and cost problems as well.

1- I&A Based on Something the User Knows

The most common form of I&A is a user ID coupled with a password. This technique is based solely on something the user knows.

1-1 Passwords

In general, password systems work by requiring the user to enter a user ID and password (or passphrase or personal identification number). The system compares the password to a previously stored password for that user ID. If there is a match, the user is authenticated and granted access.

Benefits of Passwords. Passwords have been successfully providing security for computer systems for a long time. They integrate into many operating systems, and users and system administrators are familiar with them. When properly managed in a controlled environment, they can provide effective security

Problems With Passwords. The security of a password system is dependent upon keeping passwords secret. Unfortunately, there are many ways that the secret may be divulged.

1. *Guessing or finding passwords*. If users select their own passwords, they tend to make them easy to remember. That often makes them easy to guess. The names of people's children, pets, or favorite sports teams are common examples. On the other hand, assigned passwords may be difficult to remember, so users are more likely to write them down.
   Another method of learning passwords is to observe someone entering a password or PIN. The observation can be done by someone in the same room or by someone some distance away using binoculars. This is often referred to as **shoulder surfing**.

2. *Giving passwords away*. Users may share their passwords. They may give their password to a co-worker in order to share files. In addition, people can be tricked into divulging their passwords. This process is referred to as **social engineering**.

2. *Electronic monitoring*. When passwords are transmitted to a computer system, they can be electronically monitored. This can happen on the network used to transmit the password or on the computer system itself. Simple encryption of a password that will be used again does not solve this problem because encrypting the same password will create the same ciphertext; the ciphertext becomes the password.

4. *Accessing the password file*. If the password file is not protected by strong access controls, the file can be downloaded. Password files are often protected with one-way encryption so that plain-text passwords are not available to system administrators or hackers (if they successfully bypass access controls). Even if the file is encrypted, brute force can be used to learn passwords if the file is downloaded (e.g., by encrypting English words and comparing them to the file).

2- I&A Based on Something the User Possesses

Although some techniques are based solely on something the user possesses, most of the techniques are combined with something the user knows. This combination can provide significantly stronger security than either something the user knows or possesses alone. **Objects that a user possesses for the purpose of I&A are called tokens.**
This section divides tokens into two categories: memory tokens and smart tokens.

2-1 Memory Tokens

Memory tokens store, but do not process, information. Special reader/writer devices, control the writing and reading of data to and from the tokens. The most common type of memory token is a magnetic striped card, in which a thin stripe of magnetic material is affixed to the surface of a card (e.g., as on the back of credit cards). A common application of memory tokens for authentication to computer systems is the automatic teller machine (ATM) card. This uses a combination of something the user possesses (the card) with something the user knows (the PIN).

Benefits of Memory Token Systems. Memory tokens when used with PINs provide significantly more security than passwords. In addition, memory cards are inexpensive to produce. For a hacker or other would-be masquerader to pretend to be someone else, the hacker must have both a valid token and the corresponding PIN.

Problems With Memory Token Systems.

1. *Requires special reader*. The need for a special reader increases the cost of using memory tokens. The readers used for memory tokens must include both the physical unit that reads the card and a processor that determines whether the card and/or the PIN entered with the card is valid.

3. *Token loss.* A lost token may prevent the user from being able to log in until a replacement is provided. This can increase administrative overhead costs. The lost token could be found by someone who wants to break into the system, or could be stolen or forged

3. *User Dissatisfaction*. In general, users want computers to be easy to use. Many users find it inconvenient to carry and present a token. However, their dissatisfaction may be reduced if they see the need for increased security.

2-2 Smart Tokens

A smart token expands the functionality of a memory token by incorporating one or more integrated circuits into the token itself. When used for authentication, a smart token is another example of authentication based on something a user possesses (i.e., the token itself). A smart token typically requires a user also to provide something the user knows (i.e., a PIN or password) in order to "unlock" the smart token for use. There are many different types of smart tokens.

- **Static** tokens work similarly to memory tokens, except that the users authenticate themselves to the token and then the token authenticates the user to the computer.

- A token that uses a **dynamic password generator** protocol creates a unique value, for example, an eight-digit number, that changes periodically (e.g., every minute). If the token has a manual interface, the user simply reads the current value and then types it into the computer system for authentication. If the token has an electronic interface, the transfer is done automatically. If the correct value is provided, the log-in is permitted, and the user is granted access to the system.

- Tokens that use a **challenge-response** protocol work by having the computer generate a challenge, such as a random string of numbers. The smart token then generates a response based on the challenge. This is sent back to the computer, which authenticates the user based on the response. The challenge-response protocol is based on cryptography. Challenge response tokens can use either electronic or manual interfaces.

Benefits of Smart Tokens

Smart tokens offer great flexibility and can be used to solve many authentication problems. The benefits of smart tokens vary, depending on the type used. In general, they provide greater security than memory cards.

1. *One-time passwords*. Smart tokens that use either dynamic password generation or challenge-response protocols can create one-time passwords. Electronic monitoring is not a problem with one-time passwords because each time the user is authenticated to the computer, a different "password" is used. (A hacker could learn the one-time password through electronic monitoring, but would be of no value.)

4. *Reduced risk of forgery*. Generally, the memory on a smart token is not readable unless the PIN is entered. In addition, the tokens are more complex and, therefore, more difficult to forge.
5. *Multi-application*. Smart tokens with electronic interfaces, such as smart cards, provide away for users to access many computers using many networks with only one log-in.

Problems with Smart Tokens

Like memory tokens, most of the problems associated with smart tokens relate to their cost, the administration of the system, and user dissatisfaction. Smart tokens cost more than memory cards because they are more complex, particularly challenge-response calculators.

1. *Need reader/writers or human* intervention. Smart tokens can use either an electronic or a human interface. An electronic interface requires a reader, which creates additional expense. Human interfaces require more actions from the user. This is especially true for challenge-response tokens with a manual interface, which require the user to type the challenge into the smart token and the response into the computer. This can increase user dissatisfaction.

2. *Substantial Administration*. Smart tokens, like passwords and memory tokens, require strong administration. For tokens that use cryptography, this includes key management.

3- I&A Based on Something the User Is

- Biometric authentication technologies use the unique characteristics (or attributes) of an individual to authenticate that person's identity. These include physiological attributes (such as fingerprints, hand geometry, or retina patterns) or behavioral attributes (such as voice patterns and hand-written signatures).

- Biometric authentication is technically complex and expensive, and user acceptance can be difficult. However, advances continue to be made to make the technology more reliable, less costly, and more user-friendly.

- Biometric systems can provide an increased level of security for computer systems, but the technology is still less mature than that of memory tokens or smart tokens.

Biometric Technologies

There are many biometric technologies to suit different types of applications. To choose the right biometric to be highly fit for the particular situation. Here comes a list of biometrics :

Fingerprints - A fingerprint looks at the patterns found on a fingertip. There are a variety of approaches to fingerprint verification, such as traditional police method, using pattern-matching devices, and things like moiré fringe patterns and ultrasonic. This seems to be a very good choice for in-house systems.

Hand geometry - This involves analyzing and measuring the shape of the hand. It might be suitable where there are more users or where user access the system infrequently. Accuracy can be very high if desired, and flexible performance tuning and configuration can accommodate a wide range of applications. Organizations are using hand geometry readers in various scenarios, including time and attendance recording.

Retina - A retina-based biometric involves analyzing the layer of blood vessels situated at the back of the eye. This technique involves using a low intensity light source through an optical coupler to scan the unique patterns of the retina. Retinal scanning can be quite accurate, but does require the user to look into a receptacle and focus on a given point.

Iris - An iris-based biometric involves analyzing features found in the colored ring of tissue that surrounds the pupil. This uses a fairly conventional camera element and requires no close contact between the user and the reader. Further, it has the potential for higher than average template-matching performance.

Face- Face recognition analyses facial characteristics. It requires a digital camera to develop a facial image of the user for authentication. Because facial scanning needs an extra peripheral things that are not included in basic PCs.

Signature - Signature verification analyses the way user signs his name. Signing features such as speed, velocity, and pressure are as important as the finished signature's static shape. People are used to signatures as a means of transaction-related identity verification.

Voice - Voice authentication is based on voice-to-print authentication, where complex technology transforms voice into text. Voice biometrics requires a microphone, which is available with PCs nowadays. Voice biometrics is to replace the currently used methods, such as PINs, passwords, or account names. But voice will be a complementary technique for finger-scan technology as many people see finger scanning as a higher authentication form.

Computer Security

Second semester –Fourth stage

Assist. Prof. Dr. Nada Hussein M. Ali

Lecture 4

Malicious Software (Malware)

1- Malicious Software (Malware)

Malware is defined as "a program that is inserted into a system with the intent of compromising the confidentiality, integrity, or availability of the victim's data, applications, or operating system or otherwise annoying or disrupting the victim.

There are **three characteristics** associated with malware:

1. *Self-replicating* malware actively attempts to propagate by creating new copies, or *instances,* of itself.
2. The *population growth* of malware describes the overall change in the number of malware instances due to self-replication.
3. *Parasitic* malware requires some other executable code in order to exist. "Executable" in this context should be taken very broadly to include anything that can be executed, such as boot block code on a disk, binary code in applications, and interpreted code. It also includes source code, like application scripting languages, and code that may require compilation before being executed.

2- Viruses

A computer virus is a piece of software that can "infect" other programs, or indeed any type of executable content, by modifying them. The modification includes injecting the original code with a routine to make copies of the virus code, which can then go on to infect other content. A virus that attaches to an executable program can do anything that the program is permitted to do. It executes secretly when the host program is run. Once

the virus code is executing, it can perform any function, such as erasing files and programs, that is allowed by the privileges of the current user

A computer virus has three parts. More generally, many contemporary types of malware also include one or more variants of each of these components:

- **Infection mechanism:** The means by which a virus spreads or propagates, enabling it to replicate. The mechanism is also referred to as the **infection vector.**
- **Trigger:** The event or condition that determines when the payload is activated or delivered.
- **Payload:** What the virus does, besides spreading. The payload may involve damage or may involve benign but noticeable activity**.**

3- Logic Bomb

A *logic bomb* is code which consists of two parts:

1. A *pay load*, which is an action to perform. The payload can be anything, but has the connotation of having a malicious effect.
2. A *trigger*, a boolean condition that is evaluated and controls when the payload is executed. The exact trigger condition is limited only by the imagination, and could be based on local conditions like the date, the user logged in, or the operating system version.

Logic bombs can be inserted into existing code, or could be standalone. A simple parasitic example is shown below, with a payload that crashes the computer using a particular date as a trigger.

```
legitimate code

if date is Friday the 13th:
```

In one case, a disgruntled employee rigged a logic bomb on his employer's file server to trigger on a date after he was fired from his job, causing files to be deleted with no possibility of recovery.

4- Trojan Horse

In computing, a *Trojan horse* is a program which purports to do some benign task, but secretly performs some additional malicious task.

A classic example is a password-grabbing login program which prints authentic-looking "username" and "password" prompts, and waits for a user to type in the information. When this happens, the password grabber stashes the information away for its creator, and then prints out an "invalid password" message before running the *real* login program. The unsuspecting user thinks they made a typing mistake and reenters the information, none the wiser.

5- Back Door

A **backdoor** , also known as a **trapdoor** , is a secret entry point into a program that allows someone to gain access without going through the usual security access procedures. The example back door below circumvents a login authentication process.

```
username = read_username()
password = read_password()
if username is "133t h4ck0r":
return ALLOW-LOGIN
if username and password are valid:
return ALLOW_LOGIN
else
return DENY_LOGIN
```

One special kind of back door is **a *RAT,*** which stands for Remote Administration Tool or Remote Access Trojan, depending on who's asked. These programs allow a computer to be monitored and controlled remotely; users may deliberately install these to access a work computer from home, or to allow help desk staff to diagnose and fix a computer problem from afar. However, if malware surreptitiously installs a RAT on a computer, then it opens up a back door into that machine.

6- Worm

Worm is a program that actively seeks out more machines to infect, and then each infected machine serves as an automate launching pad for attacks on other machines. Worm programs exploit software vulnerabilities in client or server programs to gain access to each new system. They can use network connections to spread from system to system.

A *worm* shares several characteristics with a virus. The most important characteristic is that worms are self-replicating too, but self-replication of a worm is distinct in two ways. ***First***, worms are standalone, and do not rely on other executable code. ***Second***, worms spread from machine to machine across networks. Like viruses, the first worms were fictional.

7- Rabbit

Rabbit is the term used to describe malware that multiplies rapidly. Rabbits may also be called ***bacteria***, for largely the same reason. There are actually ***two*** kinds of rabbit.

The ***first*** is a program which tries to consume all of some system resource, like disk space. A "fork bomb," a program which creates new processes in an infinite loop, is a classic example of this kind of rabbit.

The ***second*** kind of rabbit, which the characteristics above describe, is a special case of a worm. This kind of rabbit is a standalone program which replicates itself across a network from machine to machine, but deletes the original copy of itself after replication. In other words, there is only one copy of a given rabbit on a network; it just hops from one computer to another.

8- Spyware

Spyware is software which collects information from a computer and transmits it to someone else. The exact information spyware gathers may vary, but can include anything, which potentially has value:

1. Usernames and passwords. These might be harvested from files on the machine, or by recording what the user types using a ***keylogger***.

2. Email addresses, which would have value to a spammer.

3. Bank account and credit card numbers.

4. Software license keys, to facilitate software pirating.

Viruses and worms may collect similar information, but are not considered spyware, because spyware doesn't self-replicate. Spyware may arrive on a machine in a variety of ways, such as bundled with other software that the user installs, or exploiting technical flaws in web browsers. The latter method causes the spyware to be installed simply by visiting a web page, and is sometimes called a *drive-by download.*

9- Adware

Adware has similarities to spyware in that both are gathering information about the user and their habits. Adware is more marketing-focused, and may pop up advertisements or redirect a user's web browser to certain web sites in the hopes of making a sale. Adware may also gather and transmit information about users, which can be used for marketing purposes. As with spyware, adware does not self-replicate.

10- Hybrids, Droppers, and Blended Threats

The nature of software makes it easy to create hybrid malware which has characteristics belonging to several different types.

A classic hybrid example was presented by Ken Thompson in his ACM Turing award lecture. He prepared a *special C compiler executable* which, besides compiling C code, had two additional features:

1. When compiling the login source code, his compiler would insert a back door to bypass password authentication.
2. When compiling the compiler's source code, it would produce a special compiler executable with these same two features.

His special compiler was thus a ***Trojan horse***, which replicated like a ***virus***, and created ***back doors***. This also demonstrated the vulnerability of the compiler tool chain: since the original source code for the compiler and login programs wasn't changed, none of this nefarious activity was apparent.

There are other combinations of malware too. For example, a ***dropper*** is malware which leaves behind, or *drops,* other malware.

A worm can propagate itself, depositing a Trojan horse on all computers it compromises; a virus can leave a back door in its wake.

A ***blended threat*** is a virus that exploits a technical vulnerability to propagate itself, in addition to exhibiting "traditional" characteristics.

11-Zombies

Computers that have been compromised can be used by an attacker for a variety of tasks, unbeknownst to the legitimate owner; computers used in this way are called ***zombies***. The most common tasks for zombies are **sending spam** and participating in coordinated, **large-scale denial- of-service attacks**.

Sending spam violates the acceptable use policy of many Internet service providers, not to mention violating laws in some jurisdictions. Sites known to send spam are also blacklisted, marking sites that engage in spam-related activity so that incoming email from them can be summarily rejected. It is therefore ill-advised for spammers to send spam directly, in such a way that it can be traced back to them and their machines. Zombies provide a windfall for spammers, because they are a free, throwaway resource: spam can be relayed through zombies, which obscures the spammer's trail, and a blacklisted zombie machine presents no hardship to the spammer.

As for denials of service, one type of denial-of-service attack involves either flooding a victim's network with traffic, or overwhelming a legitimate service on the victim's network with requests. Launching this kind of attack from a single machine would be pointless, since one machine's onslaught is unlikely to generate enough traffic to take out a large target site, and traffic from one machine can be easily blocked by the intended victim. On the other hand, a large number of zombies all targeting a site at the same time can cause grief.

Computer Security

Second semester –Fourth stage

Assist. Prof. Dr. Nada Hussein M. Ali

Lecture 5

Virus

During virus lifetime, a typical virus goes through the following four phases:

1. **Dormant phase:** The virus is idle. The virus will eventually be activated by some event, such as a date, the presence of another program or file, or the capacity of the disk exceeding some limit. Not all viruses have this stage.
2. **Propagation phase:** The virus places a copy of itself into other programs or into certain system areas on the disk. The copy may not be identical to the propagating version. Each infected program will now contain a clone of the virus, which will itself enter a propagation phase.
3. **Triggering phase:** The virus is activated to perform the function for which it was intended. As with the dormant phase, the triggering phase can be caused by a variety of system events, including a count of the number of times that this copy of the virus has made copies of itself.
4. **Execution phase:** The function is performed. The function may be harmless, such as a message on the screen, or damaging, such as the destruction of programs and data files.

1- Viruses Classification

There has been a continuous arms race between virus writers and writers of anti-virus software since viruses first appeared. The viruses are classified along two orthogonal axes:

- The method the virus uses to conceal itself from detection by users and anti-virus software.
- The type of target the virus tries to infect

2- Virus classification by concealment strategy

A virus classification **by concealment strategy** includes the following categories:

- **Encrypted virus:** A typical approach is as follows. A portion of the virus creates a random encryption key and encrypts the remainder of the virus. The key is stored with the virus. When an infected program is invoked, the virus uses the stored random key to decrypt the virus. When the virus replicates, a different random key is selected

- **Stealth virus:** A form of virus explicitly designed to hide itself from detection by anti-virus software. Thus, the entire virus, not just a payload is hidden. for example compression

- **Polymorphic virus:** A polymorphic virus creates copies during replication that are functionally equivalent but have distinctly different bit patterns, in order to defeat programs that scan for viruses. In this case, the "signature" of the virus will vary with each copy. A virus that mutates with every infection, making detection by the "signature" of the virus impossible.

- **Metamorphic virus:** As with a polymorphic virus, a metamorphic virus mutates with every infection. The difference is that a metamorphic virus rewrites itself completely at each iteration, increasing the difficulty of detection. Metamorphic viruses may change their behavior as well as their appearance.

3- Virus classification by target

A virus **classification by target** includes the following categories:

- **Boot sector infector**
- **File infector**
- **Macro virus**
- **Multipartite virus**

3-1 Boot sector infector

- Infects a master boot record or boot record and spreads when a system is booted from the disk containing the virus. A *boot-sector infector*, or BSI, is a virus that infects by copying itself to the boot block. It may copy the contents of the former boot block elsewhere on the disk first, so that the virus can transfer control to it later to complete the booting process.

- ❖ In general, infecting the boot sector is strategically sound: the virus may be in a known location, but it establishes itself before any anti-virus software starts or operating system security is enabled.
- ❖ From a defensive point of view, most operating systems prevent writing to the disk's boot block without proper authorization, and many BIOS has boot block protection that can be enabled.

3-2 File infector

Infects files that the operating system or shell consider to be executable. There are two main issues for file infectors:

1. Where is the virus placed?
2. How is the virus executed when the infected file is run?

- ❖ **Beginning of File:** In this case, a virus that places itself at the start of the file gets control first when the infected file is run. This is called a ***prepending*** virus. Inserting itself at the start of a file involves some copying, which isn't difficult, but isn't the absolute easiest way to infect a file.
- ❖ **End of File:** In contrast, appending code onto the end of a file is extremely easy. A virus that places itself at the end of a file is called an ***appending*** virus. How does the virus get control? There are two basic possibilities:

  1. The original instruction(s) in the code can be saved, and replaced by a jump to the viral code. Later, the virus will transfer control back to the code it infected. The virus may try to run the original instructions directly in their saved location, or the virus may restore the infected code back to its original state and run it.
  2. Many executable file formats specify the start location in a file header. The virus can change this start location to point to its own code, then jump to the original start location when done.

- ❖ **Overwritten into File:** An *overwriting* virus places itself atop part of the original code rendering an entire file useless. This avoids an obvious change in file size that would occur with a prepending or appending virus, and the virus' code can be placed in a location where it will get control.
- ❖ **Inserted into File:** Another possibility is that a virus can insert itself into the target code, moving the target code out of the way, and even interspersing "التبعثر" small pieces of virus code with target code. This is no easy feat: branch targets in the code have to be changed, data locations must be updated, and linker relocation information needs modification.
- ❖ **Not in File**: A *companion* virus is one, which installs itself in such a way that it is naturally executed before the original code. The virus never modifies the infected code

and gains control by taking advantage of the process by which the operating system or shell searches for executable files. The easiest way to explain companion viruses is by example.

1. MS-DOS searches for an executable named foo by looking for foo.com, foo.exe, and foo.bat, in that order. If the target file is a .EXE file, then the companion virus can be a .COM file with the same name.

2. Companion viruses are possible even in GUI-based environments. A target application's icon can be overlaid with the icon for the companion virus. When a user clicks on what they *think* is the application's icon, the companion virus runs instead

3-3 Macro virus

A macro virus is programmed as a macro embedded in a document. Many applications, such as Microsoft Word and Excel, support macro languages. Once a macro virus gets on to your computer, every document you produce will become infected. This type of virus is relatively new and may slip by your antivirus software if you don't have the most recent version installed on your computer.

3-4 Multipartite virus متعدد الأجزاء

A multipartite virus is a hybrid of a Boot Sector and Program viruses. It infects program files and when the infected program is active it will affect the boot record. So the next time you start up your computer it'll infect your local drive and other programs on your computer.

4- Antivirus

The ideal solution to the threat of viruses is prevention: Do not allow a virus to get into the system in the first place. This goal is, in general, impossible to achieve, although prevention can reduce the number of successful viral attacks. The next best approach is to be able to do the following:

- **Detection:** Once the infection has occurred, determine that it has occurred and locate the virus.
- **Identification:** Once detection has been achieved, identify the specific virus that has infected a program.
- **Removal:** Once the specific virus has been identified, remove all traces of the virus from the infected program and restore it to its original state. Remove the virus from all infected systems so that the disease cannot spread further.

If detection succeeds but either identification or removal is not possible, then the alternative is to discard the infected program and reload a clean backup version.

Stephenson identifies four generations of antivirus software:

- **A first-generation** scanner (simple scanners) requires a virus signature to identify a virus. The virus may contain "wildcards" but has essentially the same structure and bit pattern in all copies. Such signature-specific scanners are limited to the detection of known viruses. Another type of first-generation scanner maintains a record of the length of programs and looks for changes in length.

- **A second-generation** scanner (heuristic scanners) does not rely on a specific signature. Rather, the scanner uses heuristic rules to search for probable virus infection. One class of such scanners looks for fragments of code that are often associated with viruses. For example, a scanner may look for the beginning of an encryption loop used in a polymorphic virus and discover the encryption key. Once the key is discovered, the scanner can decrypt the virus to identify it, then remove the infection and return the program to service.

  Another second-generation approach is integrity checking. A checksum can be appended to each program. If a virus infects the program without changing the checksum, then an integrity check will catch the change.

  To counter a virus that is sophisticated enough to change the checksum when it infects a program, an encrypted hash function can be used. The encryption key is stored separately from the program so that the virus cannot generate a new hash code and encrypt that. By using a hash function rather than a simpler checksum, the virus is prevented from adjusting the program to produce the same hash code as before.

- **Third-generation** programs (activity traps) are memory-resident programs that identify a virus by its actions rather than its structure in an infected program. Such programs have the advantage that it is not necessary to develop signatures and heuristics for a wide array of viruses. Rather, it is necessary only to identify the small set of actions that indicate an infection is being attempted and then to intervene.

- **Fourth-generation** products (full-featured protection) are packages consisting of a variety of antivirus techniques used in conjunction. These include scanning and activity trap components. In addition, such a package includes access control capability, which limits the ability of viruses to penetrate a system and then limits the ability of a virus to update files in order to pass on the infection.

Computer Security

Second semester –Fourth stage

Assist. Prof. Dr. Nada Hussein M. Ali

Lecture 6

Introduction to Firewalls

1- Introduction

A firewall is a hardware or software system that prevents unauthorized access to or from a network. It can be implemented in both hardware and software, or a combination of both. Firewalls are frequently used to prevent unauthorized Internet users from accessing private networks connected to the Internet. All data entering or leaving the intranet pass through the firewall, which examines each packet, and blocks those that do not meet the specified security criteria. Figure 1 shows the architecture of the system uses firewall and without used it.

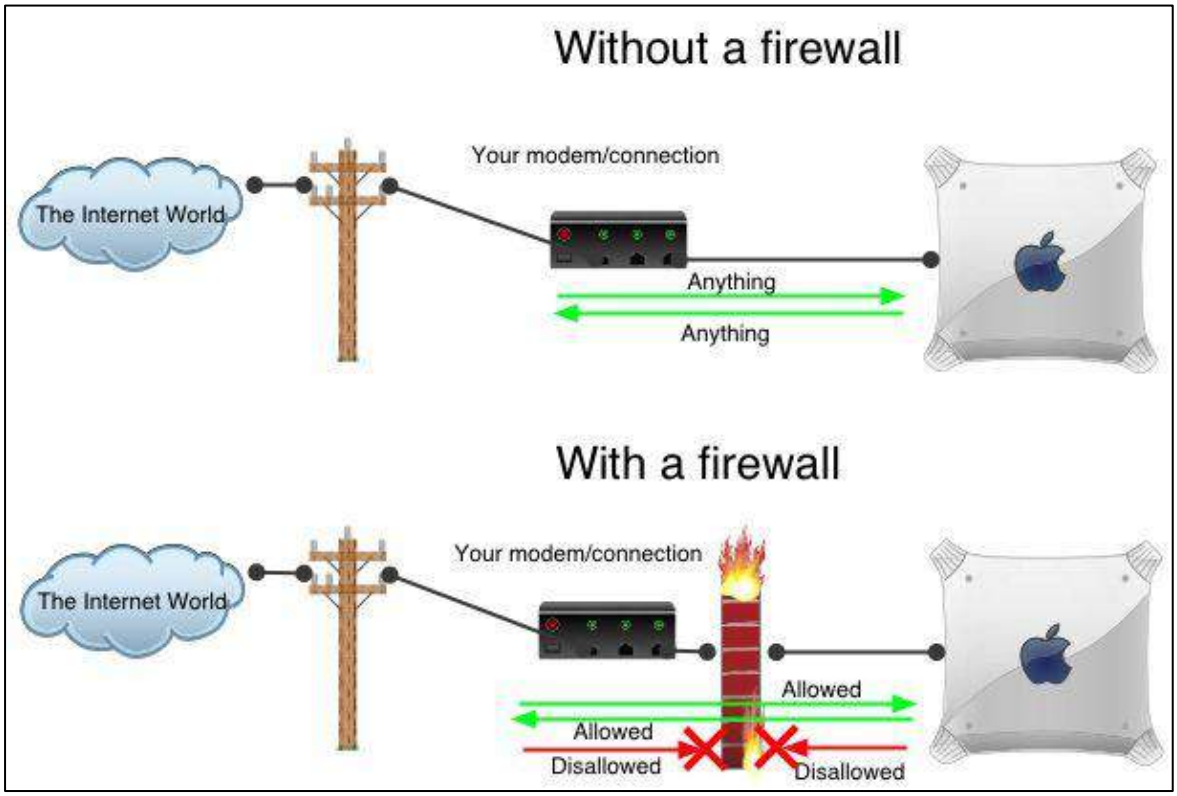


Figure (1): Simple routed network with and without firewall device

2- The Benefits of Firewall Security:

- ***Monitors Traffic***

A firewall monitors all of the traffic entering your computer network. A two-way firewall does double duty and monitors the traffic exiting your network as well. Often, they provide summaries to the administrator about what type/volume of traffic has been processed through it.

- ***Blocks Trojans***

A firewall helps block Trojan horses. These types of intruders latch onto the computer files, and when the file sends out a file, they go along for the ride to do more damage at the destination. A firewall blocks them from the outset, before they have a chance to infect your computer.

- ***Stops Hackers***

Having a firewall keeps hackers out of your network. Without firewall security, a hacker could get a hold of your computer and make it a part of what's called a botnet, which is a large group of computers used to conduct an illicit activity, such as spreading viruses. Also individuals, who you may not suspect, such as neighbors, can also take advantage of an open Internet connection you may have. A firewall prevents them.

- ***Stops Keyloggers***

Having firewall security will reduce the risk of keyloggers monitoring you. A keylogger is spyware software that cybercriminals try to put on your computer so they can target your keystrokes. After they can identify what you're typing in and where, they can use that information to do the same thing. This knowledge can help them log in to your private online accounts.

3- Types of Firewalls

Firewalls have a wide range of capabilities. Types of firewalls include

- packet filtering gateways or screening routers
- stateful inspection firewalls
- application proxies
- guards
- personal firewalls

➢ Packet Filtering Firewalls:

A packet filtering is the simplest, and in some situations, the most effective type of firewall. Packet filtering mechanisms work in the network layer of the OSI model. In packet filtering, each packet passing through a firewall is compared to a set of rules before it is allowed to pass through. Depending on the packet and the rule, the packet can be either dropped, sent through or a message can be forwarded to the originator. The packet filter is demonstrated in Figure 2.

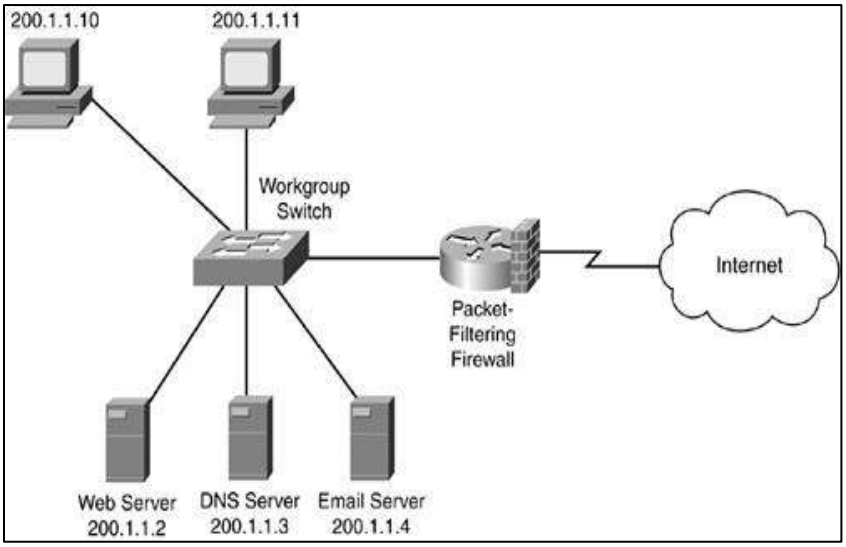


Figure (2): Packet Filtering Firewalls

Figure 3 shows a packet filter that blocks access from (or to) addresses in one network; the filter allows HTTP traffic but blocks traffic using the Telnet protocol.

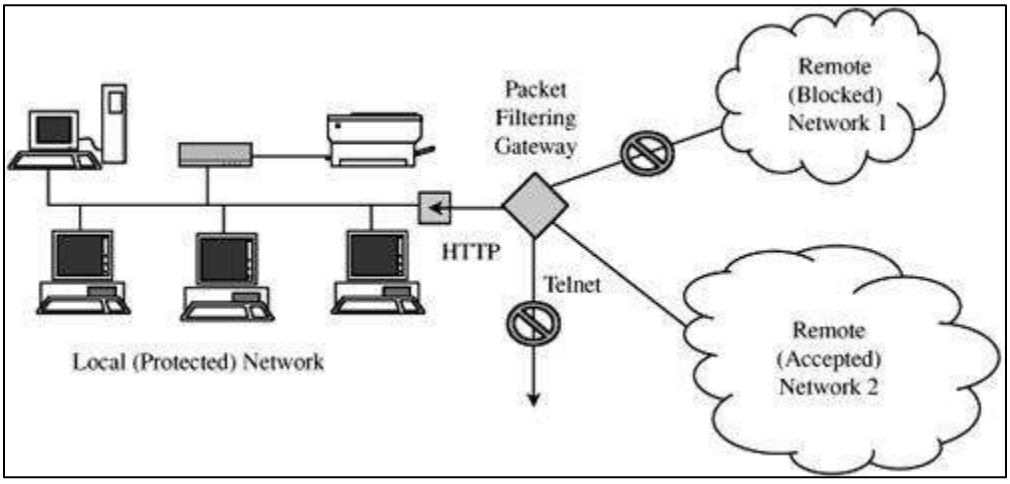


Figure (3): Packet Filter Blocking Addresses and Protocols.

*Advantage:*The Biggest Advantage of Packet Filtering Firewalls is Cost and Lower Resource Usage and best suited for Smaller Networks.

Disadvantage:

Packet Filtering Firewalls can work only on the Network Layer and these Firewalls do not support Complex rule based models. And it's also Vulnerable to Spoofing in some Cases.

- **Stateful inspection firewalls**
  - ✓ Stateful inspection, also known as **dynamic packet filtering,** is a firewall technology that monitors the state of active connections and uses this information to determine which network packets to allow through the firewall.
  - ✓ Stateful inspection monitors communications packets over a period of time and examines both incoming and outgoing packets.
  - ✓ In a firewall that uses stateful inspection, the network administrator can set the parameters to meet specific needs. In a typical network, ports are closed unless an incoming packet requests connection to a specific port and then only that port is opened. This practice prevents port scanning, a well-known hacking technique.

- **Application Proxy firewalls**
  - ✓ Application Proxy firewalls offer more security than other types of firewalls, but at the expense of speed and functionality, as they can limit which applications the network supports.
  - ✓ In application proxy firewall, computers establish a connection to the proxy, which serves as an intermediary, and initiate a new network connection on behalf of the request. This prevents direct connections between systems on either side of the firewall and makes it harder for an attacker to discover where the network is, because they don't receive packets created directly by their target system. Figure (4) illustrate the functionality of this function.

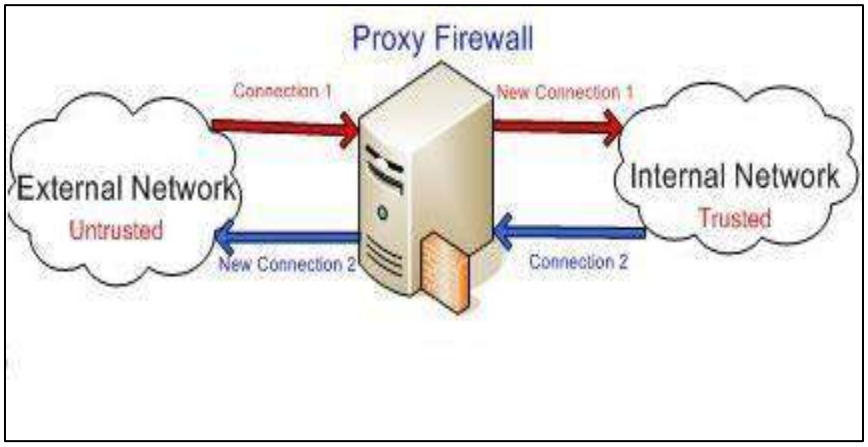


Figure (4): The connection establishment in application Proxy firewalls

- **Guard Firewalls**

A guard is a sophisticated firewall. Like a proxy firewall, it receives protocol data units, interprets them, and passes through the same or different protocol data units that achieve either the same result or a modified result. The guard decides what services to perform on the user's behalf in accordance with its available knowledge.

Guard activities can be quite sophisticated, as illustrated in the following examples:

- ✓ A university wants to allow its students to use e-mail up to a limit of so many messages or so many characters of e-mail in the last so many days. Although this result could be achieved by modifying e-mail handlers, it is more easily done by monitoring the common point through which all e-mail flows, the mail transfer protocol.

- ✓ A school wants its students to be able to access the World Wide Web but, because of the slow speed of its connection to the web, it will allow only so many characters per downloaded image (that is, allowing text mode and simple graphics, but disallowing complex graphics, animation, music, or the like).

- ✓ A library wants to make available certain documents but, to support fair use of copyrighted matter; it will allow a user to retrieve only the first so many characters of a document. After that amount, the library will require the user to pay a fee that will be forwarded to the author.
- ✓ A company wants to allow its employees to fetch files via ftp. However, to prevent introduction of viruses, it will first pass all incoming files through a virus scanner. Even though many of these files will be non-executable text or graphics, the

company administrator thinks that the expense of scanning them (which should pass) will be negligible.

Since the security policy implemented by the guard is somewhat more complex than the action of a proxy, the guard's code is also more complex and therefore more exposed to error. Simpler firewalls have fewer possible ways to fail or be subverted.

- **Personal Firewalls**

A personal firewall is an application program that runs on a workstation to block unwanted traffic, usually from the network and it screens traffic on a single workstation.

- ✓ The personal firewall is configured to enforce some policy. For example, the user may decide that certain sites, such as computers on the company network, are highly trustworthy, but most other sites are not.

- ✓ Combining a virus scanner with a personal firewall is both effective and efficient. However, leaving the virus scanner execution to the user's memory means that the scanner detects a problem only after the fact such as when a virus has been downloaded in an e-mail attachment. With the combination of a virus scanner and a personal firewall, the firewall directs all incoming e-mail to the virus scanner, which examines every attachment the moment it reaches the target host and before it is opened.

- ✓ A personal firewall runs on the very computer it is trying to protect. Thus, a clever attacker is likely to attempt an undetected attack that would disable or reconfigure the firewall for the future

Computer Security

Second semester –Fourth stage

Assist. Prof. Dr. Nada Hussein M. Ali

Lecture 7

Intrusion Detection System (IDS) and Intrusion Prevention System (IPS)

Intrusion Detection System (IDS) is meant to be a software application which monitors the network or system activities and finds if any malicious operations occur. Tremendous growth and usage of internet raises concerns about how to protect and communicate the digital information in a safe manner. Nowadays, hackers use different types of attacks for getting the valuable information. Many intrusion detection techniques, methods and algorithms help to detect these attacks.

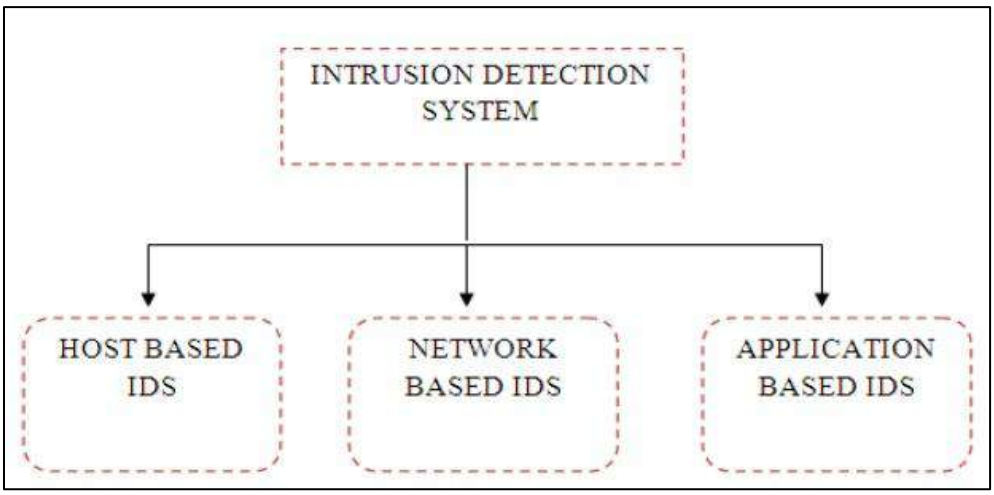
1- False Positive & False Negatives Attacks

The term false positive itself refers to security systems incorrectly seeing legitimate requests as spam or security breaches. Basically, the IDS will detect something it is not supposed to. Alternatively, IDS is prone to false negatives where the system fails to detect something it should. Both of these problematic problems are associated with IDS, but are issues vendors spend a lot of time working on, and as a result, it is not believed that IDS detects a high percentage of false positive or false negatives. Still, it is a topic worth consideration when looking at different IDS solutions.

2- Architectures Types of IDS

Figure 1 shows the different architectures types of Intrusion detection systems.

- Host based IDS (HIDS)
- Network based NIDS
- Application based APIDS



Figure(1): The three types of Intrusion Detection System

2-1 Host based IDS (HIDS)

Host based intrusion detection (HIDS) refers to intrusion detection that takes place on a single host system. Currently, HIDS involves installing an agent on the local host that monitors and reports on the system configuration and application activity. Some common abilities of HIDS systems include log analysis, event correlation, integrity checking, policy enforcement, rootkit detection, and alerting. They often also have the ability to baseline a host system to detect variations in system configuration.

Advantages of Host based Intrusion Detection Systems:

- Verifies success or failure of an attack
- Monitors System Activities
- Detects attacks that a network based IDS fail to detect
- Does not require additional hardware
- Lower entry cost

2-2 Network-Based IDS (NIDS)

A network-based intrusion detection system (NIDS) is used to monitor and analyze network traffic to protect a system from network-based threats. A NIDS reads all inbound packets and searches for any suspicious patterns. When threats are discovered, based on its severity, the system can take action such as notifying administrators, or barring the source IP address from accessing the network.

Advantages of Network based Intrusion Detection Systems:

- Lower Cost of Ownership
- Easier to deploy

- Detect network based attacks
- Retaining evidence
- Real Time detection and quick response.
- Detection of failed attacks

2-3 Application-Based IDS (APIDS)

An application-based IDS is like a host-based IDS designed to monitor a specific application (similar to antivirus software designed specifically to monitor your mail server). An application-based IDS is extremely accurate in detecting malicious activity for the applications it protects. However, this type of specialized IDS may fail to detect attacks not specifically targeted at that application. Hackers have also been known to shut down application-based IDS systems.
Most of the Intrusion Detection Systems used today are not implemented using a single approach, because all of them have their own advantages and disadvantages. Generally they use multiple approaches to gather information to detect malicious behavior in the system. When they are all used together, it also can be thought in a hierarchical order (application - host – network). Output from an IDS can be utilized by other ID systems at the same or higher levels in the hierarchy.

3- Methods of IDS

There are essentially only two methods of intrusion detection.

- **Signature-based IDSs** attempt to detect attacks based on known "signatures" or patterns. This is analogous to signature-based virus detection.

- **Anomaly-based IDSs** attempt to define a baseline, or normal, behavior of a system and provide a warning whenever the system strays too far from this baseline.

3-1 Signature-based IDS (or Heuristic Intrusion Detection)

Signature based detection works in a similar fashion to a virus scanner. This style of detection relies on rules and tries to associate possible patterns to intrusion attempts. Failed login attempts may be indicative of a password cracking attack, so an IDS might consider "N failed login attempts in M seconds" an indication, or signature, of an attack. Then anytime that N or more failed login attempts occur within M seconds, the IDS would issue a warning that a password cracking attack is suspected.

Advantages of Signature-based IDS

- Simplicity,
- Excellent ability to detect known attacks.

- Low alarm rates: All it has to do is to look up the list of known signatures of attacks and if it finds a match report it.
- Signature based is very accurate and Efficiency (provided the number of signatures is not excessive).
- Speed: The systems are fast since they are only doing a comparison between what they are seeing and a predetermined rule.
- Another major benefit is that the warning that is issued is specific. With a specific warning, an administrator can quickly determine whether the suspected attack is real or a false alarm and, if it is real, respond appropriately.

Disadvantages of Signature-based IDS

- Databases to constantly be updated to matches patterns that are not in the database.
- The system can only detect known attacks only.
- Even slight variations on known attack are likely to be missed by signature-based systems.
- If someone develops a new attack, there will be no protection.
- "Only as strong as its rule set."
- Another problem occurs when an attacker will try to modify a basic attack in such a way that it will not match the known signature of that attack. For example, the attacker may convert lowercase to uppercase letters.

Computer Security

Second semester –Fourth stage

Assist. Prof. Dr. Nada Hussein M. Ali

Lecture 8

3-2 Anomaly-based IDS

Anomaly based IDS are based on tracking unknown unique behavior pattern of detrimental activity. Anomaly-based IDSs look for unusual or abnormal behavior. There are several major challenges inherent in such an approach. First, we must determine what is normal for a system. Second, it is crucial that the definition of "normal" can adapt as system usage changes and evolves. Third, there are difficult statistical thresholding issues involved. For example, we must have a reasonable idea of how far abnormal lives from normal.

Example

Suppose that we monitor the use of the three commands

open, read, close.

We find that under normal use, Alice uses the series of commands

open, read, close, open, open, read, close.

We'll consider pairs of consecutive commands and try to devise a measure of normal behavior for Alice. From Alice's series of commands above, we observe that, of the six possible ordered pairs or commands, four pairs are normal for Alice, namely:

(open,read),(read,close),(close,open),(open,open)

And the other two pairs:

(read,open), (close,read)

Are abnormal. We can use this observation to identify potentially unusual behavior by Alice, or an intruder posing as Alice. We can then monitor the use of these three commands by Alice. If the ratio of abnormal to normal pairs is "too high," we would warn the administrator of a possible attack.

Advantages of Anomaly Based Detection

- New threats can be detected without having to worry about database being up to date

- Very little maintenance once system is installed it continues to learn about network activity and continues to build its profiles.
- The longer the system is in use the more accurate it can become at identifying threats.

Disadvantages of Anomaly Based Detection

- The network can be in an unprotected state as the system builds its profile.
- If malicious activity looks like normal traffic to the system it will never send an alarm.
- False positives can become cumbersome with an anomaly based setup. Normal usage such as checking e-mail after a meeting has the potential to signal an alarm.

3-3 Anomaly vs. Signature

- Which is the best way to defend your network?
  - Both have advantages
  - Signature can be used as a standalone system
  - Anomaly has a few weak points that prevent it from being a standalone system.
- Signature is the better of the two for defending you network
- The best way is to use both!

4- IPS — An Active Security Solution

- IPS or intrusion prevention system, is definitely the next level of security technology with its capability to provide security at all system levels from the operating system kernel to network data packets.

- It provides policies and rules for network traffic along with an IDS for alerting system or network administrators to suspicious traffic, but allows the administrator to provide the action upon being alerted.

- Where IDS informs of a potential attack, an IPS makes attempts to stop it. Another huge leap over IDS, is that IPS has the capability of being able to prevent known intrusion signatures, but also some unknown attacks due to its database of generic attack behaviors.
- Thought of as a combination of IDS and an application layer firewall for protection, IPS is generally considered to be the "next generation" of IDS.

- IPS and IDS work best when integrated with additional and existing security solutions.

- IDS is considered a passive detection monitoring system while IPS is an active prevention system
- Currently, there are two types of IPSs that are similar in nature to IDS. They consist of host-based intrusion prevention systems (HIPS) products and network-based intrusion prevention systems (NIPS).

4-1 Host-based IPS (HIPS)

- Host-based intrusion prevention systems are used to protect both servers and workstations through software that runs between your system's applications and OS kernel.
- The software is preconfigured to determine the protection rules based on intrusion and attack signatures. The HIPS will catch suspicious activity on the system and then, depending on the predefined rules, it will either block or allow the event to happen.
- HIPS monitors activities such as application or data requests, network connection attempts, and read or write attempts to name a few.

4-2 Network-based IPS (NIPS)

- A network-based intrusion prevention system (often called inline prevention systems) is a solution for network-based security.
- NIPS will intercept all network traffic and monitor it for suspicious activity and events, either blocking the requests or passing it along should it be deemed legitimate traffic.
- Network-based IPSs works in several ways; scan for intrusion signatures, search for protocol anomalies, detect commands not normally executed on the network and more.
- One interesting aspect of NIPS is that if the system finds an offending (المتسبب بالمشكلة) packet of information it can rewrite the packet so the hack attempt will fail.

4-3 HIPS vs. NIPS

While host-based IPSs are considered to be more secure than network-based intrusion prevention systems, the cost to install the software to each and every server and workstation within your organization may be quite costly. Additionally, the HIPS on each system must be frequently updated to ensure the attack signatures are up-to-date.

Problems associated with implementing NIPS exist as well. Since all data moving through the network will pass through the IPS it could cause the network performance to drop. To combat this problem, network-based IPSs that consist of appliance or hardware and software packages are available today (at a larger cost), but it will take most of the load from running a software-based NIPS off your network.

Computer Security

Second semester –Fourth stage

Assist. Prof. Dr. Nada Hussein M. Ali

Lecture 9

Authenticated Protocols

Protocols are the rules that some particular interaction. Security protocols are the communication rules followed in security applications. Suppose that Alice must prove to Bob that she's Alice, where, typically, Alice and Bob are communicating over a network. In many cases, it's sufficient for Alice to prove her identity to Bob, without Bob proving his identity to Alice. But sometimes mutual authentication is required; that is, Bob must also prove his identity to Alice. There are a vast number of security protocols:

1. PAP - Password Authentication Protocol.
2. CHAP - Challenge Handshake Authentication Protocol.
3. Symmetric Keys Authentication Protocol.
4. Public Keys Authentication and Digital Signature Protocols.
5. Kerberos (Trusted Third Party).

1- Password Authentication Protocol (PAP)

Password Authentication Protocol (PAP) (Figure 1)is a simple authentication protocol in which the user name and password is sent to the remote access server in a plaintext (unencrypted) form. Using PAP is strongly discouraged because your passwords are easily readable from the Point-to-Point Protocol (PPP) packets exchanged during the authentication process. PAP is typically used only when connecting to older UNIX-based remote access servers that do not support more secure authentication protocols. This protocol does not attempt to provide mutual authentication, which may be required in some cases.

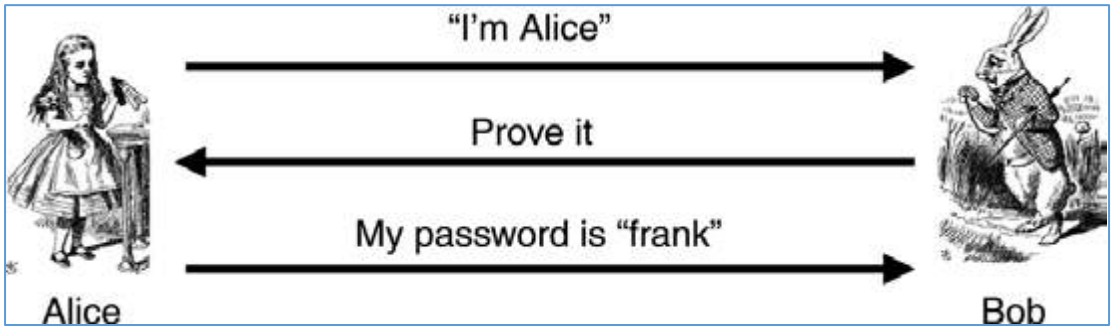


Figure (1): Simple authentication using PAP

2- CHAP (Challenge-Handshake Authentication Protocol)

CHAP (Challenge-Handshake Authentication Protocol) is a more secure procedure for connecting to a system than the Password Authentication Procedure (PAP). Here's how CHAP works:

1. After the link is made, the server sends a challenge message to the connection requestor. The requestor responds with a value obtained by using a one-way hash function.
2. The server checks the response by comparing it its own calculation of the expected hash value.
3. If the values match, the authentication is acknowledged; otherwise the connection is usually terminated.

To securely authenticate Alice, Bob will need to employ a "challenge-response" mechanism as shown in Figure 2. That is, Bob will send a challenge to Alice and the response from Alice will be something that only Alice can provide and that Bob can verify. To prevent a replay attack, Bob can employ a "number used once," or *nonce*, as the challenge. This nonce is the challenge. Alice must respond with the hash of her password, which proves that the response was generated by Alice, and the nonce, which proves that the response is fresh and not a replay. One problem with this protocol is that Bob must know Alice's password.

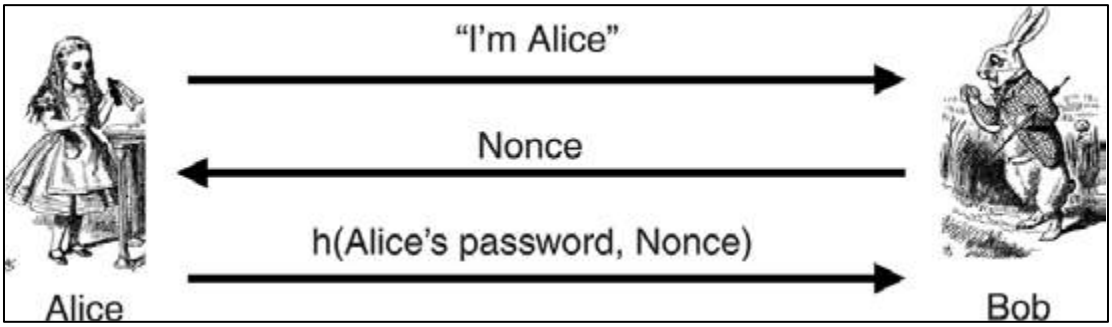


Figure (2) : Challenge-response or CHAP Protocol

At any time, the server can request the connected party to send a new challenge message. Because CHAP identifiers are changed frequently and because authentication can be requested by the server at any time, CHAP provides more security than PAP.

3- Symmetric Keys Authentication Protocol

We can design a secure authentication protocol based on symmetric key cryptography. Recall the notation for encrypting plaintext P with key K to obtain ciphertext C is: **C = E(P,K)**
and the notation for decrypting ciphertext C with key K to recover the plaintext P is: **P = D(C,K).**
Suppose that Alice and Bob share symmetric key K_{AB} and that this key is known only to Alice and Bob. Authenticate can be accomplished by proving knowledge of this shared symmetric key. Instead of hashing a nonce with a password, we'll encrypt the nonce *R* with the key K_{AB}. This protocol is illustrated in Figure 3.

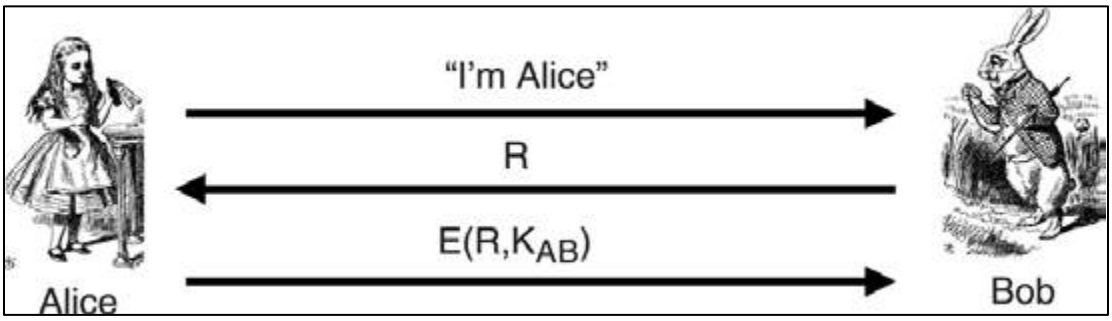


Figure (3): Symmetric key authentication protocol

The symmetric key authentication protocol in Figure above allows Bob to authenticate Alice and it prevents a replay attack and the protocol lacks mutual authentication. A more reasonable approach to mutual authentication would be to use the secure authentication protocol in Figure 3 and repeat the process twice, once for Bob to authenticate Alice and

once more for Alice to authenticate Bob. This approach is illustrated in Figure 4, where simply combined a few messages for the sake of efficiency.

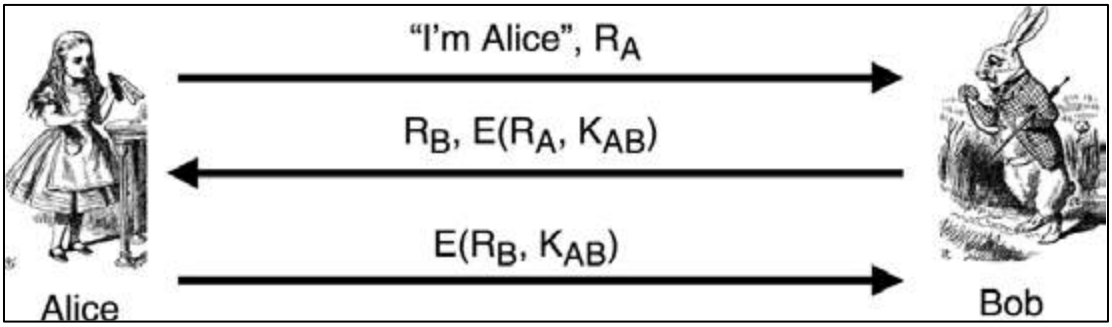


Figure (4): Secure mutual authentication using symmetric key protocol

Alice initiates a conversation with Bob and sends the challenge *RA* to Bob. Bob encrypts the challenge *RA* and sends it back along with a challenge *RB*, to Alice. At this point she encrypt RB with the same key *E(RB,K_{AB}).*

Computer Security

Second semester –Fourth stage

Assist. Prof. Dr. Nada Hussein M. Ali

Lecture 10

4- Public Keys Authentication and Digital Signature Protocols.

Public-Key Cryptosystem:

- Public-key/two-key/asymmetric cryptography involves the use of two keys:
  - A public-key, which may be known by anybody, and can be used to encrypt messages, and verify signatures
  - A private-key, known only to the recipient, used to decrypt messages, and sign (create) signatures

- Is asymmetric because those who encrypt messages or verify signatures cannot decrypt messages or create signatures.

Two of the best-known uses of public-key cryptosystem are:

- *Public-key encryption*, in which a message is encrypted and decrypted in different keys. This is used in an attempt to ensure confidentiality.

- Digital signatures, in which a message is signed with the sender's private key and can be verified by anyone who has access to the sender's public key.

The protocol, as shown in Figure 5, allows Bob to authenticate Alice where Bob encrypt R with Alice public key. Since only Alice can decrypted R with her private key, then the correct reply with R allows Bob to authenticate Alice.

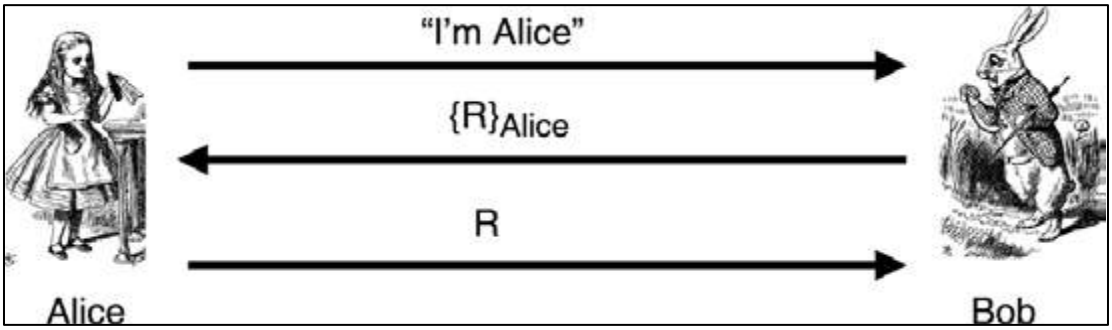


Figure (5): Authentication with public key Protocol.

The digital signature for a message is generated in two steps:

1. A *message digest* is generated. A message digest is a 'summary' of the message we are going to transmit, and has two important properties: (1) It is always smaller than the message itself and (2) Even the slightest change in the message produces a different digest. The message digest is generated using a set of hashing algorithms.

2. The message digest is encrypted using the sender's *private* key. The resulting encrypted message digest is the *digital signature*.

The digital signature is attached to the message, and sent to the receiver. The receiver then does the following:

1. Using the sender's public key, decrypts the digital signature to obtain the message digest generated by the sender.
2. Uses the same message digest algorithm used by the sender to generate a message digest of the received message.

3. Compares both message digests (the one sent by the sender as a digital signature, and the one generated by the receiver). If they are not *exactly the same*, the message has been tampered with by a third party. We can be sure that the digital signature was sent by the sender (and not by a malicious user) because *only* the sender's public key can decrypt the digital signature (which was encrypted by the sender's private key; remember that what one key encrypts, the other one decrypts, and vice versa). If decrypting using the public key renders a faulty message digest, this means that either the message or the message digest are not exactly what the sender sent. Figure 6 shows the digital signature protocol using public key cryptosystem.

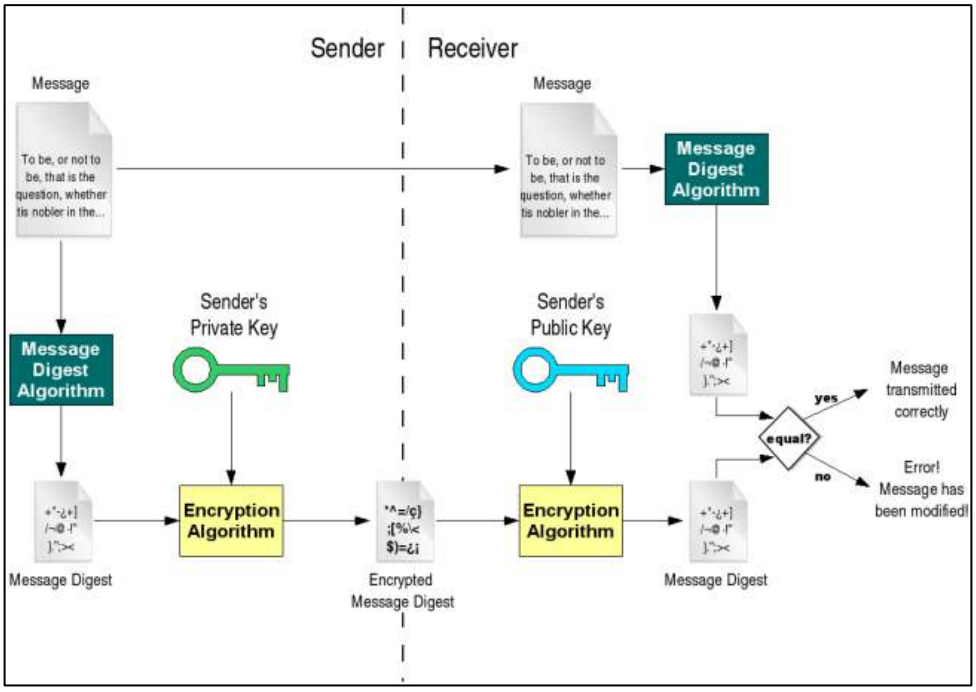


Figure (6): Digital Signature Protocol

5- Kerberos (Trusted Third Party)

A more likely scenario is that the two participants know nothing about each other, but both trust a third party. This third party is sometimes called an *authentication server*, and it uses a protocol to help the two participants authenticate each other. There are actually many different variations of this protocol. The one we describe is the one used in Kerberos, a TCP/IP-based security system developed at MIT.

There are two important principals in a Kerberos system that acts as the **Key Distribution Centre (KDC)**:

1. **Authentication Server (AS)** – client authenticate to AS based on long-term key. AS is the issuer of ticket granting ticket (TGT) and short-term keys (between client and TGS) in the protocol.
2. **Ticket Granting Server (TGS)** – client authenticate to TGS based on TGT and short-term key. TGS issues a ticket and short-term key (between client and requesting services) to the client. The client can now authenticate to any service that accept the ticket as a valid authentication token.

Figure 7 shows the details of Kerberos authenticated protocol

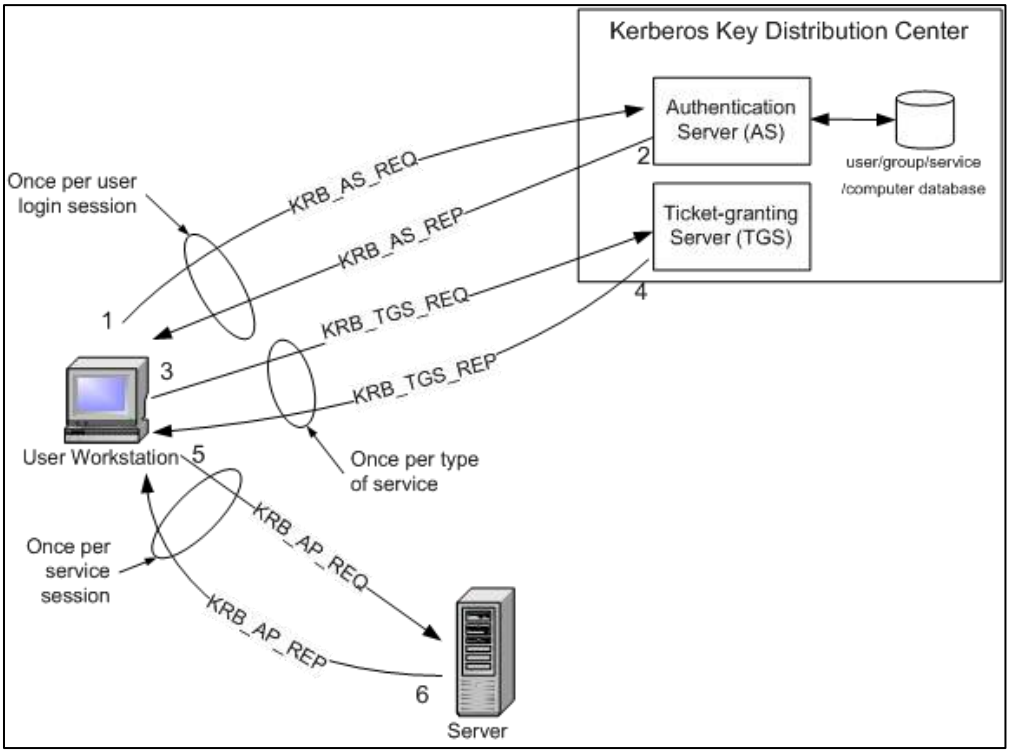


Figure (7): Kerberos authenticated Protocol

- **Step 1:** The user logs on to the workstation and requests service on the host. The workstation sends a message to the Authorization Server requesting a ticket granting ticket (TGT).

- **Step 2:** The Authorization Server verifies the user's access rights in the user database and creates a TGT and session key. The Authorization Sever encrypts the results using a key derived from the user's password and sends a message back to the user workstation. The workstation prompts the user for a password and uses the password to decrypt the incoming message. When decryption succeeds, the user will be able to use the TGT to request a service ticket.

- **Step 3:** When the user wants access to a service, the workstation client application sends a request to the Ticket Granting Service containing the client name, realm name and a timestamp. The user proves his identity by sending an authenticator encrypted with the session key received in Step 2.

- **Step 4:** The TGS decrypts the ticket and authenticator, verifies the request, and creates a ticket for the requested server. The ticket contains the client name and optionally the client IP address. It also contains the realm name and ticket lifespan. The TGS returns the ticket to the user workstation. The returned message contains two copies of a server session key – one encrypted with the client password, and one encrypted by the service password.

- **Step 5:** The client application now sends a service request to the server containing the ticket received in Step 4 and an authenticator. The service authenticates the request by decrypting the session key. The server verifies that the ticket and authenticator match, and then grants access to the service. This step as described does not include the authorization performed by the Intel AMT device, as described later.

- **Step 6**: If mutual authentication is required, then the server will reply with a server authentication message.