UNIT – 5

Basic concepts of SNMP, SNMPv1 Community facility and SNMPv3. Intruders, Viruses and related threats.

7.1 BASIC CONCEPTS OF SNMP

Network Management Architecture:

A network management system is a collection of tools for network monitoring and control that is integrated in the following senses:

- .A single operator interface with a powerful but user-friendly set of commands for performing most or all network management tasks. .
- A minimal amount of separate equipment. That is, most of the hardware and software required for network management is incorporated into the existing user equipment.

⇨ A network management system consists of incremental hardware and software additions implemented among existing network components.
⇨ The software used in accomplishing the network management tasks resides in the host computers and communications processors.
⇨ A network management system is designed to view the entire network as a unified architecture, with addresses and labels assigned to each point and the specific attributes of each element and link known to the system.

The model of network management that is used for SNMP includes the following key elements:

- Management station
- Management agent
- Management information base
- Network management protocol

⇨ The **management station** is typically a stand-alone device that serves as the interface for the human network manager into the network management system.

⇨ The **management agent** responds to requests for information from a management station, responds to requests for actions from the management station, and may asynchronously provide the management station with important but unsolicited information.

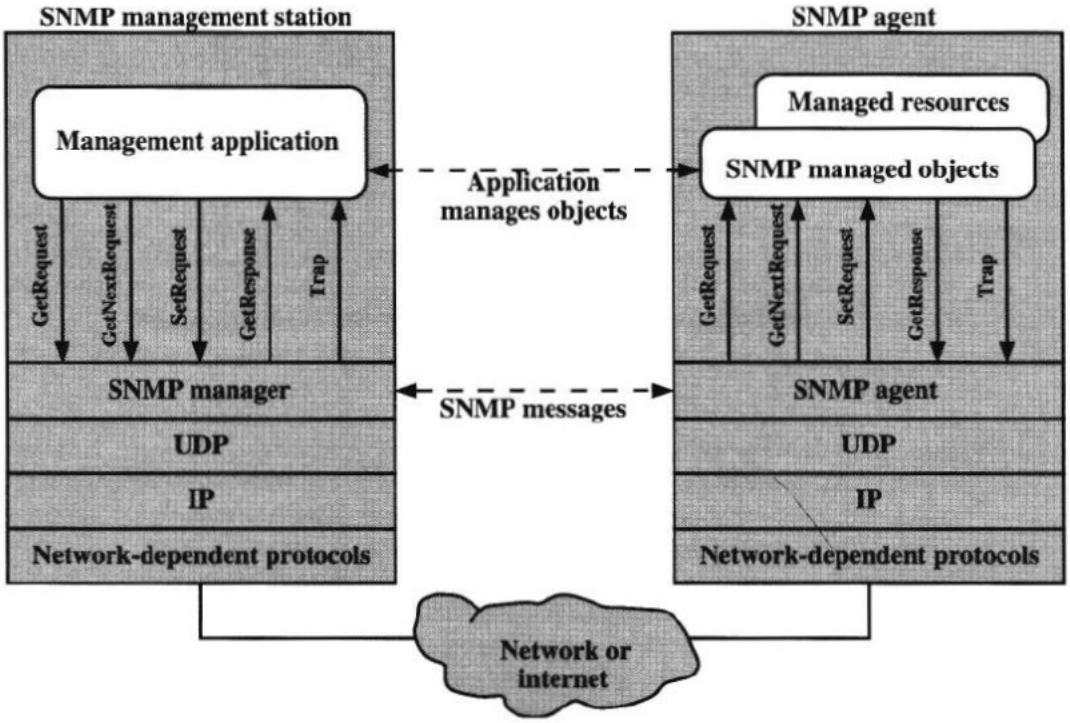
⇨ To manage resources in the network, each resource is represented as an object. An object is, essentially, a data variable that represents one aspect of the managed agent. The collection of objects is referred to as a **management information base (MIB).**

⇨ The management station and agents are linked by a **network management protocol**. The protocol used for the management of TCP/IP networks is the Simple Network Management Protocol (SNMP). This protocol includes the following key capabilities:

- **Get:** Enables the management station to retrieve the value of objects at the agent
- **Set:** Enables the management station to set the value of objects at the agent
- **Notify:** Enables an agent to notify the management station of significant events

Network Management Protocol Architecture

SNMP is a simple tool for network management. It defines a limited, easily implemented management information base (MIB) of scalar variables and two-dimensional tables, and it defines a streamlined protocol to enable a manager to get and set MIB variables and to enable an agent to issue unsolicited notifications, called *traps*.
SNMP was designed to be an application-level protocol that is part of the TCP/IP protocol suite. It is intended to operate over the User Datagram Protocol (UDP), defined in RFC 768.



7.2 SNMPv1 COMMUNITY FACILITY

SNMP network management has several characteristics not typical of all distributed applications. The application involves a one-to-many relationship between a manager and a set of agents: The manager is able to get and set objects in the agents and is able to received traps from the agents. Thus, from an operational or control point of view, the manager "manages" a number of agents. There may be a number of managers, each of which manages all or a subset of the agents in the configuration. These subsets may overlap.
Each agent controls its own local MIB ,and must be able to control the use of that MIB by a number of managers.
There are three aspects of this control:

- **Authentication service:** The agent may wish to limit access to the MIB to authorized managers.
- **Access policy:** The agent may wish to give different access privileges to different managers.
- **Proxy service:** An agent may act as proxy to other agents. This may involve implementing the authentication service and/or access policy for the other agents on the proxy system.

- An **SNMP community** is a relationship between an SNMP agent and a set of SNMP managers that defines authentication, access control, and proxy characteristics.
- The community concept is a local one, defined at the agent.
- The agent establishes one community for each desired combination of authentication, access control, and proxy characteristics.
- Each community is given a unique (within this agent) community name, and the managers within that community are provided with and must employ the community name in all get and set operations.
- The agent may establish a number of communities, with overlapping manager membership.

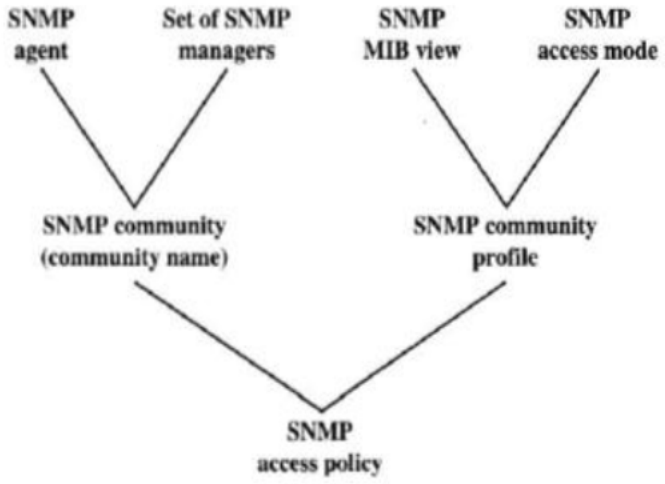
Authentication Service

The purpose of the SNMPvl authentication service is to assure the recipient that an SNMPvl message is from the source that it c1aims to be from. SNMPvl only provides far a trivial scheme far authentication. Every message (get or put request) from a manager to an agent inc1udes a community name. This name functions as a password, and the message is assumed to be authentic if the sender knows the password.

Access Policy

By defining 'a community, an agent limits access to its MIB to a selected set of managers. By the use of more than one community, the agent can provide different categories of MIB access to different managers. There are two aspects to this access control:

- **SNMP MIB view:** A subset of the objects within an MIB. Different MIB views may be defined for each community. The set of objects in a view need not belong to a single sub-tree of the MIB.
- **SNMP access mode:** An element of the set {READ-ONL Y, READ-WRITE}. An access mode is defined for each community.

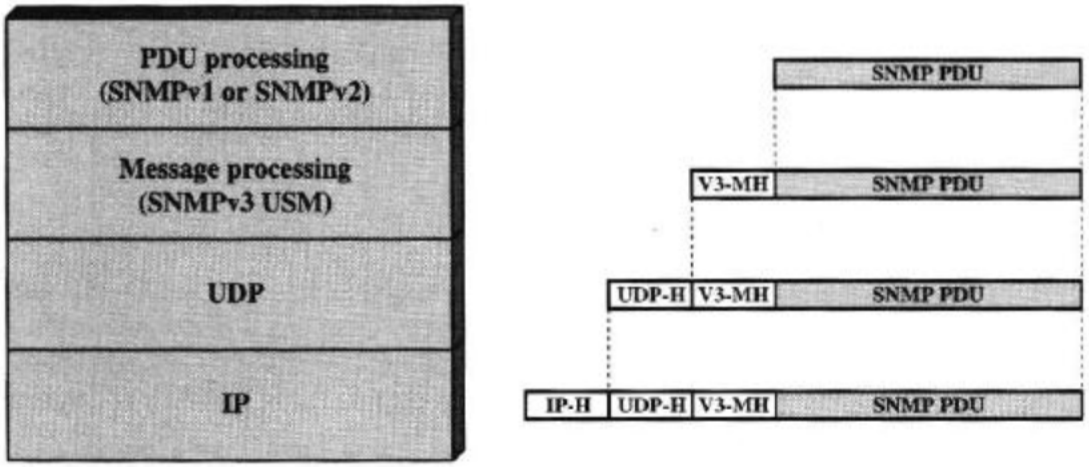


SNMPv1 Administrative Concepts

7.3 SNMPv3

SNMPv3 defines a security capability to be used in conjunction with SNMPv2 (preferred) or SNMPv1.

Fig indicates the relationship among the different versions of SNMP by means of the formats involved.



IP-H = IP header
UDP-H = UDP header
V3-MH = SNMPv3 message header
PDU = Protocol data unit

SNMP Architecture:

SNMP architecture defined in standard RFC 2571 consists of SNMP entities. These entities are interactive and are organized as an abstract set of functions and parameters that are used for passing control and data information. They act either as an agent node, manager node or both. SNMP entities comprises of collection of individual units that communicate with each other in order to provide functions.

The RFC 2571 architecture reflects a key design requirement for SNMPv3:
Design a modular architecture that

1. It allows minimum and cheaper services to be implemented over broad spectrum of functioning surrounding.
2. It is possible to move some part of the architecture forward in a conventional way even though general agreements have not reached all its part.
3. It is possible to adopt other security models.

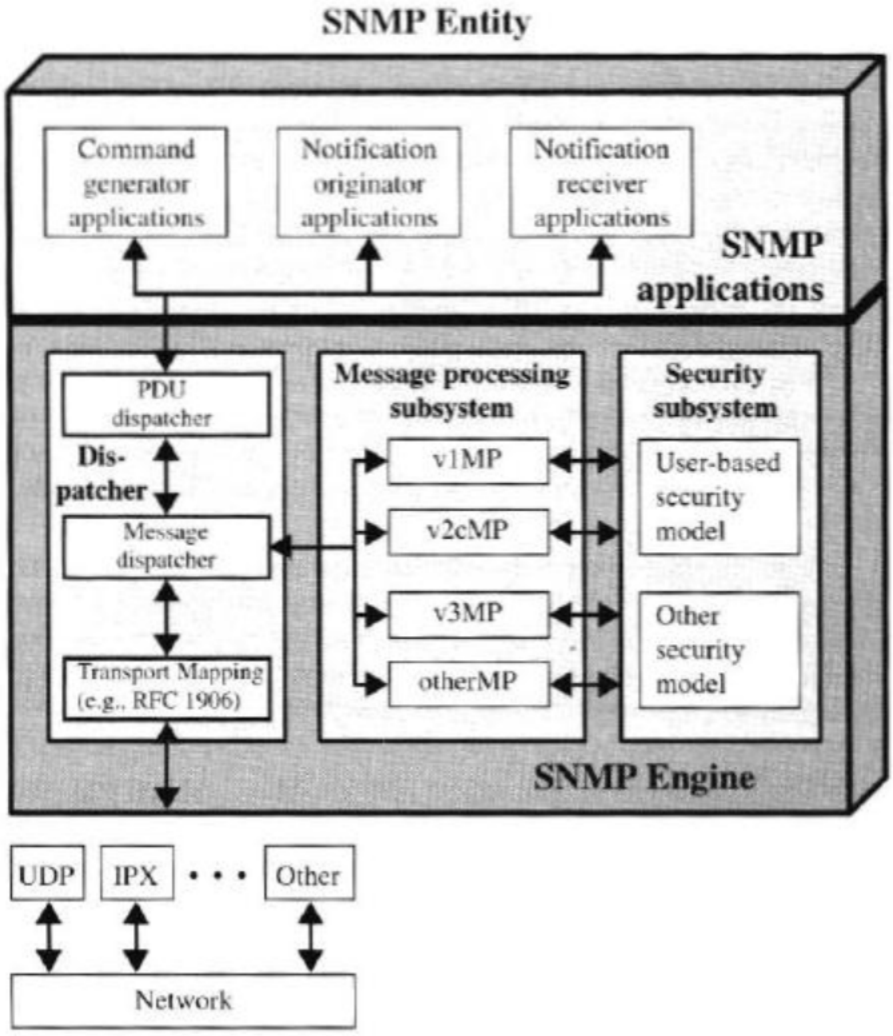
- **SNMP Entity**

Each SNMP entity includes a single SNMP engine. An SNMP engine implements functions for sending and receiving messages, authenticating and encrypting/decrypting messages, and controlling access to managed objects. These functions are provided as services to one or more applications that are configured with the SNMP engine to form an SNMP entity.

❖ **Traditional SNMP manager;**

A traditional SNMP manager interacts with SNMP agents by issuing commands (get, set) and by receiving trap messages; the manager may also interact with other managers by issuing Inform Request PDUs, which provide alerts, and by receiving Inform Response PDUs, which acknowledge Inform Requests. In SNMPv3 terminology, a traditional SNMP manager includes three categories of applications:

1. Command Generator Application
2. Notification Originator Application
3. Notification Receiver Application



Command Generator Application: This application examines and modifies the management data of remote agents. It utilizes SNMPv1 and/or SNMPv2 processing module containing Get, GetBulk, GetNext and SetMessages.

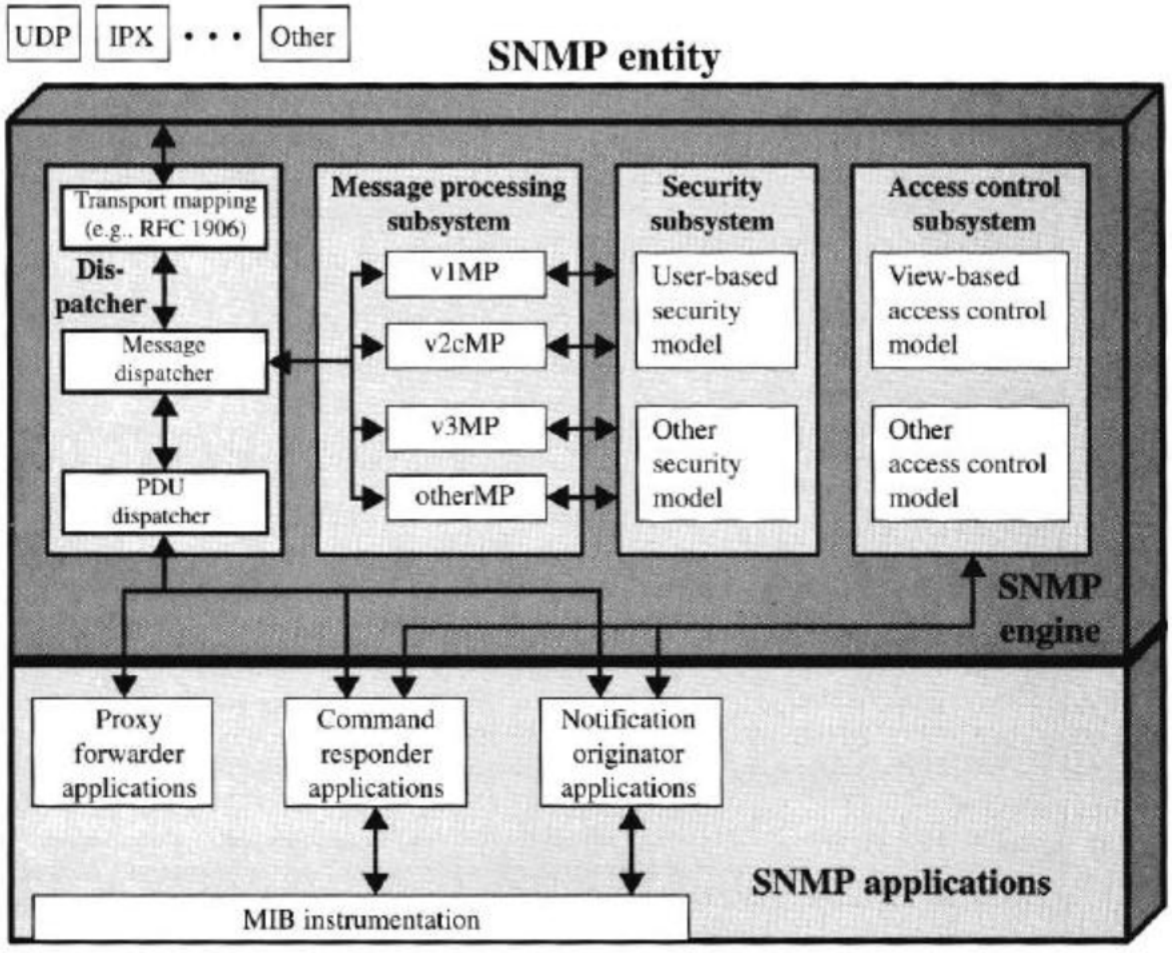
Notification Originator Application: In case of traditional SNMP manager, this application is responsible for starting the transmission of asynchronous messages like InformRequest PDU.

Notification Receiver Application: It is responsible for processing incoming asynchronous messages which may be either InformRequest PDU, SNMPv1 Trap PDU's or SNMPv2 Trap PDU.

❖ **Traditional SNMP Agent:**

SNMP agent consists of three kinds of applications. They are as follows:

1. Command Responder Application
2. Notification Originator Application
3. Proxy Forwarder Application



Command Responder Application: This application make provisions for accessing management data. It is responsible for responding to every incoming request PDU. It does this by restoring the managed entity and/or by defining the managed entity.

Notification Originator Application: In case of traditional SNMP agent, this application is used for starting the transmission of asynchronous message like, Trap PDU of both SNMPv1, SNMPv2.

Proxy Forwarder Application: This application is responsible for forwarding messages between the entities.

7.4 INTRUDERS

One of the two most publicized threats to security is the intruder (the other is viruses), generally referred to as a hacker or cracker.

- **Masquerader:** An individual who is not authorized to use the computer and who penetrates a system's access controls to exploit a legitimate user's account
- **Misfeasor:** A legitimate user who accesses data, programs, or resources for which such access is not authorized, or who is authorized for such access but misuses his or her privileges
- **Clandestine user:** An individual who seizes supervisory control of the system and uses this control to evade auditing and access controls or to suppress audit collection

The masquerader is likely to be an outsider; the misfeasor generally is an insider; and the clandestine user can be either an outsider or an insider.

INTRUSION TECHNIQUES

The objective of the intruder is to gain access to a system or to increase the range of privileges accessible on a system. Generally, this requires the intruder to acquire information that should have been protected. In some cases, this information is in the form of a user password. With knowledge of some other user's password, an intruder can log in to a system and exercise all the privileges accorded to the legitimate user.

Typically, a system must maintain a file that associates a password with each authorized user. If such a file is stored with no protection, then it is an easy matter to gain access to it and learn passwords. The password file can be protected in one of two ways:

- **One-way function:** The system stores only the value of a function based on the user's password. When the user presents a password, the system transforms that password and compares it with the stored value. In practice, the system usually performs a one-way transformation (not reversible) in which the password is used to generate a key for the one-way function and in which a fixed-length output is produced.

- **Access control:** Access to the password file is limited to one or a very few accounts.

On the basis of a survey of the literature and interviews with a number of password crackers, reports the following techniques for learning passwords:

1. Try default passwords used with standard accounts that are shipped with the system. Many administrators do not bother to change these defaults.
2. Exhaustively try all short passwords (those of one to three characters).
3. Try words in the system's online dictionary or a list of likely passwords. Examples of the latter are readily available on hacker bulletin boards.
4. Collect information about users, such as their full names, the names of their spouse and children, pictures in their office, and books in their office that are related to hobbies.
5. Try users' phone numbers, Social Security numbers, and room numbers.
6. Try all legitimate license plate numbers for this state.
7. Use a Trojan horse to bypass restrictions on access.
8. Tap the line between a remote user and the host system.

7.5 VIRUSES AND RELATED THREATS

Malicious Programs

Name	Description
Virus	Attaches itself to a program and propagates copies of itself to other programs
Worm	Program that propagates copies of itself to other computers
Logic bomb	Triggers action when condition occurs
Trojan horse	Program that contains unexpected additional functionality
Backdoor (trapdoor)	Program modification that allows unauthorized access to functionality
Exploits	Code specific to a single vulnerability or set of vulnerabilities
Downloaders	Program that installs other items on a machine that is under attack. Usually, a downloader is sent in an e-mail.
Auto-rooter	Malicious hacker tools used to break into new machines remotely
Kit (virus generator)	Set of tools for generating new viruses automatically
Spammer programs	Used to send large volumes of unwanted e-mail
Flooders	Used to attack networked computer systems with a large volume of traffic to carry out a denial of service (DoS) attack
Keyloggers	Captures keystrokes on a compromised system
Rootkit	Set of hacker tools used after attacker has broken into a computer system and gained root-level access
Zombie	Program activated on an infected machine that is activated to launch attacks on other machines

Backdoor

A backdoor, also known as a trapdoor, is a secret entry point into a program that allows someone that is aware of the backdoor to gain access without going through the usual security access procedures.

Logic Bomb

One of the oldest types of program threat, predating viruses and worms, is the logic bomb. The logic bomb is code embedded in some legitimate program that is set to "explode" when certain conditions are met. Examples of conditions that can be used as triggers for a logic bomb are the presence or absence of certain files, a particular day of the week or date, or a particular user running the application. Once triggered, a bomb may alter or delete data or entire files, cause a machine halt, or do some other damage.

Trojan Horses

A Trojan horse is a useful, or apparently useful, program or command procedure containing hidden code that, when invoked, performs some unwanted or harmful function.

Zombie

A zombie is a program that secretly takes over another Internet-attached computer and then uses that computer to launch attacks that are difficult to trace to the zombie's creator. Zombies are used in denial-of-service attacks, typically against targeted Web sites.

The Nature of Viruses

A virus is a piece of software that can "infect" other programs by modifying them; the modification includes a copy of the virus program, which can then go on to infect other programs.

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

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

Types of Viruses

The most significant types of viruses:

- **Parasitic virus:** The traditional and still most common form of virus. A parasitic virus attaches itself to executable files and replicates, when the infected program is executed, by finding other executable files to infect.
- **Memory-resident virus:** Lodges in main memory as part of a resident system program. From that point on, the virus infects every program that executes.
- **Boot sector virus:** Infects a master boot record or boot record and spreads when a system is booted from the disk containing the virus.

- **Stealth virus:** A form of virus explicitly designed to hide itself from detection by antivirus software.
- **Polymorphic virus:** A virus that mutates with every infection, making detection by the "signature" of the virus impossible.
- **Metamorphic virus:** As with a polymorphic virus, a metamorphic virus mutates with every infection. The difference is that a metamorphic virus rewrites itself completely at each iteration, increasing the difficulty of detection. Metamorphic viruses my change their behavior as well as their appearance.

Macro Viruses

In the mid-1990s, macro viruses became by far the most prevalent type of virus. Macro viruses are particularly threatening for a number of reasons:

1. A macro virus is platform independent. Virtually all of the macro viruses infect Microsoft Word documents. Any hardware platform and operating system that supports Word can be infected.
2. Macro viruses infect documents, not executable portions of code. Most of the information introduced onto a computer system is in the form of a document rather than a program.
3. Macro viruses are easily spread. A very common method is by electronic mail.

E-mail Viruses

A more recent development in malicious software is the e-mail virus. The first rapidly spreading e-mail viruses, such as Melissa, made use of a Microsoft Word macro embedded in an attachment. If the recipient opens the e-mail attachment, the Word macro is activated. Then

1. The e-mail virus sends itself to everyone on the mailing list in the user's e-mail package.
2. The virus does local damage.

Worms

A worm is a program that can replicate itself and send copies from computer to computer across network connections. Upon arrival, the worm may be activated to replicate and propagate again. In addition to propagation, the worm usually performs some unwanted function.

To replicate itself, a network worm uses some sort of network vehicle. Examples include the following:

- **Electronic mail facility:** A worm mails a copy of itself to other systems.
- **Remote execution capability:** A worm executes a copy of itself on another system.
- **Remote login capability:** A worm logs onto a remote system as a user and then uses commands to copy itself from one system to the other.

INTRUDERS

One of the most publicized attacks to security is the intruder, generally referred to as hacker or cracker. Three classes of intruders are as follows

- **Masquerader** – an individual who is not authorized to use the computer and who penetrates a system's access controls to exploit a legitimate user's account.
- **Misfeasor** – a legitimate user who accesses data, programs, or resources for which such access is not authorized, or who is authorized for such access but misuse his or her privileges.
- **Clandestine user** – an individual who seizes supervisory control of the system and uses this control to evade auditing and access controls or to suppress audit collection.

The masquerader is likely to be an outsider; the misfeasor generally is an insider; and the clandestine user can be either an outsider or an insider. Intruder attacks range from the benign to the serious. At the benign end of the scale, there are many people who simply wish to explore internets and see what is out there. At the serious end are individuals who are attempting to read privileged data, perform unauthorized modifications to data, or disrupt the system. Benign intruders might be tolerable, although they do consume resources and may slow performance for legitimate users. However there is no way in advance to know whether an intruder will be benign or malign.

Intrusion techniques The objective of the intruders is to gain access to a system or

to increase the range of privileges accessible on a system. Generally, this requires the intruders to acquire information that should be protected. In most cases, the information is in the form of a user password. Typically, a system must maintain a file that associates a password with each authorized user. If such a file is stored with no protection, then it is an easy matter to gain access to it. The password files can be protected in one of the two ways:

- **One way encryption** – the system stores only an encrypted form of user's password. In practice, the system usually performs a one way transformation (not reversible) in which the password is used to generate a key for the encryption function and in which a fixed length output is produced.
- **Access control** – access to the password file is limited to one or a very few accounts.

The following techniques are used for learning passwords.

- Try default passwords used with standard accounts that are shipped with the system. Many administrators do not bother to change these defaults.
- Exhaustively try all short passwords.
- Try words in the system's online dictionary or a list of likely passwords.

- Collect information about users such as their full names, the name of their spouse and children, pictures in their office and books in their office that are related to hobbies.
- Try user's phone number, social security numbers and room numbers.

- Try all legitimate license plate numbers.

- Use a torjan horse to bypass restriction on access.

- Tap the line between a remote user and the host system. Two principle countermeasures:
  - Detection – concerned with learning of an attack, either before or after its success.

  - Prevention – challenging security goal and an uphill bottle at all times.

INTRUSION DETECTION

Inevitably, the best intrusion prevention system will fail. A system's second line of defense is intrusion detection, and this has been the focus of much research in recent years. This interest is motivated by a number of considerations, including the following:

1. If an intrusion is detected quickly enough, the intruder can be identified and ejected from the system before any damage is done or any data are compromised.
2. An effective intrusion detection system can serve as a deterrent, so acting to prevent intrusions.
3. Intrusion detection enables the collection of information about intrusion techniques that can be used to strengthen the intrusion prevention facility.

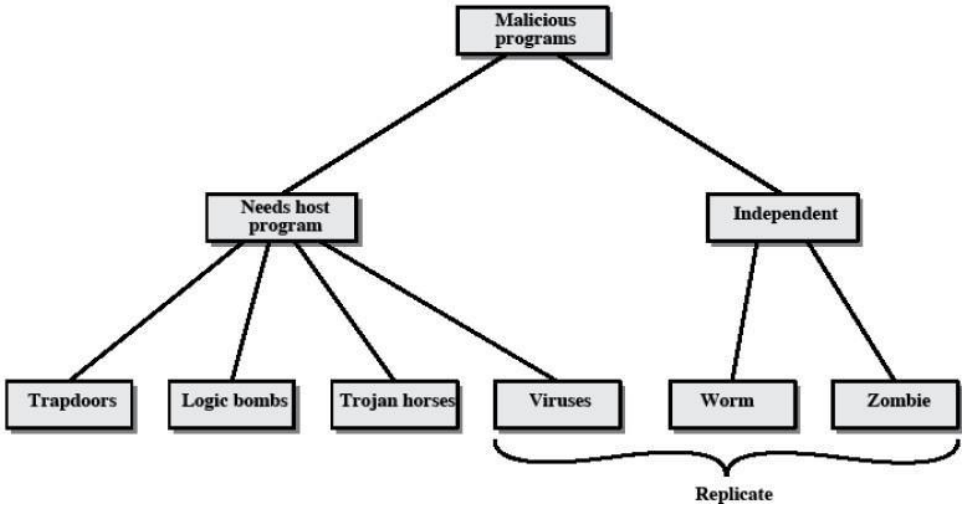
Intrusion detection is based on the assumption that the behavior of the intruder differs from that of a legitimate user in ways that can be quantified.

Figure 18.1 suggests, in very abstract terms, the nature of the task confronting the designer of an intrusion detection system. Although the typical behavior of an intruder differs from the typical behavior of an authorized user, there is an overlap in these behaviors. Thus, a loose interpretation of intruder behavior, which will catch more intruders, will also lead to a number of "false positives," or authorized users identified as intruders. On the other hand, an attempt to limit false positives by a tight interpretation of intruder behavior will lead to an increase in false negatives, or intruders not identified as intruders. Thus, there is an element of compromise and art in the practice of intrusion detection.

VIRUSES AND RELATED THREATS

Perhaps the most sophisticated types of threats to computer systems are presented by programs that exploit vulnerabilities in computing systems.

Malicious Programs



Name	Description
Virus	Attaches itself to a program and propagates copies of itself to other programs
Worm	Program that propagates copies of itself to other computers
Logic bomb	Triggers action when condition occurs
Trojan horse	Program that contains unexpected additional functionality

Backdoor (trapdoor)	Program modification that allows unauthorized access to functionality
Exploits	Code specific to a single vulnerability or set of vulnerabilities
Downloaders	Program that installs other items on a machine that is under attack. Usually, a downloader is sent in an e-mail.
Auto-rooter	Malicious hacker tools used to break into new machines remotely
Kit (virus generator)	Set of tools for generating new viruses automatically
Spammer programs	Used to send large volumes of unwanted e-mail
Flooders	Used to attack networked computer systems with a large volume of traffic to carry out a denial of service (DoS) attack
Keyloggers	Captures keystrokes on a compromised system
Rootkit	Set of hacker tools used after attacker has broken into a computer system and gained root-level access
Zombie	Program activated on an infected machine that is activated to launch attacks on other machines

Malicious software can be divided into two categories: those that need a host program, and those that are independent.
The former are essentially fragments of programs that cannot exist independently of some actual application program, utility, or system program. Viruses, logic bombs, and backdoors are examples. The latter are self-contained programs that can be scheduled and run by the operating system. Worms and zombie programs are examples.

The Nature of Viruses A virus is a piece of software that can "infect" other programs by modifying them; the modification includes a copy of the virus program, which can then go on to infect other programs. A virus can do anything that other programs do. The only difference is that it attaches itself to another program and executes secretly when the host program is run. Once a virus is executing, it can perform any function, such as erasing files and programs. During its lifetime, a typical virus goes through the following four phases:

- **Dormant phase**: The virus is idle. The virus will eventually be activated by some event, such as a date, the presence of another program or file, or the capacity of the disk exceeding some limit. Not all viruses have this stage.
- **Propagation phase:** The virus places an identical copy of itself into other programs or into certain system areas on the disk. Each infected program will now contain a clone of the virus, which will itself enter a propagation phase.
- **Triggering phase:** The virus is activated to perform the function for which it was intended. As with the dormant phase, the triggering phase can be caused by a variety of system events, including a count of the number of times that this copy

of the virus has made copies of itself.

- **Execution phase:** The function is performed. The function may be harmless, such as a message on the screen, or damaging, such as the destruction of programs and data files.

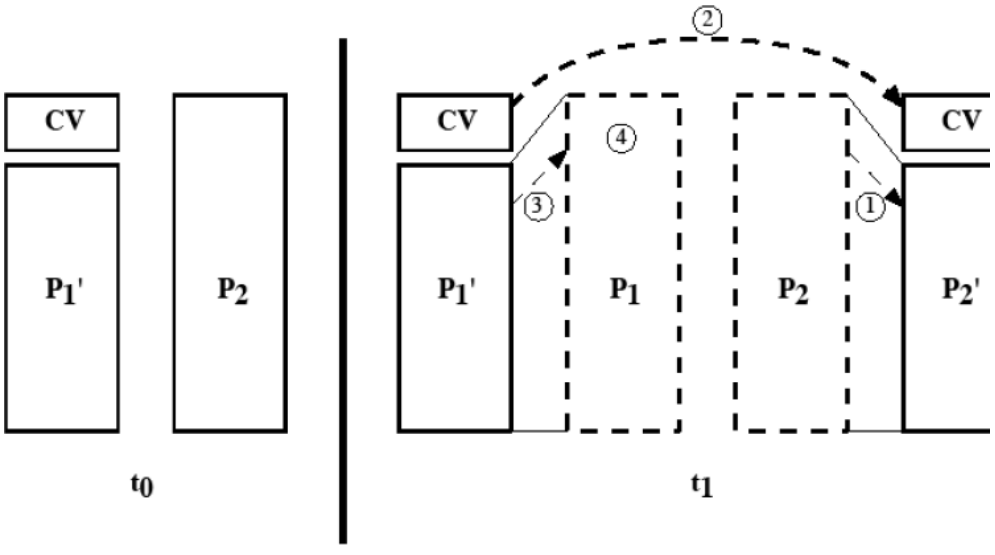
Virus Structure

A virus can be prepended or postpended to an executable program, or it can be embedded in some other fashion. The key to its operation is that the infected program, when invoked, will first execute the virus code and then execute the original code of the program. **An infected program begins with the virus code and works as follows.**

The first line of code is a jump to the main virus program. The second line is a special marker that is used by the virus to determine whether or not a potential victim program has already been infected with this virus. When the program is invoked, control is immediately transferred to the main virus program. The virus program first seeks out uninfected executable files and infects them. Next, the virus may perform some action, usually detrimental to the system. This action could be performed every time the program is invoked, or it could be a logic bomb that triggers only under certain conditions. Finally, the virus transfers control to the original program. If the infection phase of the program is reasonably rapid, a user is unlikely to notice any difference between the execution of an infected and uninfected program.

A virus such as the one just described is easily detected because an infected version of a program is longer than the corresponding uninfected one. A way to thwart such a simple means of detecting a virus is to compress the executable file so that both the infected and uninfected versions are of identical length.. The key lines in this virus are numbered, and Figure 19.3 [COHE94] illustrates the operation. We assume that program P1 is infected with the virus CV. When this program is invoked, control passes to its virus, which performs the following steps:

- For each uninfected file P_2 that is found, the virus first compresses that file to produce P'_2, which is shorter than the original program by the size of thevirus.
- A copy of the virus is prepended to the compressed program.
- The compressed version of the original infected program, P'1, is uncompressed
- The uncompressed original program is executed.



In this example, the virus does nothing other than propagate. As in the previous example, the virus may include a logic bomb.

Initial Infection

Once a virus has gained entry to a system by infecting a single program, it is in a position to infect some or all other executable files on that system when the infected program executes. Thus, viral infection can be completely prevented by preventing the virus from gaining entry in the first place. Unfortunately, prevention is extraordinarily difficult because a virus can be part of any program outside a system. Thus, unless one is content to take an absolutely bare piece of iron and write all one's own system and application programs, one is vulnerable.

Types of Viruses

Following categories as being among the most significant types of viruses:

- **Parasitic virus**: The traditional and still most common form of virus. A parasitic virus attaches itself to executable files and replicates, when the infected program is executed, by finding other executable files to infect.
- **Memory-resident virus**: Lodges in main memory as part of a resident system program. From that point on, the virus infects every program that executes.
- **Boot sector virus**: Infects a master boot record or boot record and spreads when a system is booted from the disk containing the virus.
- **Stealth virus**: A form of virus explicitly designed to hide itself from detection by antivirus software.
- **Polymorphic virus**: A virus that mutates with every infection, making detection by the "signature" of the virus impossible.
- **Metamorphic virus**: As with a polymorphic virus, a metamorphic virus mutates with every infection. The difference is that a metamorphic virus rewrites itself completely at each iteration, increasing the difficulty of detection. Metamorphic viruses my change their behavior as well as their appearance.

One example of a **stealth virus** was discussed earlier: a virus that uses compression so that the infected program is exactly the same length as an uninfected version. Far more sophisticated techniques are possible. For example, a virus can place intercept logic in disk I/O routines, so that when there is an attempt to read suspected portions of the disk using these routines, the virus will present back the original, uninfected program.

A **polymorphic virus** creates copies during replication that are functionally equivalent but have distinctly different bit patterns

Macro Viruses

In the mid-1990s, macro viruses became by far the most prevalent type of virus. Macro viruses are particularly threatening for a number of reasons:

1. A macro virus is platform independent. Virtually all of the macro viruses infect Microsoft Word documents. Any hardware platform and operating system that supports Word can be infected.

2. Macro viruses infect documents, not executable portions of code. Most of the information introduced onto a computer system is in the form of a document rather than a program.

3. Macro viruses are easily spread. A very common method is by electronic mail.

Macro viruses take advantage of a feature found in Word and other office applications such as Microsoft Excel, namely the macro. In essence, a macro is an executable program embedded in a word processing document or other type of file. Typically, users employ macros to automate repetitive tasks and thereby save keystrokes. The macro language is usually some form of the Basic programming language. A user might define a sequence of keystrokes in a macro and set it up so that the macro is invoked when a function key or special short combination of keys is input. Successive releases of Word provide increased protection against macro viruses. For example, Microsoft offers an optional Macro Virus Protection tool that detects suspicious Word files and alerts the customer to the potential risk of opening a file with macros. Various antivirus product vendors have also developed tools to detect and correct macro viruses.

E-mail Viruses

A more recent development in malicious software is the e-mail virus. The first rapidly spreading e-mail viruses, such as Melissa, made use of a Microsoft Word macro embedded in an attachment. If the recipient opens the e-mail attachment, the Word macro is activated. Then

1. The e-mail virus sends itself to everyone on the mailing list in the user's e-mail package.
2. The virus does local damage.

Worms

A worm is a program that can replicate itself and send copies from computer to computer across network connections. Upon arrival, the worm may be activated to replicate and propagate again. Network worm programs use network connections to spread from system to system. Once active within a system, a network worm can behave as a computer virus or bacteria, or it could implant Trojan horse programs or perform any number of disruptive or destructive actions. To replicate itself, a network worm uses some sort of network vehicle. Examples include the following:

- Electronic mail facility: A worm mails a copy of itself to other systems.
- Remote execution capability: A worm executes a copy of itself on another system.
- Remote login capability: A worm logs onto a remote system as a user and then uses commands to copy itself from one system to the other.

The new copy of the worm program is then run on the remote system where, in addition to any functions that it performs at that system, it continues to spread in the same fashion. A network worm exhibits the same characteristics as a computer virus: a dormant phase, a propagation phase, a triggering phase, and an execution phase.

The Morris Worm

The Morris worm was designed to spread on UNIX systems and used a number of different techniques for propagation.

1. It attempted to log on to a remote host as a legitimate user. In this method, the worm first attempted to crack the local password file, and then used the discovered passwords and corresponding user IDs. The assumption was that many users would use the same password on different systems. To obtain the passwords, the worm ran a password-cracking program that tried
    a. Each user's account name and simple permutations of it
    b. A list of 432 built-in passwords that Morris thought to be likely candidates
    c. All the words in the local system directory
2. It exploited a bug in the finger protocol, which reports the whereabouts of a remote

user.
3. It exploited a trapdoor in the debug option of the remote process that receives and sends mail.
If any of these attacks succeeded, the worm achieved communication with the operating system command interpreter.
Recent Worm Attacks In late 2001, a more versatile worm appeared, known as Nimda. Nimda spreads by multiple mechanisms:

- from client to client via e-mail
- from client to client via open network shares
- from Web server to client via browsing of compromised Web sites
- from client to Web server via active scanning for and exploitation of various Microsoft

FIREWALLS

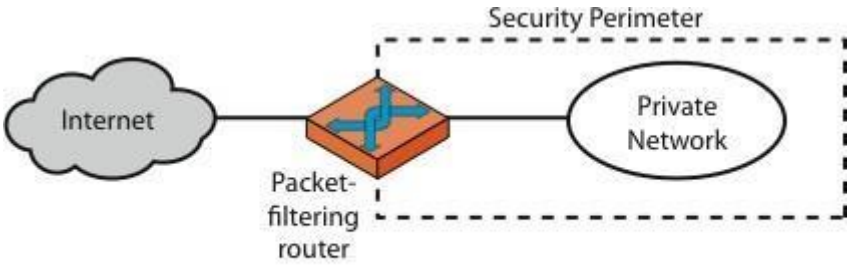
A firewall is inserted between the premises network and the Internet to establish a controlled link and to erect an outer security wall or perimeter, forming a single choke point where security and audit can be imposed. A firewall:
1. Defines a single choke point that keeps unauthorized users out of the protected network, prohibits potentially vulnerable services from entering or leaving the network, and provides protection from various kinds of IP spoofing and routing attacks.
2. provides a location for monitoring security-related events
3. is a convenient platform for several Internet functions that are not security related, such as NAT and Internet usage audits or logs
4. A firewall can serve as the platform for IPSec to implement virtual private networks.

Design Goals of Firewalls
All traffic from inside to outside must pass through the firewall (physically blocking all access to the local network except via the firewall)
Only authorized traffic (defined by the local security police) will be allowed to pass
The firewall itself is immune to penetration (use of trusted system with a secure operating system)
The four general techniques that firewalls use to control access and enforce the sites security policies are:
Service control: Determines the types of Internet services that can be accessed, inbound or outbound
Direction control: Determines the direction in which particular service requests are allowed to flow
User control: Controls access to a service according to which user is attempting to access it
Behavior control: Controls how particular services are used (e.g. filter e-mail)
The limitations of Firewalls are:
1. Cannot protect against attacks that bypass the firewall, eg PCs with dial-out capability to an ISP, or dial-in modem pool use.
2. do not protect against internal threats, eg disgruntled employee or one who cooperates with an attacker
3. cannot protect against the transfer of virus-infected programs or files, given wide variety of O/S & applications supported
Types of Firewalls

Firewalls are generally classified as three types: packet filters, application-level gateways, & circuit-level gateways.

Packet-filtering Router

A packet-filtering router applies a set of rules to each incoming and outgoing IP packet to forward or discard the packet. Filtering rules are based on information contained in a network packet such as src & dest IP addresses, ports, transport protocol & interface.



(a) Packet-filtering router

If there is no match to any rule, then one of two default policies are applied:

- that which is not expressly permitted is prohibited (default action is discard packet), conservative policy
- that which is not expressly prohibited is permitted (default action is forward packet), permissive policy

The default discard policy is more conservative. Initially, everything is blocked, and services must be added on a case-by-case basis. This policy is more visible to users, who are more likely to see the firewall as a hindrance. The default forward policy increases ease of use for end users but provides reduced security; the security administrator must, in essence, react to each new security threat as it becomes known. One advantage of a packet-filtering router is its simplicity. Also, packet filters typically are transparent to users and are very fast.

The table gives some examples of packet-filtering rule sets. In each set, the rules are applied top to bottom.

Table 20.1 Packet-Filtering Examples

A

action	ourhost	port	theirhost	port	comment
block	*	*	SPIGOT	*	we don't trust these people
allow	OUR-GW	25	*	*	connection to our SMTP port

B

action	ourhost	port	theirhost	port	comment
block	*	*	*	*	default

C

action	ourhost	port	theirhost	port	comment
allow	*	*	*	25	connection to their SMTP port

D

action	src	port	dest	port	flags	comment
allow	{our hosts}	*	*	25		our packets to their SMTP port
allow	*	25	*	*	ACK	their replies

E

action	src	port	dest	port	flags	comment
allow	{our hosts}	*	*	*		our outgoing calls
allow	*	*	*	*	ACK	replies to our calls
allow	*	*	*	>1024		traffic to nonservers

A. Inbound mail is allowed to a gateway host only (port 25 is for SMTP incoming
B. explicit statement of the default policy
C. tries to specify that any inside host can send mail to the outside, but has problem that an outside machine could be configured to have some other application linked to port 25
D. properly implements mail sending rule, by checking ACK flag of a TCP segment is set
E. this rule set is one approach to handling FTP connections

Some of the attacks that can be made on packet-filtering routers & countermeasures are:

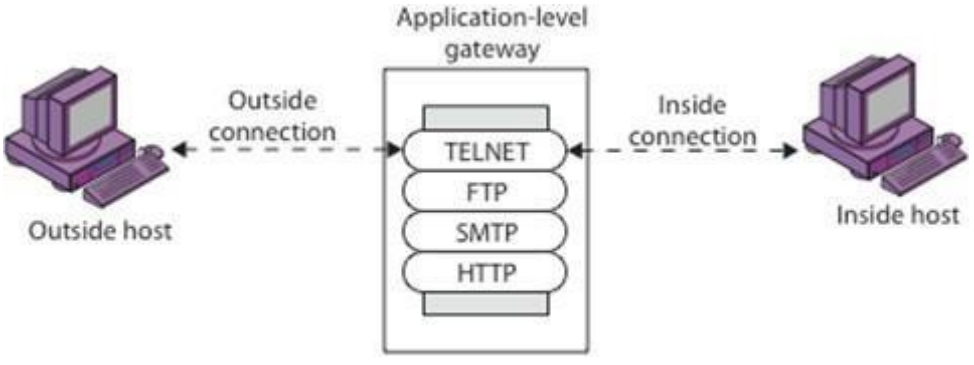
- **IP address spoofing**: where intruder transmits packets from the outside with internal host source IP addresses, need to filter & discard such packets
- **Source routing attacks**: where source specifies the route that a packet should take to bypass security measures, should discard all source routed packets
- **Tiny fragment attacks**: intruder uses the IP fragmentation option to create extremely small fragments and force the TCP header information into separate fragments to circumvent filtering rules needing full header info, can enforce minimum fragment size to include full header.

Stateful Packet Filters

A traditional packet filter makes filtering decisions on an individual packet basis and does not take into consideration any higher layer context. A stateful inspection packet filter tightens up the rules for TCP traffic by creating a directory of outbound TCP connections, and will allow incoming traffic to high-numbered ports only for those packets that fit the profile of one of the entries in this directory. Hence they are better able to detect bogus packets sent out of context.

APPLICATION LEVEL GATEWAY

An application-level gateway (or proxy server), acts as a relay of application-level traffic. The user contacts the gateway using a TCP/IP application, such as Telnet or FTP, and the gateway asks the user for the name of the remote host to be accessed. When the user responds and provides a valid user ID and authentication information, the gateway contacts the application on the remote host and relays TCP segments containing the application data between the two endpoints. If the gateway does not implement the proxy code for a specific application, the service is not supported and cannot be forwarded across the firewall.



(b) Application-level gateway

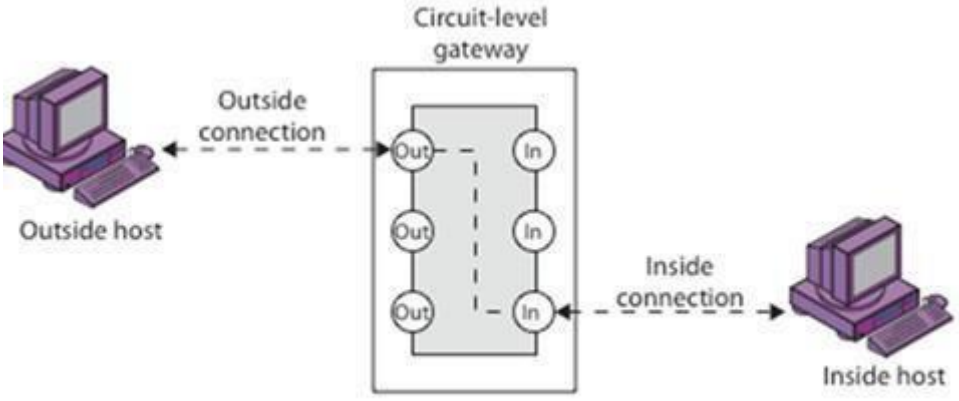
Application-level gateways tend to be more secure than packet filters. Rather than trying to deal with the numerous possible combinations that are to be allowed and forbidden at the TCP and IP level, the application-level gateway need only scrutinize a few allowable applications. In addition, it is easy to log and audit all incoming traffic at the application level. A prime disadvantage of this type of gateway is the additional processing overhead on each connection. In effect, there are two spliced connections between the end users, with the gateway at the splice point, and the gateway must examine and forward all traffic

in both directions.

CIRCUIT LEVEL GATEWAY

A circuit-level gateway relays two TCP connections, one between itself and an inside TCP user, and the other between itself and a TCP user on an outside host. Once the two connections are established, it relays TCP data from one connection to the other without examining its contents. The security function consists of determining which connections will be allowed. It is typically used when internal users are trusted to decide what external services to access.

One of the most common circuit-level gateways is SOCKS, defined in RFC 1928. It consists of a SOCKS server on the firewall, and a SOCKS library & SOCKS-aware applications on internal clients. The protocol described here is designed to provide a framework for client-server applications in both the TCP and UDP domains to conveniently and securely use the services of a network firewall. The protocol is conceptually a "shim-layer" between the application layer and the transport layer, and as such does not provide network-layer gateway services, such as forwarding of ICMP messages.



(c) Circuit-level gateway

Bastion Host

A bastion host is a critical strong point in the network's security, serving as a platform for an application-level or circuit-level gateway, or for external services. It is thus potentially exposed to "hostile" elements and must be secured to withstand this. Common characteristics of a bastion host include that it:

- executes a secure version of its O/S, making it a trusted system
- has only essential services installed on the bastion host
- may require additional authentication before a user is allowed access to the proxy services
- is configured to support only a subset of the standard application's command set, with access only to specific hosts
- maintains detailed audit information by logging all traffic
- has each proxy module a very small software package specifically designed for network security
- has each proxy independent of other proxies on the bastion host
- have a proxy performs no disk access other than to read its initial configuration file
- have each proxy run as a non-privileged user in a private and secured directory
- A bastion host may have two or more network interfaces (or ports), and must be trusted to enforce trusted separation between these network connections,

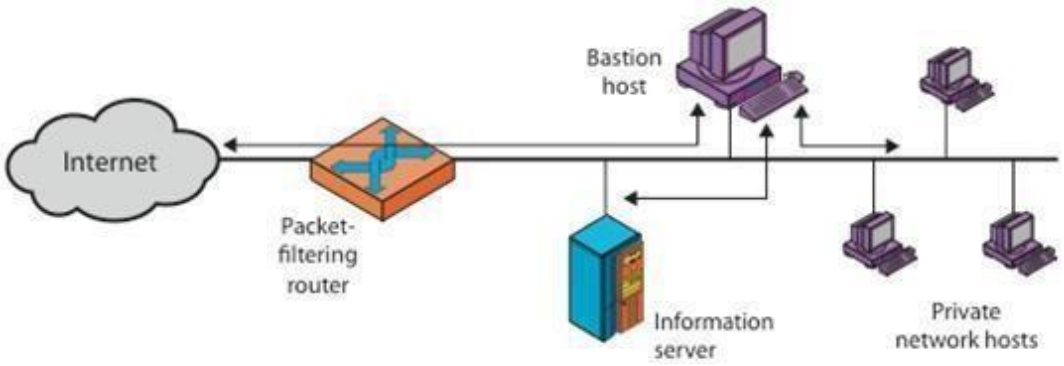
relaying traffic only according to policy.

Firewall Configurations

In addition to the use of a simple configuration consisting of a single system, more complex configurations are possible and indeed more common. There are three common firewall configurations.

The following figure shows the "**screened host firewall, single-homed bastion configuration**", where the firewall consists of two systems:

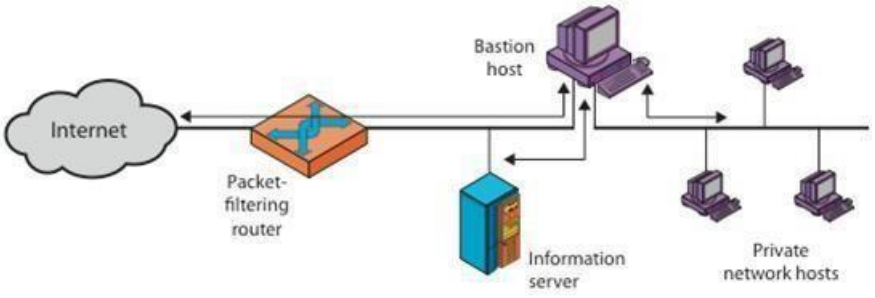
- a packet-filtering router - allows Internet packets to/from bastion only
- a bastion host - performs authentication and proxy functions



(a) Screened host firewall system (single-homed bastion host)

This configuration has greater security, as it implements both packet-level & application-level filtering, forces an intruder to generally penetrate two separate systems to compromise internal security, & also affords flexibility in providing direct Internet access to specific internal servers (eg web) if desired.

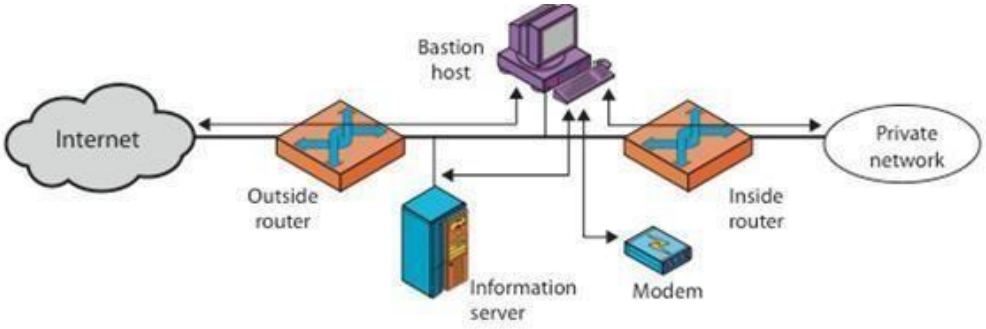
The next configuration illustrates the "**screened host firewall, dual-homed bastion configuration**" which physically separates the external and internal networks, ensuring two systems must be compromised to breach security. The advantages of dual layers of security are also present here.



(b) Screened host firewall system (dual-homed bastion host)

Again, an information server or other hosts can be allowed direct communication with the router if this is in accord with the security policy, but are now separated from the internal network.

The third configurations illustrated below shows the "**screened subnet firewall configuration**", being the most secure shown.



(c) Screened-subnet firewall system

It has two packet-filtering routers, one between the bastion host and the Internet and the other between the bastion host and the internal network, creating an isolated sub-network. This may consist of simply the bastion host but may also include one or more information servers and modems for dial-in capability. Typically, both the Internet and the internal network have access to hosts on the screened subnet, but traffic across the screened subnet is blocked.

This configuration offers several advantages:

• There are now three levels of defense to thwart intruders

• The outside router advertises only the existence of the screened subnet to the Internet; therefore the internal network is invisible to the Internet

• Similarly, the inside router advertises only the existence of the screened subnet to the internal network; hence systems on the inside network cannot construct direct routes to the Internet

16. ADDITIONAL TOPICS

COMPUTER FORENSICS

Computer security and computer forensics are distinct but related disciplines due to the degree of overlap of raw material used by both fields. In general, computer security aims to preserve a system as it is meant to be (as per the security policies) whereas computer forensics (and especially network or intrusion forensics) sets out to explain how a policy became violated. Therefore, the main difference can be seen as one of system integrity versus culpability for an event or set of events.

Whereas the two fields may use similar data sources, they have different and sometimes opposing aims. For example, security countermeasures such as encryption or data wiping tools may work against the computer forensic investigation. The security measures will complicate the investigation as the data must be decrypted prior to analysis. In addition, security functions tend to only implement minimal logging by design. Therefore, not all the information required will be available to the forensic analyst.

Computer security is an established field of computer science, whilst computer forensics is an emergent area. Increasingly, computer security will involve forensic investigation techniques, and vice versa. Therefore, both fields have much to learn from each other.