COMPUTER NETWORK SECURITY ESSENTIALS

Dr. MAZIN SAMMER

COMPUTER NETWORK SECURITY LECTURS

COMPUTER SCIENCE DEPARTMENT

UINVERSITY OF TECHNOLOGY

IRAQ

NW SECURITY LECTURES, 2008

Computer Network Security Essentials

University of Technology
Computer Science Department
Network Security
4th Class
Lecturer: Dr. Mazin Sammer

COMPUTER NETWORK SECURITY ESSENTIALS

- **Chapter 1: An Introduction to Network Security.**
- **Chapter 2: Threats in Networks.**
- **Chapter 3: Network Security Controls.**
- **Chapter 4: Network Security Solutions.**
- **Chapter 5: Advance Network Security Topics.**
- **References:**

1. **Security in Computing, 3rd Edition**

   By*: Charles P. Pfleeger, Shari Lawrence Pfleeger*

   Prentice Hall – 2003.

2. **Cryptography and Network Security**

   By: *William Stalling.*

   Prentice Hall – 1999.

3. **Several Papers Published on the Web**

Network Security Essentials

University of Technology
Computer Science Department
Network Security
4th Class
Lecturer: Dr. Mazin Sammer

Index

Chapter One: Threats in Networks

Chapter Three: Network Security Controls

4.4 Secure E-Mail

4.4.1 Security for E-Mail

4.4.2 Threats to E-Mail

4.4.3 Requirements and Solutions

4.4.4 Confidentiality

4.4.5 Other Security Features

4.4.6 Example Secure E-Mail Systems

A: PGP

B: S/MIME

4.5 Multilevel Security on Networks

4.5.1 Trusted Network Interface

4.5.2 Secure Communication

Chapter Five: Advance Network Security Topics

5.1 IP Security.

5.2 Web Security.

Chapter

1

Network Security Essentials

University of Technology
Computer Science Department
Network Security
4th Class
Lecturer: Dr. Mazin Sammer

Chapter One

Introduction to Network Security

1.1 Network Security Definition.
1.2 Computing Systems.
1.3 ISO-OSI Reference Model.
1.4 TCP/IP Model.
1.5 IP Address
1.6 Ports.
1.7 Networks are System, Too.
1.8 Security Attacks.
1.9 Active and Passive Attacks.
1.10 Methods of Defense.

1.1 Network Security Definition:

Network security address questions such as how to efficiently control access to computer networks and distributed systems, and how to securely transmit data between them.

1.2 Computing Systems:

The term of **computing system** is referring to a single main processor, its peripheral devices, its data, and its software. Such a system has a single operating system and one set of users, as shown in figure (1). All blocks of the computing system are located within a small distance of rest of the system.

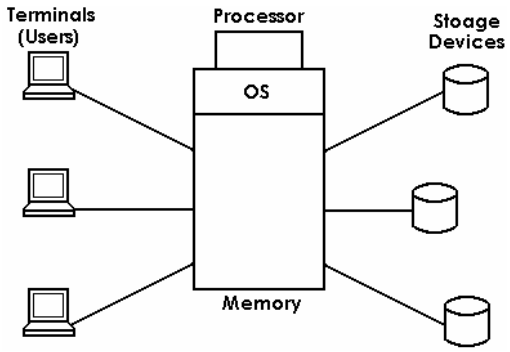


Figure (1): Computing System

While the term of **computing network** is a collection of two or more independent computing systems. The user accesses the network only indirectly through a computing system. The computing network defines to be a computing environment with more than one independent processor, as shown in figure (2).

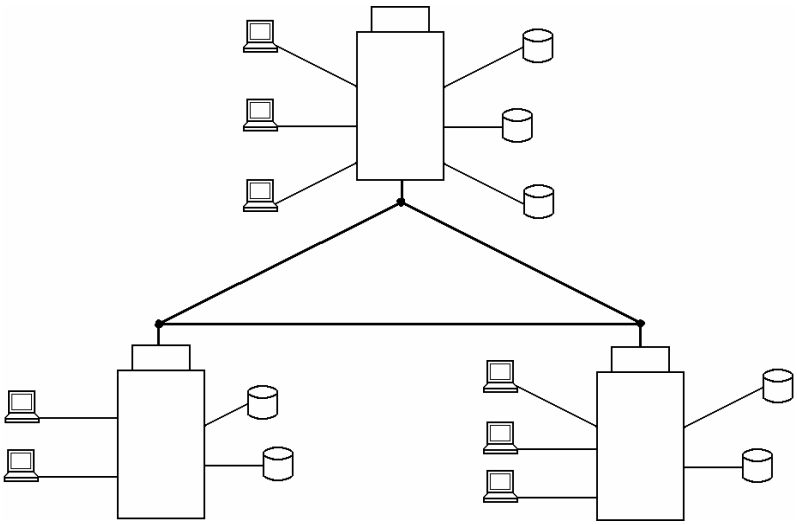


Figure (2): Computer Network

1.3 ISO-OSI Reference Model:

The ISO (International Standards Organization) has developed OSI (Open System Interconnect) model in 1982 for computer network connection. The OSI reference model specifies the seven layers of functionality, as shown in Figure (3).

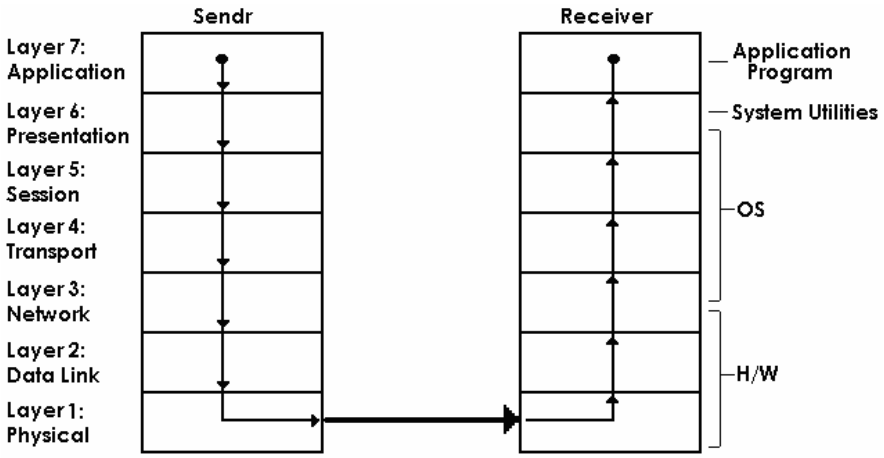
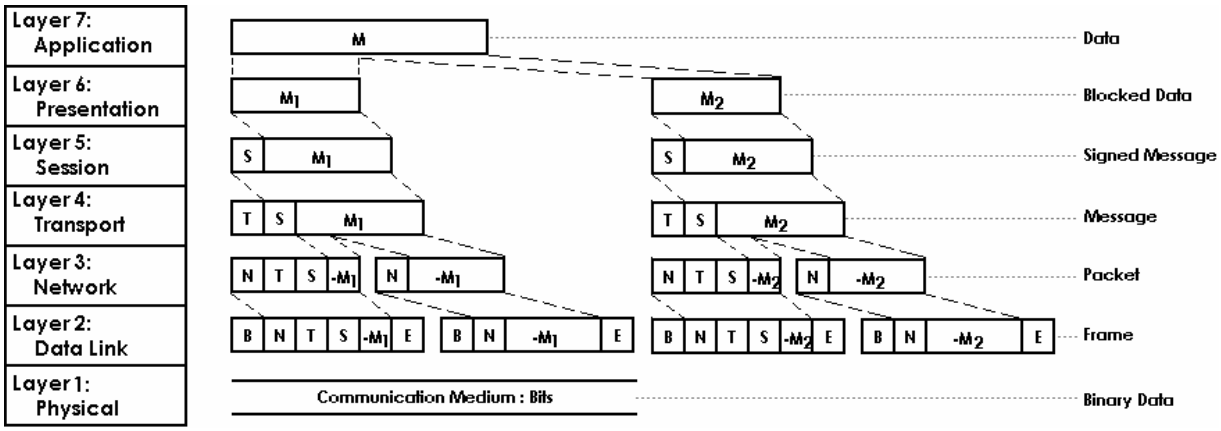


Figure (3): OSI Refernce Model.

The layers represent different actives performed in the actual transmission of a message. Figure (4) shows a typical message that has been acted upon by the seven layers to prepare it for transmission. Layer 6 breaks the original message data into blocks. At the layer 5, a session header is added to show the sender, the receiver and some sequencing information. Layer 4 adds information concerning the logical connection between the sender and receiver. At the layer 3 routing information is added, it also divides the message into unties called 'packets' which are the standard units of communication in a network. The layer 2 adds both a header and trailer to ensure correct sequencing of the message blocks, and to detect and correct transmission errors. The individual bits of the message and the control information are transmitted on the physical medium by level 1.

All the additions to the message are checked and removed by the corresponding layer on the receiving side.



S: Session Header: Sequence Info.; Sender/Receiver Identification
T: Transport Header: Connection Info.
N: Network Header: Routing Info.
B: Data Link Header: Sequence Info.
E: Data Link Trailer: Error Correction Info.

Figure (4): Message Prepared for Transmission.

1.4 TCP/IP Model:

The TCP/IP four-layer model is created with reference to the seven-layer OSI model, as shown in Figure (5). Both the OSI model and the TCP/IP layered model are based on many similarities, but there are philosophical and practical differences between the two models. However, they both deal with communications among heterogeneous computers.

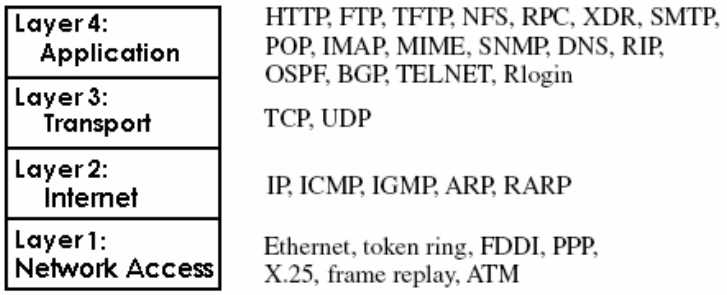


Figure (5): Message Prepared for Transmission.

1.4.1 Network Access Layer:

The network access layer contains protocols that provide access to a communication network.

1.4.2 Internet Layer:

The Internet layer provides a routing function. This layer consists of the Internet Protocol (IP) and the Internet Control Message Protocol (ICMP).

1.4.3 Transport Layer:

The transport layer delivers data between two processes on different host computers. This layer contains the Transmission Control Protocol (TCP) and the User Datagram Protocol (UDP).

1.4.4 Application Layer:

This layer provides a direct interface with users or applications. Some of the important application protocols are File Transfer Protocol (FTP) for file transfers, Hypertext Transfer Protocol (HTTP) for the World Wide Web, Simple Network Management Protocol (SNMP) for controlling network devices, Simple Mail

Transport Protocol (SMTP), Post Office Protocol (POP), Internet Mail Access Protocol (IMAP), Internet Control Message Protocol (ICMP) for email, Privacy Enhanced Mail (PEM), Pretty Good Privacy (PGP) and Secure Multimedia Internet Mail Extensions (S/MIME) for e-mail security.

1.5 IP Addresses:

IP address is the address of a computer which attached to a TCP/IP network (e.g. the Internet). Every client device (defined as a requester device of services) and server device (defined as the provider of services) must have a unique IP address.

Client workstations have either a static address or a dynamic address which assigned to them each dial-up session. IP addresses are written as four sets of numbers separated by periods; for example, (192.168.111.222) or (10.123.1.102) or (172.16.4.30), etc.

1.6 Ports:

A port represents an endpoint in the establishment of a connection between two or more computers. For the computer acting as the client, the destination port number will typically identify the type of application/service being hosted by the server.

For example:

TCP port **21** is the destination port number used when communicating with an **FTP** server.
TCP port **22** is the destination port number used when communicating with an **SSH** server.
TCP port **23** is the destination port number used when communicating with an **Telnet** server.
TCP port **25** is the destination port number used when communicating with an **SMTP** server.
TCP port **80** is the destination port number used when communicating with an **HTTP** server.
TCP port **110** is the destination port number used when communicating with a **POP3** server.
TCP port **5190** is the destination port number used when communicating with an **AOL IM** server.
TCP port **6667** is the destination port number used when communicating with an **IRC** server.

The above is a small selection from a possible 65,535 (64K) port numbers.

The port numbers are divided into three ranges: the Well Known Ports (from 0 through 1023), the Registered Ports (from 1024 through 49151), and the Dynamic and/or Private Ports (from 49152 through 65535).

1.6 Networks are System, Too:

The computer network is a large computing system containing other computing systems. The computing networks have similar characteristics.

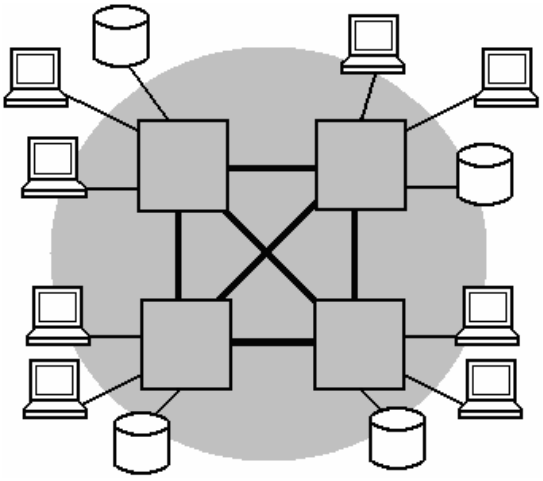


Figure (6): Distributed Computing System.

Computer networks offer several advantages over single-processor system:

1- **Resource Sharing**: Users of a network can access a variety of resources through the network.

2- **Increased Reliability**: The failure of one system in the network (which consists of more than one computing system) not block users from continuing to computer. If similar systems exist, users can move their computing tasks to other when one system fails.

3- **Distributing the Workload**: The workload can be shifted from a heavily loaded system to an underutilized one.
4- **Expandability**: Network systems can be expanded easily by adding new nodes.

1.7 Security Attacks:

The network (also computation system) must be ensuring the integrity of data, secrecy of data, and availability of service. And each user should be accesses the network through a single OS, which also includes network interface responsibilities.

- **Integrity**: means that the assets (data) can be modification (writing, changing, changing status, deleting, and creating) only by authorized parties. (*Only Authorized Disclosure*)
- **Secrecy**: means that the assets (data)of a computing systems are accessible (reading, viewing, and printing) only by authorized parties. (*Only Authorized Modification*)
- **Availability**: means that the assets (services) are available to authorized parties. An authorized party should not be prevented from accessing those services to which he or she or it has legitimate access. (*Not Denial of Services*)

There are four kinds of attacks on the security of a computer system or network (also called threats to the security of a computer system or network), as shown in figure (7) and figure (8):

1- **Interruption**: in an interruption the asset of the system becomes lost or unavailable or unusable. (*Non-availabile*)
2- **Interception**: means that some unauthorized party has gained access to an asset. (*Unsecred*)

3- **Modification**: if an unauthorized party not only accesses but tampers with an asset, the failure becomes a modification. (*Not Integrity*)

4- **Fabrication**: an unauthorized party might fabricate counter it objects for a computer system or network. (*NotAuthenticity*)

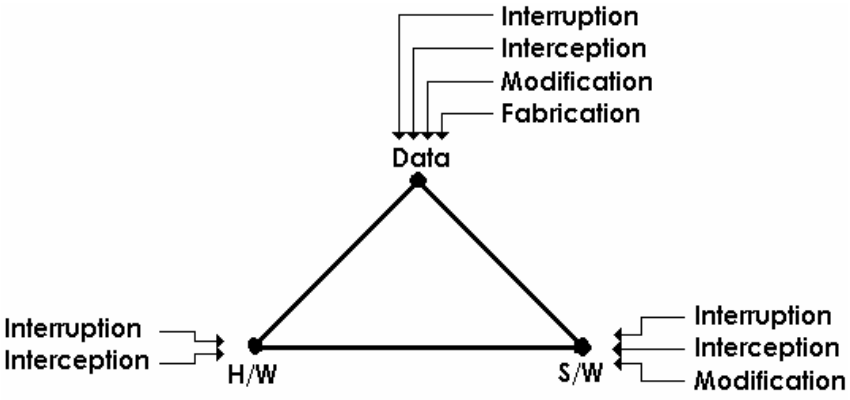


Figure (7): Attacks on (threats to) the security of a network.

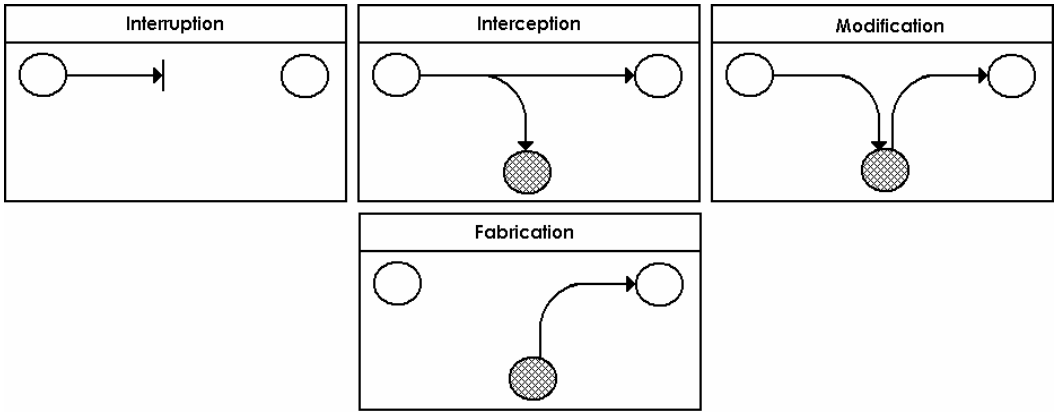
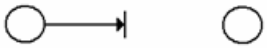
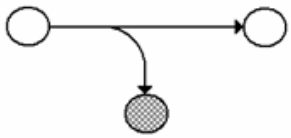
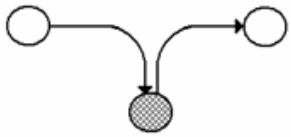
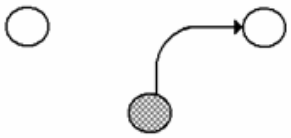


Figure (8): Security Threats (Security Attacks)

1.8 Active and Passive Attacks:

A useful categorization of the above security attacks is in terms of passive attacks and active attacks.

Security Threats	Active/Passive	Example
Interruption (Availability): 	**Active** Threats (Attacks)	- Denial of Service
Interception (Secrecy): 	**Passive** Threats (Attacks)	- Release of Message Contents - Traffic Analysis
Modification (Integrity): 	**Active** Threats (Attacks)	- Modification of message - Replay
Fabrication (Authenticity): 	**Active** Threats (Attacks)	- Fabrication of message

Passive attacks are very difficult to detect because they do not involve any aeration of the data, however, it is feasible to prevent the success of these attacks. Thus the emphasis in dealing with passive attacks is on prevention rather than detection.

While the active attacks are difficult to prevent absolutely, because to do so would require physical protection of all communications facilities and paths at all the times. Instead, the goal is to detect them and to recover from any disruption or delays caused by them.

1.9 Methods of Defense:

The methods of defense are able to detect a breach as or after it occurs and/or able to prevent attacks. There are several methods use as a defense: Software Controls, Hardware Controls, Physical Controls, Encryption, and Policies.

The defense methods must be used to be effective. They must be effective (in terms of time, memory space, human activity or other resources used), easy to use, and appropriate. However, several different methods may apply to one exposure.

Chapter

2

Network Security Essentials

University of Technology
Computer Science Department
Network Security
4th Class
Lecturer: Dr. Mazin Sammer

Chapter Two

Threats in Networks

2.1 Reasons for Network Security Problems.
2.2 Network Security Threats.
- 2.2.1 Wiretapping.
- 2.2.2 Impersonation.
- 2.2.3 Message Confidentiality Violations.
- 2.2.4 Message Integrity Violations.
- 2.2.5 Hacking.
- 2.2.6 Code Integrity Violations.
- 2.2.7 Denial of Service.
- 2.2.8 Protocol Flaws.
- 2.2.9 Spoofing.
- 2.2.10 Web Site Defacement.
- 2.2.11 Distributed Denial of Service.

2.1 Reasons for Network Security Problems:

Networks have security problems for the following reasons.

1- **Sharing**: Because of the resource and workload sharing of networks, more users have the potential to access networked systems than single computers. Perhaps worse, access is afforded to more systems, so that access controls for single systems may be inadequate in networks.

2- **Complexity of System**: A network combines two or more possibly dissimilar OSs, therefore a network operating/control system is likely to be more complex than an OS for a single computing system. This complexity confidence in the security of a network.

3- **Unknown Perimeter**: The expandability of a network also implies uncertainty about the network boundary. One host may be a node on two different networks, so that resources on one network are accessible to the users of the other network as well.

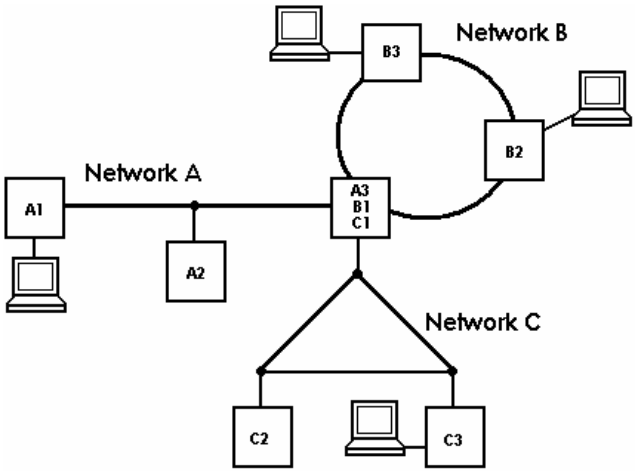


Figure (1): Unclear Network Boundaries.

4- **Many Points of Attack**: When a file is stored in a network host remote from the user, the file may pass through many host machines to get to the user. While the administrator of one host

may enforce rigorous security policies, that administrator may have no control over other hosts in the network. The user has to trust the access control mechanisms of all these systems.

5- **Unknown Path**: As shown in figure (1), there may be many paths from one host to another.

6- **Anonymity**: An attacker can mount an attack from thousand of mails away and thus never have to touch the system attacked or come into content with any of its administrators or users. The attack can be passed through many other hosts, in an effort to disguise from where the attack originated. Finally, computer-to-computer authentication is not the same as for human-to-computer.

Consider Microsoft's authentication scheme for its Windows operating systems.

In *Windows NT 4.0*, the authentication database is distributed among several domain controllers. Each domain controller is designated as a primary or backup controller. All changes to the authentication database must be made to the (single) primary domain controller; then the changes are replicated from the primary to the backup domain controllers.

In *Windows 2000*, the network views the controllers as equal trees in a forest, in which any domain controller can update the authentication database. This scheme reflects Microsoft's notion that the system is "multi-master": only one controller can be master at a given time, but any controller can be a master. Once changes are made to a master, they are automatically replicated to the remaining domain controllers in the forest.

The multi-master approach is more flexible and robust than the primary-secondary approach, because it allows any controller to take charge—especially useful if one or more controllers have failed or are out of service for some reason. But the multi-master approach introduces a new problem. Because any domain controller can initiate changes to the authentication database, any hacker able to dominate a domain controller can alter the authentication database. And, what's worse, the changes are then replicated throughout the remaining forest. Theoretically, the hacker could access anything in the forest that relies on Windows 2000 for authentication.

When we think of attackers, we usually think of threats from outside the system. But in fact the multi-master approach can tempt people inside the system, too. A domain administrator in any domain in the forest can access domain controllers within that domain. Thanks to multi-master, the domain administrator can also modify the authentication database to access anything else in the forest. For this reason, system administrators must consider how they define domains and their separation in a network.

2.2 Network Security Threats:

There are several potential threats: Wiretapping, Impersonation, Message Confidentiality Violations, Message Integrity Violations, Hacking, Code Integrity Violations, Denial of Service.

2.2.1 Wiretapping:

Wiretap means to intercept communications. Although the term has physical connection, no actual contact is necessary.

The passive wiretapping is just listening that is intercepting communication, while the active wiretapping means injecting something into the communication.

All signals sent through the cable are available for anyone to intercept. For example, each LAN connector called a 'packet sniffer' can retrieve all packets on the net, and the process called 'inductance an intruder' can pick up signals from the wire.

All signals sent through the air are available to anyone who wants to pick them up, the microwave and satellite communication are a very insecure medium.

No one can tap an optical fiber cable without detection, because, firstly the entire optical network must be tuned carefully each time a new connection is made, secondly, optical fiber carries light energy, not electricity, light does not emanate a magnetic field as electricity does. While the repeaters, splices, and taps along a optical cable are places where data may be available more easily than in the fiber cable itself. The connections from computing equipment to the fiber may also be points for penetration. By itself, fiber is much more secure than cable, but it has vulnerabilities also.

2.2.2 Impersonation:

Impersonation of another person process to obtain the data directly. In an impersonation, the attacker has several choice:

a- **Authentication Foiled by Guessing**: The attacker can guess the password because many users chose easy-to-guess passwords.

b- **Authentication Foiled by Eavesdropping**: When the some details of authentication are passed on the network, they are exposed to anyone observing the communication on the network. These same authentication details can be reused by an impersonator.

c- **Authentication Foiled by Avoidance**: A weak or flawed authentication allows access to any who can circumvent the authentication.

d- **Nonexistent Authentication**: If two computers are used by the same users and if each has authenticated its users on first access, it would seem there is no need for computer-to-computer authentication.

e- **Well-Known Authentication**: Some vendors still ship computers having a default password for its remote maintenance personnel.

f- **Trusted Authentication**: Authentication stopped by pass.

2.2.3 Message Confidentiality Violations:

a- **Missdelivery**: A destination address is modified or some handler malfunctions causing a message to be delivered to someone other than the intended recipient.

b- **Exposure**: Along the way the content of a message is exposed in temporary buffers; at switches, routers, gateways, and intermediate hosts throughout the network.

c- **Traffic Flow Analysis**.

2.2.4 Message Integrity (Correctness) Violations:

a- **Falsification of Messages**:

- Change the content of a message.
- Change any part of the content of a message.
- Replace a message entirely.
- Reuse an old message.
- Change the apparent source of a message.
- Redirect a message.
- Destroy or delete a message.

b- **Noise**: Signals sent over communications media are subject to interference from other traffic on the same media, as well as from natural sources, such as lightning, electric motors, and animals. Such unintentional interference is called noise.

2.2.5 Hacking:

The hackers can develop tools to search widely and quickly for particular weaknesses and move swiftly and stealthily to exploit those weaknesses. The hacker has unlimited time to analyze, plan, code, simulate, and test a future attack.

A hacker may impersonation to achieve access on one host, and the use a trusted relationship from that host to another to establish a session on the second host. From the second session the hacker may send spurious messages or mount a denial-of-service attack by flooding a LAN.

The network administrator should project forward the effects of the attacks, what additional capability would that give the hacker for future attacks?

2.2.6 Code Integrity (Correctness) Violations:

A serious threat in networking is damage to executable code. This threat is usually malicious (viruses, worms, trojan horses, and other malicious codes), designed to delete, modificate or replace running programs on a host.

The transfer method is similar, a user accepts or downloads a file that includes malicious code. For examples, the user typically is unaware what a downloaded file actually contains, or sometimes file downloading occurs without the user's permission (e.g. America OnLine Network change their software as often as once a day, each time the user logs onto the service, new software is download to the user's system, without the user's having been asked whether to do so).

Trusted system (mean code believed to be safe both by functional correctness and by enforcing its correctness on programs that run under it) could limit the damage possible from downloaded code, or in some cases, block the attacks entirely becaues it offers strong protection against malicious network codes.

2.2.7 Denial of Service (DoS):

a- **Connectivity**: Every point in the network must be reachable from every other points. So, most nodes are connected by multiple paths, so that when one path is unavailable, communication can be maintained using another path, but if the failure occur on a critical path or node will block the communication.

b- **Flooding**: An intruder can damage network communications by generating spurious messages. Their essential purpose is to increase the traffic on the network, thereby degrading service to the users.

c- **Routing Problems**: The attacker who wants to disable a network think first of the routing tables. Routine table modifications can either disable all communication, disable communication to a particular node, or route communication to the attacker, who can then selectively read, modify, destroy, or forward.

d- **Disruption of Service**: An active node (introder can program computers to impersonate network nodes) can also tamper with the flow of messages on the network. For examples, as shown in figure (2), the active node can flood a network with spurious messages, thereby blocking network traffic. A node can disrupt service without even being able to determine the content of legitimate messages on the network.

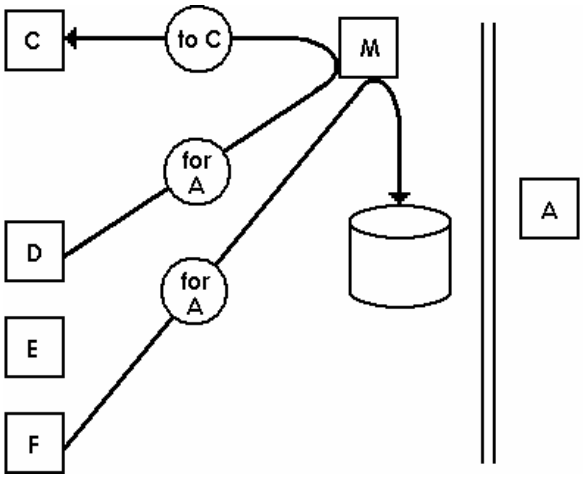


Figure (2): Malicious Disruption of Service.

2.2.8 Protocol Flaws:

Many problems with protocols have been identified by sharp reviewers and corrected before the protocol was established as a standard. But protocol definitions are made and reviewed by fallible humans. Likewise, protocols are implemented by fallible humans.

For example, TCP connections are established through sequence numbers. The client sends a sequence number to open a connection, the server responds with that number and a sequence number of its own, and the client responds with the server's sequence number. Suppose someone can guess a client's next sequence number. That person could impersonate the client in an interchange. Sequence numbers are incremented regularly, so it can be easy to predict the next number.

2.2.9 Spoofing:

a- **Masquerade:** In a masquerade one host pretends to be another. Common examples are:

1- **URL confusion**: Domain names can easily be confused, thus xyz.com, xyz.org, and xyz.net might be three different organizations, or one bona fide organization (for example, xyz.com) and two masquerade attempts from someone who registered the similar domain names. Names with or without hyphens (coca-cola.com versus cocacola.com) and easily mistyped names (l0pht.com versus lopht.com, or citibank.com versus citybank.com) are candidates for masquerading.

2- **Web page Overwrite**: The attacker exploits a flaw in the victim's web server and is able to overwrite the victim's web pages.

3- **False Site**: A clever attacker can try to build a false site that resembles the real one, perhaps to obtain sensitive information (names, authentication numbers, credit card numbers) or to induce the user to enter into a real transaction.

b- **Session Hijacking**: Session hijacking is intercepting and carrying on a session begun by another entity. Suppose two entities have

entered into a session but then a third entity intercepts the traffic and carries on the session in the name of the other. A different type of example involves an interactive session, for example, using Telnet. If a system administrator logs in remotely to a privileged account, a session hijack utility could intrude in the communication and pass commands as if they came from the administrator.

c- **Man-in-the-Middle Attack**: A man-in-the-middle attack is a similar form of attack, in which one entity intrudes between two others. The difference between man-in-the-middle and hijacking is that a man-in-the-middle usually participates from the start of the session, whereas a session hijacking occurs after a session has been established.

Suppose you want to exchange encrypted information with your friend. You contact the key server and ask for a secret key with which to communicate with your friend. The key server responds by sending a key to you and your friend. One man-in-the-middle attack assumes someone can see and enter into all parts of this protocol. A malicious middleman intercepts the response key and can then eavesdrop on, or even decrypt, modify, and reencrypt any subsequent communications between you and your friend.

This attack would be foiled with public keys, because the man-in-the-middle would not have the private key to be able to decrypt messages encrypted under your friend's public key.

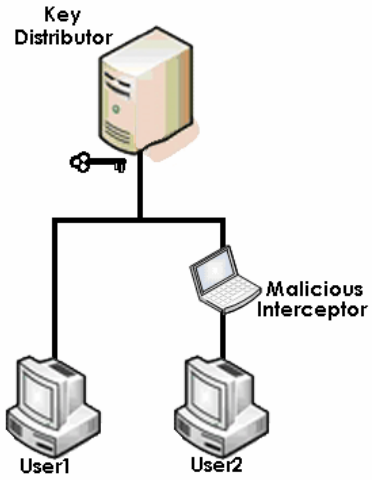


Figure (3): Man-in-the-Middle Attack

2.2.10 Web Site Defacement:

One of the most widely known attacks is the web site defacement attack. The web site vulnerabilities enable attacks known as buffer overflows, dot-dot problems, application code errors, and server-side include problems.

a- **Buffer Overflows**: Buffer overflow is alive and well on web pages, too. The attacker simply feeds a program far more data than it expects to receive. A buffer size is exceeded, and the excess data spill over into adjoining code and data locations.

b- **Dot-Dot and Address Problems**: Web server code should always run in a constrained environment (the web server should never have editors, xterm and Telnet programs, or even most system utilities), another condition for preventing an attack is to create a fence confining the web server application. With such a fence, the server application cannot escape from its area and access other potentially dangerous system areas.

The server begins in a particular directory subtree, and everything the server needs is in that same subtree. Enter the dot-dot. In both Unix and Windows, '..' is the directory indicator for predecessor. And '../..' is the grandparent of the current location. So someone who can enter file names can travel back up the directory tree one .. at a time. However, the vulnerability in the webhits.dll extension for the Microsoft Index Server. For example, passing the following URL causes the server to return the requested file, autoexec.nt, enabling an attacker to modify or delete it. http://URL/null.htw?CiWebHitsFile=/../../../../../winnt/system32/autoexec.nt

c- **Application Code Errors:** A user's browser carries on an intricate, undocumented protocol interchange with the web server. To make its job easier, the web server passes context strings to the user, making the user's browser reply with full context. A problem arises when the user can modify that context.

Assume you have selected one option on the web page and are looking at a second web page. The web server has passed you a URL similar to http://www.xyz.com/page4&option1=459012&price1=599. This URL may means you have chosen option number 459012, and its price is $15.99. You now select a second and the URL becomes http://www.xyz.com/page7&option1=459012&price1=599&option2=365217&price2=499. But if you are a clever attacker, you realize that you can edit the URL in the address window of your browser. Consequently, you change each of 599 and 499 to 99. And when the server totals up your order, your two options cost only $99 each.

d- **Server-Side Include:** The server-side include problem takes advantage of the fact that web pages can be organized to invoke a particular function automatically. For example, many pages use web commands to send an e-mail message in the "contact us" part of the displayed page. The commands, such as e-mail, if, goto, and include, are placed in a field that is interpreted in HTML.

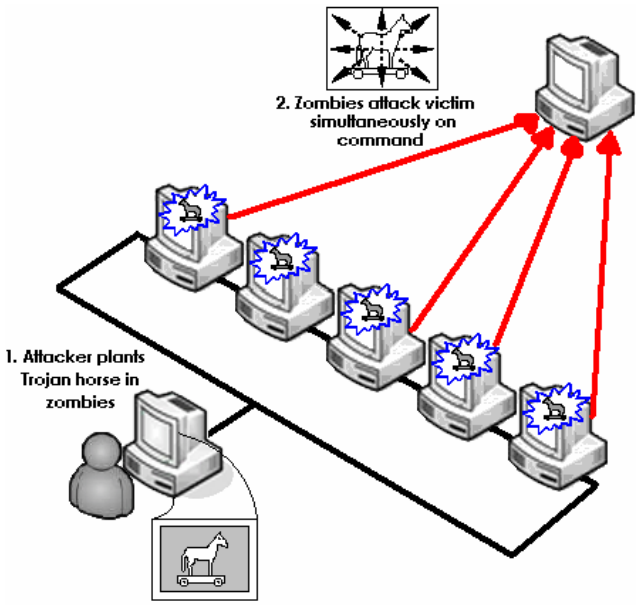
One of the server-side include commands is exec, to execute an arbitrary file on the server. For instance, the server-side include command <!- - #exec cmd="/usr/bin/telnet &"- -> will open a Telnet session from the server running in the name of the server. An attacker may find it interesting to execute commands such as

chmod (change access rights to an object), sh (establish a command shell), or cat (copy to a file).

2.2.11 Distributed Denial of Service:

Distributed denial-of-service (DDoS) attack, an attacker does two stages: plant a trojan horse (zombie), and lauch the attack.

In the first stage, the attacker uses any convenient attack to plant a Trojan horse on a target machine. That Trojan horse does not necessarily cause any harm to the target machine, so it may not be noticed. The target machine then becomes what is known as a zombie. The attacker repeats this process with many targets, the target systems carry out their normal work, unaware of the resident zombie.



At some point, as a second stage, the attacker chooses a victim and sends a signal to all the zombies to launch the attack. Then, instead of the victim's trying to defend against one denial-of-service attack from one malicious host, the victim must try to counter n attacks from the n zombies all acting at once. Not necessary all of the zombies need to use the same attack.

Some of the original DDoS tools include TFN (Tribal Flood Network), Trin00, and TFN2K (Tribal Flood Network).

2.2.12 Threats to Active or Mobile Code:

Active code or mobile code is a general name for code that is pushed to the client for execution. there are many different kinds of active code (cookies, scripts, active code, and auto exec by type):

a- **Cookies**: Cookies are not real active code, they are data files (which can store anything about a client that the browser can determine: keystrokes the user types, the machine name, connection details (such as IP address), date and type, and so forth) that can be stored and fetched by a remote server. However, cookies can be used to cause unexpected data transfer from a client to a server, so they have a role in a loss of confidentiality.

b- **Script:** Clients can invoke services by executing scripts on servers. Typically, a web browser displays a page, the browser organizes user inputs into parameters to a defined script; it then sends the script and parameters to a server to be executed. But all communication is done through HTML. The server cannot distinguish between commands generated from a user at a browser completing a web page, and a user's handcrafting a set of orders. The malicious user can monitor the communication between a browser and a server to see how changing a web page entry affects what the browser sends and then how the server reacts. With this knowledge, the malicious user can manipulate the server's actions.

A common scripting language for web servers, CGI (Common Gateway Interface), defines a machine-independent way to encode communicated data. Microsoft uses active server pages (ASP) as its scripting capability. Such pages instruct the browser on how to display files, maintain context, and interact with the server. These pages can also be viewed at the browser end, so

any programming weaknesses in the ASP code are available for inspection and attack.

c- **Active Code:** Displaying web pages started simply with a few steps: generate text, insert images, and register mouse clicks to fetch new pages. Soon, people wanted more elaborate action at their web sites: toddlers dancing atop the page, a three-dimensional rotating cube, images flashing on and off, colors changing, totals appearing. Some of these tricks, especially those involving movement, take significant computing power; they require a lot of time and communication to download from a server. But typically, the client has a capable and underutilized processor, so the timing issues are irrelevant. To take advantage of the processor's power, the server may download code to be executed on the client. This executable code is called active code. The two main kinds of active code are JavaScript and ActiveX controls.

1- **JavaScript:** Sun Microsystems designed and promoted Java as a truly machine-independent programming language. A Java program consists of Java bytecode executed on a Java virtual machine (JVM). The bytecode programs are machine independent, and only the JVM needs to be implemented on each class of machine to achieve program portability. The JVM contains a built-in security manager that enforces a security policy. A Java program runs in a Java "sandbox," a constrained resource domain from which the program cannot escape. The Java programming language is strongly typed, meaning that the content of a data item must be of the appropriate type for which it is to be used (for example, a text string cannot be used as a numeric). Although it is still difficult to

break its constraints, the Java sandbox contains many new toys, enabling more interesting computation but opening the door to exploitation of more serious vulnerabilities. A hostile applet is downloadable Java code that can cause harm on the client's system. Because an applet is not screened for safety when it is downloaded and because it typically runs with the privileges of its invoking user, a hostile applet can cause serious damage.

2- **ActiveX**: Microsoft's answer to Java technology is ActiveX. Using ActiveX, objects of arbitrary type can be downloaded to a client. If the client has a viewer or handler for the object's type, that viewer is invoked to present the object. For example, downloading a Microsoft Word .doc file would invoke Microsoft Word on a system on which it is installed. Files for which the client has no handler cause other code to be downloaded. Thus, in theory, an attacker could invent a type, called .bomb, and cause any unsuspecting user who downloaded a web page with a .bomb file also to download code that would execute .bombs.

d- **Auto Exec by Type:** Data files are processed by programs. For some products, the file type is implied by the file extension, such as .doc for a Word document, .pdf (Portable Document Format) for an Adobe Acrobat file, or .exe for an executable file. On many systems, when a file arrives with one of these extensions, the operating system automatically invokes the appropriate processor to handle it.

But this scheme presents an opportunity to an attacker. A malicious agent might send you a file named innocuous.doc,

which you would expect to be a Word document. Because of the .doc extension, Word would try to open it.

Generally, we recognize that executable files can be dangerous, text files are likely to be safe, and files with some active content, such as .doc files, fall in between. If a file has no apparent file type and will be opened by its built-in file handler, we are treading on dangerous ground. An attacker can disguise a malicious active file under a non-obvious file type.

2.2.13 Complex Attacks:

a- **Script Kiddies:** People who download and run attack scripts are called script kiddies. Attacks can be scripted. A simple denial-of-service attack is not hard to implement. But an underground establishment has written scripts for many of the popular attacks. With a script, attackers need not understand the nature of the attack nor even the concept of a network. The attackers merely download the attack script and execute it. The script takes care of selecting an appropriate (that is, vulnerable) victim and launching the attack.

b- **Building Blocks:** An attacker simply out to cause minor damage to a randomly selected site could use any of the techniques we have described. A dedicated attacker who targets one location can put together several pieces of an attack in order to compound the damage. For example, a wiretapping attack may yield reconnaissance information with which to form an ActiveX attack that transfers a Trojan horse that monitors for sensitive data in transmission. Putting the attack pieces together like building blocks expands the number of targets and increases the degree of damage.

2.3 Security Involving Programs:

Figure (5) provides an overall taxonomy of malicious programs, these threats can be divided into two categories, those that need a host program, and those that are independent.

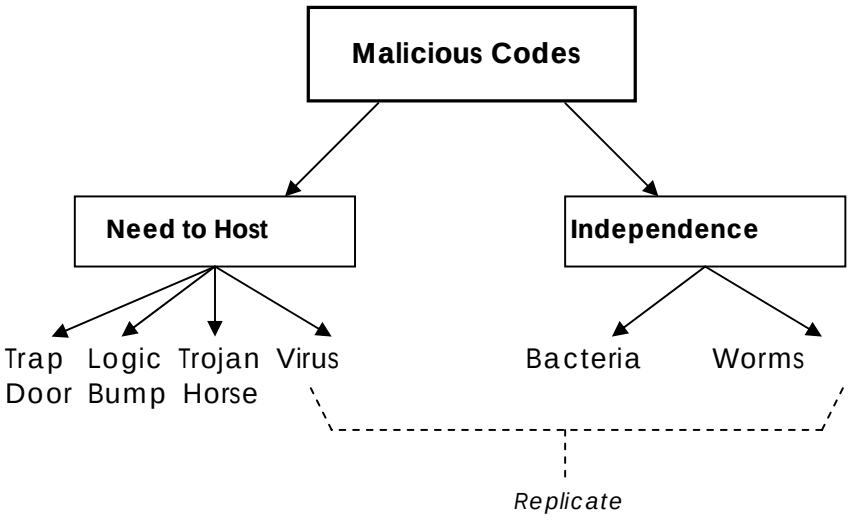


Figure (5): Taxonomy of Malicious Programs.

The malicious programs can cause two kinds of difficulties: programs can intercept or modify data, and programs can exploit service flaws in computing system or in network system, to allow system access by the intruders.

2.3.1 Information Access Problems:

a- **Trapdoors**: Trapdoor is a secret and undocumented entry point. The trapdoor is inserted sometime during code development (perhaps to assist test the module, perhaps to assist in the future modifications or enhancements, and perhaps forgets to remove them). But also trapdoor can allow access to the routine after it becomes an accepted production program, this mean that the trapdoor expose the system to modification during execution.

b- **Trojan Horse**: Trojan horse performs a hidden function in addition to its stated, obvious function. In other words, the Trojan horse is a destructive program that masquerades as a benign application. Unlike viruses, Trojan horses do not replicate themselves but they can be just as destructive.

c- **Salami Attack**: Programs that compute amounts of money may be subject to a salami attack, in this attack, a small amount of money is shaved from each computation, the amount shaved is so small that an individual case is unlikely to be noticed. However, accumulated amounts can add up.

d- **Programs that leak Information**: The programs that communicate their information to people who should not receive that information. A general name for these extraordinary paths of communication is 'covert channel'. Im many cases a programmer may want to develop a program that secretly communicates some of data on which it operates, the programmer creates what is called a covert channel, a hidden means for the program to communicate information.

2.3.2 Service Problems:

a- **Greedy Programs**: Programs that can block a computing system (infinite cycle or loop) so that no other computation can go on.

b- **Viruses**: A virus is a program that can infect other programs by modifying them (by include a copy of the virus program itself), so that the infected program then begins to act as a virus, infecting other programs.

c- **Bacteria**: Bacteria are programs that do not explicitly damage any files. Their sole purpose is to replicate themselves. Bacteria may do nothing more than execute two copies of itself simultaneously, or perhaps create two new files, both of those programs then may copy themselves twice, and so on. Bacteria

reproduce exponentially, eventually taking up all the processor capacity, memory, or disk space, denying users to those resources.

d- **Worms**: Network worms programs use network connections to spared from system to system. The worms can behave as a virus or bacteria or it could implant trojan horse programs or perform any number of disruptive actions. As with viruses, worms can be embedded in almost any other meaningful programs.

2.4 Trojan Horse Applications:

The Trojan horse applications discussed within this section are remote administration 'hacker' utilities that will allow a user to control another user's computer across the Internet using the client/server approach. (Trojan horse applications can provide control of a remote device equal than the person sitting at its keyboard).

- The Trojan Horse Application's Work:

In order to establishe the connection to another user's computer, the hacker running the 'client' portion which establishes a connection to the IP address of a known PC that has the 'server' portion installed upon it.

If the hacker running the 'client' portion and he doesn't know the IP address of the user's PC which has been compromised by the 'server' portion. The hacker usually initiates a series of connections to a large range of IP addresses on the Internet (known as 'scanning'), looking for any PC that responds back to the attempt. If a PC responds back, it responds with its IP address. Then all the hacker has to do, is to establish a connection to that IP address.

Keep in mind that 99% of the time, the hacker doesn't have a specific target to begin with, so any PC that answers back to their attempted connections satisfy their goal of hacking into another's PC.

Because the 'server' portion is configured to use (or 'listen' on) a particular port number, it's the client who attempts a connection to that specific port number to initiate the connection between computers.

Simply executing the 'server' portion of trojan, installs the software. To ease distribution, the 'server' portions can be attached ('piggy-backed') to any other windows executable which will run normally after installing the server portion.

- Trojans and the Port Numbers:

Some Trojans may use more than one port number. This is because one port is used for 'listening' and the other/s are used for the transfer of data. In their default configurations, the following trojans use:

Trojan Name	Protocol	Prot Number
Back Orifice	UDP	31337 or 31338
Deep Throat	UDP	2140 and 3150
NetBus	TCP	12345 and 12346
Whack-a-mole	TCP	12361 and 12362
NetBus 2 Pro	TCP	20034
GirlFriend	TCP	21544
Sockets de Troie	TCP	5000, 5001 or 50505
Masters Paradise	TCP	3129, 40421, 40422, 40423 and 40426

- Port Scan

The port scan supplies the attacker with very specific information for a particular IP address, reports which ports respond to messages and which of several known vulnerabilities seem to be present. For instance, an attacker can use one to find out that port 80 is open and supports HTTP.

Port scanning tells an attacker three things: which standard ports or services are running and responding on the target system, what operating system is installed on the target system, and what applications and versions of applications are present.

The famous port scanning tool is '**nmap scanner**' and anyone can download it. When given an IP address, nmap scanner will report all open ports, the service they support, and the owner (user ID) of the daemon providing the service.

Another available scanner is '**netcat**'. Commercial products are a little more costly, but not prohibitive. Well-known commercial scanners are '**Nessus scanner**', '**CyberCop scanner**', and '**Secure Scanner**' (Cisco).

- Examples of Trojan Horse Application:

There are very large range of trojan horse applications, some of these are listed below (*NetBus, NetBus Pro 2, Sockets de Troie, Deep Throat, Hack 'a' Tack, WinCrash, Devil, and Delta Source*):

***NetBus*:**

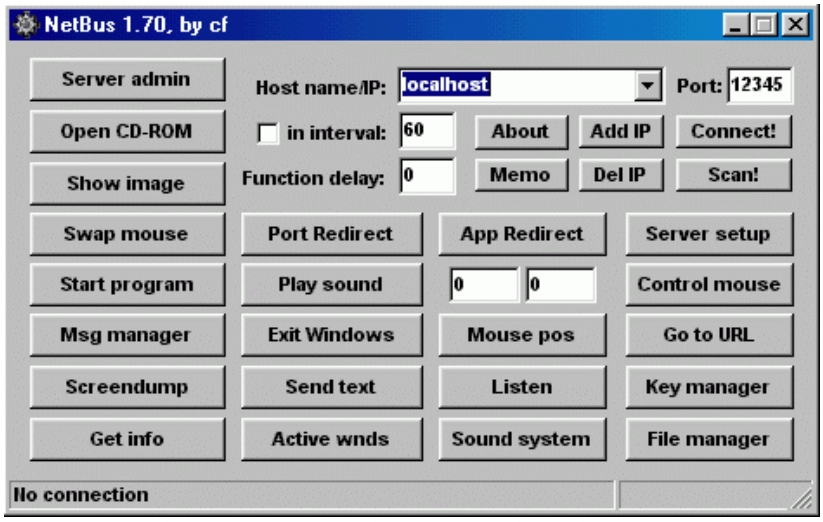
NetBus was written by Carl-Fredrik Neikter (cf@trancometer.se), a Swedish programmer. He states that it was created purely for fun.

NetBus currently affects Windows 95/98 PC's *and* Windows NT PC's.

The "server" portion (typically named "**patch.exe**") is approximately 470kb in size.

Ports **12345** and **12346** (by default) are used to establish its connection between the "client" and "server".

Once installed, it is rerun every time the computer is started by means of an entry under the "**HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run**" branch in the Registry.



Below are some of the functions that NetBus offers:

- Open/close the CD-ROM once or in intervals (specified in seconds).
- Show optional image. If no full path of the image is given it will look for it in the Patch-directory. The supported image-formats is BMP and JPG.

- Swap mouse buttons - the right mouse button gets the left mouse button's functions and vice versa.
- Start optional application.
- Play optional sound-file. If no full path of the sound-file is given it will look for it in the Patch-directory. The supported sound-format is WAV.
- Point the mouse to optional coordinates. You can even navigate the mouse on the target computer with your own!
- Show a message dialogue on the screen. The answer is always sent back to you!
- Shutdown the system, log off the user etc.
- Go to an optional URL within the default web-browser.
- Send keystrokes to the active application on the target computer! The text in the field "Message/text" will be inserted in the application that has focus. ("|" represents enter).
- Listen for keystrokes and send them back to you!
- Get a screen dump! (should not be used over slow connections)
- Return information about the target computer.
- Upload any file from you to the target computer! With this feature it will be possible to remotely update Patch with a new version.
- Increase and decrease the sound-volume.
- Record sounds that the microphone catch. The sound is sent back to you!
- Make click sounds every time a key is pressed!
- Download and deletion of any file from the target. You choose which file you wish to download/delete in a nice view that represents the hard disks on the target!
- Keys (letters) on the keyboard can be disabled.
- Password-protection management.
- Show, kill and focus windows on the system.

NetBus 2 Pro:

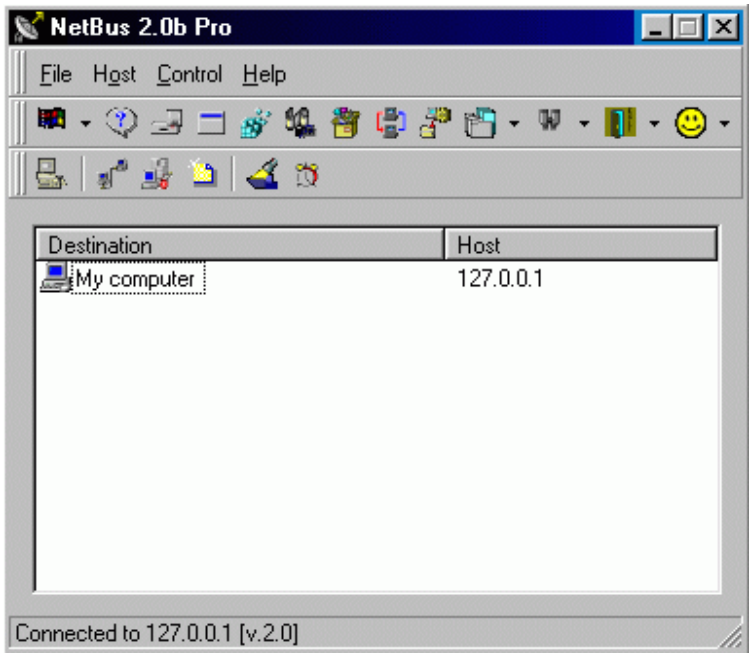
NetBus 2 Pro was written also by Carl-Fredrik Neikter (cf@trancometer.se), a Swedish programmer.

NetBus 2 Pro affects Windows 95/98 PC's *and* Windows NT PC's.

The "server" portion (typically named "**NBSvr.exe**") is approximately 599kb in size.

Port **20034** (by default) is used to establish its connection between the "client" and "server".

Once installed, it is rerun every time the computer is started by means of an entry under the
"**HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunServices**"
branch in the Registry.



Sockets de Troie (French for "Trojan Sockets"):

Sockets de Troie currently affects Windows 95/98 PC's.

The "server" portion is typically named "**mschv32.exe**".

Ports **5000** and **5001** (by default) are used to establish the connections between the "client" and "server".

There are two methods that Sockets de Troie can be performs: In the first methods, when the "server" portion is run, it shows an error dialog stating that SETUP32.DLL is missing. At the same time the "server" portion copies itself to WINDOWS\SYSTEM directory as MSCHV32.EXE and modifies the Windows Registry so it would be executed during every further Windows boot-up.

HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunLoad
MSchv32 Drv = C:\WINDOWS\SYSTEM\MSchv32.exe

In the second, when the "server" portion is run, it shows an error dialog stating that ISAPI32.DLL is missing. The "server" portion copies itself three times to the WINDOWS\ and WINDOWS\SYSTEM directories under the following names:
c:\windows\rsrcload.exe
c:\windows\system\mgadeskdll.exe
c:\windows\system\csmctrl32.exe

The virus also modifies Windows Registry to make these files be executed on every further Windows boot-up:

HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunLoad
Mgadeskdll = C:\WINDOWS\SYSTEM\Mgadeskdll.exe

HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunLoad
Rsrcload = C:\WINDOWS\Rsrcload.exe

HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServicesLoad
Csmctrl32 = C:\WINDOWS\SYSTEM\Csmctrl32.exe

***Deep Throat*:**

Deep Throat was written by an individual known as ^Cold^ KiLler, CEO of DarkLIGHT Corp.

Deep Throat currently affects Windows 95/98 PC's. It's rumored that the author is working on a Windows NT version.

The "server" portion (typically named "**systempatch.exe**") is approximately 255kb in size.

UDP Ports **2140** and **3150** are used to establish its connection between the "client" and "server".

Once installed, it is rerun every time the computer is started by means of an entry under the "**HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run**" branch in the Registry.

***Hack 'a' Tack*:**

Hack 'a' Tack was written by two persons calling themselves Da SuckA & The Bart33

Hack 'a' Tack currently affects Windows 95/98 PC's.

The server portion is named "**expl32.exe**" (236KB 5/16/99 2:49PM) and it will be found in the WINDOWS directory.

TCP ports **31785**, **31787** and UDP ports **31789**, **31791** (by default) are used to establish the connection between the "client" and "server".

Once installed, it is rerun every time the computer is started by means of an entry under the "**HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run**" branch in the Registry.

WinCrash:

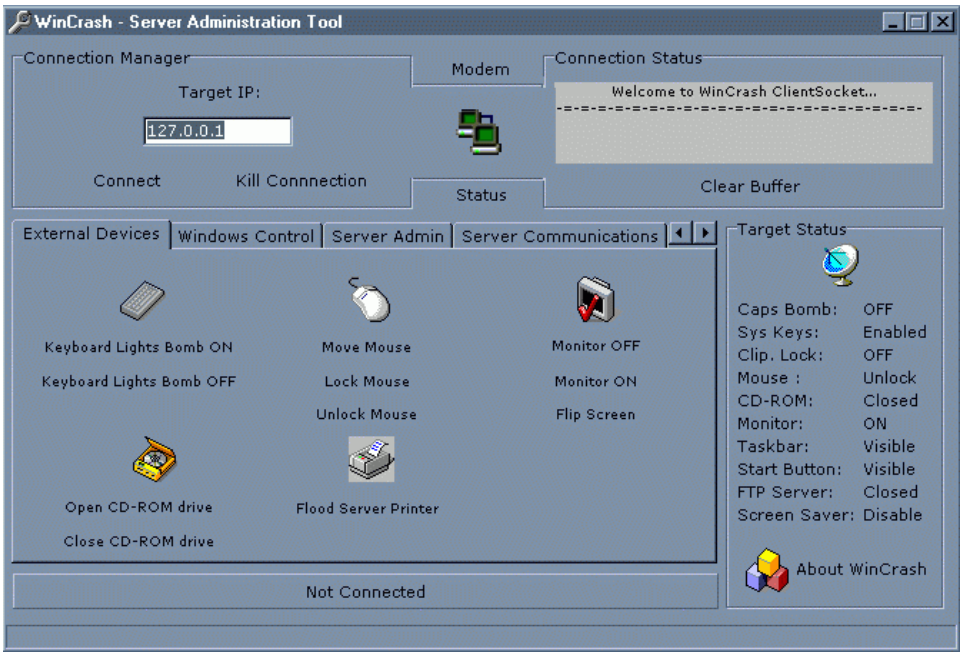
WinCrash was written by two indivudals known as Terminal Crasher and M@niac_Teen.

WinCrash currently affects Windows 95/98 PC's.

The "server" portion (typically named "**server.exe**") is approximately 290kb in size and can be found in the WINDOWS\SYSTEM directory.

TCP Port **5742** is used to establish the connection between the "client" and the "server".

Once installed, it is rerun every time the computer is started by means of an entry under the "**HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run**" branch in the Registry.



Devil:

Devil was written by an individual named JACK@$$.

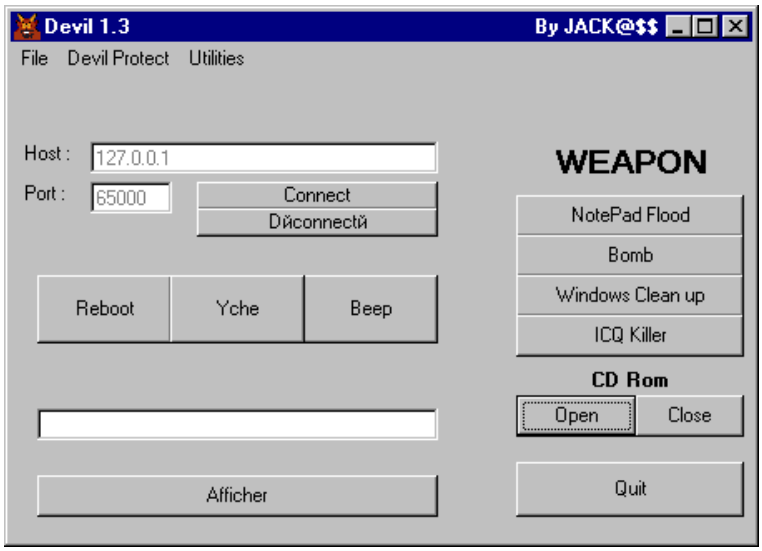
Devil currently affects Windows 95/98 PC's.

The "server" portion is named "**ICQFlood.exe**". It's approximately 25Kb in size and can usually be found in the either the WINDOWS or WINDOWS\SYSTEM directory.

Port **65000** (by default) is used to establish the connection between the "client" and "server".

Here are some of the functions that Devil offers:

- Open/Close CDROM
- Send "Beep" Signal
- Send text to Notepad
- Send Message "Yche! Yche!" with interval
- Send Applications Bomb
- Notepad Flooder
- Reboot
- Windows Clean Up
- ICQ Killer



***Delta Source*:**

Delta Source was written by an individual named DarkStar.

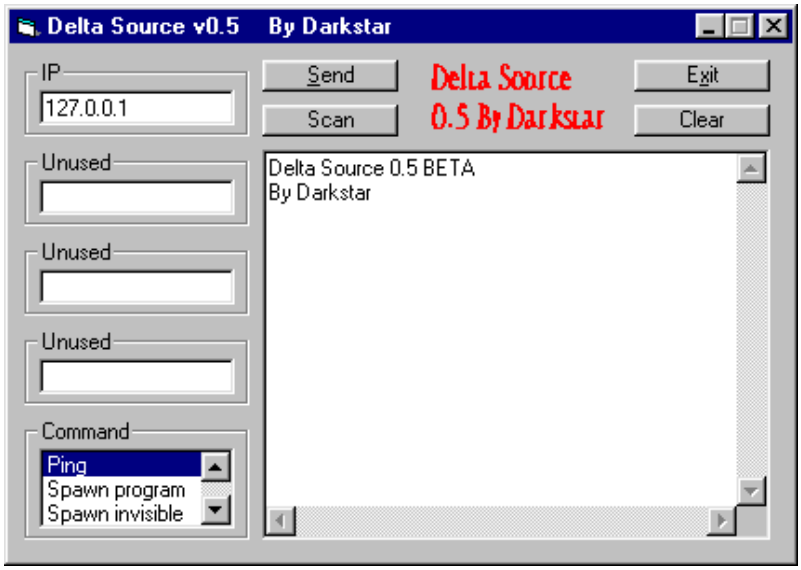
Delta Source currently affects Windows 95/98 PC's.

The "server" portion is named "**server.exe**". It's approximately 24Kb in size and can usually be found in the either the WINDOWS or WINDOWS\SYSTEM directory.

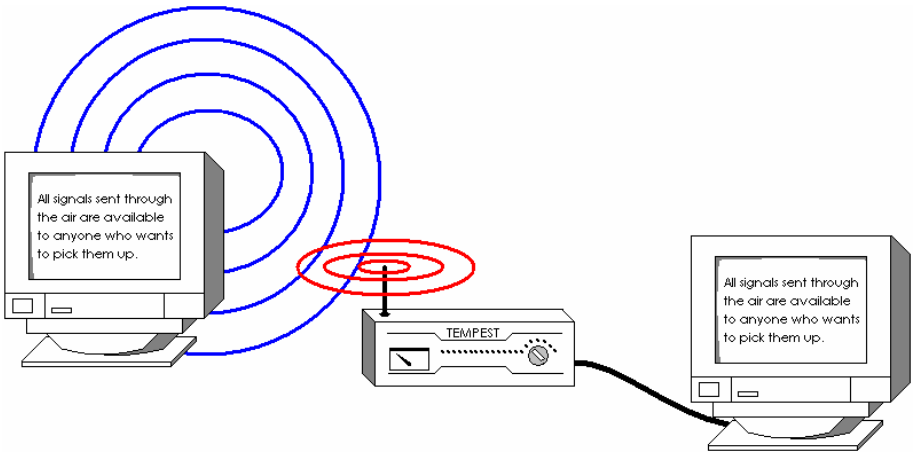
Ports **26274** and **47262** (by default) are used to establish the connection between the "client" and "server".

Here are some of the functions that Delta Source offers:

- Ping
- Spawn program
- Spawn invisible
- Delete file
- Program list
- Program kill
- Send To URL
- MsgBox
- Mouse Freeze/Unfreeze
- Mouse Swap/Unswap
- Taskbar hide/show
- Server Info
- Reboot



TEMPEST


All signals sent through the air are available to anyone who wants to pick them up.
TEMPEST
All signals sent through the air are available to anyone who wants to pick them up.

Chapter

3

Network Security Essentials

University of Technology
Computer Science Department
Network Security
4th Class
Lecturer: Dr. Mazin Sammer

Chapter Three

Network Security Controls

3.1 Introduction.
3.2 Encryption:
- 3.2.1 Encryption Methods:
  - A. Link Encryption Method.
  - B. End-to-End Encryption Method.
  - C. Comparison of Encryption Methods.
- 3.2.2 Virtual Private Networks
- 3.2.3 PKI and Certificates
- 3.2.4 SSH Encryption
- 3.2.5 SSL Encryption
- 3.2.6 IPSec
- 3.2.7 Signed Code
- 3.2.8 Key Distribution.
  - Secure Key Distribution Protocol.

- Key Server.
- Secure Cryptographic Facility.

3.3 Port Protection:

3.3.1 Automatic Call-Back.

3.3.2 Differential Access Rights.

3.3.3 Silent Modem.

3.4 Authentication:

3.4.1 User Authentication.

A: Password.

B: Token or Smart Card.

C: Personal Characteristics.

3.4.2 Non-Human Authentication.

A: Kerberos.

B: DCE.

C: SESAME.

D: CORBA.

E: Windows 2000 Authentication.

3.5 Traffic Control:

3.5.1 Pad Traffic.

3.5.2 Routing Control.

3.6 Data Integrity:

3.6.1 Protocols.

3.6.2 Checksums.

3.6.3 Digital Signature.

3.1 Introduction:

Computer security has the same characteristics. We have many controls at our disposal. Some are easier than others to use or implement. Some are cheaper than others to use or implement. And some are more difficult than others for intruders to override. Figure (1) illustrates how we use a combination of controls to secure our valuable resources. We use one or more controls, according to what we are protecting, how the cost of protection compares with the risk of loss, and how hard we think intruders will work to get what they want.

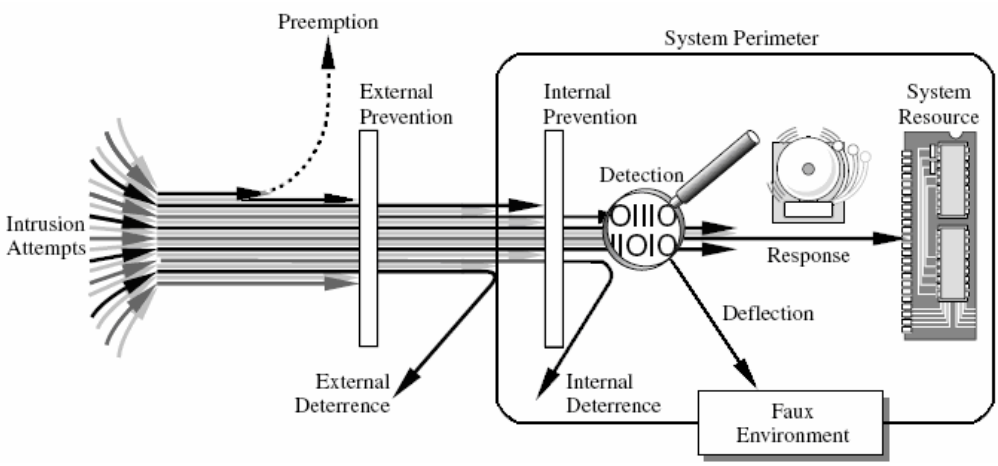


Figure (1): Multiple Controls.

There are several network security controls which can protect the network.

3.2 Encryption:

Encryption is a very powerful tool for providing privacy, authenticity, integerity, and limited access to data. Network secure data with encryption, perhaps in combination with other security controls.

3.2.1 Encryption Methods:

In network, encryption can be appiled either between two hosts or between two applications, so there are two forms of encryption 'link encryption' and 'end-to-end encryption'.

A. Link Encryption Method:

In link encryption, data is encrypted just before it is placed on the physical communication link. In this case, encryption occurs at layer 1 or layer 2 in the OSI model. Decryption occures just as the communication enters the receiving computer.

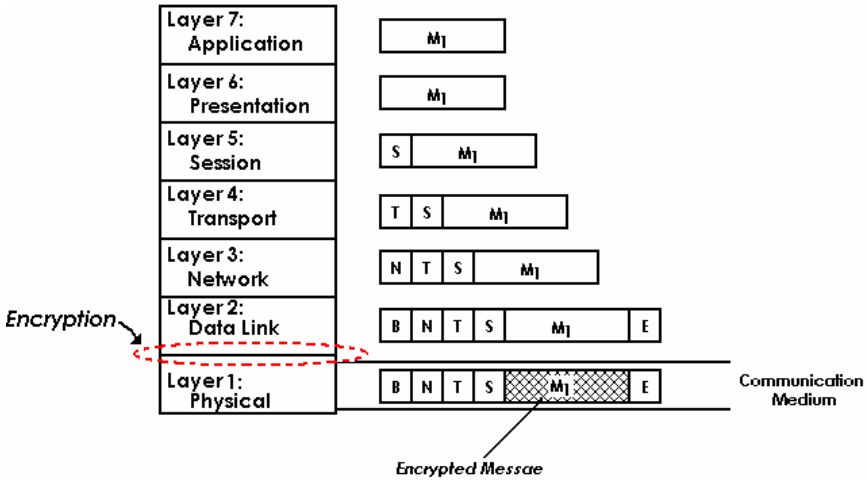


Figure (2): Link Encryption.

Encryption protects the message as it is in transit between two computers, but the message is in plaintext inside the host, as shown in figure (3). Link encryption is ecpecially vulnerable when a communication must pass through one or more additional hosts betweensender and receiver.

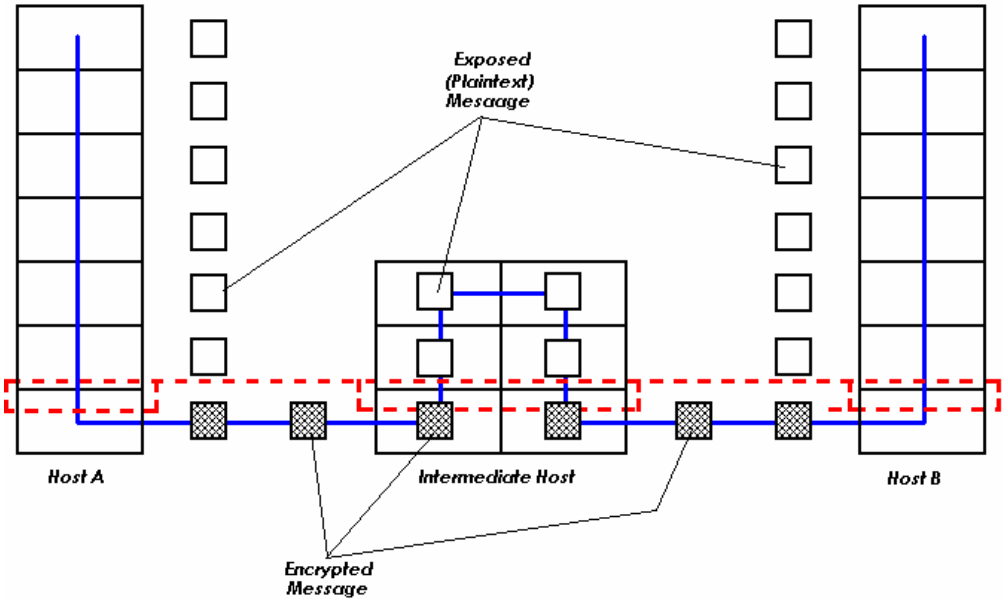


Figure (3): Link Encryption with Intermidate Host.

Link encryption is especially appropriate where the transmission line is the point of greatst vulnerability. If all hosts on a netwrok are reasonably secure, but the communicationss medium is shared with other users or is not secure, link encryption is an easy control to use.

B. End-to-End Encryption Method:

In link encryption, data is encrypted at the higher layers, either at layer 7 or perhaps at layer 6. end-to-end encyption provides security from one end of transmission through the other.

The encryption can be applied by a hardware device between the user and the host, in other case, the encryption can be done by software running on the host computer.

Since the encryption precede all routin and transmission processing of the layer, the message is transmitted in encrypted from throughout the network. The encryption covers potential flaws in lower layers in the transfer model.

Message sent through several hosts are protected, the data content of the message is still encrypted, as shown in figure (4), therefore, the encrypted message must pass through insecure nodes.

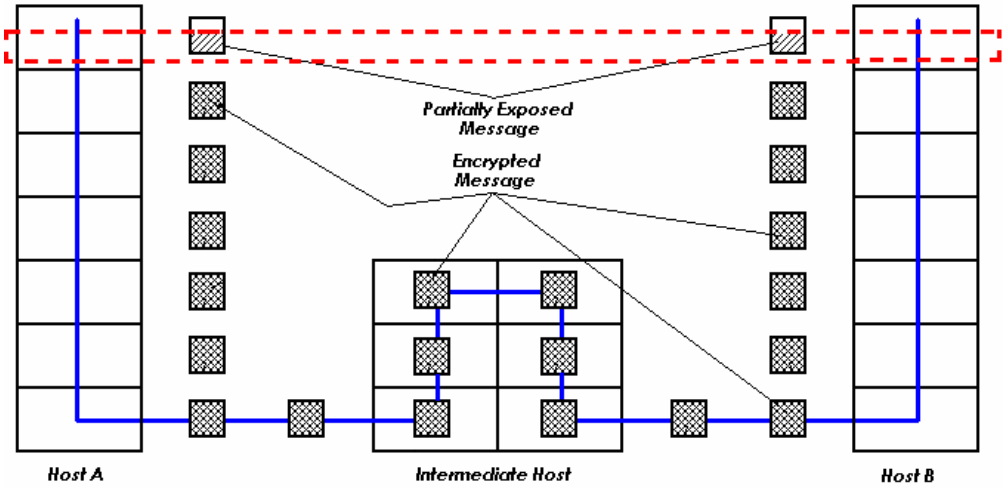


Figure (4): Encrypted Message Passing through a Host.

C. Comparison of Encryption Methods:

Link Encryption	End-to-End Encryption
Message exposed in sending host	Message encrypted in sending host
Message exposed in intermediate nodes	Message encrypted in intermediate nodes
Encryption applied by sending host (encryption invisible to user)	Encryption applied by sending process (user applies encryption)
Host maintntains encryption (one facility for all users)	User must use algorithm (user select encryption)
Encryption can be done by H/W	Encryption done by S/W
All or no message encrypted	User chooses to encrypt or not, for each message
Requires one key per host pair	Requires one key per user pair
The number of keys required is n*(n-1)/2 for n nodes	The number of keys required is n*(n-1)/2 for n users
Provides node authentication	Provides user authentication

3.2.2 Virtual Private Networks:

Link encryption can be used to give a network's users the sense that they are on a private network, even when it is part of a public network. For this reason, the approach is called a virtual private network (or VPN).

Typically, physical security and administrative security are strong enough to protect transmission inside the perimeter of a network. Thus, the greatest exposure for a user is between the user's workstation or client and the perimeter of the host network or server.

A firewall is an access control device that sits between two networks or two network segments. It filters all traffic between the protected (inside) network and a less trustworthy (outside) network or segment. (We examine firewalls in detail later in this chapter). Many firewalls can be used to implement a VPN. When a user first establishes a communication with the firewall, the user can request a VPN session with the firewall. The user's client and the firewall negotiate a session encryption key, and the firewall and the client subsequently use that key to encrypt all traffic between the two. Establishment of a VPN is shown in Figure (5).

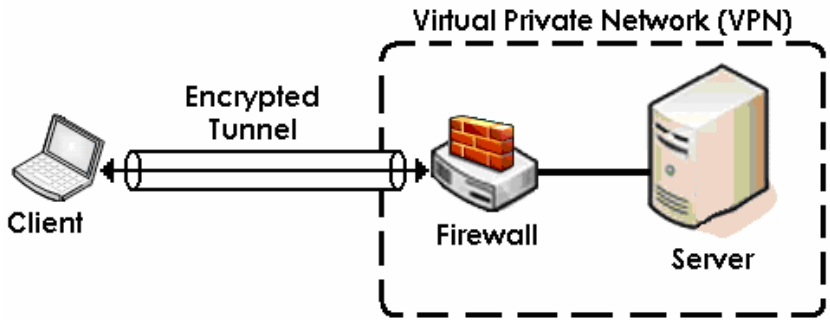


Figure (5): Virtual Private Network.

3.2.3 PKI and Certificates:

A public key infrastructure (PKI) is a process created to enable users to implement public key cryptography, usually in a large (and frequently, distributed) setting. PKI offers each user a set of services, related to identification and access control, as follows:

- Create certificates associating a user's identity with a (public) cryptographic key.
- Give out certificates from its database.
- Sign certificates, adding its credibility to the authenticity of the certificate.
- Confirm (or deny) that a certificate is valid.
- Invalidate certificates for users who no longer are allowed access or whose private key has been exposed.

3.2.4 SSH Encryption:

SSH (secure shell) is a pair of protocols (versions 1 and 2). SSH protects against spoofing attacks and modification of data in communication. The SSH protocol involves negotiation between local and remote sites for encryption algorithm (for example, DES, IDEA, AES) and authentication (including public key and Kerberos).

3.2.5 SSL Encryption:

The SSL (Secure Sockets Layer) protocol was originally designed to protect communication between a web browser and server. It is also known now as TLS, for transport layer security. SSL interfaces between applications (such as browsers) and the TCP/IP protocols to provide server authentication and an encrypted communications channel between client and server. Client and server negotiate a mutually supported suite of encryption for session encryption and hashing;

possibilities include triple DES and SHA1, or RC4 with a 128-bit key and MD5.

To use SSL, the client requests an SSL session. The server responds with its public key certificate so that the client can determine the authenticity of the server. The client returns part of a symmetric session key encrypted under the server's public key. Both the server and client compute the session key, and then they switch to encrypted communication, using the shared session key. The protocol is simple but effective, and it is the most widely used secure communication protocol on the Internet.

3.2.6 IPSec:

The address space for the Internet is running out. As domain names and equipment proliferate, the original, 32-bit address structure of the Internet is filling up. A new structure, called **IPv6** (version 6 of the IP protocol suite), solves the addressing problem.

As a part of the IPv6 suite, the **IPSec** (**IP Security Protocol** Suite) was designed to address fundamental shortcomings such as being subject to spoofing, eavesdropping, and session hijacking, the IPSec protocol defines a standard means for handling encrypted data. IPSec is implemented at the IP layer, so it affects all layers above it, in particular TCP and UDP. Therefore, IPSec requires no change to the existing large number of TCP and UDP protocols.

IPSec is somewhat similar to SSL, in that it supports authentication and confidentiality in a way that does not necessitate significant change either above it (in applications) or below it (in the TCP protocols). Like SSL, it was designed to be independent of specific

cryptographic protocols and to allow the two communicating parties to agree on a mutually supported set of protocols.

The basis of IPSec is what is called a **security association**, which is essentially the set of security parameters for a secured communication channel. It is roughly comparable to an SSL session. The fundamental data structures of IPSec are the **AH** (authentication header) and the **ESP** (encapsulated security payload). The ESP replaces (includes) the conventional TCP header and data portion of a packet, as shown in Figure (6). The physical header and trailer depend on the data link and physical layer communications medium, such as Ethernet.

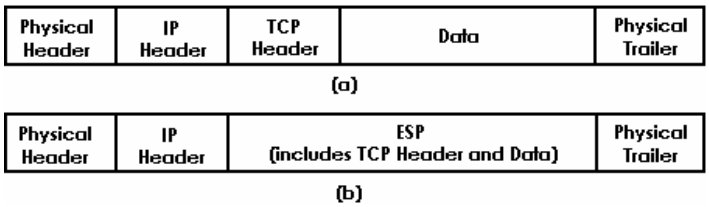


Figure (6) Packets: (a) Conventional Packet; (b) IPSec Packet.

While the ESP contains both an authenticated portion and an encrypted portion, as shown in Figure (7).

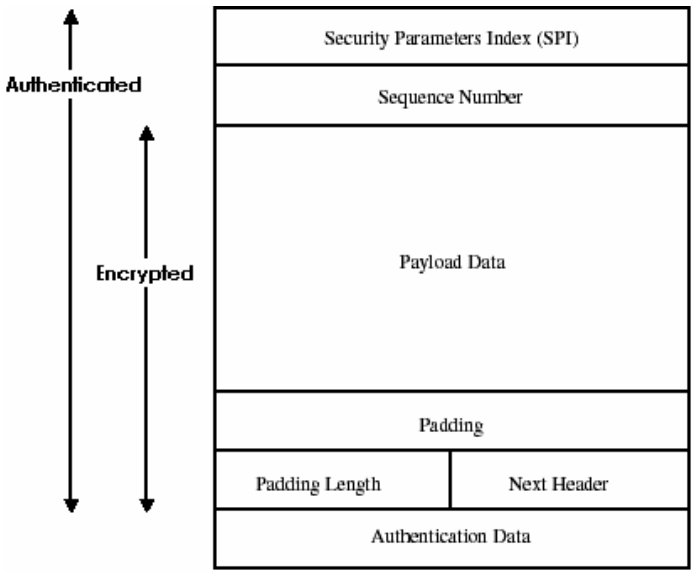


Figure (7) Encapsulated Security Packet.

IPSec can be used to establish cryptographic sessions with many purposes, including VPNs, applications, and lower-level network management (such as routing).

3.2.7 Signed Code:

As we have seen, someone can place malicious active code on a web site to be downloaded by unsuspecting users. Running with the privilege of whoever downloads it, such active code can do serious damage, from deleting files to sending e-mail messages to fetching Trojan horses to performing subtle and hard-to-detect mischief. Today's trend is to allow applications and updates to be downloaded from central sites, so the risk of downloading something malicious is growing.

A partial—not complete—approach to reducing this risk is to use signed code. A trustworthy third party appends a digital signature to a piece of code, supposedly connoting more trustworthy code. A signature structure in a PKI helps to validate the signature.

3.2.8 Key Distribution:

Key distribution is always a problem with encryption in network. Keys must be delivered to the sender and receiver in secure manner. So the problem is how to provide a secure method to distribute keys between the sender and receiver.

- Secure Key Distribution Protocol:

A key distribution method uses one key (called master key) to distribute other keys (called session keys). This technique works for any secret key cryptosystem (*although the original technique deals with distribution keys for the DES*).

The first key is called a master key and the others are called session keys. The master key is used only to encrypt session keys.

- Key Server:

A network key-Server is a process that distributes keys to users on request. The key-Server shares a unique key with each user.

Suppose that A and B want to communicate, as shown in figure(8). The session begin between A and B when A calls the key-Server. The key-Server generates a new key (K_S) (it's a secure key), and it sends to A, and it sends to B. A and B encrypt M_A and M_B ($M_A = E_{S_A}(K_S)$, and $M_B = E_{S_B}(K_S)$). A and B decrypt M_A and M_B both obtain K_S, and they transact their session using K_S. At the end of session, both A and B destroy the key.

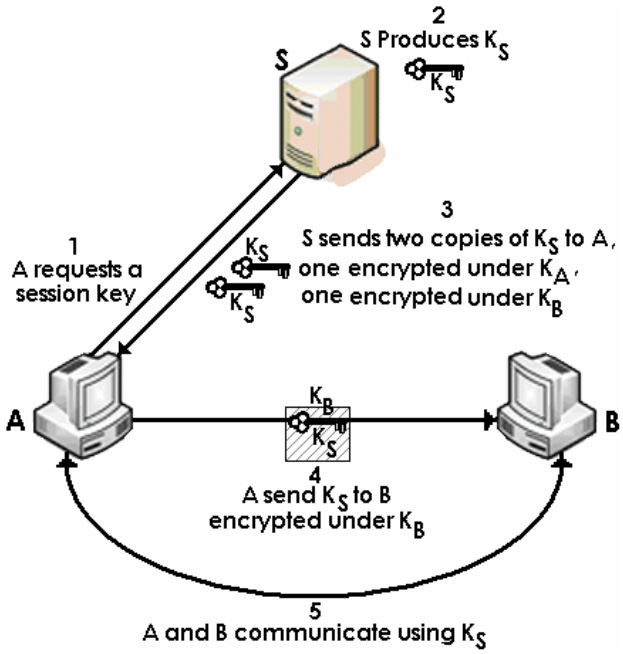


Figure (8): Session Keys with a Key Generator.

This facility can be used to provide end-to-end encryption without the massive number of keys needed. With a secure key generator, each user needs only one key by which to communicate with the key generator.

- Secure Cryptographic Facility:

An extension of the concept of a key-server is a secure cryptographic facility, it is a hardware device that performs secure encryption and decryption for users. A master key is permanently installed in the device. Operations the device can perform are encipher or decipher under the master key, accept a new working key, and encipher or decipher under the working key.

The user provides a session key that that has been enciphered under the master key, this session key becomes the working key. In this way, the session key is not in plaintext, it is communicated only under the encryption of the master key. The session key is deciphered only inside the secure facility.

A block diagram of such a facility (secure cryptographic facility) is shown in figure (9).

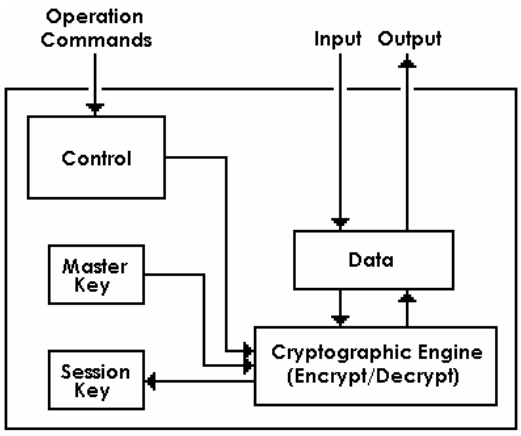


Figure (9): Secure Cryptographic Facility.

3.3 Port Protection:

A serious vulnerability to a network system is dail-in port access. User authentication is difficult in a single computing system, but it becomes far more difficult when users can dial in from network. Port protection is accomplished by several administrative and hardware techniques.

3.3.1 Automatic Call-Back:

With an automatic call-back system, an authorized user dails a computer system. After the user identifies him, the computer breaks the communication line. The computer then consults an internal table of telephone numbers and calls the user back at a predetermind number. Clearly, the table of telephone numbers must be well-protected against modification.

3.3.2 Differential Access Rights:

The sensitive data can be protected by limiting the places from which access is allowed. People can be allowed to access the most sensitive data only from secure places; even though the individuals are trusted for more sensitive access, the access path is not trusted.

For example, when dialing in, people might be allowed to obtain only less sensitive information. In this way, sensitive accesses must be made at the site, where it is more difficult to compromise data, or where it would be more noticeable if one were beign forced to reveal the data. On a network, users with access to sensitive objects can do so only by direct-connection, not through network-host. This restriction reduces the threat of malicious hosts in a network.

3.3.3 Silent Modem:

Typically, a computerreceiving an incoming call establishes the connection by sending a modem signal. However, it can also wait silently until the caller's modem sends the first tone. In this way, the computer does not reveal itself as a computer until the caller has revealed that it is a computer.

3.4 Authentication:

3.4.1 User Authentication:

Authentication mechanisms are divided into three categories: what you know (such as a password), what you possess (such as a token or a capability), and something about you (such as a picture or a fingerprint).

A: Password:

Password are very often used for authentication because they are easy to use and they provide reasonable assurance. A good password has the following characteristics:

1- Composed of letters, digits, and other chracters.
2- Long.
3- Not a common word or name.
4- Unlikley.
5- Frequently changed.
6- Not written down.

Some time, all users from a particular location or for a particular application use one password (this called a group password). For example, some systems provide a demonstration account for anyone to use to obtain an introduction to the use of the system. In another

example, some network system have one access password for all incomming network hosts.

Some time, each user from a particular location or for a particular application use one-time password (it is good for one use only), where the user and host both have access to identical lists of passwords. The user would enter the first password for the first login, the next one for the next login, and so forth. As long as the password lists remained secret and as long as no one could guess one password from another. The wiretap threat implies that a password could be intercepted from a user who enters a password across an unsecured network. A one-time password can guard against wiretapping and spoofing of a remote host.

B: Token or Smart Card:

A token is the general name for an object that authenticates its possessor. For example, royalty used to be authenticated by a signet ring, and in many applications people are authenticated by ID cards.

The 'magnetic stripe credit card' is one form of token for network communication. These cards are regular credit cards with certain information recorded in magnetic form on the back. The magnetic stripe is read by a sensing machine. These cards are not used as complete proof, since such a card might be lost or stolen, a user also has enter an identifying word or number in order to use the card.

A 'smart card' or 'chip-card' is a similar to 'magnetic stripe credit card' except it has a microprocessorembedded. The smart card can actuallt performe computation, such as performing link level encryption, of the people can carry on a secure session wth a computing network from any place in the word.

C: Personal Characteristics:

The physical characteristics of people (such as finguerprints, pronunciation, retina of eyes) can be recognize using scanners. These devices provide highly reliable assurance of authenticity. Furthermore, finguerprints or retina of eyes cannot be lost or stolen.

3.4.2 Non-Human Authentication:

Digital Equipment Corp. created a simple architecutre for authentication in distributed systems. This arhitecture was effective against the following threats:

1- Impersonation of a server.
2- Interception or modification of data exchanged between servers.
3- Replay of a previous authentication.

The architecutre assums that each server has its own private key, and that the corresponding publick key is available to or held by every other process that might need to establish an authenticated channel.

A: Kerberos (□□□□□□□□ □□ □□□□□□□ □□ □□□):

Kerberos is a network authentication protocol. Kerberos is designed to provide strong authentication for client/server applications by using secret-key cryptography. Figure (10) illustrates the process.

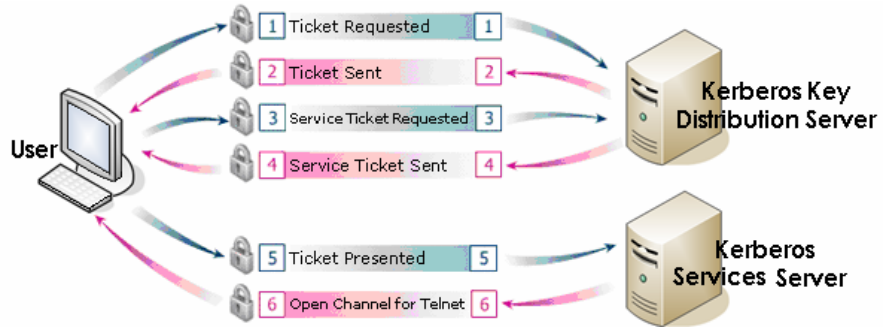


Figure (10): The Kerberos Process.

B: DCE:

DCE (Digital Equipment Corp.) presents a complete support authentication environment for building distributed applications. DCE manages controlled, shared access to remote and distributed resources. DCE is built on Kerberos technology. Instead of a separate user authentication server and ticket-granting server, DCE merges these into one security server.

C: SESAME:

SESAME is quite similar to DCE, its built on Kerberos technology, and its include a privilege attribute service.

D: CORBA:

CORBA (Common Object Request Broker Architecture) is a popular approach to distributed applications. CORBA handles a client's request of a server to perform some action on an object.

E: Windows 2000 Authentication:

Microsoft has implemented the Kerberos V5 authentication service to integrate with other Windows 2000 security services, running on the domain controller and uses the domain's active directory as its security account database.

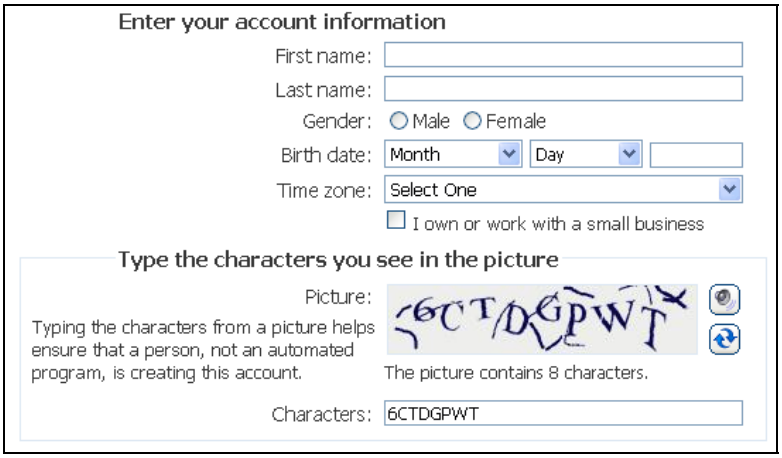
3.4.3 Case Study (Preventing Authentication Hacking Attacks):

Authentication plays a critical role in the security of web applications. When a user provides his login name and password to authenticate and prove his identity, the web application assigns the user specific privileges to the system, based on the identity established by the supplied credentials.

An important measure in stopping authentication hacking attacks is by adding random content on the page presented to the authenticating client browser. The client must be capable of successfully submitting this random content as part of the authentication process to proceed further in the web site or application.

The best way to do this is to present the random phrase in a graphic GIF, JPG or PNG format using random fonts or colours each time.

This can make it almost impossible for an automated authentication attacks process to succeed.



3.5 Traffic Control:

3.5.1 Pad Traffic:

If the attacker can detect an exceptional volume of traffic between two points, the attacker may infer the location of an event about to occur.

The countermeasure to traffic flow threats is to disguise the traffic flow. If traffic between A and B is encrypted so that the attacker can detect only the number of packets flowing, A and B can agree to pass recognizable (to them) but meaningless encrypted traffic. When A has much to communicate to B, there will be few meaningless packets; when communication is light, A will **pad the traffic** stream with many spurious packets.

3.5.2 Routing Control:

Consider a message that is covered in multiple layers, like the layers of an onion. A wants to send a message to B but doesn't want anyone in or intercepting traffic on the network to know A is communicating with B. So A takes the message to B, wraps it in a package for D to send to B. Then, A wraps that package in another package for C to send to D. Finally, A sends this package to C. The internal wrappings are all encrypted under a key appropriate for the intermediate recipient.

Receiving the package, C knows it came from A, although C does not know if A is the originator or an intermediate point. C then unwraps the outer layer and sees it should be sent to D. At this point, C cannot know if D is the final recipient or merely an intermediary. C sends the message to D, who unwraps the next layer. D knows neither where the package originally came from nor where its final destination is. D forwards the package to B, its ultimate recipient.

3.6 Data Integrity:

3.6.1 Error Correcting Codes:

We can use **error detection** and **error correction codes** to guard against modification in a transmission. The codes work as error detection codes detect when an error has occurred, and error correction codes can actually correct errors without requiring retransmission of the original message. The error code is transmitted along with the original data, so the recipient can recompute the error code and check whether the received result matches the expected value.

The simplest error detection code is a **parity check**. An extra bit is added to an existing group of data bits depending on their sum or an exclusive OR. There are other kinds of error detection codes, such as **hash codes** and **Huffman codes**.

Parity and simple error detection and correction codes are used to detect nonmalicious changes in situations in which there may be faulty transmission equipment, communications noise and interference, or other sources of spurious changes to data.

The two kinds of parity are called even and odd:

Even parity the extra bit is **0** if the sum of the data bits is even and **1** if the sum is odd; that is, the parity bit is set so that the sum of all data bits plus the parity bit is even.

Odd parity is the same except the sum is odd. For example, the data stream 01101101 would have an even parity bit of 1 (and an odd parity bit of 0) because 0+1+1+0+1+1+0+1 = 5 + 1 = 6 (or 5 + 0 = 5 for odd parity).

A parity bit can reveal the modification of a single bit. However, parity does not detect two-bit errors—cases in which two bits in a group are changed.

3.6.2 Cryptographic Checksum:

Malicious modification must be handled in a way that prevents the attacker from modifying the error detection mechanism as well as the data bits themselves. One way to do this is to use a technique that shrinks and transforms the data, according to the value of the data bits.

A **cryptographic checksum** (sometimes called a **message digest**) is a cryptographic function that produces a checksum. The cryptography prevents the attacker from changing the data block (the plaintext) and also changing the checksum value (the ciphertext) to match.

3.6.3 Digital Signature:

A digital signature is a means to certify the authenticity of a set of data (document, message, file, or something else). The person affixing the digital signature confirms that the data is authentic.

Chapter

4

Network Security Essentials

University of Technology
Computer Science Department
Network Security
4th Class
Lecturer: Dr. Mazin Sammer

Chapter Four

Network Security Solutions

- How does it work?
- What Windows Firewall does and does not do

4.3 Intrusion Detection Systems
- 4.3.1 Introduction
- 4.3.2 Types of IDSs
  - A: Signature-Based Intrusion Detection
  - B: Heuristic Intrusion Detection
- 4.3.3 Stealth Mode
- 4.3.4 Goals for Intrusion Detection Systems
- 4.3.5 Responding to Alarms
- 4.3.6 False Results

4.4 Secure E-Mail
- 4.4.1 Security for E-Mail
- 4.4.2 Threats to E-Mail
- 4.4.3 Requirements and Solutions
- 4.4.4 Confidentiality
- 4.4.5 Other Security Features
- 4.4.6 Example Secure E-Mail Systems
  - A: PGP
  - B: S/MIME

4.5 Multilevel Security on Networks
- 4.5.1 Trusted Network Interface
- 4.5.2 Secure Communication

4.1 Kerberos Authentication System:

The name 'Kerberos' comes from Greek mythology in which a three-headed dog guards the gates to Hades (Hades is the home of the dead beneath the earth, otherwise known as hell).

Kerberos authenticaion system is a system that supports authentication in distributed systems (i.e. in the client-server system). Originally designed to work with secret key encryption, but in the latest version, it uses public key technology to support key exchange.

Kerberos is ,based on the idea that, a central server provides authenticated tokens, called **tickets**, to requesting applications.

1. A ticket is an unforgeable, nonreplayable, authenticated object.
2. A ticket is an encrypted data structure naming a user and a service that user is allowed to obtain.
3. A ticket also contains a user's authenticated identity, an identification of file, the access rights (for example, to read), a session key for the file server to use this file, and an expiration date for the ticket.

Kerberos is a complete solution, and all applications must use Kerberos authentication and access control. But, currently a few applications use Kerberos authentication, so integration of Kerberos into an existing environment requires modification of existing applications, which is not feasible.

4.1.1 Kerberos System's Work:

The first step in using Kerberos system is to establish a session with the Kerberos server, as shown in Figure (1). A user's workstation sends the user's identity to the Kerberos server when a user logs in. The Kerberos server verifies that the user is authorized. The Kerberos server sends two messages:

1. to the user's workstation, a session key (*SG*) (for use in communication with the ticket-granting server), and a ticket (*TG*) (for the ticket-granting server); session key (*SG*) is encrypted under the user's password (E(SG + TG, pw)).
2. to the ticket-granting server, a copy of the session key (*SG*), and the identity of the user (encrypted under a key shared between the Kerberos server and the ticket-granting server).

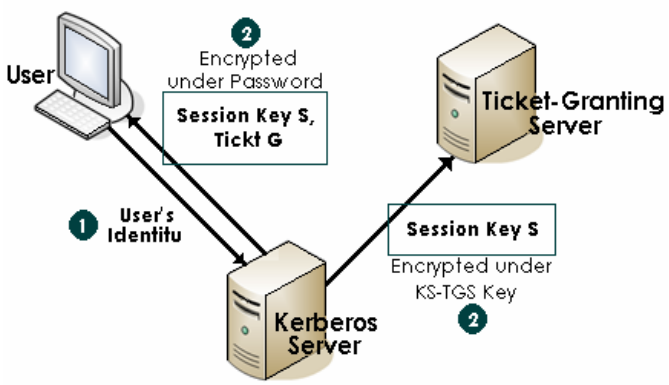


Figure (1): Initiating a Kerberos Session.

Next, the user will want to exercise some other services of the distributed system (such as accessing a file). Using the key SG, the user requests a ticket to access file from the ticket-granting server, as shown in Figure (2). After the ticket-granting server verifies User's access permission, it returns a ticket and a session key.

The ticket is encrypted under a key shared exclusively between the ticket-granting server and the file server. This ticket cannot be read, modified, or forged by the user (or anyone else). The ticket-granting server must, also provide user with a copy of SF (session key for the file server).

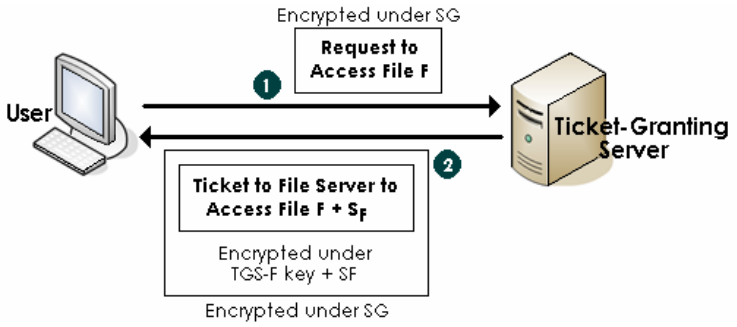


Figure (2): Obtaining a Ticket to Access a File.

4.1.2 Kerberos is withstanding Attacks:

Kerberos was carefully designed to withstand attacks in distributed environments because:

- *No passwords communicated on the network*. As already described, a user's password is stored only at the Kerberos server. The user's password is not sent from the user's workstation when the user initiates a session.
- *Cryptographic protection against spoofing*. Each access request is mediated by the ticket-granting server (which knows the identity of the requester) based on the authentication (which performed initially by the Kerberos server) and based on the fact that the user was able to present a request encrypted under a key (that had been encrypted under the user's password).

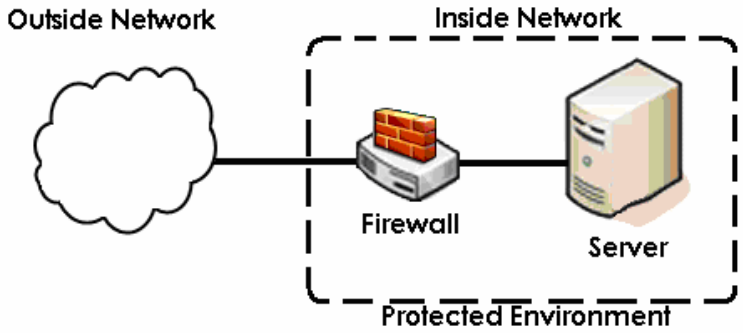
- *Limited period of validity*. Each ticket is issued for a limited time period; the ticket contains a timestamp with which a receiving server will determine the ticket's validity.

  In this way, certain long-term attacks will usually be neutralized because the attacker will not have time to complete the attack.
- *Timestamps to prevent replay attacks*. Kerberos requires reliable access to a universal clock. Each user's request to a server is stamped with the time of the request. A server receiving a request will compare this time to the current time and fulfill the request only if the time is reasonably close to the current time. This time-checking prevents most replay attacks, since the attacker's presentation of the ticket will be delayed too long.
- *Mutual authentication*. The user of a service can be assured of any server's authenticity by requesting an authenticating response from the server. The user sends a ticket to a server and then sends the server a request encrypted under the session key for that server's service; the ticket and the session key were provided by the ticket-granting server. The server can decrypt the ticket only if it has the unique key it shares with the ticket-granting server. Inside the ticket is the session key, which is the only means the server has of decrypting the user's request. If the server can return to the user a message encrypted under this same session key but containing 1 + the user's timestamp, the server must be authentic. Because of this mutual authentication, a server can provide a unique channel to a user and the user may not need to encrypt communications on that channel to ensure continuous authenticity. Avoiding encryption saves time in the communication.

4.2 Firewalls:

4.2.1 Firewall Definition:

A firewall is a code (it runs on a dedicated device) that filters all traffic between a protected (inside) network and a less trustworthy (outside) network. Usually a firewall is implemented on a separate computer, with direct connections only to the outside and inside networks.



The purpose of a firewall is to keep 'bad' things outside a protected environment. To accomplish that, firewalls implement a security policy that is specifically designed to address what bad things might happen.

For example, the policy might be to prevent any access from outside. Alternatively, the policy might permit accesses only from certain places, from certain users, or for certain activities. Part of the challenge of protecting a network with a firewall is determining which security policy meets the needs of the installation.

4.2.2 Design of Firewalls:

By carefully positioning a firewall within a network, we can ensure that all network accesses that we want to control must pass through it. A firewall is typically well isolated, making it highly immune to modification. Firewall designers strongly recommend keeping the functionality of the firewall simple.

4.2.3 Types of Firewalls:

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

a- Packet filtering gateways or screening routers
b- Stateful inspection firewalls
c- Application proxies
d- Guards
e- Personal firewalls

Each type does different things; no one is necessarily "right" and the others "wrong."

A: Packet Filtering Gateway

A packet filtering gateway or screening router is the simplest, and in some situations, the most effective type of firewall. A packet filtering gateway controls access to packets based on packet address (source or destination) or specific transport protocol type (such as HTTP web traffic).

Figure (4) 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.

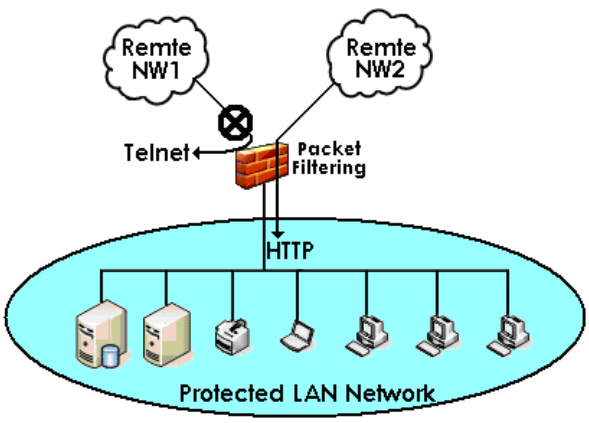


Figure (4): Packet Filter Blocking Addresses and Protocols.

Packet filters block or accept packets solely on the basis of the IP addresses and ports. Thus, any details in the packet's data field (for example, allowing certain Telnet commands while blocking other services) is beyond the capability of a packet filter.

Packet filters can perform the very important service of ensuring the validity of inside addresses. A screening packet filter might be configured to block all packets from the outside that claimed their source address was an inside address. In this example, the packet filter blocks all packets claiming to come from any address of the form 100.50.25.x (but, of course, it permits in any packets with destination 100.50.25.x).

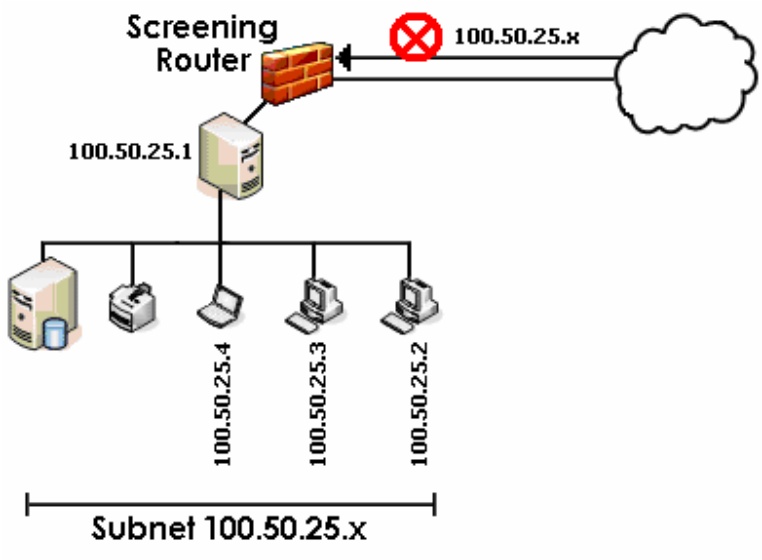


Figure (5): Filter Screening Outside Addresses.

B: Stateful Inspection Firewall:

Filtering firewalls work on packets one at a time, accepting or rejecting each packet and moving on to the next. They have no concept of "state" or "context" from one packet to the next. A stateful inspection firewall maintains state information from one packet to another in the input stream.

One classic approach used by attackers is breaking an attack into multiple packets by forcing some packets to have very short lengths so that a firewall will not be able to detect the signature of an attack split across two or more packets. A stateful inspection firewall would track the sequence of packets and conditions from one packet to another to thwart such an attack.

C: Application Proxy:

Packet filters look only at the headers of packets, not at the data inside the packets. Therefore, a packet filter would pass anything to port 25, assuming its screening rules allow inbound connections to that port. But applications are complex and sometimes contain errors. Worse, applications often act on behalf of all users, so they require privileges of all users. A flawed application, running with all users' privileges, can cause much damage.

An application proxy gateway, also called a bastion host, is a firewall that simulates the effects of an application so that the application will receive only requests to act properly. A proxy gateway is a two-headed device: It looks to the inside as if it is the outside (destination) connection, while to the outside it responds just as the insider would.

An application proxy runs pseudo-applications. The distinction between a proxy and a screening router is that the proxy interprets the protocol stream to an application, to control actions through the firewall on the basis of things visible within the protocol, not just on external header data.

D: Guard:

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, such as whatever it can reliably know of the (outside) user's identity, previous interactions, and so forth. The degree of control a guard can provide is limited only by what is computable. But guards and proxy firewalls are

similar enough that the distinction between them is sometimes fuzzy. That is, we can add functionality to a proxy firewall until it starts to look a lot like a guard.

E: Personal Firewalls:

Increasingly, home users, individual workers, and small businesses use cable modems or DSL connections with unlimited, always-on access. These people need a firewall, but a separate firewall computer to protect a single workstation can seem too complex and expensive. These people need a firewall's capabilities at a lower price.

A personal firewall is an application program that runs on a workstation to block unwanted traffic, usually from the network. A personal firewall can complement the work of a conventional firewall by screening the kind of data a single host will accept, or it can compensate for the lack of a regular firewall, as in a private DSL or cable modem connection.

Just as a network firewall screens incoming and outgoing traffic for that network, a personal firewall screens traffic on a single workstation. A workstation could be vulnerable to malicious code or malicious active agents (ActiveX or Java applets), leakage of personal data stored on the workstation, and vulnerability scans to identify potential weaknesses. Commercial implementations of personal firewalls include Norton Personal Firewall from Symantec, McAfee Personal Firewall, and Zone Alarm from Zone Labs.

Combining a virus scanner with a personal firewall is both effective and efficient. Typically, users forget to run virus scanners daily, but they do remember to run them occasionally, such as sometime during the week. 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. A personal firewall can provide reasonable protection to clients that are not behind a network firewall.

4.2.4 Example Firewall Configurations:

The simplest use of a firewall is shown in Figure (6). This environment has a screening router positioned between the internal LAN and the outside network connection. In many cases, this installation is adequate when we need only screen the address of a router.

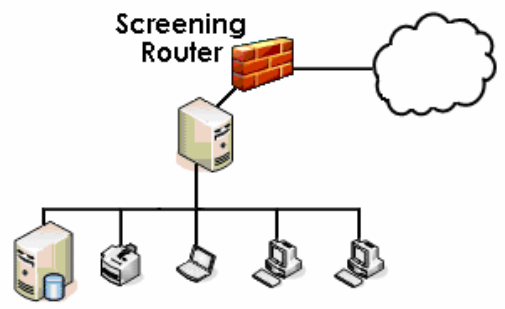


Figure (6): Firewall with Screening Router.

However, to use a proxy machine, this organization is not ideal. Similarly, configuring a router for a complex set of approved or rejected addresses is difficult. If the firewall router is successfully attacked, then all traffic on the LAN to which the firewall is connected is visible. To reduce this exposure, a proxy firewall is often installed on its own LAN, as shown

in Figure (7). In this way the only traffic visible on that LAN is the traffic going into and out of the firewall.

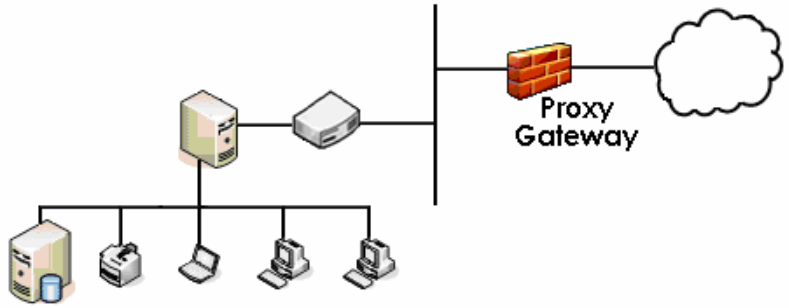


Figure (7): Firewall on Separate LAN.

For even more protection, we can add a screening router to this configuration, as shown in Figure (8). Here, the screening router ensures address correctness to the proxy firewall, the proxy firewall filters traffic according to its proxy rules. Also, if the screening router is subverted, only the traffic to the proxy firewall is visible—not any of the sensitive information on the internal protected LAN.

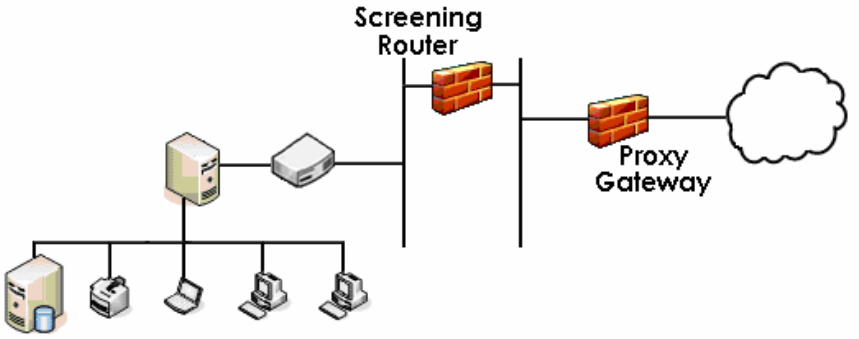


Figure (8): Firewall with Proxy and Screening Router.

4.2.5 Windows Firewall:

Firewall: *A combination of hardware and software that provides a security system, usually to prevent unauthorized access from outside to an internal network or intranet. A firewall prevents direct communication between network and external computers by routing communication through a proxy server outside of the network. The proxy server determines whether it is safe to let a file pass through to the network. A firewall is also called a security-edge gateway.*

A firewall helps to keep your computer more secure. It restricts information that comes to your computer from other computers, giving you more control over the data on your computer and providing a line of defense against people or programs (including viruses and worms) that try to connect to your computer without invitation.

You can think of a firewall as a barrier that checks information (often called traffic) coming from the Internet or a network and then either turns it away or allows it to pass through to your computer, depending on your firewall settings.

- How does it work?

When someone on the Internet or a network tries to connect to computer, we call that attempt an "unsolicited request." When your computer gets an unsolicited request, Windows Firewall blocks the connection. If you run a program such as an instant messaging program or a multiplayer network game that needs to receive information from the Internet or a network, the firewall asks if you want to block or unblock (allow) the connection. If you choose to unblock the

connection, Windows Firewall creates an exception so that the firewall won't bother you when that program needs to receive information in the future.

For example, if you are exchanging instant messages with someone who wants to send you a file (a photo, for example), Windows Firewall will ask you if you want to unblock the connection and allow the photo to reach your computer. Or, if you want to play a multiplayer network game with friends over the Internet, you can add the game as an exception so that the firewall will allow the game information to reach your computer.

Although you can turn off Windows Firewall for specific Internet and network connections, doing this increases the risk that the security of your computer might be compromised.

- What Windows Firewall does and does not do:

It Does	It Does Not
Help block computer viruses and worms from reaching your computer.	**Detect or disable computer viruses and worms** if they are already on your computer. For that reason, you should also install antivirus software and keep it updated to help prevent viruses, worms, and other security threats from damaging your computer or using your computer to spread viruses to others.
Ask for your permission to block or unblock certain connection requests.	**Stop you from opening e-mail with dangerous attachments**. Don't open e-mail attachments from senders that you don't know. Even if you know and trust the source of the e-mail you should still be cautious.
Create a record (a security log), if you want one, that records successful and unsuccessful attempts to connect to your computer.	**Block spam or unsolicited e-mail** from appearing in your inbox. However, some e-mail programs can help you do this.

4.3 Intrusion Detection Systems

4.3.1 Introduction:

Firewalls are important tools in protecting an environment connected to a network. However, the environment must be viewed as a whole, all possible exposures must be considered, and the firewall must fit into a larger, comprehensive security strategy. Firewalls alone cannot secure an environment.

After the perimeter controls, firewall, and authentication and access controls block certain actions, some users are admitted to use a computing system. Most of these controls are preventive: they block known bad things from happening.

Many studies have shown that most computer security incidents are caused by insiders, people who would not be blocked by a firewall. And insiders require access with significant privileges to do their daily jobs. The vast majority of harm from insiders is not malicious; it is honest people making honest mistakes. Then, too, there are the potential malicious outsiders who have somehow passed the screens of firewalls and access controls. Prevention, although necessary, is not a complete computer security control; detection during an incident copes with harm that cannot be prevented in advance.

Intrusion detection systems complement these preventive controls as the next line of defense.

An intrusion detection system (IDS) is a code that run on device, typically on another separate computer, that monitors activity to identify malicious or suspicious events.

An IDS is a sensor, like a smoke detector, that raises an alarm if specific things occur. A model of an IDS is shown in Figure (9). An IDS

receives raw inputs from sensors. It saves those inputs, analyzes them, and takes some controlling action.

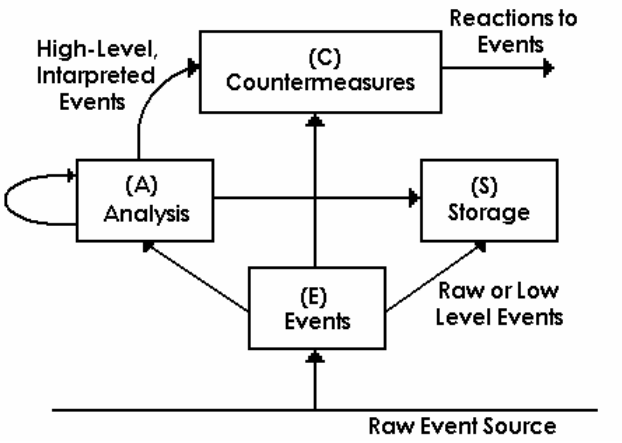


Figure (9): Common Components of an Intrusion Detection Framework.

IDSs perform a variety of functions:

- Monitoring users and system activity
- Auditing system configuration for vulnerabilities and misconfigurations
- Assessing the integrity of critical system and data files
- Recognizing known attack patterns in system activity
- Identifying abnormal activity through statistical analysis
- Managing audit trails and highlighting user violation of policy or normal activity
- Correcting system configuration errors
- Installing and operating traps to record information about intruders

No one IDS performs all of these functions. Let us look more closely at the kinds of IDSs and their use in providing security.

4.3.2 Types of IDSs:

The two general types of intrusion detection systems are signature based and heuristic:

- **Signature-based** intrusion detection systems perform simple pattern-matching and report situations that match a pattern corresponding to a known attack type.
- **Heuristic** intrusion detection systems, also known as **anomaly based**, build a model of acceptable behavior and flag exceptions to that model; for the future, the administrator can mark a flagged behavior as acceptable so that the heuristic IDS will now treat that previously unclassified behavior as acceptable.

Intrusion detection devices can be network based or host based:

- A **network-based IDS** is a stand-alone device attached to the network to monitor traffic throughout that network.
- A **host-based IDS** runs on a single workstation or client or host, to protect that one host.

Early intrusion detection systems worked after the fact, by reviewing logs of system activity to spot potential misuses that had occurred. The administrator could review the results of the IDS to find and fix weaknesses in the system. Now, however, intrusion detection systems operate in real time (or near real time), watching activity and raising alarms in time for the administrator to take protective action.

A: Signature-Based Intrusion Detection:

A simple signature for a known attack type might describe a series of TCP SYN packets sent to many different ports in succession and at times close to one another, as would be the case for a port scan. An intrusion detection system would probably find nothing unusual in the first SYN, say, to port 80, and then another (from the same source address) to port 25. But as more and more ports receive SYN packets, especially ports that are not open, this pattern reflects a possible port scan. Similarly, some implementations of the protocol stack fail if they receive an ICMP packet with a data length of 65535 bytes, so such a packet would be a pattern for which to watch.

The problem with signature-based detection is the signatures themselves. 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 or convert a symbol such as "blank space" to its character code equivalent %20. The IDS must necessarily work from a canonical form of the data stream in order to recognize that %20 matches a pattern with a blank space. The attacker may insert malformed packets that the IDS will see, to intentionally cause a pattern mismatch; the protocol handler stack will discard the packets because of the malformation. Each of these variations could be detected by an IDS, but more signatures require additional work for the IDS, which reduces performance.

Of course, signature-based IDSs cannot detect a new attack for which a signature is not yet installed in the database. Every attack starts as a new attack at some time, and the IDS is helpless to warn of its existence.

Ideally, signatures should match every instance of an attack, match subtle variations of the attack, but not match traffic that is not part of an attack. However, this goal is grand but unreachable.

B: Heuristic Intrusion Detection:

Because signatures are limited to specific, known attack patterns, another form of intrusion detection becomes useful. Instead of looking for matches, heuristic intrusion detection looks for behavior that is out of the ordinary. The original work in this area focused on the individual, trying to find characteristics of that person that might be helpful in understanding normal and abnormal behavior. For example, one user might always start the day by reading e-mail, write many documents using a word processor, and occasionally back up files. These actions would be normal. This user does not seem to use many administrator utilities. If that person tried to access sensitive system management utilities, this new behavior might be a clue that someone else was acting under the user's identity. The approach has been extended to networks in. Later work sought to build a dynamic model of behavior, to accommodate variation and evolution in a person's actions over time. The technique compares real activity with a known representation of normality.

Alternatively, intrusion detection can work from a model of known bad activity. For example, except for a few utilities (login, change password, create user), any other attempt to access a password file is suspect. This form of intrusion detection is known as misuse intrusion detection. In this work, the real activity is compared against a known suspicious area.

All heuristic intrusion detection activity is classified in one of three categories: good/benign, suspicious, or unknown. Over time, specific

kinds of actions can move from one of these categories to another, corresponding to the IDS's learning whether certain actions are acceptable or not.

4.3.3 Stealth Mode:

An IDS is a network device (or, in the case of a host-based IDS, a program running on a network device). Any network device is potentially vulnerable to network attacks. How useful would an IDS be if it itself were deluged with a denial-of-service attack? If an attacker succeeded in logging in to a system within the protected network, wouldn't trying to disable the IDS be the next step?

To counter those problems, most IDSs run in stealth mode, whereby an IDS has two network interfaces: one for the network (or network segment) being monitored and the other to generate alerts and perhaps other administrative needs. The IDS uses the monitored interface as input only; it never sends packets out through that interface. Often, the interface is configured so that the device has no published address through the monitored interface; that is, a router cannot route anything to that address directly, because the router does not know such a device exists. It is the perfect passive wiretap. If the IDS needs to generate an alert, it uses only the alarm interface on a completely separate control network. Such an architecture is shown in Figure (10).

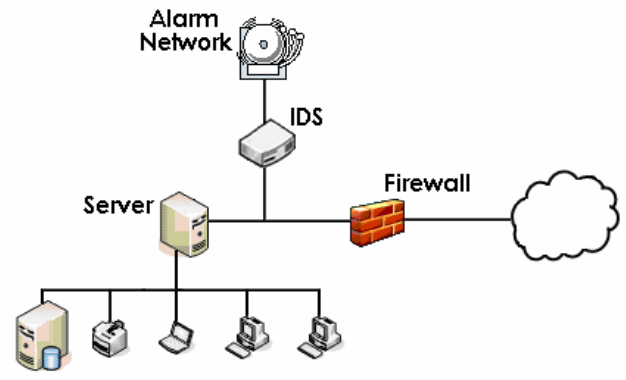


Figure (10): Stealth Mode IDS Connected to Two Networks.

4.3.4 Goals for Intrusion Detection Systems:

The two styles of intrusion detection—pattern-matching and heuristic—represent different approaches, each of which has advantages and disadvantages. Actual IDS products often blend the two approaches.

Ideally, an IDS should be fast, simple, and accurate, while at the same time being complete. It should detect all attacks with little performance penalty. An IDS could use some—or all—of the following design approaches:

- Filter on packet headers
- Filter on packet content
- Maintain connection state
- Use complex, multipacket signatures
- Use minimal number of signatures with maximum effect
- Filter in real time, online
- Hide its presence
- Use optimal sliding time window size to match signatures

4.3.5 Responding to Alarms:

Whatever the type, an intrusion detection system raises an alarm when it finds a match. The alarm can range from something modest, such as writing a note in an audit log, to something significant, such as paging the system security administrator. Particular implementations allow the user to determine what action the system should take on what events.

43.6 False Results:

Intrusion detection systems are not perfect, and mistakes are their biggest problem. Although an IDS might detect an intruder correctly most of the time, it may stumble in two different ways:

- Raising an alarm for something that is not really an attack (called a false positive, or type I error in the statistical community), or
- Not raising an alarm for a real attack (a false negative, or type II error). Too many false positives means the administrator will be less confident of the IDS's warnings, perhaps leading to a real alarm's being ignored.

But false negatives mean that real attacks are passing the IDS without action. We say that the degree of false positives and false negatives represents the sensitivity of the system. Most IDS implementations allow the administrator to tune the system's sensitivity, to strike an acceptable balance between false positives and negatives.

4.4 Secure E-Mail

4.4.1 Security for E-Mail:

E-mail is vital for today's commerce, as well a convenient medium for communications among ordinary users. But e-mail is very public, exposed at every point from the sender's workstation to the recipient's screen. Just as you would not put sensitive or private thoughts on a postcard, you must also acknowledge that e-mail messages are exposed and available for others to read.

Sometimes we would like e-mail to be more secure. To define and implement a more secure form, we begin by examining the exposures of ordinary e-mail.

4.4.2 Threats to E-Mail:

Consider threats to electronic mail:

- message interception (confidentiality)
- message interception (blocked delivery)
- message interception and subsequent replay
- message content modification
- message origin modification
- message content forgery by outsider
- message origin forgery by outsider
- message content forgery by recipient
- message origin forgery by recipient
- denial of message transmission

Confidentiality and content forgery are often handled by encryption. Encryption can also help in a defense against replay,

although we would also have to use a protocol in which each message contains something unique that is encrypted. Symmetric encryption cannot protect against forgery by a recipient, since both sender and recipient share a common key; however, public key schemes can let a recipient decrypt but not encrypt. Because of lack of control over the middle points of a network, senders or receivers generally cannot protect against blocked delivery.

4.4.3 Requirements and Solutions:

If we were to make a list of the requirements for secure e-mail, our wish list would include the following protections.

- *Message confidentiality* (the message is not exposed en route to the receiver)
- *Message integrity* (what the receiver sees is what was sent)
- *Sender authenticity* (the receiver is confident who the sender was)
- *Nonrepudiation* (the sender cannot deny having sent the message)

Not all of these qualities are needed for every message, but an ideal secure e-mail package would allow these capabilities to be invoked selectively.

4.4.4 Confidentiality:

Because the protection has several aspects, we begin our description of them by looking first at how to provide confidentiality enhancements. The sender chooses a (random) symmetric algorithm encryption key. Then, the sender encrypts a copy of the entire message to be transmitted, including FROM:, TO:, SUBJECT:, and DATE: headers. Next, the sender prepends plaintext headers. For key management, the sender encrypts the message key under the recipient's public key, and

attaches that to the message as well. The process of creating an encrypted e-mail message is shown in Figure (11).

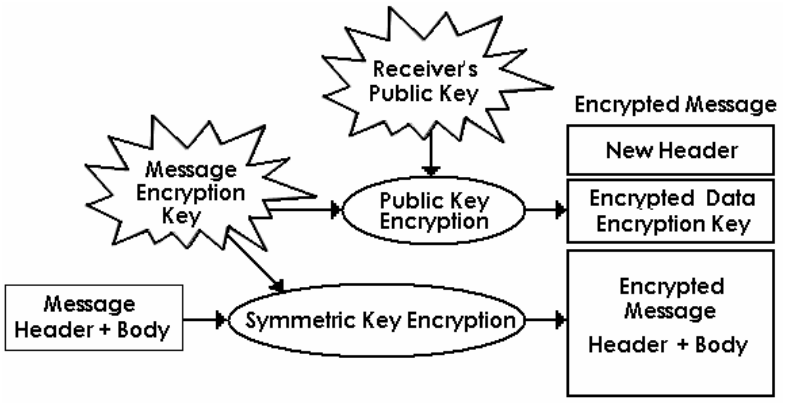


Figure (11): Overview of Encrypted E-Mail Processing.

Encryption can potentially yield any string as output. Many e-mail handlers expect that message traffic will not contain characters other than the normal printable characters. Network e-mail handlers use unprintable characters as control signals in the traffic stream. To avoid problems in transmission, encrypted e-mail converts the entire ciphertext message to printable characters. An example of an encrypted e-mail message is shown in Figure (12). Notice the three portions: an external (plaintext) header, a section by which the message encryption key can be transferred, and the encrypted message itself. (The encryption is shown with shading.)

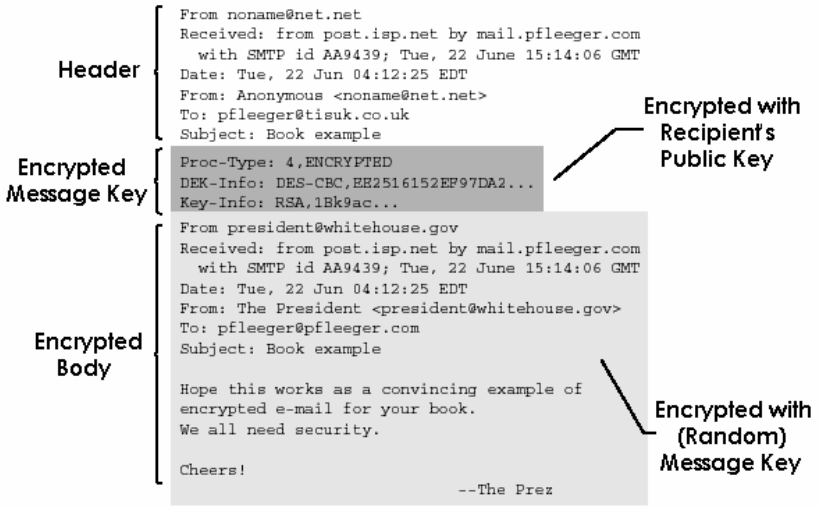


Figure (12): Encrypted E-Mail–Secured Message.

The encrypted e-mail standard works most easily as just described, using both symmetric and asymmetric encryption. The standard is also defined for symmetric encryption only: To use symmetric encryption, the sender and receiver must have previously established a shared secret encryption key. The processing type ("Proc-Type") field tells what privacy enhancement services have been applied. In the data exchange key field ("DEK-Info"), the kind of key exchange (symmetric or asymmetric) is shown. The key exchange ("Key-Info") field contains the message encryption key, encrypted under this shared encryption key. The field also identifies the originator (sender) so that the receiver can determine which shared symmetric key was used. If the key exchange technique were to use asymmetric encryption, the key exchange field would contain the message encryption field, encrypted under the recipient's public key. Also included could be the sender's certificate (used for determining authenticity and for generating replies).

The encrypted e-mail standard is designed to support multiple encryption algorithms, using popular algorithms such as DES, triple DES,

and AES for message confidentiality, and RSA and Diffie–Hellman for key exchange.

4.4.5 Other Security Features:

We may want various forms of integrity for secure e-mail. Encrypted e-mail messages always carry a digital signature, so the authenticity and nonrepudiability of the sender is assured. The integrity is also assured because of a hash function (called a message integrity check, or MIC) in the digital signature. Optionally, encrypted e-mail messages can be encrypted for confidentiality.

Notice in Figure (13) that the header inside the message (in the encrypted portion) differs from that outside. A sender's identity or the actual subject of a message can be concealed within the encrypted portion.

The encrypted e-mail processing can integrate with ordinary e-mail packages, so a person can send both enhanced and nonenhanced messages, as shown in Figure 7-46. If the sender decides to add enhancements, an extra bit of encrypted e-mail processing is invoked on the sender's end; the receiver must also remove the enhancements. But without enhancements, messages flow through the mail handlers as usual.

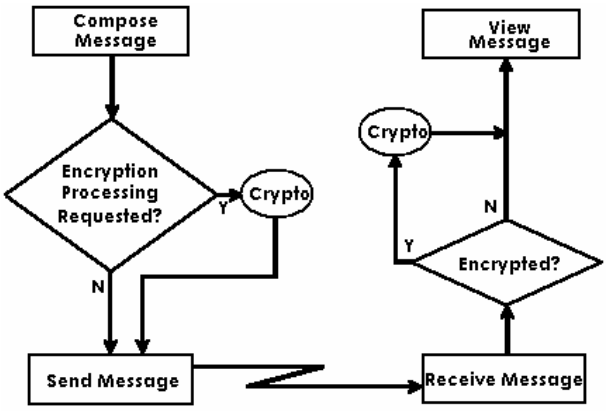


Figure (13): Encrypted E-Mail Processing in Message Transmission.

S/MIME (discussed later in this section) can accommodate the exchange of other than just text messages: support for voice, graphics, video, and other kinds of complex message parts.

4.4.6 Example Secure E-Mail Systems:

Encrypted e-mail programs are available from many sources. Several universities and companies have developed either prototype or commercial versions of encrypted e-mail.

A: PGP:

PGP (Pretty Good Privacy) was invented in 1991. Originally a free package, it became a commercial product after being bought by Network Associates in 1996. A freeware version is still available. PGP is widely available, both in commercial versions and freeware, and it is heavily used by individuals exchanging private e-mail.

PGP addresses the key distribution problem with what is called a "ring of trust" or a user's "keyring". One user directly gives a public key to another, or the second user fetches the first's public key from a server. Some people include their PGP public keys at the bottom of e-mail

messages. And one person can give a second person's key to a third (and a fourth, and so on). Thus, the key association problem becomes one of caveat emptor: If I am reasonably confident that an e-mail message really comes from you and has not been tampered with, I will use your attached public key. If I trust you, I may also trust the keys you give me for other people. The model breaks down intellectually when you give me all the keys you received from people, who in turn gave you all the keys they got from still other people, who gave them all their keys, and so forth.

You sign each key you give me. The keys you give me may also have been signed by other people. I decide to trust the veracity of a key-and-identity combination, based on who signed the key.

PGP does not mandate a policy for establishing trust. Rather, each user is free to decide how much to trust each key received.

The PGP processing performs some or all of the following actions, depending on whether confidentiality, integrity, authenticity, or some combination of these is selected:

- Create a random session key for a symmetric algorithm.
- Encrypt the message, using the session key (for message confidentiality).
- Encrypt the session key under the recipient's public key.
- Generate a message digest or hash of the message; sign the hash by encrypting it with the sender's private key (for message integrity and authenticity).
- Attach the encrypted session key to the encrypted message and digest.
- Transmit the message to the recipient.

The recipient reverses these steps to retrieve and validate the message content.

B: S/MIME:

An Internet standard governs how e-mail is sent and received. The general MIME specification defines the format and handling of e-mail attachments. S/MIME (Secure Multipurpose Internet Mail Extensions) is the Internet standard for secure e-mail attachments.

S/MIME is very much like PGP and its predecessors, PEM (Privacy-Enhanced Mail) and RIPEM. S/MIME has been adopted in commercial e-mail packages, such as Eudora and Microsoft Outlook.

The principal difference between S/MIME and PGP is the method of key exchange. Basic PGP depends on each user's exchanging keys with all potential recipients and establishing a ring of trusted recipients; it also requires establishing a degree of trust in the authenticity of the keys for those recipients. S/MIME uses hierarchically validated certificates, usually represented in X.509 format, for key exchange. Thus, with S/MIME, the sender and recipient do not need to have exchanged keys in advance as long as they have a common certifier they both trust.

S/MIME works with a variety of cryptographic algorithms, such as DES, AES, and RC2 for symmetric encryption.

S/MIME performs security transformations very similar to those for PGP. PGP was originally designed for plaintext messages, but S/MIME handles (secures) all sorts of attachments, such as data files (for example, spreadsheets, graphics, presentations, movies, and sound). Because it is integrated into many commercial e-mail packages, S/MIME is likely to dominate the secure e-mail market.

4.5 Multilevel Security on Networks

With a multilevel security network two or more people wish to share network access. These people work on projects with different sensitivity levels. Thus, a hierarchy of security levels is imposed on the network. It is common to use the military security model, in which data is classeified according to level of sensitivity and further identified by topic. Thus, only cleared people wit a need to know have access to network data.

A multilevel secure network must be preserve two properties of access to data:

1- The simple security property states that no user read data at level higher than that for which the person is authorized.
2- The *-propery stat that no person may write data to a level lower tha the person has accessed.