UNIT 1 TRANSPORT SERVICES AND MECHANISM

Structure **Page Nos.**

1.0 INTRODUCTION

The transport layer is the core of the OSI model. It is not just another layer but the heart of the whole protocol hierarchy. Protocols at this layer oversee the delivery of data from an application program on one device to an application program on another device. It is the first end-to-end layer in the OSI model. It provides its services to the upper layer to use the services of the network layer and other lower layer protocols.

You must be aware that an Internet is comprised of several physical networks such as the LANs, MANs, and WANs that are connected together through a variety of media (wired as well wireless) to facilitate the transfer of data from a computer on one network to another computer on another network. As a transmission moves from one network to another, the data on the way may be transformed i.e., it may be encapsulated in different types and lengths of packets.

The upper-layer protocols are unaware of the intricacy of the physical networks the transport layer. To the upper layers, the individual physical networks are a simple homogeneous network that somehow takes data and delivers it to its destination, reliably. For example, even if an Ethernet in the LAN part of internet is replaced by a FDDI, the upper layers remain unaware of it. To them, the Internet is a single and essentially unchanging network. The transport layer provides this transparency.

Examples of transport layer protocols are Transmission Control Protocol (TCP) and User Datagram Protocol (UDP).

In this unit, we will first discuss types of services we might expect from a transport layer. Next, we will examine several mechanisms to support these services.

1.1 OBJECTIVES

After going through this unit, you should be able to:

- list and describe transport services, and
- list and describe transport mechanisms.

1.2 TRANSPORT SERVICES

We begin by looking at the kinds of services that a transport protocol can or should provide to upper layer protocols. *Figure 1* outlines the environment for transport services. There is a transport entity that provides services to the upper layer users, which might be an application process such as http, telnet etc. This local transport entity communicates with some remote-transport entity, based on the services provided by the network layer.

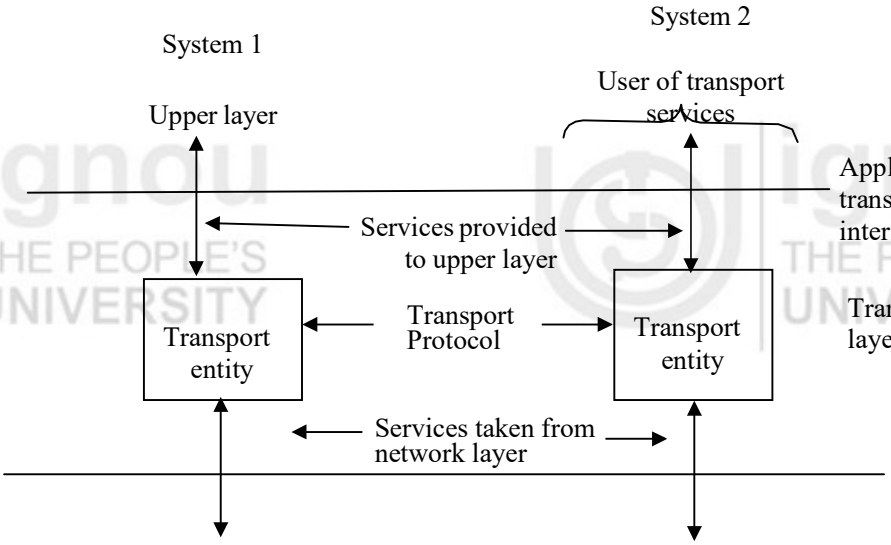


Figure 1: Transport entity environment

As discussed earlier the general service provided by a transport protocol is the end-to-end transport of data in a way that shields the upper layer user from the details of the underlying networks infrastructure. The following categories of services are generally used for describing the services provided by the transport layers.

- Types of services
- Quality of service
- Data transfer
- Connection management
- Expedited delivery

1.2.1 Types of Services

There are generally two basic types of services: (i) Connection-oriented and (ii) Connectionless, or datagram services provided by the Internet. A connection-oriented service is the most common type of protocol service available. It provides the establishment, maintenance, and termination of a logical connection between users. This is also **called handshaking** procedures. The connection-oriented service generally implies that the service is reliable which includes features such as flow control, error control and sequence delivery. Flow control makes sure that neither side of a connection overwhelms the other side by sending too many pockets too quickly.

Connection-Oriented: The Internet connection oriented service is accomplished using TCP (Transmission Control Protocol). Reliable transport, flow control and congestion control are types of services that are provided to an application by TCP. These services are acknowledged. The TCP's congestion control mechanism is used to maintain throughput.

Connectionless Service: There is no handshaking here before sending the packet. There are also no flow control and congestion control mechanisms. Since there is no handshaking procedure, data can be transferred faster. But there is no reliable data transfer either as these services are acknowledged. The Internets connectionless service is called UDP (User Datagram Protocol). Some of the applications of connectionless service are internet telephony and video conferencing.

The connection-oriented transport service is similar to the connection oriented network service. So the question is, why do we require two different layers to provide the same type of service. The answer to this question is that the transport code runs on the user's machine whereas the network layer runs mostly on the routers, which are controlled by the carrier which provide poor service. Therefore, the only possibility is to put another layer on top of the network layer, that improves the quality of service.

It is due to the transport layer that application programmers can write programs according to standard sets of library procedure calls and have these programs run on networks without worrying about dealing with different subnet interfaces and unreliable transmission.

Thus, there is a place at the transport level for both a connection-oriented and a connectionless type of service.

1.2.2 Quality of Service

The transport protocol entity should allow the upper layer protocol users to specify the types of quality of transmission service to be provided. Based on these specifications the transport entity attempts to optimise the use of the underlying link, network, and other resources to the best of its ability, so as to provide the collective requested services. But these services are limited to the internet capabilities of the network layer services.

Examples of services that might be requested are:

- Expedited delivery of packet
- Acceptable error and loss levels of packet
- Desired average and maximum delay in transmission of a packet
- Desired average and minimum throughout
- Priority levels.

You are aware that IP is a standard protocol for the network layer. IP does provide a quality-of-service parameter such as priority as well as a binary specification for normal or low delay, normal or high throughput, and normal or high reliability. Thus, the transport entity can make a request to the internetwork entity. The network may alter flow control parameters and the amount of network resources allocated on a virtual circuit to achieve desired throughput. The transport layer may also split one transport connection among multiple virtual circuits to enhance throughput. This will be explained in section 1.3.2.

You must be aware that different applications may require different quality of services. For example:

- A file transfer protocol might require high throughput. It may also require high reliability to avoid retransmissions at the file transfer level.
- A transaction protocol (e.g., web browser-web server) may require low delay.
- An electronic mail protocol may require multiple priority levels.

One approach to providing a variety of qualities of service is to include a quality-of-service facility within the protocol; the transport protocols typically follow the same approach. An alternative is to provide a different transport protocol for different

classes of traffic; this is to some extent the approach taken by the ISO-standard family of transport protocols.

An application layer protocols need from the Transport layer [Ref.2]. These are:

(i) Reliable data transfer
(ii) Bandwidth
(iii) Timing/delay.

(i) **Reliable data transfer**: By reliable data transfer means that there is not a single bit of loss of data during transmission. There are certain types of application, which does not tolerate any loss of data, such as financial applications. In particular, a loss of file data, or data in a financial transaction, can have devastating consequences (in the latter case, for either the bank or the customer). Other applications in the category are transfer of web documents, electronic mail, file transfer and remote host access. But there are some applications which can tolerate some amount of data loss. For example, multimedia applications such as real-time audio/video. In these multimedia applications, lost data might result in a small glitch while playing the audio/video. The effects of such a loss on application quality and actual amount of tolerable packet loss, will depend strongly on the application and the coding scheme used.

(ii) **Bandwidth:** The concept of bandwidth has been explained in the first block. Just to recall, higher the bandwidth more the channel capacity. There are certain applications, which are **bandwidth sensitive**. For example, the Internet telephony, requires a given amount of bandwidth. But there are some other types of application called elastic application which can make use of as much or as little bandwidth as happens to be available. Electronic mail, file transfer, and Web transfers are all elastic application [Ref.2] of course; the more bandwidth, and the better would be transport capacity.

(iii) **Timing/delay:** The final service requirement is that of timing. There are certain applications, which require tight timing constraints on data delivery in order to be effective. For example, interactive real-time applications, such as Internet telephony, virtual environments, teleconferencing, and multiplayer games. Many of these applications require that end-to end delays be on the order of a few hundred milliseconds or less. Long delays in Internet telephony, and a long delay between taking an action and judging the response from the environment in a multiplicity game tends to result in unnatural pauses in the conversation.

1.2.3 Data Transfer

The whole purpose, of course, of a transport protocol is to transfer data between two transport entities. Both user data and control data must be transferred from one end to the other end either on the same channel or separate channels. Full-duplex service must be provided. Half-duplex and simplex modes may also be offered to support peculiarities of particular transport users.

Here, you may ask the question if the network layer does a similar task, why it is necessary at the transport layer. The network layer overseas the hop by hop delivery of the individual packet but does not see any relationship between those packets even those belonging to a single message. Each packet is treated as an independent entity. The transport layer on the other hand makes sure that not just a single packet but the entire sequence of packets.

1.2.4 Connection Management

When connection-oriented service is provided, the transport entity is responsible for establishing and terminating connections. Both symmetric and asymmetric procedure for connecting establishment may be provided.
Connection termination can be either *abrupt* or *graceful*. With an abrupt termination, data in transmit may be lost. A graceful termination prevents either side from shutting down until all data has been delivered.

1.2.5 Expedited Delivery

A service similar to that provided by priority classes is the expedited delivery of data. Some data submitted to the transport service may supersede data submitted previously. The transport entity will endeavour to have the transmission facility transfer the data as rapidly as possible. At the receiving end, the transport entity will interrupt the transport service user to notify it of the receipt of urgent data. Thus, the expedited data service is in the nature of an interrupted mechanism, and is used to transfer occasional urgent data, such as a break character from a terminal or an alarm condition. In contrast, a priority service might dedicate resources and adjust parameters such that, on average, higher priority data are delivered more quickly.

☞ Check Your Progress 1

1) List the three types of services provided by Transport layer to Applications layer.

   ……

   ……

   ……

2) Describe the presence of data loss and time sensitiveness for the following applications:

   (a) File Transfer
   (b) E-mail
   (c) Web surfing
   (d) Stored audio/video.

1.3 ELEMENTS OF TRANSPORT LAYER PROTOCOLS

The services provided by the transport layer are implemented by the transport layer protocols such as TCP and UDP. We will talk about them in the next unit. In this section, we will take up important components of the transport layer for discussion. A transport layer deals with hop-by-hop processes.

1.3.1 Addressing

The issue concerned with addressing is simply this-how a user process will set up a connection to a remote user process? How the address of a remote process should be specified? The method generally used is to define transport address to which the process can listen for connection requests. In Internet jargon these end points are called **port or TSAP** (Transport Services Access Points). Similarly, these end points at the network layer are then called NSAP (Network Services Access Points). The following illustration (*Figure 2)* shows the connection between a user processes as host 1 with a remote process (server) as host 2.

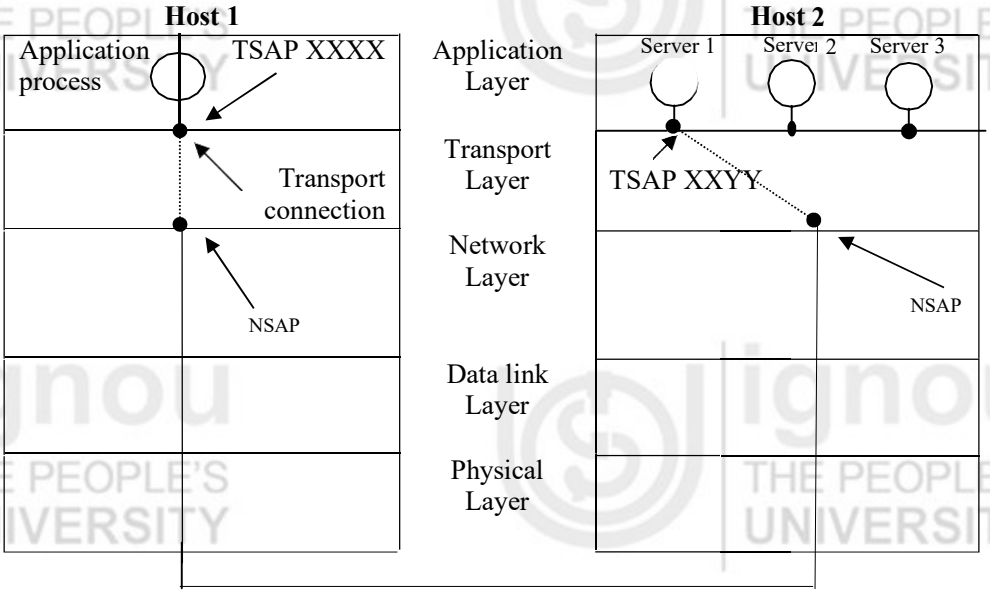


Figure 2: Transport connections between a client and a server

The following steps may be needed for establishing a transport connection [Ref. 1]:

1) Just assume that an application process running on 1 wants to find out the next day's weather forecast, so, it issues a CONNECT request specifying its transport connection point number TSAP XXXX as well as the TSAP number XXYY of the weather forecasting server which will establish transport connection with the destination. This action ultimately results in a transport connection between the application process on host 1 and a server 1 on host 2.

2) Assume that a weather forecasting server is attached to a transport connection at XXYY.

3) The application process then sends a request for the next 10 days weather report.

4) The weather server process responds with the current weather report.

5) The transport connection is then released.

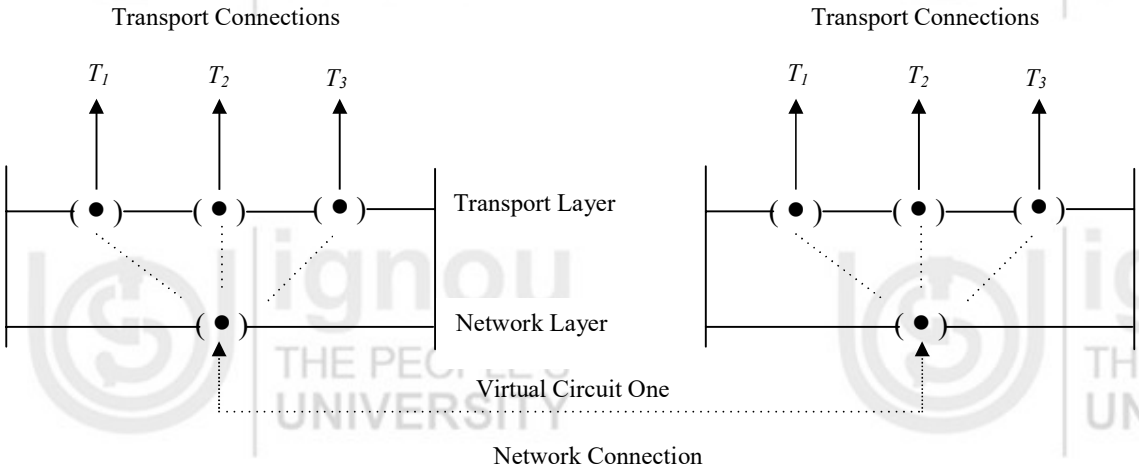
The question, that needs to be addressed: How does the host user know that the address of the destination server process is attached to a particular transport connection. To resolve this issue various strategies are suggested [Refer 1]:

1) **Initial connection protocol:** In the scheme a process server acts as a proxy server for less heavily used servers. Instead of every server listening at a well known port address waiting for a connection request, a process server listens to sets of ports at the same time and informs a particular server to act upon getting a connecting request to perform the required work*. The new server then does the request work, while the process server goes back to listening for new request. Therefore, through a process server a rarely used server should not be listening to ports all the time.

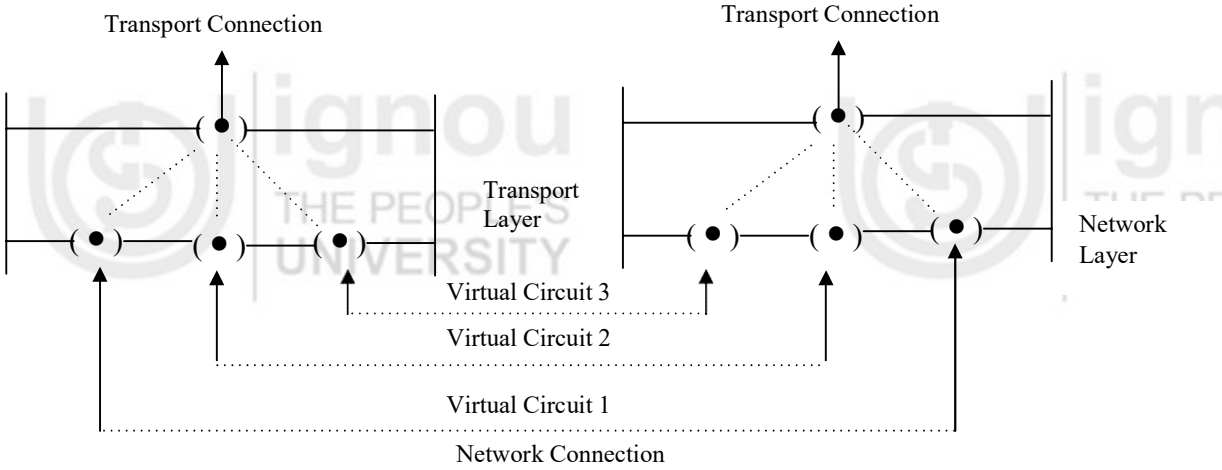
2) Some commonly used services are assigned "well-known address" (for example, time sharing and word processing).
3) A name server or sometimes a directory server is provided. The Transport Service user requests a service through some generic or global name. The request is sent to the name server, which does a directory lookup and returns an address. The transport entity then proceeds with the connection. This service is useful for commonly used applications that change location from time to time. It is also used for those stations in which services do exist independently of a process server. For example, a file server that needs to run on a special hardware (a machine with disk) that cannot be created dynamically when someone want to talk to it [Ref. 1].

1.3.2 Multiplexing

The transport entity may also perform a multiplexing function with respect to the network services that it uses. There are two types of multiplexing techniques that define **upward multiplexing** as the multiplexing of multiple transport connections on a single network connection, and **downward multiplexing** as the splitting of a single transport connection among multiple lower-level connections. This is illustrated in *Figure 3*.



(a) Upward Multiplexing



(b) Downward Multiplexing

Figure 3: Multiplexing

Consider, for example, a transport entity making use of an X.25 service. X.25 is a network layer protocol like IP. Why should the transport entity employ upward multiplexing? There are, after all, 4905 virtual circuits available. In the typical case, this is a more than enough to handle all active Transport Service user. However most X.25 networks base part of their charge on virtual-circuit connect time, as each virtual circuit consumes some node buffer resources. Thus, if a single virtual circuit provides sufficient throughput for multiple TS users, upward multiplexing may be used [*Figure 3*].

On the other hand, **downward multiplexing** or splitting might be used to provide more bandwidth than a single Virtual Circuit can manage. A way out is to open multiple network connections and distribute traffic among them on a round-robin basis, as indicated in *Figure (3b).* This *modus operandi* is called **downward multiplexing.** With *k* network connections open, the effective bandwidth is increased by a factor of *k*. A common example of downward multiplexing occurs with home users who have an ISDN line. This line provides for two separate connections of 64 kbps each. Using both of them to call an Internet provider and dividing the traffic over both lines makes it possible to achieve an effective bandwidth of 128 kbps.

1.3.3 Flow Control and Buffering

There are similarities as well as differences between data link layer and transport layer in order to support flow control mechanism. The similarity is in using the sliding window protocol. The main difference is that a router usually has relatively few lines, whereas a host may have numerous connections. Due to this reason it is impractical to implement the data link buffering strategy in the transport layer.

Whereas flow control is a relatively simple mechanism at the data link layer, it is a rather complex mechanism at the transport layer, for two main reasons:

- Flow control at the transport level involves the interaction of three components: TS users, transport entities, and the network service.
- The transmission delay between transport entities is variable and generally long compared to actual transmission time.

The following delay may arise during the interaction of the two transport entities A and B:

(i) Waiting time to get permission from its own transport entity (interface flow control).

(ii) Waiting time by A to have permission to send the data to B.

(iii) Waiting time as network layer services.

In any case, once the transport entity has accepted the data, it sends out a segment. Some time later, it receives an acknowledgement that the data has been received at the remote end. It then sends a confirmation to the sender.

Now let us come to the flow control problem.

First, we present two ways of coping with the flow control requirement [Ref.3] by using a fixed sliding-window protocol and by using a credit scheme.

The first mechanism is already familiar to us from our discussions of data link layer protocols. The key ingredients are:

- The use of sequence numbers on data units.

- The use of a window of fixed size.
- The use of acknowledgements to advance the window.

This scheme really works well. For example, consider a protocol with a window size of 3. Whenever the sender receives an acknowledgement from a particular segment, it is automatically authorised to send the succeeding seven segments. (Of course, some may already have been sent). Now, when the receiver's buffer capacity comes down to 7 segments, it can withhold acknowledgement of incoming segments to avoid overflow. The sending transport entity can send, at most, seven additional segments and then must stop. Because the underlying network service is reliable, the sender will not time-out and retransmit. Thus, at some point, a sending transport entity may have a number of segments outstanding, for which no acknowledgement has been received. Because we are dealing with a reliable network, the sending transport entity can assume that the segments will come through and that the lack of acknowledgement is a flow control tactic. Such a strategy would not work well in an unreliable network, as the sending transport entity would not know whether the lack of acknowledgement is due to flow control or a lost segment.

The second alternative, a credit scheme, provides the receiver with a greater degree of control over data flow.

The credit scheme decouples acknowledgement from flow control. In fixed sliding-window protocols, used as the data Line layer, the two are synonymous. In a credit scheme, a segment may be acknowledged without granting a new credit, and vice versa. *Figure 4* illustrates the protocol. For simplicity, we have depicted data flow in one direction only. In this example, data segments are numbered sequentially module 8 (e.g., SN 0 = segment with sequence number 0). Initially, through the connection-establishment process, the sending and receiving sequence numbers are synchronised, and A is granted a credit allocation of 7. A advances the trailing edge of its window each time that it transmits, and advances the leading edge only when it is granted a credit.

Figure 4 shows a view of this mechanism from the sending and receiving sides; of course, both sides take both views because data may be exchanged in both directions.

From the receiving point of view, the concern is for received data and for the window of credit that has been allocated. Note that the receiver is not required to immediately acknowledge incoming segments, but may wait and issue a cumulative acknowledgement for a number of segments; this is true for both TCP and the ISO transport protocol.

In both the credit allocation scheme and the sliding window scheme, the receiver needs to adopt some policy concerning the amount of data it permits the sender to transmit. The conservative approach is only to allow new segments up to the limit of available buffer space.

A conservative flow control scheme may limit the throughput of the transport connection in long-delay situations. The receiver could potentially increase throughput by optimistically granting credit for space it does not have. This is called dynamic buffer allocation scheme. For example, if a receiver's buffer is full but it anticipates that it can release space for two segments within a round-trip propagation time, it could immediately send a credit of 2. If the receiver can keep up with the sender, this scheme may increase throughput and can do no harm. If the sender is faster than the receiver, however, some segments may be discarded, necessitating a retransmission. Because retransmissions are not otherwise necessary with a reliable network service, an optimistic flow control scheme will complicate the protocol.

The optimum trade-off between source buffering and destination buffering depends on the type of traffic carried out by the connection. For low bandwidth bursty traffic, such as that produced by an interactive terminal, it is better not to dedicate any buffers, but rather to acquire them dynamically at both ends. Since the sender cannot be sure the receiver will be able to acquire a buffer, the sender must retain a copy of the TPDU(Transport Protocol Data Unit) until it is acknowledged. On the other hand, for file transfer and other high bandwidth traffic, it is better if the receiver dedicates a full window of buffers, to allow the data to flow at maximum speed. Thus, for low-bandwidth bursty traffic, it is better to buffer at the sender, and for high-bandwidth smooth traffic, it is better to buffer at the receiver.

As connections are opened and closed and as the traffic pattern changes, the sender and receiver needs to dynamically adjust their buffer allocations. Consequently, the transport protocol should allow a sending host to request buffer space at the other end. Buffers could be allocated per connection, or collectively, for all the connections running between the two hosts. Alternatively, the receiver, knowing its buffer station (but not knowing the offered traffic) could tell the sender "I have reserved X buffers for you". If the number of open connections should increase, it may be necessary for an allocation to be reduced, so the protocol should provide for this possibility.

In summary, a reasonably general way to manage dynamic buffer allocation is to decouple the buffering from the acknowledgements, in contrast to the sliding window protocols of data link layer.

1.3.4 Connection Establishment

End-to-end delivery can be accomplished in either of two modes: connection-oriented or connectionless. Of these two, the connection-oriented mode is the more commonly used. A connection-oriented protocol establishes a virtual circuit or pathway through the Internet between the sender and receiver. All of the packets belonging to a message are then sent over this same path. Using a single pathway for the entire message facilities, the acknowledgement process and retransmission of damaged or lost frames. Connection-oriented services, therefore, are generally considered reliable.

Connection-oriented transmission has three stages: connection establishment, data transfer, and connection termination.

The network is generally unreliable and congested. Even with a reliable network service, there is a need for connection establishment and termination procedures to support connection-oriented service. Connection establishment serves three main purposes:

- It allows each end to assure that the other exists.
- It allows negotiation of optional parameters (e.g., maximum segment size, maximum window size, quality of service).
- It triggers allocation of transport entity resources (e.g., buffer space, entry in connection table).

To solve the problems arising out of unreliable and congested networks 3-way handshake procedure is used to establish transport connection, which is generally comprised of three steps.

(i) Connection initiation / requirement

(ii) Connection confirmation

(iii) Acknowledgement of confirmation and data transfer.

Similar to the database this is an important issue in computer network. In a network a host as well as a router can crash. Therefore, the issue is its recovery. In case of the loss of Virtual Circuit (In case the network layer provides connection oriented service) due to a crash delivery, a new virtual circuit may be established then probing the transport entity to ask which TPDU it has received and which one it has not received so that the latter can be retransmitted. In case of datagram subnet, the frequency of the lost TPDU is high but the network layer knows how to handle that. The real problem is the recovery from the host crash. Students are requested to read *Computer Network by* ***Tanenbaum*** on this method and also to relate to how a similar problem is resolved in the database (MCS-043).

☞ Check Your Progress 2

1) List the important multiplexing mechanism at the Transport Layer and also explain how they are different from each other.

……

……

……

2) Describe the similarities as well as differences between Data Link Layer and Transport Layer in order to support Flow Control mechanism.

……

……

……

1.4 SUMMARY

In this unit, we have discussed two important components with respect to transport layer namely, transport services and the elements of transport layer protocols. As a part of transport services we discussed type of services, quality of services, connection mechanism. We also outlined three types of services provided by the transport layer through the applications layer. We listed the various types of protocols (applications layer) and the kind of services. We also differentiated between the data link layer and transport layer with respect to flow control mechanism. Similarly, we also differentiated between two types of multiplexing mechanism at the transport layer.

1.5 SOLUTIONS/ANSWERS

Check Your Progress 1

1) Reliable data transfer, Bandwidth, timing delay

2)

	Application	**Data Loss**	**Time-sensitive**
a.	File Transfer	No Loss	No
b.	e-mail	No Loss	No
c.	Web Surfing	No Loss	No
d.	Stored Audio/Video	Loss tolerant	Yes

Check Your Progress 2

1) There are two multiplexing mechanism in the transport layer namely, upward multiplexing and downward multiplexing. Upward multiplexing as the multiplexing of multiple transport connections on a single network connection, and downward multiplexing as the splitting of a single transport connection among multiple lower-level connections. Downward multiplexing or splitting might be used to provide more bandwidth than a single Virtual Circuit can manage. A way out is to open multiple network connections and distribute the traffic among them on a round-robin basis. This *modus operandi* is called downward multiplexing.

2) The similarity is in using the sliding window protocol. The main difference is that an intermediate node usually has relatively few lines, whereas a host may have numerous connections. Due to this reason it is impractical to implement the data link buffering strategy in the transport layer. Whereas flow control is a relatively simple mechanism at the data link layer, it is a rather complex mechanism at the transport layer, for two main reasons:

   - Flow control at the transport level involves the interaction of three components: TS users, transport entities, and the network service.
   - The transmission delay between transport entities is variable and generally long compared to actual transmission time.

1.6 FURTHER READINGS

1) *Computer Networks,* A.S. Tanenbaum, 4th Edition, Prentice Hall of India, New Delhi, 2002.

2) *Computer Networking, A Top down approach featuring the Internet,* James Kurose and Keith W. Ross, Pearson education, New Delhi.

3) *Data and Computer Communications*, William Stalling, 6th Edition, Pearson Education, New Delhi.

UNIT 2 TCP/UDP

Structure **Page Nos.**

2.0 INTRODUCTION

In the previous unit, we discussed the fundamentals of the transport layer which covered topics related to the quality of services, addressing, multiplexing, flow control and buffering. The main protocols which are commonly used such as TCP (Transmission Control Protocol and UDP (User Diagram Protocol) were also discussed. TCP is the more sophisticated protocol of the two and is used for applications that need connection establishment before actual data transmission. Applications such as electronic mail, remote terminal access, web surfing and file transfer are based on TCP.

UDP is a much simpler protocol than TCP because, it does not establish any connection between the two nodes. Unlike TCP, UDP does not guarantee the delivery of data to the destination. It is the responsibility of application layer protocols to make UDP as reliable as possible.

In this unit we will go into details on these two protocols.

2.1 OBJECTIVES

After going through this unit, you should be able to:

- list and explain the various services provided by the Transport Layer;
- establish and release TCP connections;
- describe TCP and UDP header formats;
- describe TCP Flow Control mechanism and how it is different from data link layer, and
- discuss the Congestion mechanism in TCP.

2.2 SERVICES PROVIDED BY INTERNET TRANSPORT PROTOCOLS

The Internet provides two service models: TCP and UDP. The selection of a particular service model is left to the application developers. TCP is a connection oriented and reliable data transfer service whereas UDP is connectionless and provides unreliable data transfer service.

2.2.1 TCP Services

When an application invokes TCP for its transport protocol, the application receives the following services from it:

- Connection oriented service
- Reliable transport service
- Congestion-control mechanism.

Now let us describe these services in brief:

Connection-oriented service: As you are aware from your understanding of the previous unit that the connection oriented service is comprised of the handshake procedure which is a full duplex connection in that, two processes can send messages to each other over the connection at the same time. When the application has finished sending the message, it must remove the connection. The service is referred to as a "connection oriented" service. We are also aware from the discussion on the network layer that this service is implemented through the virtual circuit mechanism.

Reliable transport service: Communicating processes can rely on TCP to deliver all the data that is sent, without error and in the proper order. When one side of the application passes a stream of bytes into a socket, it can count on TCP to deliver the same stream of data to the receiving socket, with no missing or duplicate bytes. Reliability in the Internet is achieved with the use of acknowledgement and retransmissions.

Congestion-control mechanism: TCP also includes a congestion-control mechanism for the smooth functioning of Internet processes. When the packet load offered to the network exceeds its handling capacity congestion builds up.

One point to be noted is that, although, the Internet connection oriented service is bundled with suitable data transfer, flow control and congestion control mechanisms, they are not the essential components of the connection oriented service [Ref. 2].

A connection oriented service can be provided with bundling these services through a different type of a network.

Now we will look at the services TCP does not provide? Some of these are:

i) It does not guarantee a minimum transmission rate,

ii) It does not provide any delay guarantee. But it guarantees delivery of all data. However, it provides no guarantee to the rate of data delivery.

2.2.2 UDP Services

Some of the important features of UDP are as follows:

i) UDP is connectionless, so there is no hand shaking before the two processes start communications.

ii) It provides unreliable data transfer service therefore, there is no guarantee that the message will reach. Due to this, the message may arrive as the receiving process at a random time.

iii) UDP does not provide a congestion-control service. Therefore, the sending process can pump data into a UDP socket at any rate it pleases. Then why do we require such a protocol at the transport layer? There are certain types of applications such as real time. Real-time applications are applications that can tolerate some loss but require a minimum rate. Developers of real-time applications often choose to run their applications over the UDP because their applications cannot wait for acknowledgements for data input. Now coming back to TCP, since it guarantees that all the packets will be delivered to the host, many application protocols, layer protocols are used for these services. For example, SMTP (E-mail), Telnet, HTTP, FTP exclusively uses TCP whereas NFS and Streaming Multimedia may use TCP or UDP. But Internet telephony uses UDP exclusively.

2.3 INTRODUCTION TO UDP

So you might have guessed from the previous section, that UDP is and unreliable transport protocol. Apart from multiplexing /demultiplexing and some error correction UDP adds little to the IP protocol. In this section, we take a look at UDP, at how it works and at what it does.

In case, we select UDP instead of TCP for application development, then the application is almost directly talking to the IP layer. UDP takes messages from the application process, attaches the source and destination port number fields for the multiplexing/demultiplexing service, adds two other small fields, and passes the resulting segment to the network layer. Without establishing handshaking between sending and receiving transport-layer entities, the network layer encapsulates the segment into an IP datagram and then makes a **best-effort** attempt to deliver the segment to the receiving host. If the segment arrives at the receiving host, UDP uses the destination port number to deliver the segment's data to the desired application process.

So you might ask a question then why is UDP required? TCP should be the choice for all types of application layer/protocol. DNS stands for domain name system. It provides a directory service for the internet. It is commonly used by other application protocols (HTTP, FTP etc.) to translate user given host names to IP address. But before we answer your question let-us look at another application called the DNS, which runs exclusively in UDP only but unlike other protocols DNS is not an application with which users interact directly. Instead DNS is a core Internet function that translates the host name to IP addresses. Also unlike other protocols, DNS typically uses UDP. When the DNS application in a host wants to make a query, it constructs a DNS query message and passes the message to the UDP. Without performing any handshaking with the UDP entity running on the destination end system, UDP adds header fields to the message and passes the resulting segment to the network layer. The network layer encapsulates the UDP segment into a datagram and sends the datagram to a name server. The DNS application at the querying host then waits for a reply to its query. If it doesn't receive a reply (possibly because the underlying network lost the query or the reply), either it tries sending the query to another name server, or it informs the invoking application that it can't get a reply.

Like DNS, there are many applications, which are better suited for UDP for the following reasons [Ref 2]:

- *No connection establishment:* Since UDP does not cause any delay in establishing a connection, this is a probably the principal reason why DNS runs

over UDP rather than TCP-DNS which would be much slower if it runs over TCP. HTTP uses TCP rather than UDP, since reliability is critical for Web pages with text.

- *More Client support:* TCP maintains connection state in the end systems. This connection state includes receiving and sending buffers, congestion-control parameters, and sequence and acknowledgement number parameters. We will see in Section 3.5 that this state information is needed to implement TCP's reliable data transfer service and to provide congestion control. UDP, on the other hand, does not maintain connection state and does not track any of these parameters. A server devoted to a particular application can typically support many more active clients when the application runs over UDP rather than TCP. Because UDP, (unlike TCP) does not provide reliable data service and congestion control mechanism, therefore, it does not need to maintain and track the receiving and sending of buffers (connection states), congestion control parameters, and sequence and acknowledgement number parameters.

- *Small packet header overhead.* The TCP segment has 20 bytes of header over-head in every segment, whereas UDP has only eight bytes of overhead.

Now let us examine the application services that are currently using the UDP protocol. For example, remote file server, streaming media, internet telephony, network management, routing protocol such as RIP and, of course, DNS use UDP. Other applications like e-mail, remote terminal access web surfing and file transfer use TCP. Please see reference [2] for further details.

Before discussing the UDP segments structure, now, let us try to answer another question. Is it possible to develop a reliable application on UDP? Yes, it may be possible to do it by adding acknowledgement and retransmission mechanisms, at the application level. Many of today's proprietary streaming applications do just this – they run over UDP, but they have built acknowledgements and retransmissions into the application in order to reduce packet loss. But you understand that it will lead to complete application software design.

UDP Segment Structure

UDP is an end to end transport level protocol that adds only port addresses, checksum error control and length information to the data from the upper layer.

The application data occupies the data field of the UDP segment. For example, for DNS, the data field contains either a query message or a response message. For a streaming audio application, audio samples fill the data field. The packet produced by UDP is called a user datagram.

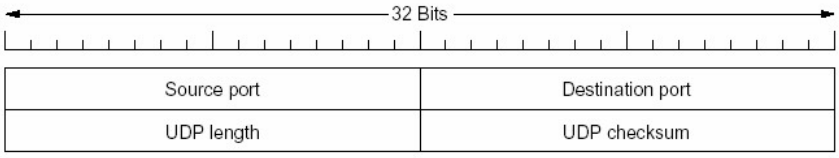


Figure 1: UDP segment structure

The UDP header has only four fields, each consisting of two bytes [*Figure 1*]. Let us discuss each field separately:

- **Source port address**: It is the address of the application program that has created the message.

- **Destination port address:** It is the address of the application program that will receive the message.

- **Total length:** It specifies the length of UDP segment including the header in bytes.

- **Checksum:** The checksum is used by the receiving host to check whether errors have been introduced into the segment. In truth, the checksum is also calculated over a few of the fields in the IP header in addition to the UDP segment.

☞ **Check Your Progress 1**

1) Is it true that if IP is the network layer protocol, so TCP must be the transport layer 1?

……

……

……

2) There are two protocols at the transport layer, which of the two is better?

……

……

……..

2.4 INTRODUCTION TO TCP

TCP is designed to provide reliable communication between pairs of processes (TCP users) across a variety of reliable and unreliable networks and Internets. TCP is stream oriented (Connection Oriented). Stream means that every connection is treated as stream of bytes. The user application does not need to package data in individual datagram as it is done in UDP. In TCP a connection must be established between the sender and the receiver. By creating this connection, TCP requires a VC at IP layers that will be active for the entire duration of a transmission. The data is placed in allocated buffers and transmitted by TCP in segments. In addition, TCP provides two useful facilities for labelling data, push and urgent:

[When the PSH (data push) bit is set, this is an indication that the receiver should pass the data to the upper layer immediately. The URG (Urgent) bit is used to indicate that there is data in this segment that the sending side upper layer entity has marked as urgent. The location of the last byte of this urgent data is indicated by the 16 bit urgent data pointer field 7).

Both IP and UDP treat multiple datagrams belongings to a single transmission as entirely separate units, un-related to each other. TCP on the other hand is responsible for the reliable delivery of entire segments. Every segment must be received and acknowledged before the VC is terminated.

2.5 TCP SEGMENT HEADER

TCP uses only a single type of protocol data unit, called a TCP segment. The header is shown in *Figure 2*. Because one header must perform all protocol mechanisms, it is rather large, with a minimum length of 20 octets. A segment beginning with 9 fixed format 20 byte header may be followed by header option [Ref. 1]. After the options, if any, up to 65,535 – 20 (IP header) – (TCP header) = 65,445 data bytes may follow. A Segment with no data is used, for controlling messages and acknowledgements.

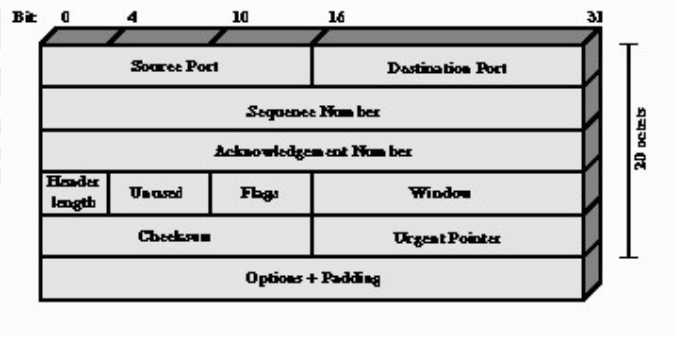


Figure 2: TCP header format

The fields are:

- **Source port (16 bits):** Source service access points and identify local end points of connection.

- **Sequence number (32 bits):** Sequence number of the first data octet in this segment except when SYN is present. If SYN is present, it is the initial sequence number (ISN), and the first data octet is ISN +1.

- **Acknowledgement number (32 bits):** A piggybacked acknowledgement contains the sequence number of the next byte that the TCP entity expects to receive and not for last byte correctly received. Separate number field and acknowledgement number fields are used by the TCP sender and the receiver has to implement a reliable data service.

- **Data offset (4 bits):** Number of 32-bit words in the header.

- **Reserved (6 bits):** Reserved for future use.

- **Flags (6 bits):**

  URG: Used to indicate that there is data in this segment which sends the at the upper layer has marked urgent. The location of the last byte of this urgent data is indicated by the 16 bit urgent data pointer field.

  ACK: Acknowledgement field indicates that the value carried in the ACK field is valid.

  PSH: Push function. The receiver is represented to deliver the data to the application upon arrival, and not buffer it until a full buffer has been received.

  RST: Reset the connection due to host crash or some other reason.

  SYN: Synchronise the sequence numbers.

  FIN: No more data from sender.

- **Receive Window (16 bits):** Used for credit based flow control scheme, in bytes. Contains the number of data bytes beginning with the one indicated in the acknowledgement field that the receiver is willing to accept.

- **Checksum (16 bits):** It provides extra reliability. It Checksums the header, the data and conceptual pseudo header. The Checksum algorithm simply adds up all the 16 bit words in one's complement and then takes one's complement of the sum. As a consequence, when the receiver performs calculations on the entire segment including the checksum field, the result should be 0 [Ref. 1].

- **Urgent data Pointer (16 bits):** Points to the octet following the urgent data; this allows the receiver to know how much urgent data is coming.

- **Options (Variable):** At present, only one option is defined, which specifies the maximum segment size that will be accepted.

Several of the fields in the TCP header warrant further elaboration. The *source port* and *destination port* specify the sending and receiving users of TCP. As with IP, there are a number of common users of TCP that have been assigned numbers; these numbers should be reserved for that purpose in any implementation. Other port numbers must be arranged by agreement between communicating parties.

2.6 TCP CONNECTION ESTABLISHMENT

Before data transmission a connection must be established. Connection establishment must take into account the unreliability of a network service, which leads to a loss of ACK, data as well as SYN. Therefore, the retransmission of these packets have to be done. Recall that a connection establishment calls for the exchange of SYNs.

Figure 3 illustrates typical three-way [Ref. 3] handshake operations.

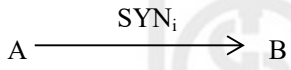
1) Transport entity A (at the sender side) initiates the connection to transport to entity B by setting SYN bit.

2) Transport entity B (at the receiver side) acknowledges the request and also initiates a connection request.

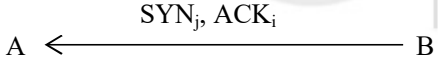
3) A acknowledges to connection request of B and B sends a retransmission and B retransmits.

Now, let us test how this mechanism handles delayed SYN and ACK packets. It is shown that old SYN *X* arrives at B after the close of the relevant connection as shown in *Figure 3* (b). B assumes that this is a fresh request and responds with SYN *j,* ACK *i.* When A receives this message, it realises that it has not requested a connection and therefore, sends an RST, ACK *j*. Note that the ACK *j* portion of the RST message is essential so that an old duplicate RST does not abort a legitimate connection establishment. The final example *Figure 3 (c)* shows a case in which an old SYN, ACK arrives in the middle of a new connection establishment. Because of the use of sequence numbers in the acknowledgements, this event causes no harm.

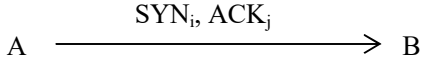
Sender A indicates a connection



Receiver B accepts and acknowledges

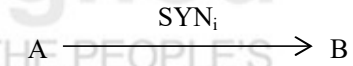


The sender also acknowledges the connection request from the receiver and begins transmission

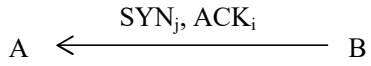


(a) Normal Operation

Obsolete SYN arrives from the previous connection at B



B accepts and acknowledges



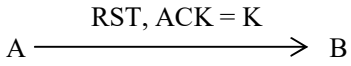
Sender A rejects Receiver B's connection

(b) Delayed arrived of SYN packet

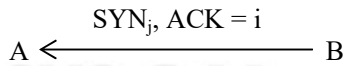
A initiates a connection

old SYN arrives at A

A rejects it



B accepts and acknowledge it



A acknowledges and begins transmission

(c) Delayed SYN and ACK

Figure 3: Three-way handshake mechanism

The three-way handshake procedure ensures that both transport entities agree on their initial sequence number, which must be different. Why do we need different sequence numbers? To answer this, let us assume a case, in which, the transport entities have

picked up the same initial sequence number. After a connection is established, a delayed segment from the previous connection arrives at B, which will be accepted because the initial sequence number turns out to be legal. If a segment from the current connection arrives after some time, it will be rejected by host B, thinking that it is a duplicate. Thus host B cannot distinguish a delayed segment from the new segment.

2.7 TCP CONNECTION TERMINATION

TCP adopts a similar approach to that used for connection establishment. TCP provides for a graceful close that involves the independent termination of each direction of the connection. A termination is initiated when an application tells TCP that it has no more data to send. Each side must explicitly acknowledge the FIN parcel of the other, to be acknowledged. The following steps are required for a graceful close.

- The sender must send FIN *j* and receive an ACK *j*
- It must receive a FIN *j* and send an ACK *j*
- It must wait for an interval of time, to twice the maximum expected segment lifetime.

Acknowledgement Policy

When a data segment arrives that is in sequence, the receiving TCP entity has two options concerning the timing of acknowledgment:

- **Immediate:** When data are accepted, immediately transmit an empty (no data) segment containing the appropriate acknowledgment number.

- **Cumulative:** When data are accepted, record the need for acknowledgment, but wait for an outbound segment with data on which to piggyback the acknowledgement. To avoid a long delay, set a window timer. If the timer expires before an acknowledgement is sent, transmit an empty segment containing the appropriate acknowledgement number.

2.8 TCP FLOW CONTROL

Flow Control is the process of regulating the traffic between two end points and is used to prevent the sender from flooding the receiver with too much data. TCP provides a flow control service to its application to eliminate the possibility of the sender overflowing. At the receivers buffer TCP uses sliding window **with credit scheme** to handle flow control. The scheme provides the receiver with a greater degree of control over data flow. In a **credit scheme** a segment may be acknowledged without the guarantee of a new credit and vice-versa. Whereas in a fixed sliding window control (used at the data link layer), the two are interlinked (tied).

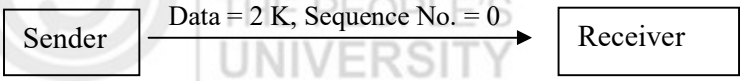
Now, let us understand this scheme with the help of a diagram.

Assume that the sender wants to send application data to the receiver. The receiver has 4 K byte buffer which is empty as shown below:

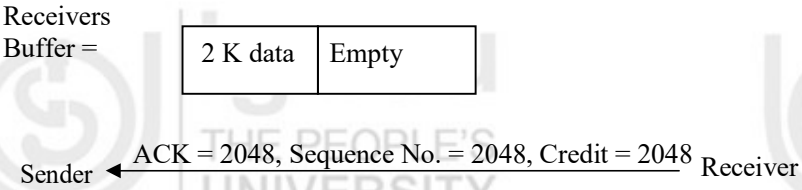
0 4 K byte

1) Initially buffer size = | Empty |

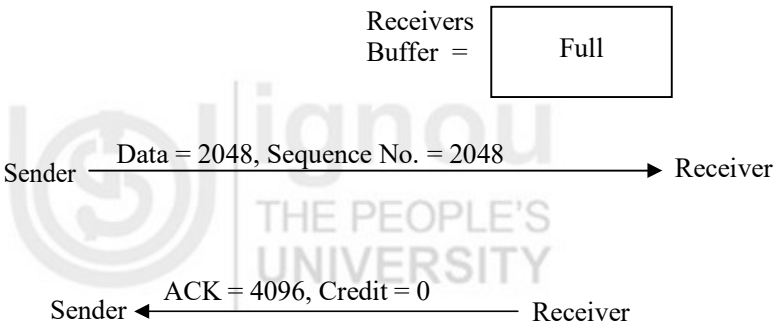
2) Sender transmits 2 K byte segment (data) with sequence number 0 as shown below :



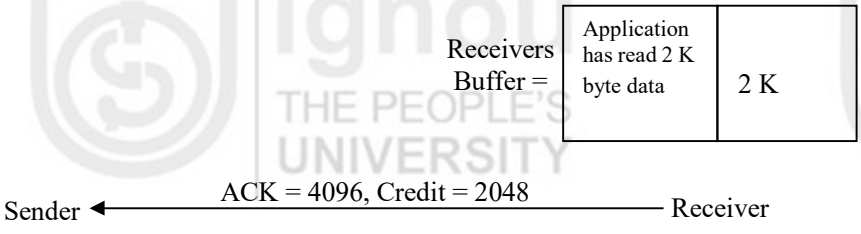
3) The packet is examined at the receiver. After that it will be acknowledged by it. It will also specify the credit value (window size) of the receiver. Until the application process running at the receiver side removes some data from buffer, its buffer size remains fixed at 2048. Therefore, credit value (window size) is 2048 byte.



4) Now the sender transmits another 2048 bytes, which is acknowledged, but the advertised window (credit) is 0.



5) The sender must stop sending data until the application process on the receiving host has removed some data from the buffer, at which time TCP can advertise, a large window (credit value).



When the credit size is zero, normally there is no transmission from the sender side except in two situations:

(i) Urgent data requires to be sent

(ii) Sender wants to know the credit size and the next byte expected by the receiver.

Both senders and receivers can delay the transmission from their side to optimise resources. If a sender knows that the buffer capacity of a receiver window is 8 K and currently it has received just 2 K, then it may buffer it at the sender side till it gets more data from the application process. Similarly, the receiver has to send some data to the sender it can delay the acknowledgement till its data is ready for that the acknowledgement can be **piggybacked.** Therefore, the basic reason of delaying the acknowledgement is to reduce the bandwidth.

Let us consider an example of a log-in session, for example, **Telnet** Session in which the user types one character at a time and the server (log-in server) reacts to every keystroke. You will notice that one character requires four exchanges of IP packets between the log-in session client and a Telnet server which is illustrated below:

Assume that TCP and IP header are 20 bytes each).

1st exchange (at the sender side)

- Whenever application data comes to TCP sender, it creates 20 bytes of a segment, which it gives to IP to create a IP datagram.

Total no. of bytes sent = 20 bytes (TCP) + 20 bytes (IP) + 1 byte for a character (a key stroke) = 41 byte.

2nd exchange (at the receiver side)

- The receiver (Telnet a log-in server) immediately sends an acknowledgement to the sender.

Total no. of bytes sent by the server = 20 bytes (TCP) +20 byte (IP) = 40 bytes. No extra byte is required for an acknowledgement.

3rd exchange (From the log-in server)

- Window update-related message which moves to window one byte right.

Total no. of byte sent – 20bytes (TCP) + 20byte (IP) = 40 bytes.

4th exchange (From the log-in server)

- Echos Character back to the client

Total no. bytes – 20 bytes (TCP) + 20 byte (IP) + 1 byte (0 char) = 41 bytes.

Therefore, the total number of bytes exchanged = 41 + 40 + 40 + 41 = 162 bytes.

So what is the solution to reduce the wastage of bandwidth? The solution has been proposed by **Nagle** and is known as **Nagle's algorithm.**

The algorithm works as follows: When data comes the sender one byte at a time, just send the first byte and buffer all the rest until the outstanding byte is acknowledged. In the meantime, if the application generates some more characters before the acknowledgement arrives, TCP will not transmit the character but buffer them instead. After the acknowledgement arrives TCP transmits all the characters that have been waiting in the buffer in a single segment.

Nagle's algorithm is widely used for the TCP implementation. But in certain cases, the algorithm might not be applicable. For example, in highly interactive applications where every keystroke or a curser movement is required to be sent immediately.

Another problem that wastes network bandwidth is when the sender has a large volume of data to transmit and the receiver can only process its receiver buffer a few bytes at a time. Sooner or later the receiver buffer becomes full. When the receiving application reads a few bytes from the receive buffer, the receiving TCP sends a small advertisement window to the sender, which quickly transmits a small segment and fills the receiver buffer again. This process goes on and on with many small segments being transmitted by the sender for a single application message. This problem is called the **silly window syndrome. It can be avoided if the receiver does not advertise the window until the window size is at least as large as half of the receiver buffer size, or the maximum segment size.** The sender side can cooperate by refraining from transmitting small segments.

To summarise, the silly window syndrome can be explained stepwise as:

Step I: Initially the receiver buffer is full

Step II: Application process reads one byte at the receiver side.

Step III: The TCP running at the receiver sends a window update to the sender.

Step IV: The TCP sender sends another byte.

Step V: The receiver buffer is filled up again.

Steps III, IV and V continue forever. **Clark's** solution is to prevent the receiver from sending a window update for 1 byte. In the solution the receiver window is forced to wait until it has a sufficient amount of space available and then only advertise. Specifically, the receiver should not send a window update until it can handle the maximum segment size it advertised when the connection was established, or its buffer half empty, whichever is smaller.

Nagle's algorithm and **Clark's** solution to the silly window syndrome are complementary. Both solutions are valid and can work together. The goal is for the sender not to send small segments and the receiver not to ask for them. **Nagle** was trying to solve the problem caused by the sending application delivering data to TCP a byte at a time. **Clark** was trying to solve the problem of the receiving application.

2.9 TCP CONGESTION CONTROL

As discussed in the previous section, TCP provides many services to the application process. Congestion Control is one such service. TCP uses end-to-end Congestion Control rather than the network supported Congestion Control, since the IP Protocol does not provide Congestion released support to the end system. The basic idea of TCP congestion control is to have each sender transmit just the right amount of data to keep the network resources utilised but not overloaded. You are also aware that TCP flow control mechanism uses a sliding-window protocol for end-to-end flow control. This protocol is implemented by making the receiver advertise in its acknowledgement the amount of bytes it is willing to receive in the future, called the *advertised window* to avoid the receiver's buffer from overflow. By looking at the advertised window, the sender will resist transmitting data that exceeds the amount that is specified in the advertised window. However, the advertised window does not prevent the buffers in the intermediate routers from overflowing due to which routers

get overloaded. Because IP does not provide any mechanism to control congestion, it is up to the higher layer to detect congestion and take proper action. It turns out that TCP window mechanism can also be used to control congestion in the network.

The protocols designers have to see that the network should be utilised very efficiently (i.e., no congestion and no underutilisation). If the senders are too aggressive and send too many packets, the network will experience congestion. On the other hand, if TCP senders are too conservative, the network will be underutilised. The maximum amount of bytes that a TCP sender can transmit without congesting the network is specified by another window called the *congestion window*. To avoid network congestion and receiver buffer overflow, the maximum amount of data that the TCP sender can transmit at any time is the minimum of the advertised window (receiver window) and the congestion window. Thus, the effective window is the minimum of what the sender thinks is OK and what the receiver thinks is OK. If the receiver advertises for 8 K window but the sender sends 4 K size of data if it thinks that 8 K will congest the network, then the effective windows size is 4K.

The approach used in TCP is to have each sender limit the rate of data traffic into the network as a function of **perceived network congestion**. If a TCP sender perceives that there is small congestion along the path, then it increases its send rate otherwise it reduce it.

The TCP congestion control algorithm dynamically adjusts the congestion window according to the network state. The algorithm is called slow start. The operation of the TCP congestion control algorithm may be divided into three phases: **slow start, congestion avoidance** and **congestion occurrence**. The first phase is run when the algorithm starts or restarts, assuming that the network is empty. The technique, **slow start,** is accomplished by first setting the congestion window to one maximum-size segment. Each time the sender receives an acknowledgement from the receiver, the sender increases the congestion window by one segment. After sending the first segment, if the sender receives an acknowledgement before a time-out, the sender increases the congestion window to two segments. If these two segments are acknowledged, the congestion window increases to four segments, and so on. As shown in the *Figure 4,* the congestion window size grows exponentially during this phase. The reason for the exponential increase is that slow start needs to fill an empty pipe as quickly as possible. The name "slow start" is perhaps a misnomer, since the algorithm ramps up very quickly.

Slow start does not increase the congestion window exponentially forever, since the network will be filled up eventually. Specifically, slow start stops when the congestion window reaches a value specified as the **congestion threshold**, which is initially set to 65,535 bytes. At this point a **congestion avoidance** (2^{nd} phase) phase takes over. This phase assumes that the network is running close to full utilisation. It is wise for the algorithm to reduce the rate of increase so that it will not overshoot excessively. Specifically, the algorithm increases the congestion window **linearly** rather than exponentially when it tries to avoid congestion. This is realised by increasing the congestion window by one segment for each round-trip time.

Obviously, the congestion window cannot be increased indefinitely. The congestion window stops increasing when TCP detects that the network is congested. The algorithm now enters the third phase.

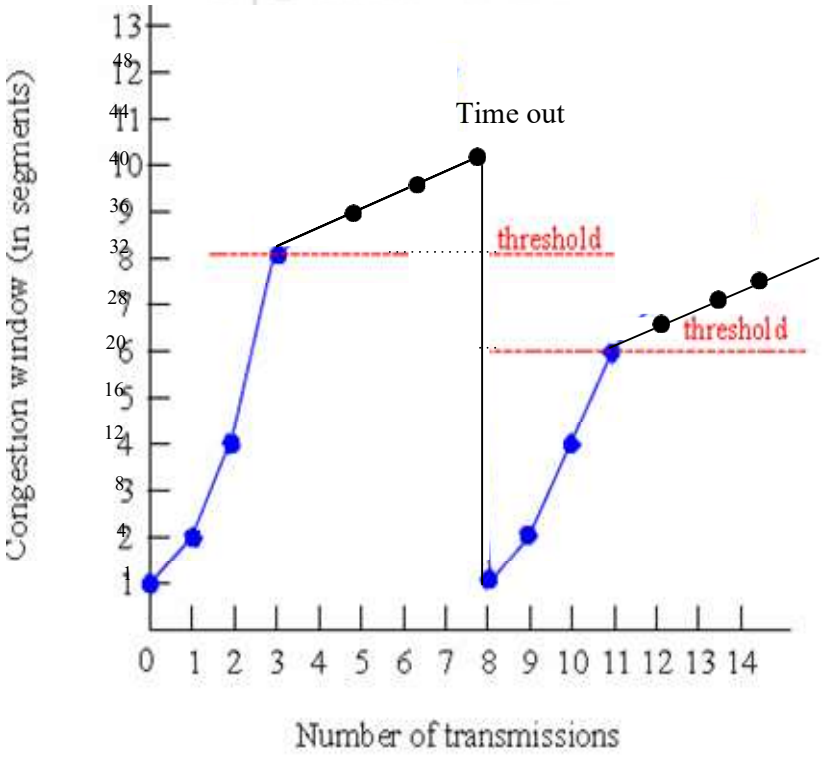


Figure 4: Dynamics of TCP congestion window

At this point the congestion threshold is first set to one-half of the current window size (the minimum of the congestion window and the advertised window, but at least two segments). Next the congestion window is set to one maximum-sized segment. Then the algorithm restarts, using the slow start technique.

How does TCP detect network congestion? There are two approaches:

i) Acknowledgement does not arrive before the timeout because of a segment loss.

ii) Duplicate acknowledgement is required.

The basic assumption the algorithm is making is that, a segment loss is due to congestion rather than errors. This assumption is quite valid in a wired network where the percentage of segment losses due to transmission errors is generally low (less than 1 per cent). Duplicate ACK can be due to either segment reordering or segment loss. In this case the TCP decreases the congestion threshold to one-half of the current window size, as before. However, the congestion window is not reset to one. If the congestion window is less than the new congestion threshold, then the congestion window is increased as in slow start. Otherwise, the congestion window is increased as in congestion avoidance. However, the assumption may not be valid in a wireless network where transmission errors can be relatively high.

The *Figure 4* illustrates the dynamics of the congestion window as time progresses. Thus, assumes that maximum segment size is 1024, threshold to 32K and congestion window to 1 K for the first transmission. The congestion window then grows exponentially (slow start phase) until it hits the threshold (32 K). After that it grows linearly (congestion avoidance phase I). Now, when congestion occurs, the threshold is set to half the current window (20 K) and slow start initiated all over again. In case there is no congestion, the congestion window will continue to grow.

In short, TCP's congestion control algorithm operates as follows:

1) When the Congestion Window is below the threshold, the sender uses for slow start phase and congestion window grows exponentially.

2) When the congestion window is above the threshold the sender is in the congestion avoidance phase and congestion window grows linearly.

You may wish to refer the references [1], [2] and [4] for further clarification on the subject.

2.10 REMOTE PROCEDURE CALL

In this section we will look at two other files access mechanisms such as Network File System and Remote Procedure Call used for accessing files from a server at the client machine. These two mechanism allow programs to call procedures located on remote machines.

Network File System (NFS)

The network file system (NFS) is a file access protocol. FTP and TFTP transfer entire files from a server to the client host. A file access service, on the other hand, makes file systems on a remote machine visible, though they were on your own machine but without actually transferring the files. NFS provides a number of features:

(1) NFS allows you to edit a file on another machine exactly as you would if it were on your own machine.

(2) It even allows you to transfer files from the server to a third host not directly connected to either of you.

Remote Procedure Call (RPC)

NFS works by invoking the services of a second protocol called remote procedure call (RPC). *Figure 5* shows a local procedure call (in this case, a C program calls the open function to access a file stored on a disk).

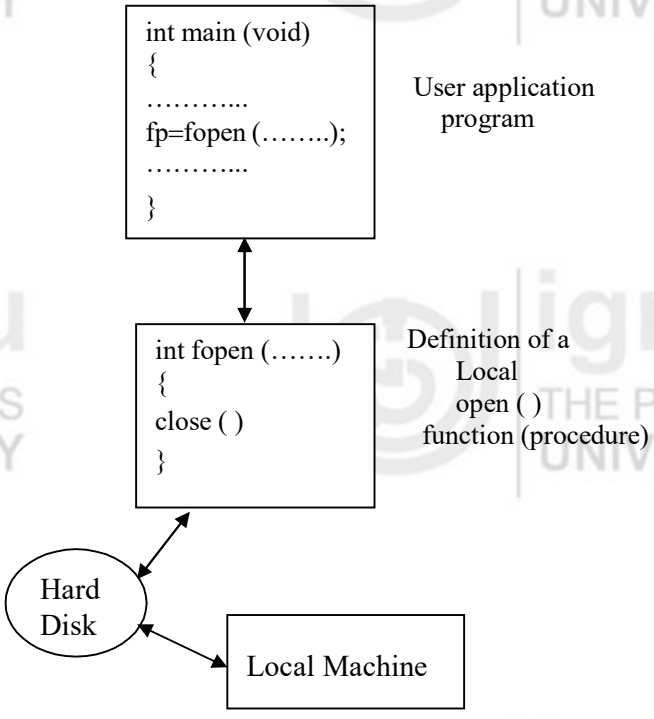


Figure 5: Concept of local procedure call

RPC transfers the procedure call to another machine. Using RPC, local procedure calls are mapped onto appropriate RPC function calls. *Figure 6* illustrates the process: a program issues a call to the NFS client process. The NFS client formats the call for the RPC client and passes it along. The RPC client transforms the data into a different format and provides the interface with the actual TCP/IP transport mechanisms. At the remote host, the RPC server retrieves the call, reformats, and passes it to the NFS server. The NFS server relays the call to the remote disk, which responds as if to a local call and opens the file to the NFS server. The same process is followed in reverse order to make the calling application believe that the file is open on its own host.

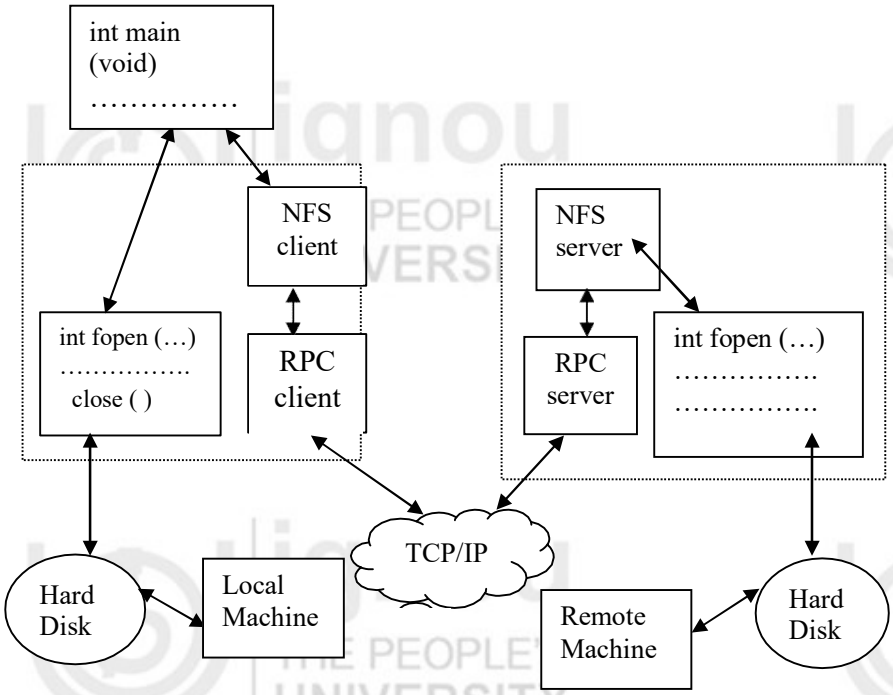


Figure 6: Concept of remote procedure call

There are three independent components in NFS protocol:

(i) NFS itself

(ii) General purpose RPC

(iii) Data representing format (DRF).

In order to make the NFS computer independent what can be done is to use RPC and DRF in other software, (including application), programme by dividing a program into a client side and a server side that use RPC as the chief transport mechanism. On the client side, the transfer procedures may be designated as remote, forcing the compiler to incorporate RPC code into those procedures. On the server side, the desired procedures are implemented by using other RPC facilities to declare them as a part of a server. When the executing client program calls one of the remote procedures, RPC automatically collects values for arguments, from a message, sends the message to the remote server, awaits a response, and stores returned values in the designated arguments. In essence, communication with the remote server occurs automatically as a side-effect of a remote call. The RPC mechanism hides all the details of protocols, making it possible for programmers who know little about the underlying communication protocols but who can write distributed application programs.

☞ **Check Your Progress 2**

1) What is the difference between FTP and NFS?

……

……

……

2) What is a flow control problem? What is the basic mechanism at the transport layer to handle flow control problem?

……

……

……

2.11 SUMMARY

In this unit, we discussed the two transport layer protocols (TCP & UDP) in detail. The transport port can be light weight (very simple) which provide very little facility. In such cases, the application directly talks to the IP. UDP is an example of a transport protocol, which provides minimum facility, with no control, no connection establishment, no acknowledgement and no congestion handling feature. Therefore, if the application is to be designed around the UDP, then such features have to be supported in the application itself. On the other hand, there is another protocol-TCP which provides several feature such as reliability, flow control, connection establishment to the various application services. Nevertheless, the services that the transport layer can provide often constrain the network layer protocol.

We had a close look at the TCP connection establishment, TCP flow control, TCP congestion handling mechanism and finally UDP and TCP header formats. With respect to congestion control, we learned that congestion control is essential for the well-being of the network. Without congestion control the network can be blocked.

2.12 SOLUTIONS/ANSWERS

Check Your Progress 1

1) No, TCP is only part of the TCP/IP transport layer. The other part is UDP (user datagram protocol).

2) It depends on the type of an application. TCP provide a connection-oriented, reliable service (extra overheads). Stream means that the connection is treated as stream of bytes. The user application does not need to package data in individual datagrams.

   Reliable means that every transmission of data is acknowledged by the receiver. UDP offers minimum datagram delivery service.

Check Your Progress 2

1) FTP transfers entire files from a server to the client host. NFS, on the other hand makes file systems on a remote machine visible as they are on your own machine but without actually transferring the files.

2) Flow control is the process of regulating the traffic between points and is used to present the sender from flooding the receiver with too much data.

   The basic mechanism to handle flow control at the transport layer is the sliding window with credit scheme. The credit scheme provides the receiver with a greater degree of control over data flow. In decouples acknowledgement from flow control. In fixed sliding window control such as x.25 and HDLC, the two are synonymous. In a credit scheme, a segment may be acknowledged without granting new credit and vice-versa.

2.13 FURTHER READINGS

1) *Computer Networks,* A.S Tanebaum, 4th Edition, PHI, New Delhi, 2003.

2) *Computer Networking, Computer Networking, A top down approach featuring the Internet,* J.F. Kurose & K.W. Ross, Pearson Edition, New Delhi, 2003.

3) *Data and Computer Communication,* William Stallings, 6th Edition, Pearson Education, New Delhi. 2002.

4) *Communications Networks,* Leon Garcia, and Widjaja, Tata McGraw Hill, 2000.

UNIT 3 NETWORK SECURITY-I

Structure **Page Nos.**

1.0 INTRODUCTION

Cryptography plays an important role in network security. The purpose of cryptography is to protect transmitted information from being read and understood by anyone except the intended recipient. In the ideal sense, unauthorised individuals can never read an enciphered message.

Secret writing can be traced back to 3,000 B.C. when it was used by the Egyptians. They used hieroglyphics to conceal writing from unintended recipients. Hieroglyphics is derived from the Greek word hieroglyphica, which means “sacred carvings”. Hieroglyphics evolved into hieratic, which was easier to use script. Around 400 B.C., military cryptography was employed by the Spartans in the form of strip of papyrus or parchment wrapped around a wooden rod. This system is called a Scytale. The message to be encoded was written lengthwise down the rod on the wrapped material. Then, the material was unwrapped and carried to the recipient. In unwrapped form, the writing appeared to random characters, and to read again the material was rewound on a rod of the same diameter and length.

During 50 B.C., Julius Caesar, the emperor of Rome, used a substitution cipher to transmit messages to Marcus Tullius Cicero. In this, letters of the alphabets are substituted for other letters of the same alphabet. Since, only one alphabet was used, this is also called monoalphabetic substitution. This particular cipher involved shifting the alphabet by three letters and substituting those letters. This is sometimes known as C3 (for Caesar shifting three places).

In this unit, we provide details of cryptography and its needs. In section 3.2 we define cryptography, Section 3.3 presents a brief review of symmetric cryptography. In section 3.4 we discuss the Asymmetric Cryptography. We summarise the unit in section 3.6 followed by the Solutions/Answers.

1.1 OBJECTIVES

After going through this unit, you should be able to:

- learn about the principles and practice of cryptography, and
- various symmetric and public key algorithms.

1.2 CRYPTOGRAPHY

Cryptography is the science of writing in secret code and is an ancient art; the first documented use of cryptography in writing dates back to circa 1900 B.C. when an Egyptian scribe used non-standard hieroglyphics in an inscription. Some experts argue that cryptography appeared simultaneously sometime after writing was invented, with applications ranging from diplomatic missives to war-time battle plans. The new form of cryptography emerges with the widespread development of computer and communications technologies. Cryptography is a key technology when communicating over any un-trusted medium, particularly the Internet. Rivest defines cryptography as simply "Communication in the presence of adversaries". Modern cryptographic techniques have many uses, such as access control, digital signatures, e-mail security, password authentication, data integrity check, and digital evidence collection and copyright protection.

Cryptographic systems are generically classified among three independent dimensions:

- **Types of Operation:** All encryption algorithms are based on two general principles: substitution and transposition. The fundamental requirements are that no information is lost and all operations are reversible.

- **Key Used:** If both sender and the receiver use the same key, then it is referred to as symmetric, single-key, secret-key, or conventional encryption. And if the encryption and decryption key are different, the system is asymmetric key, two-key, or public key encryption.

Within the context of any application-to-application communication, there are some specific security requirements: (1) Authentication, (2) Privacy/confidentiality, (3) Integrity, and (4) Non-repudiation. There are, in general, three types of cryptographic schemes typically used to accomplish these goals: (1) secret key (or symmetric) cryptography, (2) public key (or asymmetric) cryptography, and (3) hash functions, each of which is described in the subsequent subsections.

☞ Check Your Progress 1

1) What is cryptography?

……

……

……

2) List the various requirements of application-to-application communication.

……

……

……

3) List various types of cryptographic techniques.

……

……

……

1.3 SYMMETRIC KEY CRYPTOGRAPHY

About twenty years ago, cryptography was the art of making and breaking codes. The codes were used to transfer messages over an unsecured channel. The channel should be protected from intruders who read, insert, delete or modify messages, as depicted in *Figure 1*. The transmission of a message is done using an encryption function E that converts the message or plaintext using the key into a cipher text. The receiver does the reverse of this operation, using the decryption function D and a decryption key to recover the plaintext from the cipher text. The key is distributed in advance over a secure channel, for example by courier.

Normally, the encryption and decryption function, but not the key, are considered known to the adversary, so the protection of the information depends on the key only. If the enemy knows the key, the whole system is useless until a new key is distributed. In secret Key Cryptography, a single key is used for both encryption and decryption, as shown in *Figure 1.* The main difficulty with this approach is the distribution of the key.

Secret key cryptography schemes are generally categorised as being either stream ciphers or block ciphers. Stream ciphers operate on a single byte (or computer word) at a time, and implement some form of feedback mechanism so that the key is constantly changing. Block cipher scheme encrypts one block of data at a time using the same key on each block.

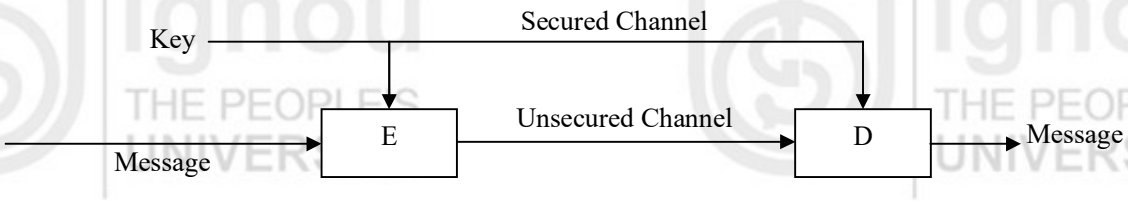


Figure 1: Secret Key Cryptography

The most common secret key cryptography scheme is Data Encryption Standard (DES), designed by IBM in 1970s and adopted by the National Bureau of Standard (NBS) of USA [now the National Institute for Standards and Technology (NIST)] in 1977 for commercial and unclassified government applications. DES is a block-cipher employing a 56-bit key operating on 64 bit blocks of data. DES has a complex set of rules and transformations designed basically for fast hardware implementation and slow software implementation. The latter point is significant in today's context, as the speed of CPU is several orders of magnitude faster than twenty years ago. IBM also proposed a 112-bit key for DES in the 1990s, which was never implemented seriously.

In 1997, NIST initiated the task of developing a new secure cryptosystem for U.S government applications. This effort has resulted in AES (Advanced Encryption Standard). AES became the official successor to DES in December 2001. AES is based on Rjndael Algorithm and employs a 128-, 192-, or 256-bit key.

The following terminology is often used when symmetric ciphers are examined:

Block Ciphers

A block of cipher transforms *n*-bit plaintext blocks to *n*-bit ciphertext blocks on the application of a cipher key *k*. The key is generated at random using various mathematical functions as shown in *Figure 2.*

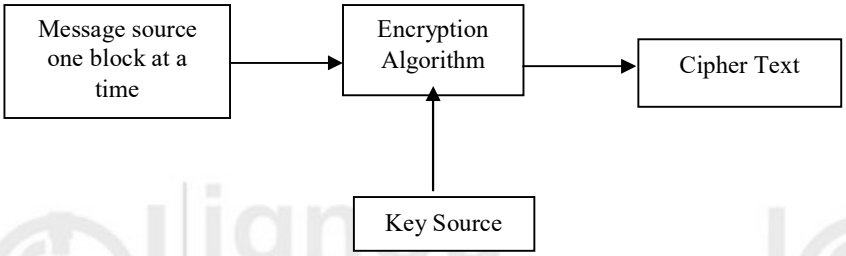


Figure 2: Block cipher

Stream Ciphers

A stream cipher consists of a state machine that outputs at each state transition one bit of information. This stream of output bits is called the *running key*. Just XORring the running key to the plaintext message can implement the encryption. The state machine is a pseudo-random number generator. For example, we can build one from a block cipher by encrypting repeatedly its own output. Typically, more elaborate constructions (for high speed) are used for stream ciphers. Some of the more well-known stream ciphers are RC4 and SEAL. Several stream ciphers are based on linear-feedback shift registers (LFSR) (*Figure 3*), such as A5/1 used in the GSM. These have the benefit of being very fast (several times faster than usual block ciphers).

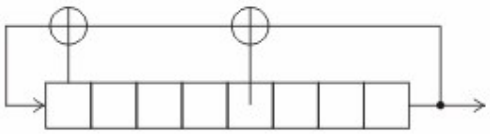


Figure 3: Stream cipher linear feedback shift register (LFSR)

SSSC (Self-Synchronising Stream Ciphers)

The class of *self-synchronsing stream ciphers* has property that it corrects the output stream after the bit flips or even drops bits after a short time.
SSSCs can be constructed using block ciphers in a CFB-related way. Assume, that we have already encrypted *n* bits of the message and know that much of the ciphertext (where *n* denotes the block length of the cipher). Then we produce a new running key bit by encrypting the *n* ciphertext bits. Take one bit of the output of the cipher to be the new running key bit. Now moving one bit further we can iterate this procedure for the whole length of the message.

One bit error in a ciphertext cannot affect the decrypted plaintext after *n* bits. This makes the cipher self-synchronising.

The block cipher used should have sufficiently large block size to avoid substitution attacks.

Substitution and Transposition

A substitution (*Figure 4*) means replacing symbols or group of symbols by other symbols or groups of symbols. Transposition (*Figure 5*), on the other hand,

means permuting the symbols in a block. Neither of these operations alone provide sufficient security, but strong ciphers can be built by combining them. A Substitution-Permutation Network (SPN) means a cipher composed of a number of stages, each involving substitutions and permutations. The prominent examples of SPNs are DES and CAST-128.

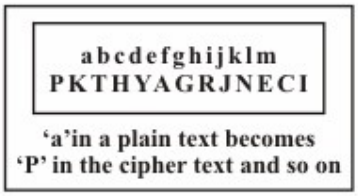


Figure 4: Substitution

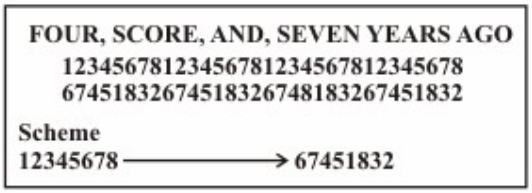


Figure 5: Transposition

S-Boxes

Lookup tables that map *n* bits to *m* bits, where *n* and *m* are often equal. There are several ways of constructing and measuring good S-boxes for ciphers.

a) S-boxes, which are resistant against well known attacks, can be created using rigorous mathematical approach by applying bent functions (or related).

b) Other designers apply heuristic approaches that result in S-boxes that are more difficult to handle in mathematical proofs, but can have additional benefits.

The S-box may even be the only non-linear part of the cipher (e.g., DES) and thus, may be considered as the single most important part of the cipher. In fact, DES's S-boxes are so good that it is used in many other cipher design (for example, Serpent).

Feistel Networks

The original idea was used in the block cipher, Lucifer, invented by **Horst Feistel.** Several variations have been devised from the original version. A **Feistel** network (*Figure 6*) is a general way of constructing block ciphers from simple functions. The standard **Feistel** network takes a function from *n* bits to *n* bits and produces an invertible function from *2n* bits to *2n* bits. The structure of **Fiestel** network is based on round function. The essential property of **Feistel** networks that makes them so useful in cipher design is that the round function need not be invertible, but always is the resulting function. If the round function depends on, say, *k* bits of a key, then the **Feistel** cipher requires *rk* bits of the key where *r* is the number of rounds used. The security of the **Feistel** structure is not obvious. It is compulsory that a **Feistel cipher** has enough number of rounds, but just adding more rounds does not always guarantee security.

The process of taking the user key and expanding it into *rk* bits for the **Feistel** rounds is called **key scheduling**. This is often a non-linear operation and hence does not

directly provide any information about the actual user key. There are many ciphers that have this basic structure; Lucifer, DES, and Twofish etc.

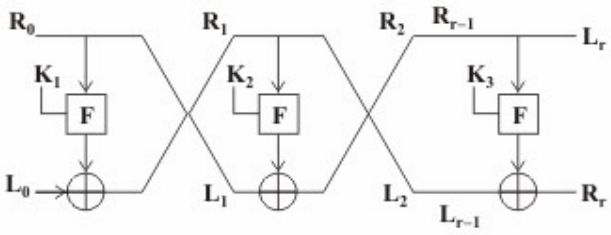


Figure 6: Feistal Cipher

Expansion, Permutation

They are linear operations, and thus, not sufficient to guarantee security. They are common tools in mixing bits in a round function and when used with good non-linear S-boxes (as in DES) they are vital for the security because they propagate the non-linearity uniformly over all bits.

Bitslice Operations (Bitwise Logic Operations XOR, AND, OR, NOT and Bit Permutations)

The idea of bitslice implementations of block ciphers is due to **Eli Biham.** It is common in vector machines to achieve parallel operation. However, **Biham** applied it on serial machines by using large registers as available in modern computers. The term "bitslice" has been the contribution of Matthew Kwan.

All block ciphers can be designed in bitslice manner, but this could affect the speed of operations such as addition and multiplication they may become very slow. On the other hand, permutations are almost free as they only require a *renaming* of the registers and this can be done at the coding level. Thus, for example, in DES exhaustive key search using bitslice techniques, one can increment the current key in, a fraction of a time than is usually needed for key scheduling.

The AES finalist *Serpent* is designed to be implemented using bitslice operations only. This makes it particularly efficient on modern architectures with many registers.

Modes of Operation

Block ciphers are the most commonly used cipher. Block ciphers transform a fixed-size block of data (usually 64 bits) into another fixed-size block (possibly *64* bits wide again) using a function selected by the key. If the key, input block and output block have all *n* bits, a block cipher basically defines a one-to-one mapping from *n*-bit integers to permutations of *n*-bit integers.

If the same block is encrypted twice with the same key, the resulting ciphertext blocks are also the same (this *mode* of encryption is called *electronic code book*, or **ECB**). This information could be useful for an attacker. For identical plaintext blocks being encrypted to different ciphertext blocks, three standard modes are generally used:

1) **CBC (Cipher Block Chaining):** First XORing the plaintext block with the previous ciphertext block obtains a ciphertext block, and then the resulting value needs to be encrypted. In this, leading blocks influence all trailing blocks, which increases the number of plaintext bits one ciphertext bit depends on, but this may also leads to synchronisation problems if one block is lost. The Process of Cipher Block Chaining shown in *Figure 7.* In this Figure m_1, m_2, m_3 are messages and C_1, C_2 and C_3 are ciphertexts.

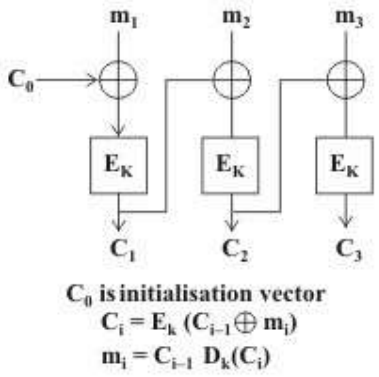


Figure 7: Cipher block chaining

2) **CFB (Cipher Feedback):** In this the *k*th ciphertext block is obtained by encrypting the *(k-1)*th ciphertext block and XORing the result onto the plaintext. The CFB mode possess self-synchronising property: A CFB feedback loop can also be used as a pseudo-random number generator if one simply feeds one block of true random data with trailing blocks of zeroes into the encryption routine (although the expected period of this PRNG would be only about $2^{n/2}$ where *n* is the block size of the cipher). CFB is shown in *Figure 8*.

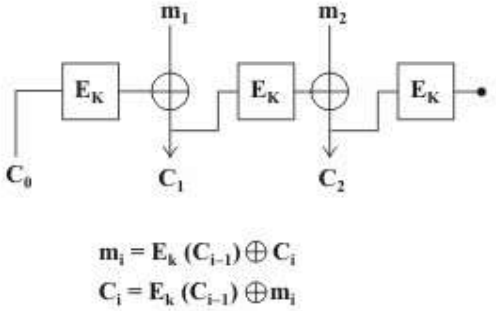


Figure 8: Cipher feedback model

3) **OFB (Output Feedback Block):**It is used as a synchronous key-stream generator, whose output is XORed with the plaintext to obtain ciphertext, block by block. The key-stream is generated by riterative encryption, starting with an initialisation vector (IV) which, along with the key, is agreed upon beforehand. OFB is shown in *Figure 9*.

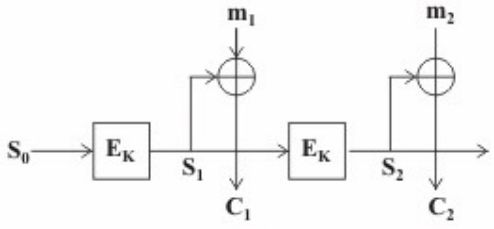


Figure 9: Output feedback block

The One-Time Pad

The one-time pad (OTP) is the only cipher that has been proven to be unconditionally secure, i.e., unbreakable in practice. Further, it has been proven that any unbreakable, unconditionally secure cipher must be a one-time pad.

The Vernam cipher (invented by **G. Vernam** in 1917) in one time pad and is very simple: take a stream of bits that contain the plaintext message, and a key, which is the secret random bit-stream of the same length as the plaintext. [To encrypt the plaintext with the key, sequentially exclusive-or each pair of key bit and plaintext bit to obtain the ciphertext bit]. If the key is truly random, the attacker or adversary has no means of deciding whether some guessed plaintext is more likely than any other when, only the ciphertext and no knowledge of the plaintext is available.

The main practical problem is that the key does not have a small constant length, but the same length as the message, and one part of a key should never be used twice (or the cipher can be broken). However, this cipher has allegedly been in widespread use since its invention, and even more since the security proof by **C. Shannon** in 1949. Although, admittedly the security of this cipher had been conjectured earlier, it was **Shannon** who actually found a formal proof for it.

DES

The Data Encryption Standard (**DES**) is an algorithm developed in the mid-1970s and turned into a standard by the US National Institute of Standards and Technology (NIST), and was also adopted by several other governments worldwide. It was and still is widely used, especially in the financial industry.

DES is a block cipher with a 64-bit block size. It uses 56-bit keys. This makes it suspectible to exhaustive key search with modern computing powers and special-purpose hardware. DES is still strong enough to keep most random hackers, adversaries and individuals out, DES is easily breakable with special hardware by, government, criminal organizations etc. DES is getting too weak, and should not be used in new applications. NIST proposed in 2004 to withdraw the DES standard.

A variant of DES, Triple-DES (also **3DES**) is based on using DES three times (normally in an encrypt-decrypt-encrypt sequence with three different, unrelated keys). The Triple-DES is arguably much stronger than (single) DES, however, it is rather slow compared to some new block ciphers.

Although, at the time of DES's introduction, its design philosophy was held secret. Some information has been published about its design, and one of the original designers, **Don Coppersmith**, has said that they discovered ideas similar to differential crypt analysis while designing DES in 1974. However, it was just a matter of time that these fundamental ideas were re-discovered.

Although DES is no longer considered a practical solution, it is many a time used to describe new crypt analytical methods. Even today, there are no crypt analytical techniques that would completely break DES in a structural way; indeed, the only real weakness known is the short key size (and perhaps the small block size).

DES uses the:

- Data Encryption Algorithm (DEA),
- a secret key block-cipher employing a 56-bit key operating on 64-bit blocks.
- FIPS 81 describes four modes of DES operation: Electronic Codebook (ECB), Cipher Block Chaining (CBC), Cipher Feedback (CFB), and Output Feedback (OFB). Despite all these options, ECB is the most commonly deployed mode of operation.

DES uses a 56-bit key, which is divided into eight 7-bit blocks and an 8th odd parity bit is added to each block (i.e., a "0" or "1" is added to the block so that there are an odd number of 1 bit in each 8-bit block). By using the 8 parity bits for rudimentary error detection, a DES key is actually 64 bits in length for computational purposes (although it only has 56 bits worth of randomness, or *entropy*).

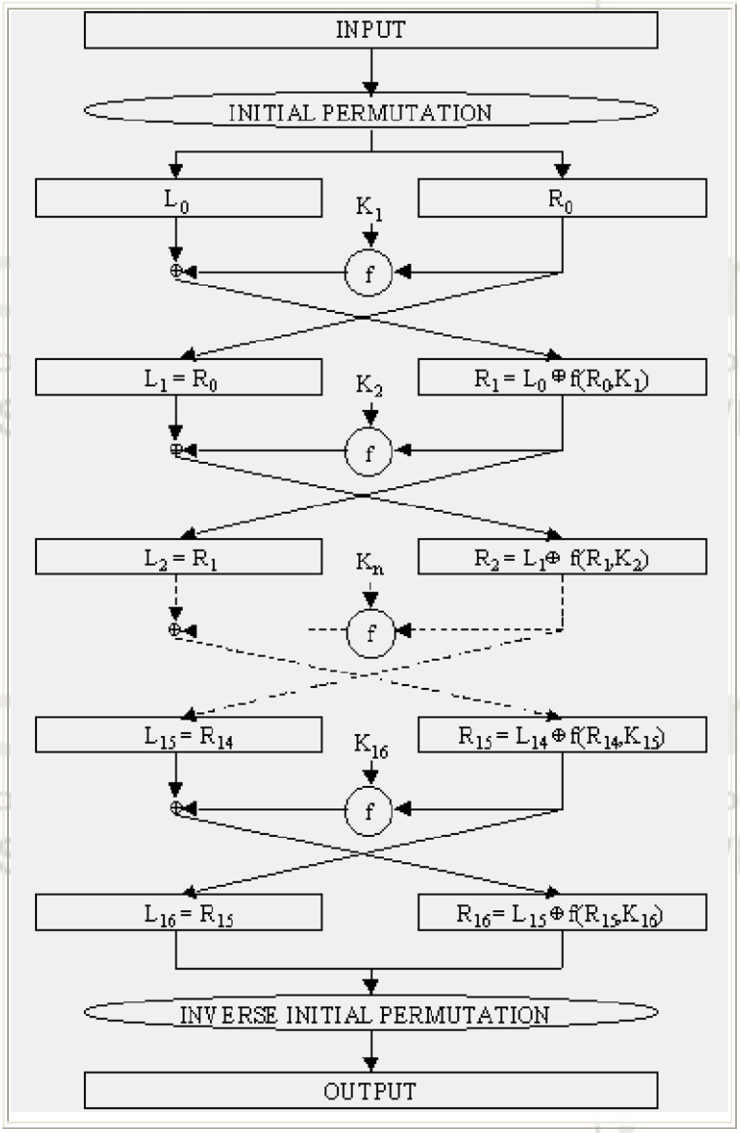


Figure 10: DES algorithm

DES process on 64-bit blocks of the plaintext, invoking 16 rounds of permutations, swaps, and substitutes, as shown in *Figure 10*. The main DES steps are:

1) The 64-bit block undergoes an initial permutation (IP), where each bit is moved to a new bit position; e.g., the 1st, 2nd, and 3rd bits are moved to the 58th, 50th, and 42nd position, respectively.
2) The permuted 64-bit input is divided into two 32-bit blocks, called *left* and *right*, respectively. The initial values of the left and right blocks are denoted as L_0 and R_0.
3) There are then 16 rounds of operation on the L and R blocks.

During each iteration (where *n* ranges from 1 to 16), the following formula is used:

$$L_n = R_{n-1}$$
$$R_n = L_{n-1} \text{ XOR } f(R_{n-1}, K_n)$$

The new L block value is taken from the prior R block value. The new R block is calculated by taking the bit-by-bit exclusive-OR (XOR) of the prior L block with the results of applying the DES cipher function, *f*, to the prior R block and K_n. (K_n is a 48-bit value derived from the 64-bit DES key). Each round uses a different 48 bits according to the standard's Key Schedule algorithm.

The cipher function, f, combines the 32-bit R block value and the 48-bit subkey in the following way:

- First, the 32 bits in the R block are expanded to 48 bits by an expansion function (E); the extra 16 bits are found by repeating the bits in 16 predefined positions.
- The 48-bit expanded R-block is then ORed with the 48-bit subkey.
- The result is a 48-bit value that is then divided into eight 6-bit blocks.
- These are fed as input into 8 selection (S) boxes, denoted $S_1,...,S_8$. Each 6-bit input yields a 4-bit output using a table lookup based on the 64 possible inputs; this results in a 32-bit output from the S-box. The 32 bits are then rearranged by a permutation function (P), producing the results from the cipher function.

The results from the final DES round — i.e., L_{16} and R_{16} — are recombined into a 64-bit value and fed into an inverse initial permutation (IP^{-1}). At this step, the bits are rearranged into their original positions, so that the 58th, 50th, and 42nd bits, for example, are moved back into the 1st, 2nd, and 3rd positions, respectively. The output from IP^{-1} is the 64-bit ciphertext block.

Breaking DES

DES's 56-bit key was too short to withstand a brute-force attack from computing power of modern computers. Remember Moore's Law: computer power doubles every 18 months. Keeping this law in mind, a key that could withstand a brute-force guessing attack in 2000 could hardly be expected to withstand the same attack a quarter of a century later.

DES is even more vulnerable to a brute-force attack as it is commonly used to encrypt words, meaning that the entropy of the 64-bit block is greatly reduced. That is, if we are encrypting random bit streams, then a given byte might contain any one of 2^8 (256) possible values and the entire 64-bit block has 2^{64}, or about 18.5 quintillion, possible values. If we are encrypting words, however, we are most likely to find a limited set of bit patterns; perhaps 70 or so if we account for upper and lower case letters, the numbers, space, and some punctuation. That is, only about ¼ of the bit combinations of a given byte are likely to occur. Despite this, the U.S. government insisted throughout the mid-1990s that 56-bit DES was secure and virtually unbreakable if appropriate precautions were employed. In response, RSA Laboratories sponsored a series of cryptographic challenges to prove that DES was no longer secure and appropriate for use.

- **DES Challenge I** was launched in March 1997. R. Verser completed it in 84 days in a collaborative effort to break DES using thousands of computers on the Internet. The first DES II challenge lasted 40 days during early 1998. This problem was solved by distributed.net, a worldwide distributed computing

network using the spare CPU cycles of computers around the Internet. The participants in distributed.net's activities load a client program that runs in the background, conceptually similar to the SETI @Home "Search for Extraterrestrial Intelligence" project). By the end of the project, distributed.net systems were checking 28 *billion* keys per second.

- The second **DES II challenge** lasted just less than 3 days. On July 17, 1998, the Electronic Frontier Foundation (EFF) announced the construction of a hardware that could brute-force a DES key in an average of 4.5 days. The device called Deep Crack, could check 90 billion keys per second and cost only about $220,000 including design. As the design is scalable, this suggests that an organisation could develop a DES cracker that could break 56-bit keys in an average of a day for as little as $1,000,000.
- The **DES III challenge**, launched in January 1999, was broken in less than a day by the coordinated efforts of Deep Crack and distributed.net.

The Deep Crack algorithm is to launch a brute-force attack by guessing every possible key. A 56-bit key yields 2^{56}, or about 72 quadrillion, possible values. The DES cracker team initially assumed that *some* recognisable plaintext would appear in the decrypted string even though they didn't have a specific known plaintext block. They then applied all 2^{56} possible key values to the 64-bit block. The system checked to find if the decrypted value of the block was "interesting", which they defined as bytes containing one of the alphanumeric characters, space, or some punctuation. As the likelihood of a single byte being "interesting" is about ¼, then the likelihood of the entire 8-byte stream being "interesting" is about $¼^8$, or 1/65536 ($½^{16}$). This dropped the number of possible keys that might yield positive results to about 2^{40}, or about a trillion.

After the assumption that an "interesting" 8-byte block would be followed by another "interesting" block. Therefore, if the first block of ciphertext decrypted to something interesting, they decrypted the next block; otherwise, they abandoned this key. Only if the second block was also "interesting" did they examine the key closer. Looking for 16 consecutive bytes that were "interesting" meant that only 2^{24}, or 16 million, keys required to be examined further. This further examination was primarily to see if the text made any sense.

It is important to mention mentioning a couple of forms of crypt analysis that have been shown to be effective against DES. *Differential cryptanalysis*, invented in 1990 by **E. Biham** and **A. Shamir** (of RSA fame), is a chosen-plaintext attack. By selecting pairs of plaintext with particular differences, the crypt analyst examines the differences in the resultant ciphertext pairs. *Linear plaintext*, invented by **M. Matsui**, uses a linear approximation to analyse the actions of a block cipher (including DES). Both these attacks one be more efficient than brute force.

DES Variants

After DES algorithm was "officially" broken, several variants appeared. In the early 1990s, there was a proposal to increase the security of DES by effectively increasing the key length by using multiple keys with multiple passes etc. But for this scheme to work, it had to first be shown that, the DES function is not a *group*, as defined in mathematics. If DES was a group, then we could show that for two DES keys, X1 and X2, applied to some plaintext (P), we can get a single equivalent key, X3, that would provide the same result; i.e.,

$$E_{X2}(E_{X1}(P)) = E_{X3}(P)$$

where $E_X(P)$ represents DES encryption of some plaintext *P* using DES key *X*. If DES were a group, it wouldn't matter how many keys and passes we applied to some plaintext; we could always find a single 56-bit key that would provide the same result.

As DES was proven not to be a group, we apply additional keys and passes to increase the effective key length. One choice, then, might be to use two keys and two passes, yielding an effective key length of 112 bits. This can be called Double-DES. The two keys, Y1 and Y2, might be applied as follows:

$$C = E_{Y2}(E_{Y1}(P))$$
$$P = D_{Y1}(D_{Y2}(C))$$

where $E_Y(P)$ and $D_Y(C)$ represent DES encryption and decryption, respectively, of some plaintext *P* and ciphertext *C*, respectively, using DES key *Y*.

But an interesting attack can be launched against this "Double-DES" scheme. The applications of the formula above can be with the following individual steps (where C' and P' are intermediate results):

$$C' = E_{Y1}(P) \text{ and } C = E_{Y2}(C')$$
$$P' = D_{Y2}(C) \text{ and } P = D_{Y1}(P')$$

Unfortunately, C'=P', which leaves it vulnerable to a simple *known plaintext* attack (or "Meet-in-the-middle") where the attacker or adversary knows some plaintext (P) and its matching ciphertext (C). To obtain C', the attacker or adversary needs to try all 2^{56} possible values of Y1 applied to P; to obtain P', the attacker needs to try all 2^{56} possible values of Y2 applied to C. Since C'=P', the attacker knows when a match has been achieved — after only $2^{56} + 2^{56} = 2^{57}$ key searches, only twice the work of brute-forcing DES.

Triple-DES (3DES) is based upon the Triple Data Encryption Algorithm (TDEA), as described in FIPS 46-3. 3DES, which is not susceptible to a meet-in-the-middle attack, employs three DES passes and one, two, or three keys called K1, K2, and K3. Generation of the ciphertext (C) from a block of plaintext (P) is accomplished by:

$$C = E_{K3}(D_{K2}(E_{K1}(P)))$$

where $E_K(P)$ and $D_K(P)$ represent DES encryption and decryption, respectively, of some plaintext *P* using DES key *K*. This is also sometimes referred to as an *encrypt-decrypt-encrypt mode* operation.

Decryption of the ciphertext into plaintext is accomplished by:

$$P = D_{K1}(E_{K2}(D_{K3}(C)))$$

The use of three, independent 56-bit keys provides 3DES with an effective key length of 168 bits. The specification also defines use of two keys where, in the operations above, K3 = K1; this provides an effective key length of 112 bits. Finally, a third keying option is to use a single key, so that K3 = K2 = K1 (in this case, the effective key length is 56 bits and 3DES applied to some plaintext, P, will yield the same ciphertext, C, as normal DES would with that same key). With the relatively low cost of key storage and the modest increase in processing due to the use of longer keys, the best recommended practices are that 3DES are employed with three keys.

Another variant of DES, called DESX, is the contribution of Ron Rivest. Developed in 1996, DESX is a very simple algorithm that greatly increases DES's resistance to brute-force attacks without increasing its computational complexity. In DESX, the plaintext input is XORed with 64 additional key bits prior to encryption and the output is likewise XORed with the 64 key bits. By adding just two XOR operations, DESX has an effective keylength of 120 bits against an exhaustive key-search attack. It is pertinent to note that DESX is not immune to other types of more sophisticated

attacks, such as differential or linear crypt analysis, but brute-force is the primary attack vector on DES.

AES

In response to cryptographic attacks on DES, search for a replacement to DES started in January 1997. In September 1997, a formal Call for Algorithms was initiated and in August 1998 announced that 15 candidate algorithms were being considered (Round 1). In April 1999, NIST announced that the 15 had been filtered down to five finalists (Round 2): *MARS* (multiplication, addition, rotation and substitution) from IBM; **Ronald Rivest's** *RC6*; *Rijndael* from a Belgian team; *Serpent*, developed jointly by a team from England, Israel, and Norway; and *Twofish*, developed by Bruce **Schneier**. In October 2000, NIST announced their selection: Rijndael.

In October 2000, NIST published the Report on the Development of the Advanced Encryption Standard (AES) that compared the five Round 2 algorithms in a number of categories. The table below summarises the relative scores of the five schemes (1=low, 3=high):

	Algorithm				
Category	**MARS**	**RC6**	**Rijndael**	**Serpent**	**Twofish**
General security	3	2	2	3	3
Implementation of security	1	1	3	3	2
Software performance	2	2	3	1	1
Smart card performance	1	1	3	3	2
Hardware performance	1	2	3	3	2
Design features	2	1	2	1	3

NIST selected Rijndael as its performance in hardware and software across a wide range of environments in all possible modes. It has excellent key setup time and has low memory requirements, in addition its operations are easy to defend against power and timing attacks.

In February 2001, NIST published the Draft Federal Information Processing Standard (FIPS) AES Specification for public review and comment. AES contains a subset of Rijndael's capabilities (e.g., AES only supports a 128-bit block size) and uses some slightly different nomenclature and terminology, but to understand one is to understand both. The 90-day comment period ended on May 29, 2001 and the U.S. Department of Commerce officially adopted AES in December 2001, published as FIPS PUB 197.

AES (Rijndael) Overview

Rijndael (pronounced as in "rain doll" or "rhine dahl") is a block cipher designed by **Joan Daemen** and **Vincent Rijmen**, both cryptographers in Belgium. Rijndael can operate over a variable-length block using variable-length keys; the version 2 specification submitted to NIST describes use of a 128-, 192-, or 256-bit key to encrypt data blocks that are 128, 192, or 256 bits long. The block length and/or key length can be extended in multiples of 32 bits and it is specifically designed for efficient implementation in hardware or software on a range of processors. The design

of Rijndael was strongly influenced by the block cipher called *Square*, also designed by **Daemen** and **Rijmen.** Rijndael is an iterated block cipher, meaning that the initial input block and cipher key undergoes multiple rounds of transformation before producing the output. Each intermediate cipher result is called *State*.

For simplicity, the block and cipher key are often given as an array of columns where each array has 4 rows and each column represents a single byte (8 bits). The number of columns in an array representing the state or cipher key, then, can be calculated as the block or key length divided by 32 (32 bits = 4 bytes). An array representing a State will have Nb columns, where Nb values of 4, 6, and 8 correspond to a 128-, 192-, and 256-bit block, respectively. Similarly, an array representing a Cipher Key will have Nk columns, where Nk values of 4, 6, and 8 correspond to a 128-, 192-, and 256-bit key, respectively. An example of a 128-bit State (Nb = 4) and 192-bit Cipher Key (Nk = 6) is shown below:

$$\begin{matrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{matrix} \qquad \begin{matrix} k_{0,0} & k_{0,1} & k_{0,2} & k_{0,3} & k_{0,4} & k_{0,5} \\ k_{1,0} & k_{1,1} & k_{1,2} & k_{1,3} & k_{1,4} & k_{1,5} \\ k_{2,0} & k_{2,1} & k_{2,2} & k_{2,3} & k_{2,4} & k_{2,5} \\ k_{3,0} & k_{3,1} & k_{3,2} & k_{3,3} & k_{3,4} & k_{3,5} \end{matrix}$$

The number of transformation rounds (Nr) in Rijndael is a function of the block length and key length, and is given in the table below:

Rounds Nr		**Block Size**		
		128 bits Nb = 4	**192 bits Nb = 6**	**256 bits Nb = 8**
Key Size	**128 bits Nk = 4**	10	12	14
	192 bits Nk = 6	12	12	14
	256 bits Nk = 8	14	14	14

The AES version of Rijndael does not support all nine combinations of block and key lengths, but only the subset using a 128-bit block size. NIST calls these supported variants AES-128, AES-192, and AES-256 where the number refers to the key size. The Nb, Nk, and Nr values supported in AES are:

	Parameters		
Variant	**Nb**	**Nk**	**Nr**
AES-128	4	4	10
AES-192	4	6	12
AES-256	4	8	14

The AES/Rijndael cipher itself has three operational stages:

- AddRound Key transformation
- **Nr**-1 Rounds comprising: SubBytes transformation; ShiftRows transformation; MixColumns transformation; AddRoundKey transformation
- A final Round comprising: SubBytes transformation; ShiftRows transformation; AddRoundKey transformation.

The nomenclature used below is taken from the AES specification, although, references to the Rijndael specification are made for completeness. The arrays *s* and *s'* refer to the State before and after a transformation, respectively (NOTE: The Rijndael specification uses the array nomenclature *a* and *b* to refer to the before and after States, respectively). The subscripts *i* and *j* are used to indicate byte locations within the State (or Cipher Key) array.

The SubBytes Transformation

The substitute bytes (called *ByteSub* in Rijndael) transformation operates on each of the State bytes independently and changes the byte value. An S-box, or *substitution table*, controls the transformation. The characteristics of the S-box transformation as well as a compliant S-box table are provided in the AES specification; as an example, an input State byte value of 107 (0x6b) will be replaced with a 127 (0x7f) in the output State and an input value of 8 (0x08) would be replaced with a 48 (0x30).

One simple way to think of the SubBytes transformation is that a given byte in State s is given a new value in State s' according to the S-box. The S-box, then, is a function on a byte in State s so that:

$$s'_{i,j} = \text{S-box}\ (s_{i,j})$$

The more general depiction of this transformation is shown by:

$$\begin{matrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{matrix} \quad \Longrightarrow \text{S-box} \Longrightarrow \quad \begin{matrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{matrix}$$

The ShiftRows Transformation

The shift rows transformation cyclically shifts the bytes in the bottom three rows of the State array. According to the more general Rijndael specification, rows 2, 3, and 4 are cyclically left-shifted by C1, C2, and C3 bytes, respectively, per the table below:

Nb	C1	C2	C3
4	1	2	3
6	1	2	3
8	1	3	4

The current version of AES, of course, only allows a block size of 128 bits (Nb = 4) so that C1=1, C2=2, and C3=3.

The diagram below shows the effect of the Shift Rows transformation on State s:

State s	**---------- no shift ---------->**	**State s'**
$s_{0,0}\ s_{0,1}\ s_{0,2}\ s_{0,3}$	----> left-shift by C1 (1)---- >	$s_{0,0}\ s_{0,1}\ s_{0,2}\ s_{0,3}$
$s_{1,0}\ s_{1,1}\ s_{1,2}\ s_{1,3}$	----> left-shift by C2 (2)---- >	$s_{1,1}\ s_{1,2}\ s_{1,3}\ s_{1,0}$
$s_{2,0}\ s_{2,1}\ s_{2,2}\ s_{2,3}$	----> left-shift by C3 (3)---- >	$s_{2,2}\ s_{2,3}\ s_{2,0}\ s_{2,1}$
$s_{3,0}\ s_{3,1}\ s_{3,2}\ s_{3,3}$		$s_{3,3}\ s_{3,0}\ s_{3,1}\ s_{3,2}$

The MixColumns Transformation

The mix columns (called *MixColumn* in Rijndael) transformation uses a mathematical function to transform the values of a given column within a State, acting on the four values at one time as if they represented a four-term polynomial.

$$s'_{i,c} = \text{MixColumns}\ (s_{i,c})$$
$$\text{for } 0<=i<=3 \text{ for some column, c.}$$

The column position does not change, however, the values within the column change.

Round Key Generation and the AddRoundKey Transformation

The Cipher Key is used to derive a different key to be applied to the block during each round of the encryption operation. These keys are known as the Round Keys and each will be the same length as the block, i.e., Nb 32-bit words.

The AES defines a key schedule by which the original Cipher Key (of length Nk 32-bit words) is used to form an *Expanded Key*. The *Expanded Key* size is equal to the block size multiplied by the number of encryption rounds plus 1, which will provide Nr+1 different keys. (Note that there are Nr encipherment rounds but Nr+1

AddRoundKey transformations.). For example, AES uses a 128-bit block and either 10, 12, or 14 iterative rounds depending upon key length. With a 128-bit key, we would need 1408 bits of key material (128×11=1408), or an *Expanded Key* size of 44 32-bit words (44×32=1408). Similarly, a 192-bit key would require 1664 bits of key material (128×13), or 52 32-bit words, while a 256-bit key would require 1920 bits of key material (128×15), or 60 32-bit words. The key expansion mechanism, then, starts with the 128-, 192-, or 256-bit Cipher Key and produces a 1408-, 1664-, or 1920-bit *Expanded Key*, respectively. The original Cipher Key occupies the first portion of the *Expanded Key* and is used to produce the remaining new key material.

The result is an *Expanded Key* that can be 11, 13, or 15 separate keys, each used for one AddRoundKey operation. These, then, are the *Round Keys*. The diagram below shows an example using a 192-bit Cipher Key (Nk=6), shown in *magenta italics*:

Expanded Key:	W_0	W_1	W_2	W_3	W_4	W_5	W_6	W_7	W_8	W_9	W_{10}	W_{11}	W_{12}	W_{13}	W_{14}	W_{15}	...	W_{44}	W_{45}	W_{46}	W_{47}	W_{48}	W_{49}	W_{50}	W_{51}
Round keys:	Round key 0				Round key 1				Round key 2				Round key 3				...	Round key 11				Round key 12			

The AddRoundKey (called *Round Key addition* in Rijndael) transformation merely applies each Round Key, in turn, to the State by a simple bit-wise exclusive OR operation.

MARS by Zunic et al., IBM.

This new design uses a special type of a Feistel network, which depends heavily on the instruction sets available on modern 32-bit processors. This has the benefit that on these target machines it is efficient, but it may lead to implementation problems/difficulties in cheaper architectures like smart cards.

RC6 by Rivest, Robshaw, and Yin, RSA Laboratories.

RC6 follows the ideas of RC5 with improvements. It attempts to avoid some of the differential attacks against RC5's data dependent rotations. However, there are some attacks that are not yet employed, and it is unclear whether RC6 is well enough analysed yet.

Serpent by Anderson, Biham, and Knudsen.

Serpent has a basically conservative but, in many ways innovative design. It may be implemented by bitslice (or vector) gate logic throughout. This makes it rather complicated to implement from scratch, and writing it in a non-bitslice way involves an efficiency penalty. The *32* rounds lead to probably the highest security margin on all AES candidates, while it is still fast enough for all practical purposes.

Twofish by Schneier et al., Counterpane Security.

Twofish is a block cipher designed by Counterpane, whose founder and CTO is **Bruce Schneier**. The design is highly delicate, supported with many alternative ways of implementation. It is crypt analysed in much detail, by the very authoritative "extended Twofish team". It is basically a Feistel cipher, but utilises many different ideas. This cipher has key dependent S-boxes like **Blowfish** (another cipher by Bruce Schneier).

Other Symmetric Ciphers

Blowfish

Bruce Schneier designed blowfish and it is a block cipher with a 64-bit block size and variable length keys (which may vary up to 448 bits). It has gained of acceptance in a number of applications, including Nautilus and PGPfone.

Blowfish utilises the idea of randomised S-boxes: while doing key scheduling, it generates large pseudo-random lookup tables through several encryptions. These tables depend on the user supplied key in a very complex manner. This makes blowfish highly resistant against many attacks such as differential and linear crypt analysis. But is not the algorithm of choice for environments where large memory space (something like *4096* bytes) is not available. The known attacks against Blowfish are based on its weak key classes.

CAST-128

CAST-128, Substitution-Permutation Network (SPN) cryptosystem, is similar to DES, which have good resistance to differential, linear and related-key crypt analysis. CAST-128 has Feistel structure and utilises eight fixed S-boxes. CAST-128 supports variable key lenghts between 40 and 128 bits (described in RFC2144).

IDEA

IDEA (International Data Encryption Algorithm), was developed at ETH Zurich in Switzerland by **Xuejia Lai** uses a 128-bit key, and is considered to be very secure. The best attacks against IDEA uses the impossible differential idea of **Biham, Shamir** and **Biryukov**. They can attack only *4.5* rounds of IDEA and this poses no threat to the total of *8.5* rounds used in IDEA. It is pertinent to note that IDEA is patented in the United States and in most European countries.

Rabbit

Rabbit is a stream cipher, based on iterating a set of coupled nonlinear function. It is characterised by a high performance in software.

RC4

RC4 is a stream cipher by Ron Rivest at RSA Data Security, Inc. It accepts keys of arbitrary length. It used to be a trade secret, until someone posted source code for an

algorithm on the usenet, claiming it to be equivalent to RC4. The algorithm is very fast. Its security is unknown, but breaking it does not seem trivial either. Because of its speed, it may have uses in certain particular applications.

RC4 is essentially a pseudo random number generator, and the output of the generator is exclusive-ored with the data stream. For this reason, it is very important that the same RC4 key should not be used to encrypt two different data streams.

☞ Check Your Progress 2

1) What is the result of the Exclusive OR operation 1XOR0?

   (a) 1
   (b) 0
   (c) Indeterminate
   (d) 10

2) A block cipher:

   (a) Is an asymmetric key algorithm
   (b) Converts variable-length plaintext into fixed-length ciphertext
   (c) Breaks a message into fixed length units for encryption
   (d) Encrypts by operating on a continuous data stream.

3) What is the block length of the Rijndael Cipher?

   (a) 64 bits
   (b) 128 bits
   (c) Variable
   (d) 256 bits.

4) What is the key length of the Rijndael Block Cipher

   (a) 56 or 64 bits
   (b) 512 bits
   (c) 128, 192, or 256 bits
   (d) 512 or 1024 bits.

5) The NIST Advanced Encryption Standard uses the:

   (a) 3 DES algorithm
   (b) DES algorithm
   (c) Rijndael algorithm
   (d) IDEA algorithm.

6) The modes of DES do not include:

   (a) Electronic code book
   (b) Variable block feedback
   (c) Cipher block chaining
   (d) Output feedback.

1.4 PUBLIC KEY CRYPTOGRAPHY

Modern cryptography deals with communication between many different parties, without using a secret key for every pair of parties. In 1976, **Whitfield Diffie** and **Martin Hellman** published an invited paper in the ÏEEE Transactions on Information Theory titled “New Directions in Cryptography”. This paper can be considered as the

beginning of modern cryptography. Their paper described a two – key crypto system in which two parties can perform a secure communication over a non-secure channel without having to share a secret key. PKC depends upon the existence of so-called one-way function, or mathematical functions that are easy to calculate but the calculation of the inverse function is relatively difficult.

Generic PKC uses two keys that are mathematically related and knowledge of one key does not allow someone to easily determine the other key. One key is used to encrypt and the other key is used to decrypt. It does not matter which key is applied first and because a pair of keys are used, this is called asymmetric key cryptography. In PKC, one of the keys is designated as the public key and the other key is designated as the private key. The public key may be advertised but the private key is never revealed to another party. It is straightforward enough to send messages under this scheme. Suppose Alice wants to send Bob a message, Alice encrypts some information using Bob's public key; Bob decrypts the ciphertext using his private key.

The most common PKC implementation is RSA, named after the three MIT mathematicians who developed it — **Ronald Rivest**, **Adi Shamir, and Leonard Adleman.** RSA can be used for key exchange, digital signatures, or encryption of small blocks of data. The *Figures 11 and 12* highlights how digital signature are created and verified. The detailed discussion will be made in the next unit. RSA uses a variable size encryption block and variable size key. The key pair is derived from a very large number, *n*, that is the product of two large prime numbers selected through special rules; these primes may be 100 or more digits in length each, yielding an *n* with roughly twice as many digits as the prime factors. The public key includes *n* and a derivate of one of the factor of *n*; an adversary cannot determine the prime factor of *n* (and, therefore, the private key) from this information alone and that is what makes the RSA algorithm so secure.

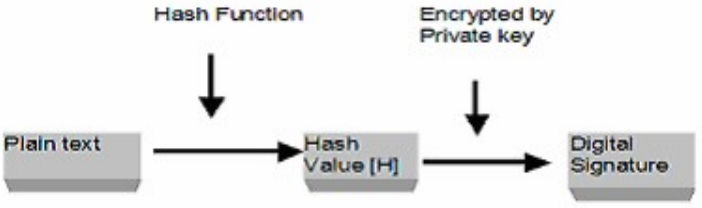


Figure11: Digital Signature

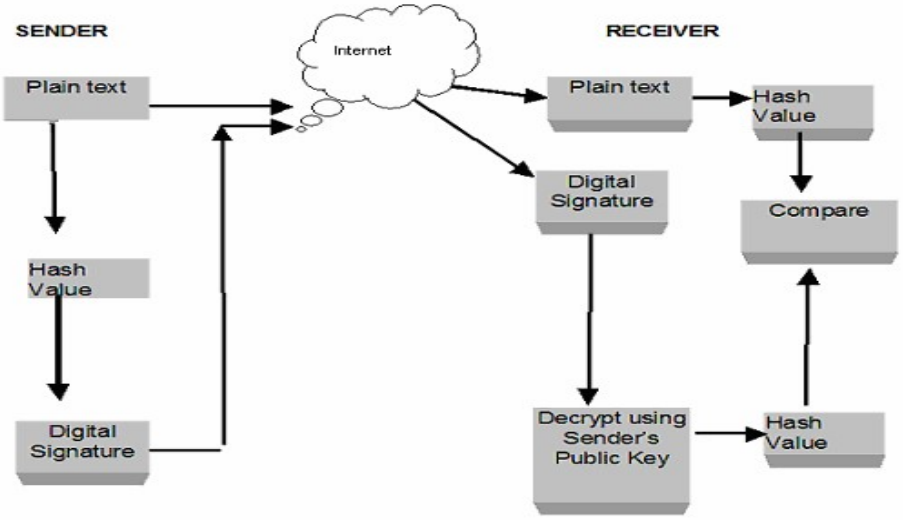


Figure12: Digital signature verification process

Goldwasser, Micali, and Rivest gave the rigorous definition of security for signature scheme and provided the first construction that probably satisfied that definition, under a suitable assumption. The assumption that claw-free pair of permutations exist, is implied by the assumption that integer factorisation is hard. The earlier signature scheme due to **Merkle** was also shown to satisfy this definition. **Rompel, Naor** and **Yung** showed how to construct a digital signature scheme using any one-way function. The inefficiency of the above mentioned schemes, and due to the fact that these schemes require the signer's secret key to change between invocations of the signing algorithm make these solutions impractical.

Several other signature schemes have been shown to be secure in the so-called random-oracle model. The random-oracle model is a model of computation in which it is assumed that a given hash function behaves as an ideal random function. An ideal random function is a function where the input domain is the set of binary strings, such that each binary string is mapped to a random binary string of the same length. Although this assumption is evidently false, as it formalises the notion that the hash function is like a black box and its output cannot be determined until it is evaluated. A proof of security in the random oracle model gives some evidence of resilience to attack. The RSA signatures with special message formatting, the Fiat-Shamir, and the Schnorr signature schemes have been analysed and proven secure in the random oracle model. However, it is known that security in the random oracle model does not imply security in the plain model.

Gennaro, Halevi, Rabin, Cramer Shoup proposed the first signature schemes whose efficiency is suitable for practical use and whose security analysis does not assume an ideal random function. These schemes are considered to be secure, under RSA assumption.

PKC depends upon the existence of so-called *one-way functions*, or mathematical functions that are easy to compute whereas their inverse function is relatively difficult to compute. Let me give you two simple examples:

1) Multiplication vs. factorisation
2) Exponentiation vs. logarithms

The important Public-key cryptography algorithms are:

- ***RSA***
- ***Diffie-Hellman:*** After the RSA algorithm was published, **Diffie** and **Hellman** came up with their own algorithm. D-H is used for secret-key key exchange only, and not for authentication or digital signatures.
- ***Digital Signature Algorithm (DSA):*** The algorithm specified in NIST's Digital Signature Standard (DSS), provides digital signature capability for the authentication of messages.
- ***ElGamal:*** Designed by **Taher Elgamal**, is a PKC system similar to Diffie-Hellman and used for key exchange.
- ***Elliptic Curve Cryptography (ECC):*** A PKC algorithm based upon elliptic curves. ECC can offer levels of security with small keys comparable to RSA and other PKC methods. It was designed for devices with limited compute power and/or memory, such as smartcards and PDAs.
- ***Public-Key Cryptography Standards (PKCS):*** A set of interoperable standards and guidelines for public-key cryptography, designed by RSA Data Security Inc.
    - PKCS #1: RSA Cryptography Standard (Also RFC 3447)

- PKCS #2: *Incorporated into PKCS #1.*
- PKCS #3: Diffie-Hellman Key-Agreement Standard
- PKCS #4: *Incorporated into PKCS #1.*
- PKCS #5: Password-Based Cryptography Standard (PKCS #5 V2.0 is also RFC 2898)
- PKCS #6: Extended-Certificate Syntax Standard (being phased out in favor of X.509v3)
- PKCS #7: Cryptographic Message Syntax Standard (Also RFC 2315)
- PKCS #8: Private-Key Information Syntax Standard
- PKCS #9: Selected Attribute Types (Also RFC 2985)
- PKCS #10: Certification Request Syntax Standard (Also RFC 2986)
- PKCS #11: Cryptographic Token Interface Standard
- PKCS #12: Personal Information Exchange Syntax Standard
- PKCS #13: Elliptic Curve Cryptography Standard
- PKCS #14: Pseudorandom Number Generation Standard is no longer available
- PKCS #15: Cryptographic Token Information Format Standard

- ***Cramer-Shoup:*** A public-key cryptosystem proposed by **R. Cramer** and **V. Shoup** of IBM in 1998.
- ***Key Exchange Algorithm (KEA):*** A variation on Diffie-Hellman; proposed as the key exchange method for Capstone.
- ***LUC:*** A public-key cryptosystem designed by **P.J. Smith** and based on Lucas sequences. Can be used for encryption and signatures, using integer factoring.

Public Key Infrastructure (PKI) in India

The Act provides the Controller of Certifying Authorities to license and regulate the working of CAs and also to ensure that CAs does not violate any of the provisions of the Act. CCA has created the framework for Public Key Infrastructure in India. It has prescribed technical standards for cryptography and physical security based on international standards/best practices of ITU, IETF, IEEE etc. CAs has to prove compliance with these standards through a stringent audit procedure that has been put in place. CAs also needs to get their CPS (Certification Practice Statement) approved by the CCA. India has seven Certifying Authorities: (1) SafeScrypt-Certifying Authority; (2) National Informatics Centre-Certifying Authority; (3) Tata Consultancy- Services Certifying Authorities; (4) Institute for Research and Development in Banking Technology-Certifying Authority; (5) Customs and Central Excise-Certifying Authority; (6) Mahanagar Telephone Nigam Limited-Certifying Authority; and (7) n(Code) Solutions CA (GNFC).

The CCA also maintains the National Repository of Digital Certificates (NRDC), as required under the IT Act 2000 that contains all the certificates issued by all the CAs in India. The CCA operates its signing activity through the Root Certifying Authority of India (RCAI), which is operational under stringent controls.

Public Key Cryptosystem

Asymmetric key algorithms or Public key algorithm use a different key for encryption and decryption (*Figure 13*), and the decryption key cannot (practically) be derived from the encryption key. Public key methods are important because they can be used to distribute encryption keys or other data securely even when the parties have no opportunity to agree on a secret key in private. All known public key methods are quite slow, and they are usually only used to encrypt session keys, that are then used to encrypt the bulk of the data using a symmetric encryption.

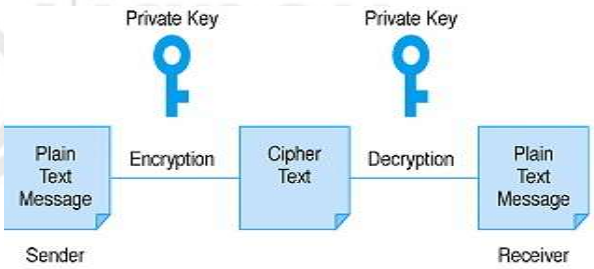


Figure13: Encryption and decryption process

1.4.1 RSA Public Key Algorithm (Rivest-Shamir-Adelman Algorithm)

RSA Public Key Algorithm is the most commonly used public key algorithm and this algorithm can be used both for encryption and for signing. It is generally considered to be secure when sufficiently long keys are used (512 bits is insecure, 768 bits is moderately secure, and 1024 bits is good). The security of RSA relies on the difficulty of factoring large integers. The recent advances in factoring large integers would make RSA vulnerable. It is patented in US and the Patent expired in year 2000.

RSA Algorithm

Key Generation Algorithm

1) Generate two large random primes, p and q, of approximately equal size such that their product $n = pq$ is of the required bit length, e.g., 1024 bits.

2) Compute n = pq and (φ) phi = (p –1)(q –1).

3) Choose an integer e, 1 < e < phi, such that gcd(e, phi) = 1.

4) Compute the secret exponent d, 1 < d < phi, such that ed ≡ 1 (mod phi).

5) The public key is (n, e) and the private key is (n, d). The values of p, q, and phi should also be kept secret.

Where

- n is known as the *modulus*.
- e is known as the *public exponent* or *encryption exponent*.
- d is known as the *secret exponent* or *decryption exponent*.

Encryption

Sender A does the following:

1) Obtains the recipient B's public key (n, e).

2) Represents the plaintext message as a positive integer m.

3) Computes the ciphertext $c = m^{\wedge e} \bmod n$.

4) Sends the ciphertext c to B.

Decryption

Recipient B does the following:

1) Uses his private key (n, d) to compute $m = c^{\wedge d} \bmod n$.

2) Extracts the plaintext from the integer representative m.

Summary of RSA

- n = pq where p and q are distinct primes.
- phi, φ = (p-1)(q-1)
- e < n such that gcd(e, phi)=1
- $d = e^{\wedge -1} \bmod phi$.
- $c = m^{\wedge e} \bmod n$, 1<m<n.
- $m = c^{\wedge d} \bmod n$.

Example RSA Algorithm

Let us select two prime numbers, p = 7 and q= 17. Keys are generated as follows:

1. Two prime numbers, p = 7 and q = 17
2. n = pq = 7 x17 = 119
3. P(n) = (p-1) (q-1) = (7-1) (17-1) = 6 x 16 = 96
4. Find e which is relatively prime to ϕ(n) = 96 and less than ϕ(n). e = 5 for this case
5. Find d such that d.e = 1 mod 96 and d<96 d = 77
6. So, public key in = [5,119] and private key is = [77, 119]

Encryption and Decryption using RSA algorithm

Encryption: Let plain text input of M = 19
Cipher text $C = M^e \pmod n$

$$= 19^5 \pmod{119} = \frac{2476099}{119}$$

= 20807 with remainder of 66
Cipher Tex t= 66

Decryption:

$M = c^d \pmod n$
$= 66^{77} \pmod{119}$
= 19

1.4.2 Diffie-Hellman

Diffie-Hellman is a commonly used public-key algorithm for key exchange. It is generally considered to be secure when sufficiently long keys and proper generators are used. The security of Diffie-Hellman relies on the difficulty of the discrete logarithm problem (which is believed to be computationally equivalent to factoring large integers). Diffie-Hellman is claimed to be patented in the United States, but the patent expired on April 29, 1997. There are also strong rumours that the patent might in fact be invalid (there is evidence of it having been published over an year before the patent application was led). Diffie-Hellman is sensitive to the selection of the strong prime, size of the secret exponent, and the generator.

The "Diffie-Hellman Method For Key Agreement" allow two hosts to create and share a secret key.

1) First the hosts must get the "Diffie-Hellman parameters". A prime number, 'p' (larger than 2) and "base", 'g', an integer that is smaller than 'p'. They can either be hard coded or fetched from a server.

2) The hosts each secretly generate a private number called 'x', which is less than "p – 1".

3) The hosts next generate the public keys, 'y'. They are created with the function:

   y = g^x % p

4) The two host now exchange the public keys ('y') and the exchanged numbers are converted into a secret key, 'z'.

   z = y^x % p

   'z' can now be used as the key for whatever encryption method is used to transfer information between the two hosts. Mathematically, the two hosts should have generated the same value for 'z'.

   z = (g^x % p)^x' % p = (g^x' % p)^x % p

All of these numbers are positive integers

x^y	means: x is raised to the y power
x%y	means: x is divided by y and the remainder is returned

Example:

Prime Number p = 353 and primitive root of 353, in this case is g = 3. Let A is sender B is receives A & B select secret key as XA = 97 and XB = 233. Now A & B computes their public keys. YA = 3^{233}. Now A & B computes their public keys.
YA = 3^{97} mod 353 = 40
YB = 3^{233} mod 353 = 248

After exchanging their public keys, A & B can compute the common secret key:
A computes:

Z=(YB) mod 253
B computes = 248^{97} mod 353 = 160
Z = $(YA)^{xb}$mod353 = 40^{233} mod 353 = 160

1.4.3 Elliptic Curve Public Key Cryptosystems

Elliptic Curve Public Key Cryptosystems is an emerging field. They have been slow to execute, but have become feasible with modern computers. They are considered to be fairly secure, but haven't yet undergone the same scrutiny as done on RSA algorithm.

The Public-Key cryptography systems use hard-to-solve problems as the basis of the algorithm. The most predominant algorithm today for public-key cryptography is RSA, based on the prime factors of very large integers. While RSA can be successfully attacked, the mathematics of the algorithm has not been comprised, per se; instead, computational brute-force attacks have broken the keys. The protection

mechanism is "simple" — keep the size of the integer to be factored ahead of the computational curve!

In 1985, cryptographers **Victor Miller** (IBM) and **Neal Koblitz** (University of Washington) proposed the Elliptic Curve Cryptography (ECC) independently (as shown in *Figure 14*). ECC is based on the difficulty of solving the Elliptic Curve Discrete Logarithm Problem (ECDLP). Like the prime factorisation problem, ECDLP is another "hard" problem that is deceptively simple to state: Given two points, P and Q, on an elliptic curve, find the integer *n*, if it exists, such that P = nQ.

Elliptic curves combine number theory and algebraic geometry. These curves can be defined over any field of numbers (i.e., real, integer, complex), although, we generally see them used over finite fields for applications in cryptography. An elliptic curve consists of the set of real numbers (x, y) that satisfies the equation:

$y^2 = x^3 + ax + b$

The set of all the solutions to the equation forms the elliptic curve. Changing *a* and *b* changes the shape of the curve, and small changes in these parameters can result in major changes in the set of (x,y) solutions.

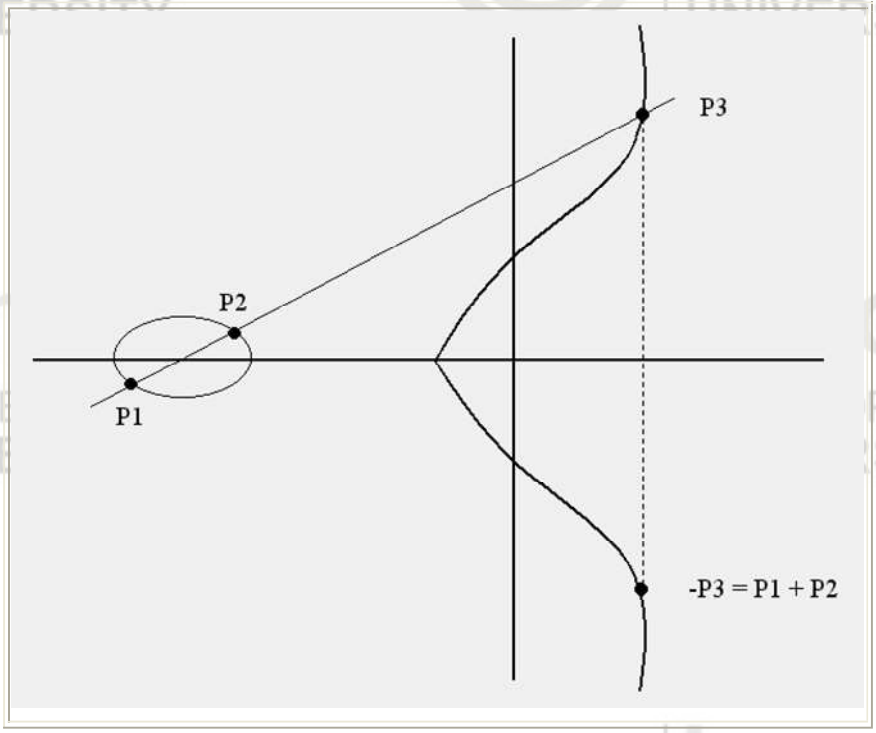


Figure14: Elliptic Curve Cryptography

The *figure* above shows the addition of two points on an elliptic curve. Elliptic curves have the interesting property that adding two points on the elliptic curve yields a third point on the curve. Therefore, adding two points, P1 and P2, gets us to point P3, also on the curve. Small changes in P1 or P2 can cause a large change in the position of P3.

Let us analyse the original problem as stated above. The point Q is calculated as a multiple of the starting point, P, *or*, Q = nP. An attacker might know P and Q but finding the integer, *n*, is a difficult problem to solve. Q is the public key, then, and *n* is the private key.

RSA has been the mainstay of PKC for over two decades. But ECC is exciting because of their potential to provide similar levels of security compared to RSA but

with significantly reduced key sizes. Certicom Corp. (http://www.certicom.com/), one of the major proponents of ECC, suggests the key size relationship between ECC and RSA as per the following table:

ECC and RSA Key Comparison

RSA Key Size	Time to Break Key (MIPS Years)	ECC Key Size	RSA:ECC Key-Size Ratio
512	10^4	106	5:1
768	10^8	132	6:1
1,024	10^{11}	160	7:1
2,048	10^{20}	210	10:1
21,000	10^{78}	600	35:1

Since the ECC key sizes are so much shorter than comparable RSA keys, the length of the public key and private key is much shorter in elliptic curve cryptosystems. Therefore, this results in faster processing, and lower demands on memory and bandwidth. In practice, the final results are not yet in; RSA, Inc. notes that ECC is faster than RSA for signing and decryption, but slower than RSA for signature verification and encryption.

Nevertheless, ECC is particularly useful in applications where memory, bandwidth, and/or computational power is limited (e.g., a smartcard) and it is in this area that ECC use is expected to grow.

1.4.4 DSA

DSS (Digital Signature Standard): A signature-only mechanism endorsed by the United States Government. Its design has not been made public, and many people have found potential problems with it (e.g., leaking hidden data, and revealing your secret key if you ever happen to sign two different messages using the same random number).

The DSA is used by a signatory to generate a digital signature on data and by a verifier to verify the authenticity of the signature. Each signatory has a public and private key. The private key is used in the signature generation process and the public key is used in the signature verification process. For both signature generation and verification, the data which is referred to as a message, M, is reduced by means of the Secure Hash Algorithm (SHA) specified in FIPS YY. An adversary, who does not know the private key of the signatory, cannot generate the correct signature of the signatory. In other words, signatures cannot be forged. However, by using the signatory's public key, anyone can verify a correctly signed message.

A means of associating public and private key pairs for the corresponding users is required. That is, there must be a binding of a user's identity and the user's public key. A mutually trusted party may certify this binding. For example, a certifying authority could sign credentials containing a user's public key and identity to form a certificate. Systems for certifying credentials and distributing certificates are beyond the scope of this standard. NIST intends to publish separate document(s) on certifying credentials and distributing certificates.

The DSA makes use of the following parameters:

1) p = a prime modulus, where $2^{L-1} < p < 2^{L}$ for 512 = < L = <1024 and L a multiple of 64

2) q = a prime divisor of p - 1, where $2^{159} < q < 2^{160}$

3) $g = h^{(p-1)/q}$ mod p, where h is any integer with 1 < h < p - 1 such that $h^{(p-1)/q}$ mod p > 1(g has order q mod p)

4) x = a randomly or pseudorandomly generated integer with 0 < x < q

5) $y = g^{x}$ mod p

6) k = a randomly or pseudorandomly generated integer with 0 < k < q

The integers p, q, and g can be public and can be common to a group of users. A user's private and public keys are x and y, respectively. They are normally fixed for a period of time. Parameters x and k are used for signature generation only, and must be kept secret. Parameter k must be regenerated for each signature.

The signature of a message M is the pair of numbers r and s computed according to the equations below:

$$r = (g^{k} \bmod p) \bmod q \text{ and}$$

$$s = (k^{-1}(SHA(M) + xr)) \bmod q.$$

In the above, k^{-1} is the multiplicative inverse of k, mod q; i.e., $(k^{-1} k) \bmod q = 1$ and $0 < k^{-1} < q$. The value of SHA (M) is a 160-bit string output by the Secure Hash Algorithm specified in FIPS 180. For use in computing s, this string must be converted to an integer.

As an option, one may wish to check if r = 0 or s = 0. If either r = 0 or s = 0, a new value of k should be generated and the signature should be recalculated (it is extremely unlikely that r = 0 or s = 0 if signatures are generated properly).

The signature is transmitted along with the message to the verifier.

Prior to verifying the signature in a signed message, p, q and g plus the sender's public key and identity are made available to the verifier after proper authentication.

Let M', r' and s' be the received versions of M, r, and s, respectively, and let y be the public key of the signatory. To verify first check to see that 0 < r' < q and 0 < s' < q; if either condition is violated, the signature shall be rejected. If these two conditions are satisfied, the verifier computes:

$$w = (s')^{-1} \bmod q$$

$$u1 = ((SHA(M')w) \bmod q$$

$$u2 = ((r')w) \bmod q$$

$$v = (((g)^{ul} (y)^{u2}) \bmod p) \bmod q.$$

If v = r', then the signature is verified and the verifier can have high confidence that the received message was sent by the party holding the secret key x corresponding to y. For a proof that v = r' when M' = M, r' = r, and s' = s, see Appendix 1.

If v does not equal r', then the message may have been modified, the message may have been incorrectly signed by the signatory, or the message may have been signed by an impostor. The message should be considered invalid.

ElGamal public key cryptosystem. Based on the discrete logarithm problem.

ElGamal consists of three components: the key generator, the encryption algorithm, and the decryption algorithm.

The key generator works as follows:

- Alice generates an efficient description of a cyclic group G of order q with generator g. See below for specific examples of how this can be done.
- Alice chooses a random x from $\{0,...,q - 1\}$.
- Alice computes $h = g^x$.
- Alice publishes h, along with the description of G,q,g, as her public key. Alice retains x as her secret key.

The encryption algorithm works as follows: to encrypt a message m to Alice under her public key (G,q,g,h),

- Bob converts m into an element of G.
- Bob chooses a random k from $\{0,...,q - 1\}$, then calculates $c_1 = g^k$ and $c_2 = m \cdot h^k$.
- Bob sends the ciphertext (c_1,c_2) to Alice.

The decryption algorithm works as follows: to decrypt a ciphertext (c_1,c_2) with her secret key x,

- Alice computes $c_2(c_1^x)^{-1}$ as the plaintext message.

Note that the decryption algorithm does indeed produce the intended message since:

$$c_2(c_1^x)^{-1} \equiv \frac{m \cdot h^k}{g^{xk}} \equiv \frac{m \cdot g^{xk}}{g^{xk}} \equiv m \pmod{q}$$

If the space of possible messages is larger than the size of G, then the message can be split into several pieces and each piece can be encrypted independently. Typically, however, a short key to a symmetric-key cipher is first encrypted under ElGamal, and the (much longer) intended message is encrypted more efficiently using the symmetric-key cipher — this is termed *hybrid encryption*.

1.5 MATHEMATICAL BACKGROUND

In this section, we will find out mathematical background from mathematical Algorithms.

1.5.1 Exclusive OR (XOR)

Exclusive OR (XOR) is one of the fundamental mathematical operations used in cryptography (and many other applications). **George Boole**, a mathematician in the late 1800s, invented a new form of "algebra" that provides the basis for building electronic computers and microprocessor chips. **Boole** defined a bunch of primitive

logical operations where there are one or two inputs and a single output depending upon the operation; the input and output are either TRUE or FALSE. The most elemental **Boolean** operations are:

- NOT: The output value is the inverse of the input value (i.e., the output is TRUE if the input is false, FALSE if the input is true).
- AND: The output is TRUE if all inputs are true, otherwise FALSE. (e.g., "the sky is blue AND the world is flat" is FALSE while "the sky is blue AND security is a process" is TRUE.)
- OR: The output is TRUE if either or both inputs are true, otherwise FALSE. (e.g., "the sky is blue OR the world is flat" is TRUE and "the sky is blue OR security is a process" is TRUE).
- XOR (Exclusive OR): The output is TRUE if exactly one of the inputs is TRUE, otherwise FALSE. (e.g., "the sky is blue XOR the world is flat" is TRUE while "the sky is blue XOR security is a process" is FALSE.)

In computers, Boolean logic is implemented in *logic gates*; for design purposes, XOR has two inputs and a single output, and its logic diagram looks like this:

XOR	0	1
0	0	1
1	1	0

So, in an XOR operation, the output will be a 1 if one input is a 1; otherwise, the output is 0. The real significance of this is to look at the "identity properties" of XOR. In particular, any value XORed with itself is 0 and any value XORed with 0 is just itself. Why does this matter? Well, if I take my plaintext and XOR it with a key, I get a jumble of bits. If I then take that jumble and XOR it with the same key, I return to the original plaintext.

1.5.2 The Modulo Function

The *modulo* function is, simply, the remainder function. It is commonly used in programming and is critical to the operation of any mathematical function using digital computers.

To calculate *X modulo Y* (usually written *X mod Y*), you merely determine the remainder after removing all multiples of Y from X. Clearly, the value X mod Y will be in the range from 0 to Y-1.

Some examples should clear up any remaining confusion:

- 15 mod 7 = 1
- 25 mod 5 = 0
- 33 mod 12 = 9
- 203 mod 256 = 203

Modulo arithmetic is useful in crypto because it allows us to set the size of an operation and be sure that we will never get numbers that are too large. This is an important consideration when using digital computers.

☞ **Check Your Progress 3**

1) Which of the following is an example of a symmetric key algorithm?

(a)
(b) RSA
(c) Diffie-Hellman
(d) Knapsack

2) Which of the following is a problem with symmetric key algorithms?

(a) It is slower than asymmetric key encryption
(b) Most algorithms are kept proprietry
(c) Work factor is not a function of the key size.
(d) It provides secure distribution of the secret key.

3) Which of the following is an example of an asymmetric key algorithms?

(a) ELLIPTIC CURVE
(b) IDEA
(c) 3 DES
(d) DES

4) In public key cryptography:

(a) Only the public key can encrypt, and only the private can decrypt
(b) Only the private key can encrypt, and only the public can decrypt
(c) The public key is used to encrypt and decrypt
(d) If the public key encrypts, and only the private can decrypt.

5) In a block cipher, diffusion:

(a) Conceals the connection between the ciphertext and plaintext
(b) Spread the influence of a plaintext character over many ciphertext characters.
(c) Cannot be accomplished
(d) Is usually implemented by non-linear S-boxes.

6) State whether following statements are True or False. T ☐ F ☐

(a) Work factor of double DES is the same as for single DES
(b) Elliptic curse cryptosystem have a lower strength per bit than RSA.
(c) In digitally-signed message transmission using a hash function the message digest is encrypted in the public key of the sender.

1.6 SUMMARY

Information security can be defined as technological and managerial procedures applied to computer systems to ensure the availability, integrity, and confidentiality of the information managed by computer. Cryptography is a particularly interesting field because of the amount of work that is, by necessity, done in secret. The irony is that today, secrecy is not the key to the goodness of a cryptographic algorithm. Regardless of the mathematical theory behind an algorithm, the best algorithms are those that are well known and well documented because they are also well tested and well studied! In fact, time is the only true test of good cryptography; any cryptographic scheme that stays in use year after year is most likely to be a good one. The strength of cryptography lies in the choice and management of the keys; longer keys will resist attack better than shorter keys.

With the introduction of IT Act 2000, the electronic transactions and electronically signed documents are valid under the court of law. Use of PKI in India is expected to increase rapidly and for activation of global e-commerce and so also the use of digital signatures within domestic and other Global PKI domains. This requires the interoperability of PKI based applications.

Digitisation of information, networking, and WWW (world wide web), has changed the economics of information. The Copyright Act 1957 as amended up to 1999 takes care of the technological changes that still require major amendments in order to deal with the challenges posed by the computer storage, Internet and the digital revolution. With India being party to Trade Related Aspects of Intellectual Properties (TRIPs), electronic transactions of IPRs and consultancy services related to them is expected to be a new possibility in India. This may also promote our technology base to keep pace with global trends. As India has already enacted IT Act 2000, this allows transactions signed electronically for e-commerce primarily to be enforced in a court of law. Several issues arise when we consider using digital documents and exchanging them over the Internet, such as eavesdropping, tampering, impersonation etc. All these can be remedied with the use of public key infrastructure (PKI). However, the question of the time when a document was created may be answered by using Digital Time stamping service, which is based on PKI. This information may prove to be crucial for most e-commerce legally binding transactions, in particular for supporting non-repudiation of digitally signed transactions.

1.7 SOLUTIONS/ANSWERS

Check Your Progress 1

1) Cryptography is the service of writing in secret code. It is the key technology used when communicating over any un-trusted medium, particularly for Internet.

2) (i) Authentication,
(ii) Privacy/confidentiality,
(iii) Integrity, and
(iv) Non-repudiation

3) (i) Secret key (or symmetric) cryptography,
(ii) Public key (or asymmetric) cryptography, and
(iii) Hash functions

Check Your Progress 2

1) (a) 1
(b) 0
(c) Indeterminated
(d) 10

2) Breaks a message into fixed length units for encryption.

3) Variable

4) 128, 192, or 256 bits

5) Rijndael algorithm

6) Variable block feedback

Check Your Progress 3

1) Rijndael

2) It is slower than asymmetric key encryption

3) Elliptic Curve

4) If the public key encrypts, and only the private can decrypt

5) Spread the influence of a plaintext character over many ciphertext characters.

6) (a) True
   (b) False
   (c) False

1.8 FURTHER READINGS

1) *Network Security Essential - Application and standard*, Willam Stallings, Pearson Education, New Delhi.

2) *Computer Networks,* A.S. Tanenbaum, 4th Edition, Practice Hall of India, New Delhi, 2002.

UNIT 4 NETWORK SECURITY-II

4.0 INTRODUCTION

Network security - A term which can be defined in different ways; Securing the hardware present in the network; securing the software/application or most importantly securing the information that is going to be exchanged among devices present in the network. In short, the security aspects that can be attributed to use of computer networks is known as network security. One must understand the term use of computer networks before understanding its security aspects.

Let's understand this with an example of distributed network of an organization. Suppose there is an organization XYZ which has several branches across the world. The data center of this organization is situated in its head office at New-York. All the other branches share their data and extract the information from this data center. Now, think about the following questions:

- Who can share and access the data from this data center?
- What data can be shared and accessed from this data center?
- How will you ensure that the data which you are receiving at your end, is exactly the same which is stored at the data center?

Let's talk about the first question: you might be thinking the employees which are the part of organization. Yes, you are right, but can only employees are the

stakeholders of the company. No, all the people are the stakeholders paying for the benefit of the services provided by this organization. So, it means those who have associated with the organization's services may share or access the data center. But, how can we find out whether some person is associated or not. Here, comes the term authentication which is the first part in network security. We'll discuss this in detail in further sections.

Coming to second question: Suppose a scheme for authenticating the user is adapted at data center and now only valid users are allowed to access the information. But, can each user access every information available on the data center. No, because we have many stakeholders one may be the employee, manager or client. They may be interested in variety of information, but complete data available on the data center may not be relevant to all of them. So, here comes the term accessibility/ permission/ rights which is the second part of network security. These rights ensure fetching or sharing the relevant information to the respective stakeholders. Many access mechanisms are there which can be used to provide these rights, we'll discuss it later.

Considering first two problems are solved by authentication and access rights. Now the last part of this scenario is to ensure the integrity of the data which means the data is not tempered in between the network. Let's understand this: Suppose you have authenticated by giving your identity and also shown your access rights to the data center for accessing the information. Data center has permitted you to access the data. You have acquired the relevant data which is coming through the public network/ Internet. It is quite possible that you may lose some of the data since it travels in form of packets in the network. There may be different reasons congestion, packet loss, delay or some kind of cyber-attacks. Therefore, the integrity of the data need to be ensured when our data travels through network. Here come the third term data-integrity mechanisms under network security. Thankfully, our researchers have developed so many mechanisms to ensure data loss due to congestion, packet loss and other network based problems. However, cyber-attacks are not very limited and increasing day by day with the rapid growth of internet and their users. Moreover, aforementioned reasons for impacting data-integrity only affects the information whereas these cyber-attacks possess capability to hamper the data as well as hardware and software inside the network. Therefore, everyone in today era who is using internet should have the understanding of such cyber-attacks and their counter-measures.

We have come across different problems in network security, but scope of this chapter is limited to the third question where we have to ensure the security of the networks from these cyber-attacks.

Thus, we'll focus on to exploring the various cyber threats, attacks and their impact in the network. We'll also cover the ideas to handle these cyber-attacks to ensure the security of the network.

4.1 OBJECTIVES

After going through this unit, you should be able to understand the following:

- Basic understanding of the cyber-crimes and cyber-attacks

- The different types of cyber-attacks in computer networks, and
- The mechanisms to ensure the data integrity and security of the networks encountering these cyber-attacks.

4.2 CYBER THREATS, ATTACKS AND COUNTER MEASURES

Let's start understanding these concepts with the history of computer crimes. In 1983, as the internet was introduced to the world, computer crimes also came into the existence. The momentous Morris worm, the first denial of service (DoS) attack came into 1988. This worm contains a few dozen lines of code that replicated rapidly and hampered almost 10% of all the computers over the internet. Apart from these attacks, flash and browser add on vulnerabilities were introduced in mid-90's by the hackers to control the computers from remote location. As the software industry grew rapidly in early 2000s, they were less bothered about the security concerns, and thus the multi-accessed and insecure software were more vulnerable. The hackers and attackers used this opportunity to manipulate these software including some of the Microsoft software. As people started spending more time over internet, phishing (a type of social engineering where an attacker sends a fraudulent message designed to trick a human victim into revealing sensitive information to the attacker or to deploy malicious software on the victim's infrastructure like ransomware.) has taken place large in number. Mobilization has created a massive market for spyware and monitoring. Mobile devices have expanded public networks and wireless connections, where hackers lurk. Since 2014, automotive and other machine software has also fallen victim. IoT, a market that continues to grow exponentially, has faced many security concerns as it sits at the center of software, cloud, network and physical access concerns.

In today's world of internet, everybody is prone to cyber-attacks. So, we need to understand the various cyber threats, attacks and its counter measures. In this section, we'll define these terms one by one.

4.2.1 Cyber-Threats

As per the definition available on Oxford Dictionary, cyber threat is "the possibility of a malicious attempt to damage or disrupt a computer network or system." But the scope of cyber threat is not only limited to damaging the networked systems. It also includes an attempt made for unauthorized access of networked systems as well as infiltrating or stealing the data from those systems.

In this definition, the threat is defined as a possibility. However, in the cybersecurity community, the threat is more closely identified with the actor or adversary attempting to gain access to a system. Or a threat might be identified

by the damage being done, what is being stolen or the Tactics, Techniques and Procedures (TTP) being used.

Types of Cyber Threats

The list of most common threats published by R. A. Grimes in 2012 consists of following unwanted means of attacks in user's system or account

(i) Unpatched Software (such as Java, Adobe Reader, Flash)
(ii) Phishing
(iii) Social Engineered Trojans
(iv) Network traveling worms
(v) Advanced Persistent Threats

However, these threats are still commonly occurring, but the various game changing technologies like big data, cloud computing, Internet of things and wide use of mobile device usage have also contributed and widened the landscape of these types of attacks. Although, these new technologies have made things easier and provide more mobility to people, some adverse effect can also be seen in terms of the number of threats possible after 2016.

Cyber threats typically consist, but not limited to, one or more of the following types of attacks:

- Advanced Persistent Threats
- Phishing
- Trojans
- Botnets
- Ransomware
- Distributed Denial of Service (DDoS)
- Wiper Attacks
- Intellectual Property Theft
- Theft of Money
- Data Manipulation
- Data Destruction
- Spyware/Malware
- Man in the Middle (MITM)
- Drive-By Downloads
- Malvertising
- Rogue Software

You will get a thorough knowledge of some of the attacks in later section. Before going to discuss these threats, we need to understand the source of such threats so that we can counter them. For prevention of such threats, one can argue that, first we must know the tactics, techniques and procedure (TTPs) being used by the attackers. However, digging deep into the TTPs can't give the significant way of prevention since these TTPs are evolving day by day. Since the way of such threats will change continuously, but the source of them

will remain same. Therefore, it is more important to know the real source of threat. For this researchers and the domain experts have clearly stated, "Behind any cyber threat, there is always a human element". So, for identifying the cyber threat, we need to focus on to identifying the person with a motive behind all such activities. As we have seen at present every field is technology-driven. Cyber threats are taking advantage of this high technology and malicious attempt to damage the computer network & computer information system. These attackers also attack personal computers or applications.

Now the question arises, how do we react to these types of attacks. Instigators every time come up with new ideas/technology to target computers with malicious attacks. To mitigate this type of risk we should have a good plan with a listed set of actions to respond to these attacks. However, we should always keep in mind finding what type of cyber-attack it is, we should also emphasize the person behind this attack.

For instance, according to the details given by SecureWorks, in June 2016 Hillary Clinton's presidential campaign emails were attacked by Russian Threat Group-4127. Later. in September Hillary Clinton's emails were presumed to be attacked by some foreign intruders. However, in both the cases, the attack was not done as it was presumed. So, we can see in both cases target was the same, however, the technology behind this cyberattack was different.

There are following major sources of Cyber Threats which can be explored with respect to target being attacked:

- Nation states or national governments
- Terrorists
- Industrial spies
- Organized crime groups
- Hacktivists and hackers
- Business competitors
- Disgruntled insiders

4.2.2 Cyber Attacks

In simple words, the term Cyber Attack can be defined as the threat which has already taken place. Although, various definitions can be found in literature, but all of them state the common aim to compromise the data integrity, data confidentiality and availability. As far as new technologies are concerned, new ways are being discovered to attack and remain untraced. Despite continuous growing new ways of attacks, traditional threats defined above are still the main source of such attacks.

There are various attacks given in the literature which results into different set of attacks: For example: Man in the middle attack (MITM): Suppose X and Y are two communication ends and Z is an external entity that wrongly enters in between X and Y. The real communicating entities X and Y don't have any idea about the presence of Z. In this way, Z receives every

message sent from X or Y before its designated recipient. This leads to other attacks which breaches sensitive information and unauthorized access or it may also result into altering the messages.

- **Social engineering** can be defined as techniques which can be used to gain un-authentic access to information through human interaction.
- **DDoS** (Distributed Denial of Service) can be defined as unavailability of data or services by flooding the server with number of fake requests or commands, thus it leads to in-operational service or application.
- **Brute force attack**: When the repeated attempts are made by the attacker to get the sensitive information viz. passkeys, techniques of encryption or decryption etc.
- **Phishing** is a technique which aim to get personal and protected information from users through pretending that message is coming from the authentic sources for example: any repudiated website or mail.
- **Malware** is a generic term describing types of malicious software, used by the attacker to compromise the confidentiality, availability and integrity of data. Most common types of malware are: viruses, worms, trojans, spyware, ransomware, adware and scareware/rogware.

Although, it is hard to determine the exact number or percentage of different attack types, however the most common attacks are: denial of service, malicious codes, viruses, worms and trojans, malware, malicious insiders, stolen devices, phishing and social engineering, web-based attacks. Nevertheless, the results could easily be split into four categories, depending on the objective of the attack: cyber-crime, cyber espionage, cyber war and hacktivism.

4.2.3 Counter Measures

Generally, each organization issues some counter measures to prevent its systems from various attacks. Out of those counter measures, following is the list of actions that can be performed

1. **Train your Staff members** against fraudulent emails impersonating someone from the organization itself and revealing the secrecy of the organization in terms of sharing the key information for example access codes of internal servers, employee id and passwords etc. One of the powerful ways to protect the organizational privacy is to make your staff aware about current attacks and possible threats based on the designation and level.
   The possible steps to prevent these attacks:
   - One can verify the email addresses and then send the sensitive information.
   - Do not click the links which seems to inappropriate or one can verify by calling the person who has sent it.

2. **Fully Updated system and software** Its always a good practice for organizational people to keep their system and software up to date. Since, weaknesses in your systems provide an easy way to attackers to capture, access and stealing the information. Therefore, once must ensure to have the patch management systems that automatically updates the systems and software.

3. **Ensuring endpoint protection for remote devices** In today's era where IT is the backbone of most of the industries, remote access of organizational network is very common to ensure distributed nature of work among employees. On one side it provides parallel and distributed working environment, but on the other side it opens up the ways to security threats. So, these open paths need end to end security that can be achieved with endpoint protection software.

4. **Firewall** The most effective way to protect our network from any kind of cyber-attack is to use Firewall. It can be considered as a shield for our network and/or systems which helps to prevent brute force attacks.

5. **Always backup your data** The network of computer systems become more prone to cyber threats when they have always on connection with internet. In such a case, one must have backed up all the data for avoiding financial loss as well as personal loss

6. **Access management for your systems** Protecting your systems/networks by utilizing the access control mechanism is most important as you may believe or not, one of the possible attacks is the physical one and i.e. the most dangerous attack. Since these physical attacks are possible via capturing and injecting some infectious files in one of your system which can give access to the whole network. Therefore, one must ensure to have access control mechanisms to limit the perimeter of physical security of devices and networks.

7. **Enable WiFi Security Options** In 21st century, most of the devices are WiFi enabled. Due to this option, every device connects with different networks at different times. Out of those networks one may be infectious e. So, your device may get affected/infected by connecting with that network and once you connect your infectious device to your organizational network, it may also get infected if it is not secured. Thus, you must enable Wifi security options in your devices all the time.

8. **Secure Employee personal information** The systems and networks become more vulnerable to attacks when multiple users use the same credentials. It is the best practice in an organization that every

employee should have separate login credentials for enhancing the security up fronts.

9. **Passwords** Generally, in an organizational network/system or personal systems, people use to keep same password for various applications which cannot be considered a wise idea. Since, once an attacker gets your password, he/she can damage your system completely or can steal all the information from various applications. Therefore, it is always advised not to keep your passwords uniform across the applications and change your passwords frequently to make your system more secure.

   However, these are few guidelines/steps that can be followed to make our system/network secure, but a number of other steps can be taken into consideration based on the need of the organization. Basically, it depends on the size of the organization that what kind of security parameters are needed to fully secure our systems.

Check you progress 1

- ✓ How would you differentiate cyber threat and attacks?
  ……
  ……
  ……
  ……
- ✓ What are the necessary counter measures that would be preferred for big organization?
  ……
  ……
  ……
  ……

4.3 TAXONOMY OF VARIOUS CYBER ATTACKS

The term taxonomy can be defined as scientific classification of anything which means arranging things into groups. Cyber-attacks came into the existence in 1956, since then a variety of attacks are known in the field of computer security. Hence, a proper classification is needed to identify the type of attacks, its behavior and the impact. This not only helps in detecting the attacks, it also supports in identifying the counter measures that need to be adopted to mitigate and remediate these cyber vulnerabilities. So, in this section, we will see some known cyber-attacks, the possible attackers, the motive behind the attacks and consequences.

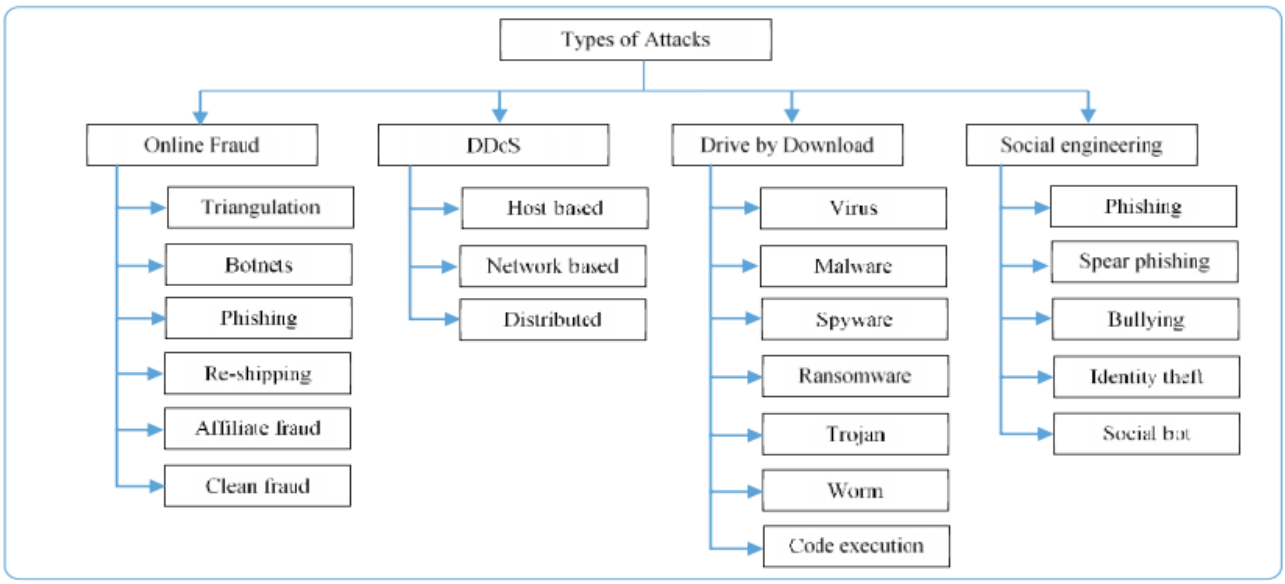


Figure 1: Classification of Attacks

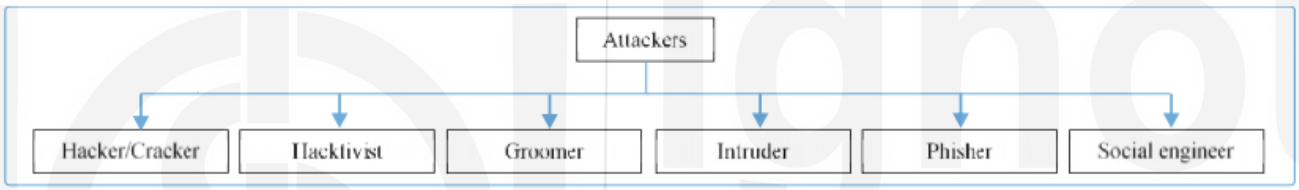


Figure 2: Possible Attackers

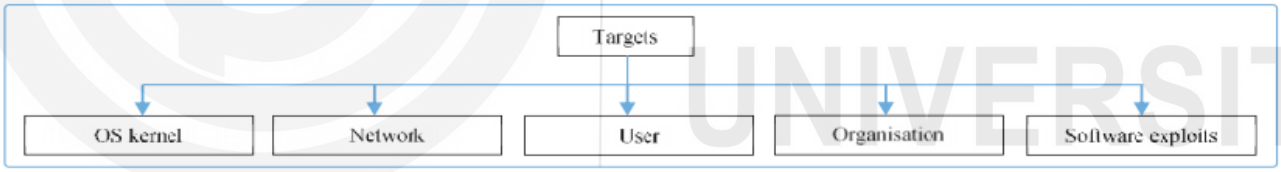


Figure 3: Targets

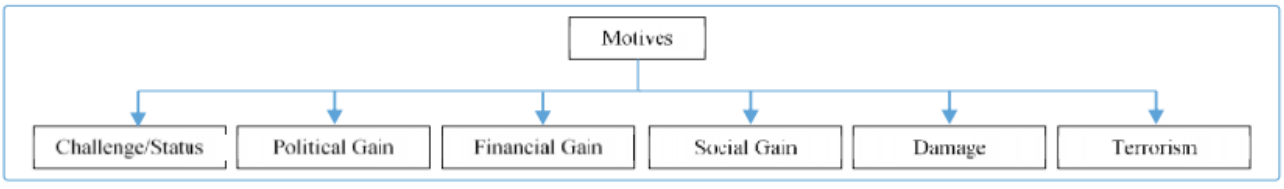


Figure 4: Motives

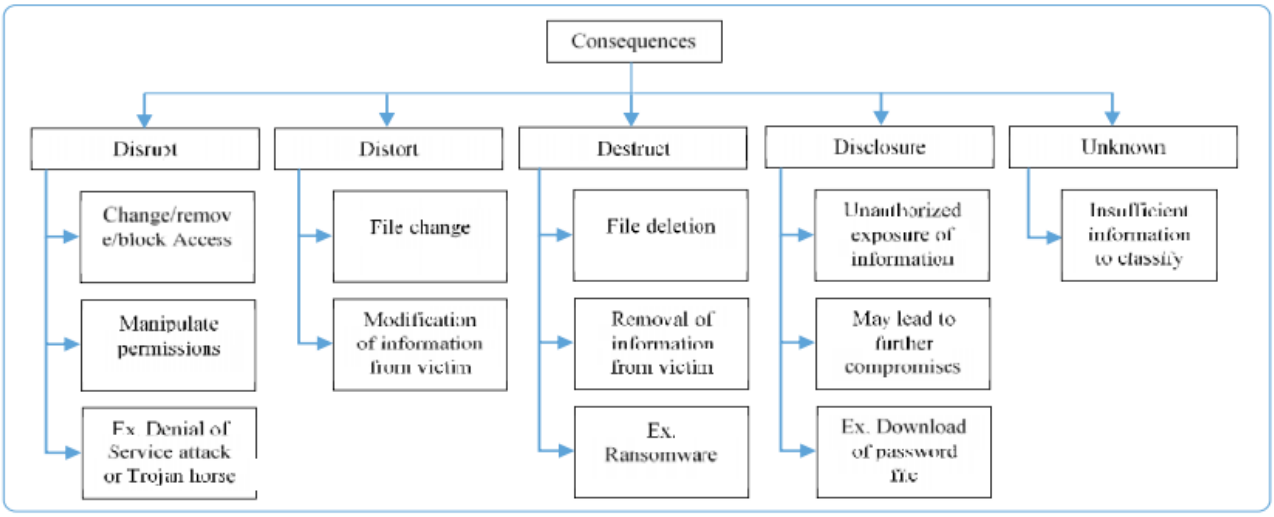


Figure 5: Consequences

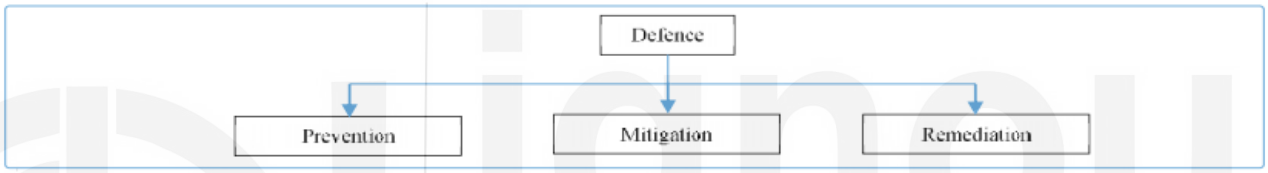


Figure 6: Defense Mechanism

4.4 VIRUS, WORM AND TROJAN, DOS ATTACK, DDOS ATTACK, PHISHING ATTACKS, MALWARE, RANSOM

Virus A self-replicating malicious computer program which affects the other programs in the system/network is known as computer virus. Alternatively, it can be defined as a software entity which is self-executable and holds the capability of affecting any code, files, systems or networks. The main objective of creating such an entity is to gain administrative rights and steal sensitive information of user by infecting the vulnerable systems. These entities can be easily spread through attaching the malicious code in the emails- a vulnerable user open this attachment and get easily infected or it can traverse through the infected sites, malicious executable code, the advertisements etc. There is one more very common way to get affected with such malicious code i.e. through the already infected storage devices such as external Hard Drive, USB etc.

There are following categories of computer viruses which are mentioned below:

(i) **Boot sector virus:** This category of virus attack directly when you start your system. It affects the boot process and prevents the system to start.

Systems may get affected with such type of viruses through external devices such as USB drive, external hard disk etc.

(ii) **Browser Hijacker:** You may have encountered one problem i.e. redirection from one site to another website several times while browsing over the internet. Such websites which redirect you to new websites are potential link for such viruses. These viruses first hijack your browser and then performs some unintended functionalities such as opening new links and applications frequently.

(iii) **Web Scripting Virus:** Such viruses use the codes written for web pages or browsers to control the functionality of any system. Once clicked on such webpages, it automatically gets into your systems and may infect your system.

(iv) **Resident Virus:** It is a different category of virus which resides in the memory of any computer and acts once the system starts.

(v) **Polymorphic Virus:** As the name suggests, Poly means many and morph means different forms; such viruses are very hard to detect since it changes its code each time when they are executed with an infected file.

(vi) **File Infector Virus:** One category of virus which generally hamper the designed functionality of the executable files by attaching some malicious codes.

(vii) **Multipartite Virus:** The category of virus works in many ways. It is not only able to infect the system boot sectors, it can also infect program files, application lead to temporarily or completely shut-down the applications or systems.

(viii) **Macro Virus:** This type of virus is written in the same macro language as software applications. It spreads when the infected document sent generally by email is opened.

There are various signs which show you that your system has been infected with one of the aforementioned viruses- homepage changes every time, slowing down the system, password changes every time you open an application, bulk emailing from your email address, frequent window popups. To protect yourself against such virus, make a good habit of adopting few steps while browsing over internet. Following are the steps that can save you from getting infected from viruses:

1) Use of any trusted Antivirus.
2) Always scan the files downloaded from internet before opening it.
3) Scan the external devices every time before use.
4) Do not click on unnecessary links/ pop ups.
5) Special attention need to pay when you are using the file sharing options using any application. Always scan such files.

Worm It is a type of computer malware. The malwares are self-replicating viruses. However, malwares known as worms have main functionality to replicate as much as it can and infect the other systems. While infecting the other systems it stays in its source system too and create each system capable of infecting other systems. It often spreads through exploiting the functionality of operating systems which are automated and not visible to user with bare eyes. It can only be noticed in systems when user experiences a lot of resource consumption without doing any activities. The other signs may be slowing down the systems or halting of the tasks.

The main difference between a computer virus and worm stated by "Security of Internet" report published in 1996 by the CERT: A software Engineering Institute at Carnegie Mello University, is that worms, once are in action, need no human intervention for its replication and spreading all around the network of systems.

On the contrary, viruses may also be self-replicating but they usually need some kind of action from the user for spreading and infecting the systems or programs.

The computer worms often use the vulnerabilities in networking protocols for propagating through the network. As soon as a computer worm infects any system, its main directive is to infect as many systems as it can. For instance, in Windows, there is a protocol known as resource sharing protocol which is a networking protocol. The very first version of one of such protocols was Server Message Block (SMBv1). WannaCry_ransomware was a worm which exploited the vulnerability of this protocol to infect the systems which were using this protocol for file sharing. As it uses the networking protocol for its propagation, it immediately starts a network search for finding out the potential systems. The systems in the same network who respond to such network query, get infected by this worm.

There are following categories of computer worms which are mentioned below:

(i) **Email worms** Generally, this type of computer worms use the user's email address book for sending out the emails carrying malicious code. Once the recipient of the mail receives such mail,

he/she thinks that the mail is from one of his/her colleague, friend or some authentic user. As soon as the mail is opened up, it infects the recipient's system. However, for encouraging to open up such type of malicious mails, it needs to perform strong social engineering and phishing techniques.

(ii) **File sharing worms** These worms are usually impersonate itself as a media files. These spread through the infected source files shared with using some external device like USB drive, Hard drive etc. It is mainly used to target the big industrial environments such as electricity supply services, water supply services etc. For example, Stuxnet is the well-known file sharing worm till date which has the capability to target big SCDA systems.

(iii) **Crypto-worms** The actions of these worms are defined to encrypt the files in infected system. One may wonder how all the files get encrypted, and is not able to open it. This comes under the category of ransomware attacks where perpetrators ask for money to decrypt the files.

(iv) **Internet worms** These are special type of worms that mainly target those websites which either not uses or uses a poor security protocols. Once such website is accessed by any user, first the system gets infected and thereafter, the private network attached to this system may get affected.

(v) **Instant messaging worms** This is similar to email worms. These worms are also come in the form of attachments or links though which the complete contacts may get infected. However, the propagating nature is different as compared to email worm, it propagates through any chat/instant messaging service.

For protecting one's system from such worms, these are the following measure that can be taken:

1) Regular updating the OS and its software patches
2) One can use the firewalls to prevent the unauthorized access to systems.
3) Use of any trusted Antivirus
4) Click only those links which you get from authentic people. Avoid checking of every link in messaging applications.
5) Encrypt your data.

Trojan It can be understood by a simple scenario- Everyone must have noticed, sometimes when you log-in in your system, everything gets popped-up or system behaves inappropriately. If such situation arises, it can be a possibility that a Trojan virus is there in your system. The Trojans are one of the most dangerous viruses which are not only able to steal your valuable and sensitive information, they can also open you up for potential cybercrimes by identity theft.

This type of virus generally comes under the category of malwares that impersonate itself as real/operational executable programs. It can be so destructive for your system once it is inside. It can also download and install other malwares to breach your security. Some other Trojans may directly try to breach the security. Based on this, there are two types of Trojans: one sits idle until it does not get any instructions from its host hacker, however others start reacting maliciously once they reached into your system.

There are following potential sources which make you vulnerable to Trojan attacks:

(i) **File sharing websites:** Every one of us must have used Torrent for sharing the files, movies, songs etc. There are other file sharing websites too over the Internet. The most appealing thing of these websites are we can have anything for free like games, movies, software etc. However, it makes you vulnerable to easily hacked by the hacker. For instance, a hacker has uploaded a cracked version of VLC media player or any popular software with a hidden Trojan on a file sharing website. Now, the clients who will download this file, may think that it is VLC software and execute it in their systems. But, unknowingly they are also installing hidden Trojan. Once this Trojan is installed, your system might be controlled by the hacker for performing any type of cyber-crime.

(ii) **Email Attachments:** This is a very common way to infect other's system. A hacker sends an email with attachment. As soon as anyone click on this attachment for downloading it to open, one may found himself infected.

(iii) **Spoofed messages:** The various desktop applications provides you the platform for chatting with others. This makes you vulnerable to Trojan attacks. Since the hackers can send you a spoofed message which seems like coming from a trustworthy person, but it is not. Apart from this, the attackers may also create a fake profile very similar to your trustworthy person. If you do not pay close attention to slightest variations in accounts, you may be victim of Trojan attacks.

(iv) **Unsecured/poorly secured Websites and Hacked WiFi Networks:** Sometimes hackers do not target the specific users. In that case they try to hijack the unsecured or poorly secured websites from where the files can be downloaded. Thereafter, redirect the clients request to fake websites and servers. Once the client downloads the file from fake/redirected website, it also downloads the Trojan. Hence, client's system can be compromised then by the hacker to perform various illegal activities.

Similarly, the fake WiFi networks can be created to compromise the client system. For instance, a hacker can create the hotspot with the same name on which a client can rely upon. When a client need to use Internet, he/she will

think that connection is going to be established with the right hotspot. However, he/she will be connected to hacker's hotspot. In that case, all the request will be forwarded through Hacker's WiFi. In such a case, all the information shared by you over that network may be compromised to perform any activity.

For protecting one's system from such Trojan attacks, these are the following measure that can be taken:

1) One should use the cloud services which supports the secure services and provide the recovery options.
2) One can use VPNs over public Wifi
3) Use any trusted Antivirus with real time protection
4) Make sure before opening any email attachments.

Denial of Service (DoS)/ Distributed Denial of Service (DDoS) attack

DoS An attack by a single system on a single system/network to reduce and restrict its access rights for the authorized users is known as DoS. There are two categories of DoS attacks:

1) **Flooding services:** In this type of DoS attack, generally system encounters with a lot of traffic which cannot be buffered in the server which leads to slowing down the system and eventually it stops working. The well-known flood attacks are as follows:

    (i) Buffer overflow attacks – Every system/network is designed to handle some finite/limited traffic. In such attacks, the idea is to increase the traffic to the network address of the system till that extent it crosses the limit so that it cannot be handled by the network administrator.

    (ii) ICMP flood – This type of attack occurs when there are some misconfigured network devices which can be leveraged by sending spoofed packets that ping every computer on the targeted network, instead of just one specific machine. The network is then triggered to amplify the traffic. This attack is also known as the smurf attack or ping of death.

    (iii) SYN flood – In such type of flooding attack, a continuous request made for establishing connection with server, but the handshaking never been completed

from the attacker side. This leads to saturate all the open ports with connection establishment request, consequently no legitimate user will be able to connect due to lack of open ports.

2) **Crashing services:** In these DoS attacks, attacker adopts those methods with which he/she can exploit vulnerabilities to cause system/service crash. Attacker takes an advantage of bugs present in the system and send that particular input which create this bug. Therefore, system crashes in some time and not available to access by the legitimate users.

DDoS An attack which is initiated simultaneously from multiple compromised systems to attack a single system/network is known as **DDoS**. The authorized users of such system/network find themselves not able to access the targets system/network. In other words, DDoS can be defined as DoS which are performed by multiple systems simultaneously

For performing the DDoS attack, an attacker selects the zombie systems to attack on a single system/network. Once these zombie systems are compromised, the attacker sets up the command and controller which controls these zombie systems to attack. These zombies are controlled with the use of bot which is a special type of malicious software. A group of zombies installed with bot is known as botnet. Figure 7 shows the flow of DDoS attack.

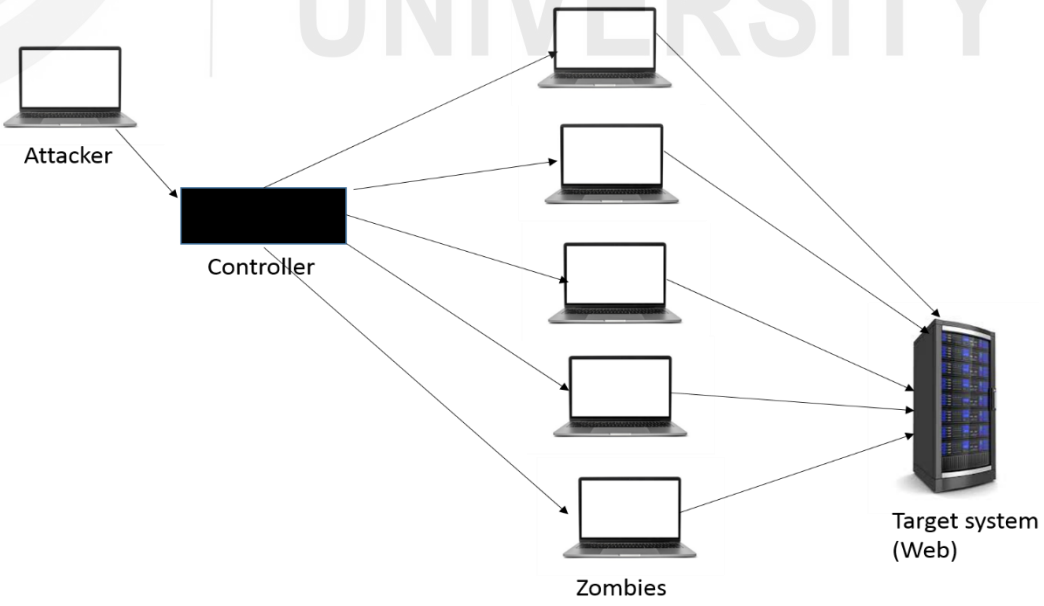


Figure 7 Distributed Denial of Service Attack (DDoS)

Phishing Attack It requires a social engineering techniques to target some specific users. Attackers performs such attacks by establishing the fraud communication with the users which seems to come from an authentic source. The main objective is to get the sensitive information of users such as debit/credit card information or login details and installing any malware over victim's system. Generally, it is performed through emails. There are two categories of phishing attacks based on impact of attack on individual:

1) For financial gain, the attackers try to get the victim's sensitive information like debit/credit card details or any other personal data which may be helpful in carrying out the fraud transactions.
2) For attack against an organization, attackers try to get the login information of employees for doing some advanced attacks. Advanced Persistent Threats (APTs) and Ransomware usually being performed with Phishing.

   Below are the following measures that can be taken to protect from such attacks:
   1) Educate the users/employees to recognize such phishing emails.
   2) Adopting the network security technologies such as email and web security services, antivirus to protect from malwares, access control mechanisms and behavior monitoring.

Malware It a malicious software, once installed on victim's device opens up for breaching the cybersecurity and make your device prone to several threats. For gaining unauthorized access or personal information, cyber-attackers create such malicious executable codes which can be run without the user's consent. It is usually performed for financial gain or completely damaging the device. It can affect any device whether it is macOS, iOS or windows. There are different malwares which are as follows:

1) Viruses
2) Spyware
3) Ransomware
4) Trojan horses

Apart from these malwares, there are different categories of malware attacks under which these malwares can be classified:

1) Exploit Kit It is kind of malicious toolkits which usually used by the attackers to find out loop-holes/vulnerabilities in victim's system/device. The task of pre-written codes in such kits is to get the details of software vulnerabilities. Once the task is performed successfully in one's computer, this kit injects the malware into the system using the loop-holes. In order to protect from such type of attacks, one has to always install software updates and security patches which make itself unavailable to such exploit kits.

2) **Malicious Websites and Drive-by-Downloads** Some malicious websites are specifically designed for malware attacks also use these exploit kits. Such websites work as a host for these kits. Once a user visits such website, exploit kit hosted start working and search for vulnerabilities present on the browser. In this way, malwares can found their way and driven by downloads to infect the system.
3) **Malvertising** As the name suggest- the attacks which are performed using malicious advertising are comes under this category. In malvertising, attackers purchase some advertising space on repudiated websites and embed the malicious code in their advertisements. Once clicked, it starts infecting without the knowledge of user. This is very similar to Malicious websites and drive-by-download.
4) **Man in the Middle attack (MitM)** This type of attack can easily be understood with real life scenario such as two persons are communicating with each other but they are at some distance. Third person comes in between the communication and changes the information passed from one to other person. In computer network, this is performed with the help of employing the poorly secured network or over the weak WiFi network. First, attackers find out the vulnerabilities for example, default passcodes or weak keys. Then, they intercept the communication between two entities over that network by putting themselves in between. Therefore, the sensitive information like credit/debit card details, employee login details can be captured easily.

5) **Man in the Browser attack (MitB)**This is very much similar to MitM. However, in this type of attack, the attackers need not to present nearby the router as in case of MitM. Here, attackers inject the malwares in target system, and once it is installed over the browser all the information will be collected by malware. Then the programmed malware sends this data to attacker, and thus it can be used for any cyber-crime.

Below mentioned measures can be taken to be protected from such malware attacks:

(i) One should always keep their software updated
(ii) One should take back-up of files at regular interval
(iii) One should always scan the executable files before opening it.

Ransom It is also a malicious software which is used by the cyber criminals to infect the target system. When any target system is infected with ransomware, it completely blocks the system for use or it encrypts all the data available on the system. The main aim of such type of attack is to get handsome amount of ransom from victim to release its data or provide access to his/her own computer. There are three steps that can be carried out by the victim after getting infected: a) One option is to give the money asked by the attacker and get access of files and system; b) Completely remove the malware from your

system by using any Antivirus software; c) Restart the device in safe mode. For protecting your system against such attacks, cyber security experts have suggested to use trustful security software. There are well-known examples of ransomware which recently came into existence like Locky, WannaCry, Bad Rabbit, Crypto Locker, GoldenEye, Jigsaw, MADO ransomware and Fair Ransomware.

There are mainly two categories of ransomware which are popular nowadays:

1) **Locker Ransomware:** As the name suggested, it works like a locker. Until and unless, you get the key, you cannot open the system. In other words, this type of attacks has the capability to block some of the system functionalities such as interaction with I/O devices with the OS. After getting infected, one can pay the amount asked by the attacker to allow you to use your system. Despite the fact that it stops operations of computer device, it doesn't affect the files stored on your system. Thus, damage to your files and sensitive information is still safe in such attacks.
2) **Crypto Ransomware:** This works exactly opposite to locker ransomware. It attacks on your data rather than system. Once a target system is infected with such ransomware, all the files, documents and other data files get encrypted. In this case, user may able to see their data but not able to use it. This leads to panic in user's mind. With this type of attack, cyber criminals usually add a countdown for paying the amount asked, otherwise you may lose all your data since they can delete all the captured files. One way to secure your system and data from such cyber threats is to always backup your important data in cloud.

Check you progress 2

- ✓ List out the main differences between MiTM and MiTB attacks?
  ……
  ……
  ……
  ……
- ✓ What does make ransomware more dangerous in comparison to other attacks?
  .....
  .....
  .....
  .....

4.5 Vulnerabilities

Cyber criminals are nowadays very active to take an advantage of weaknesses present in the systems/networks. These weaknesses/loop holes are known as

vulnerabilities. Though, using these vulnerabilities, objectives of different types of attacks may vary from one to other. Attackers may perform such activities for financial gain, personal gain or due to some political motives. Knowing these vulnerabilities plays a vital role in protection of the system. As one having knowledge of these vulnerabilities can effectively work against modern cyber-attacks. Due to lack of such knowledge and awareness, people do not get the specific mechanisms to adopt which can work against the attacks.

Before going to dive into the details of these system vulnerabilities, lets first understand the term vulnerability: Computer security vulnerabilities can be defined as flaws and weaknesses present inside the system which are exploited by the attackers to manipulate or damage the system. These vulnerabilities are differentiated from cyber-threats as they reside in the system itself. In the absence of any possible threat, there is no harm with these weaknesses or flaws. However, they maybe used by the cyber threat i.e. usually considered as an external entity.

There are different security vulnerability types which one should know before going to analyze any type of cyber-attack. The knowledge of these vulnerabilities provide the clear picture how an attacker can approach to your system for financial or personal gain. Following are the few major categories of vulnerabilities, we will discuss few in later sections in detail.

1) **Network Vulnerabilities:** This type of vulnerabilities is usually found in hardware or software managing the network such as WiFi routers, switches, hubs in big organizations or the poorly configured firewalls. The attackers exploit these weaknesses to intrude inside the network first, and then infect all the system connected inside the network.
2) **Operating System Vulnerabilities:** These are user specific vulnerabilities which can be exploited by attackers to attempt to damage a particular's system. This type of flaws generally present in the one's operating system. For instance, default admin account created in some OS through which anyone can install any software hidden programs.
3) **Human Vulnerabilities:** This is the most common vulnerability present in the area of computer networking. Since, people those who are not from the networking background also use Internet for different tasks in their day to day life. Due to the lack of adequate knowledge they leave their network open/public. In case of individual system, they don't worry about the firewalls, antivirus software etc. that leads to being an easy target for attackers. So, in the context of such vulnerabilities, humans are the weakest link in overall network security architecture.
4) **Process Vulnerabilities:** There are various vulnerabilities which exist in specific processes such as process for setting up the login id and passwords for any application. However, most of the process vulnerabilities are the result of human error/unawareness.

4.6 BUFFER OVERFLOW

Buffer overflow is a specific type of process vulnerability which is caused by constrained resources availability in software development phases. Specifically, buffer overflow is a situation where temporary space supported by any software consumed to its defined storage capacity. In such a case, the excess data take place into already occupied memory locations which leads to corrupting or overwriting the data present earlier at those locations. This flaw is generally used by the hackers to infect the users of the software. Moreover, it can be described as a vulnerability which is caused by software coding error at run time. Every software has some limitation, and those limitations are exploited by the attackers for unauthorized access to damage any corporate system. Buffer overflow is one of the well-known software vulnerabilities since it can occur in so many ways. Every time hackers find out the different way to attack, thus there is no specific strategy to prevent from such attack. Therefore, usually cybersecurity team works along with the development team for handling such cases.

The hackers exploit this vulnerability to manipulate the coding error to use infected system for unauthorized access to perform malicious activities. However, manipulating the error is not an easy process. This includes changing the complete execution path of the software and overwriting the contents already present at the location. It typically occurs when:

- The code is based on some outside data for controlling the functionality of the software.
- The complexity of code creates problems in accurately predicting the functionality of the software.

As a result of such attack, there are following impacts that may occur in your system:

1) Complete System Crash: The consequence of buffer flow attack may result into full system crash. Moreover, such attack may also lead to temporary unavailability of services provided by the software.

2) Access control: The arbitrary code is generally used for commencing a buffer overflow attack which is usually not defined in any software policies. Thus, losing the access rights.

3) Additional Security concerns: When an attacker uses such arbitrary code it also leads to breaching the other security policies defined in the program. Consequently, other vulnerabilities also exposed to be exploited.

The most common buffer flow attacks are:

1) Stack based: In this type of attack, an attacker replaces and send the data required by the application with a malicious code. This contagious data is stored in stack buffer of the application which

changes/overwrites the actual data on the stack, and then application returns a pointer to this data in the stack. As soon as attacker gets this return pointer, he/she get control over the complete stack and misuses the application on the target system.

2) Heap based: It is more difficult in comparison to stack based buffer flow attack. It floods the memory of the program once program gets into the execution state which is not a simple task.
3) Format input string attack: The attack with formatting input stack takes place when the program takes input data as a command and not able to validate it. This exposes opportunities to the attacker to modify the input string and takes control over the process and results into segmentation fault etc. Hence, triggers malicious actions in the system.

4.7 SQL Injection

SQL stands for structured query language which is used for querying and modifying the databases. In the internet world, any web service hosted over the internet uses database to store the information it's users and other details. Cyber criminals mainly try to intrude in database to capture the information from a web service. This intrusion is databases are possible with some sort of query language. Therefore, SQL injection is a way to intrude into the databases through which hackers try to gain unauthorized access and sensitive information. SQL injection is nothing but a simple piece of code to manipulate the database. It's one of the most prevalent and threatening types of attack because it can potentially be used against any web application or website that uses an SQL-based database.

For example, consider a website which give access rights to its clients only when user enter the login information over the website. These login details are generally taken through webforms hosted by the website. Once a user fills these login details in the form, the information entered by user is matched with the entries already existing in database at the backend. In case of mismatch, user asked to enter the correct login details. Otherwise, user allowed to use the website.

However, instead of these login details one can fill any information in these web forms. This provides a way to hackers/attackers to intrude in the database through their own requests. These web applications are generally designed in such a way that every request must be processed by checking through the database. So, these false request also go for search in the databases which leads to malicious activities such stealing valuable information of clients or modifying already existing entries.

Preventive measures that can be taken to handle SQL injection attack:

1) Use a trusted web application firewall

2) Use a trusted Antivirus software which provides the real-time security

4.8 BROWSER Vulnerabilities

Revolution in world of Internet came with the introduction of web browsers. Every employee, be it from small or large organization, use these browsers to access internet to perform their jobs. However, it has dramatically increased the productivity, but also exposes the organization's security breach points. Cyber-criminals found the easiest way to enter any system i.e. browsers used by its employees. Since every browser contains the information of cookies (simply a file which keeps records of last visit of a user) and uses plugins, so by entering into a browser an attacker can use these vulnerabilities to get the sensitive information of any user. Using the login details of any user, an attacker can enter into the complete system of an organization.

Apart from these cookies, cyber criminals also try to attack by injecting a malware into the browsers. These malwares reside in some malicious websites which is specially designed to do false activities. As soon as a user visits such websites, it automatically gets injected in user's browser and handed over the control to hacker. By exploiting the information transmitted from the browser, an attacker steals the sensitive information, and thus performs the malicious activities.

In addition to cookies and plugins, many other vulnerabilities are making things worse due to which various threats such as MItM, MiTB etc, are hard to detect.

4.9 OS Vulnerabilities

In an operating system, user uses various application software which exposes any system through many source of vulnerabilities such as browser vulnerabilities, client application vulnerabilities etc. Though, OS vulnerabilities are classified as follows:

1) **Client side vulnerabilities:** In this category, an attacker tries to infiltrate via different applications installed/supported by operating system. These applications may be any web-browsers, different office software where client interaction is mandatory, email client applications or media players.
2) **Server side vulnerabilities:** These vulnerabilities generally found on server side computer system or in between the client and server. For instance, this can be understood by a simple scenario in which an attacker can attack thorough SQL injection method on any web application. This leads to get access of the database and employing any malicious activity at the sever side. In other words, a server side vulnerability is exposed due to vulnerabilities present at the client side, web browser as mentioned in the above scenario.

We know popular operating systems such as Windows 7/8/X, Linux, Unix, MacOS etc. All the operating system carries some vulnerabilities; however, vulnerabilities present in Windows OS are easy to understand since most users are well familiar with functionality of windows. All of us must have used Remote desktop connection (RDC) in our Windows OS. This RDC client is known as popular OS vulnerability since they allow to execute the external codes for establishing the connection. Attackers exploits this for running their malicious code, and thus infiltrate in the user's system and damage the system or steal their sensitive information. Similarly, one more very popular functionality provided by Windows OS i.e. Windows Remote Desktop Gateway (RDG). An attacker can run their arbitrary code similar to RDC, but there is no interaction required from the user in RDG. Hence, it can be used to attack and get unauthorized access of the system with specially designed requests. In contrast to this, client based OS vulnerability are supposed to exploit by a compromised user for connecting with malicious server.

Check you progress 3

- ✓ Which type of vulnerability is found as the easiest way for intrusion by the cyber-criminals, and why?
  ……
  ……
  ……
  ……
- ✓ List out some of the OS vulnerabilities in Linux and MacOS.
  ……
  ……
  ……
  ……

4.10 BASIC COMPUTER FORENSICS

Computer forensics is an art which describes the present state of a digital artifact such as mobile phones/devices, computer systems, hard drives, pen drives. Moreover, it also includes investigating and linking the data present in the computer systems, email applications etc. for describing any event. The field of computer forensics is comparatively new than cyber security, however recent advancements done in the past twenty years have overwhelmed this field to contribute in various legal proceedings in the court. A formal definition of computer forensics is given by Kruse and Heiser in year 2002. According to the authors, it is defined as involving

"the preservation, identification, extraction, documentation and interpretation of computer data".

The existence of computer forensics starts from 1980s when computers/digital devices have started revolutionizing. This revolution in digital world has increased productivity of employees and industries along with various criminal activities such as online frauds, hacking and cracking. This leads to several cyber security experts to design specific techniques to investigate the fraud and crimes. Specifically, these come under the emerging field of computer forensics. Although, it started in 1980s, but a rapid growth has been recorded in year 2002-2003 when cyber-crimes were increased by 67%. Currently, it is being used in investigating various frauds and crimes such as murders, rapes, child pornography, cyber stalking etc. forensic investigations are generally performed on static system rather dynamic. In this, the process of investigations usually follows a set of digital forensic phases which are as follows:

- Acquisition of digital artifacts.
- Examination of acquired these digital artifacts.
- Analysis of examined artifacts.
- Preparation of reports for legal issues.

Following are some major techniques being used in investigations by forensics team:

1) **Cross-Drive Analysis:** This technique is used where typical source of information (from where the data is obtained) remains distributed in nature and forensic teams do cross examination of data to correlate and analyze the findings. Researchers are still working to develop such techniques which can give more insightful results. This is generally used in identification of social networks and detecting anomalies present in the digital artifacts.
2) **Live Analysis:** This technique is used where computer systems are on and in working condition. Forensic team uses such techniques to extract the evidences with the help of custom forensics or SysAdmin tools. These tools examine the systems starting from operating system and are able to deal with encrypted files.
3) **Analysis by recovery:** Techniques used in recovering deleted files from the system for finding out the cause of incident/fraud happened. Forensic team exploits the nature of some operating systems which do not always allow to delete data from physical disk sectors. File carving is a technique which can be used to find out the headers representing the disk sectors of such files in physical space. After getting such headers, team reconstruct the deleted files for further analysis.
4) **Stochastic forensics:** This technique is used in case of data theft. Forensic team uses the probability theory or stochastic properties of systems for investigating the digital artifacts.
5) **Stenography analysis:** The term stenography defines a way to hide the data in image files Attackers used this technique to hide pornographic pictures of children or the information they do not want to get

discovered by anyone. However, forensic team exploits the technical details of an image where pixel information gets hash information. When any change occurs in original picture, this hash file changes which leads to detect alteration or hidden information.

4.11 Recent Cyber Attacks

In 21st century, cyber security is essential for every individual and organization due to vast use of Internet. Although, various self-learning/experience based solutions and software solutions such as antiviruses exists, but cyber criminals are in continuous search for getting success in breaching out these solutions. Various vulnerabilities that still exist in system/network exposes them to attackers to think about an alternative way to infect. Thus, it becomes more important to educate people about these vulnerabilities which are discussed in previous sections. Now in this section, we will see few recent cyber-attacks which happened against well-known organizations. The impact of attacks on an individual or organization can be understood with the details given in each of the attack. Herein, the following are some recent attacks from last two years:

1) **Attack on Popular Social media website Facebook:** In the month of APRIL 2019, it was reported that around 540 million user data exposed and published over Amazon's cloud service. It was found that two third party application developer were involved in this activity. At later stage it was reported that there were some political firms behind this attack. The captured data from millions of Facebook profile users were then used for false political advertisements. In the end of the month, Facebook again revealed about one more such leak. However, it was not done by attackers, whereas millions of user mail ids were made public unintentionally by the Facebook itself.
   From these two incidents, one can understand that how much data these large organizations are handling- and one small mistake can put the information of millions of people at risk.
2) **Attack on graphic design website CANVA:** This is another recent example of cyber-attack where around 140 million user's data got exposed. The hackers intruded into main servers of this website and get access of the information such as user login IDs, passkeys and other profile information. However, these details were encrypted but still put millions of users at risk.
3) **Attack on Servers of MGM hotel:** Similar to CANVA attack was recorded in February 2020 with MGM hotel. More than 10 million people's information got exposed who stayed there in past. The hocking agency got several personal information such as name of the customers, address, contact details, email addresses, DOB. Few of those information belongs to well-known business personals,

celebrities, employees working in government agencies and tourists. However, there was no loss occurred in monetary terms to any customer. Hence, it can be inferred that no card details were leaked in this attack.

4) **Ransomware attack on California university:** In June 2020, attackers had attacked the university servers with a malware known as Ransomware. They had targeted various servers with aim to get a handsome amount from the university to release their data store on those servers. With this all data became inaccessible due to encryption posed by the malware. This results into paying the amount hackers had asked for. According to the reports, university had given around $1.25 million to the hackers for releasing their servers.
5) **Attack on World Health Organization:** In march 2019, when COVID pandemic broke across the world, WHO had played a crucial role to find out the ways to fight with this novel corona virus. Many of the international agencies were working together to fight against COVID. But, in april 2020, it was revealed that around 25000 email addresses were hacked by the hackers. All the email addresses and passwords leaked online along with other information such as the names of the different agencies and groups who were fighting against the pandemic. The names in the list includes National Institute of Health (NIH), Gates Foundation and US centers for Disease control and prevention (CDC).
6) **Attack on Zoom video conferencing service:** During pandemic in 2019, the various industries had adopted work from home culture. This resulted in high demands of applications like zoom video conferencing, webex, google meet etc. People started using these applications for their personal and professional meetings. Now, it became easy for cyber criminals to attack on an individual or organization using the vulnerabilities present in such applications. For instance, ZoomBombing was one such attack which enabled attackers to join personal/professional meetings and gain access of the conversation made. Moreover, they had distracted the host and other members by sharing the offensive picture and videos. However, the security levels were increased in zoom after this attack.

4.12 FIREWALLS AND INTRUSION DETECTION SYSTEMS

In this section, you will study about the most important preventive measures that can be considered to protect any individual or an organization from cyber-attacks. A firewall and Intrusion detection system both works for protecting you from various cyber-attacks at different levels. However, there are some similarities and differences in the working of both firewall and an IDS.A firewall can be considered as security watchman standing at the front door of

your home whereas an IDS can be considered as security cameras installed inside your home. As watchman standing at the door can stop any person entering into the home, firewall can stop/block the incoming connection in the network/system. Similarly, the security cameras installed cannot block any person from entering in the house, but it can detect any intrusion attempts made by the person already there in the house. In the same way, an IDS cannot block the incoming connection, but reports or generates alarm in case of intrusion. Before going to explore the various firewalls and IDS available in the market, first we need to understand the basic concepts of firewall and IDS.

Firewall A firewall can be a software or hardware both, it depends on the nature of the system where security restrictions are required. The main functionality of a firewall is to provide access rights to authorized individuals whereas it denies/blocks the permission to unauthorized users. This mainly works in networked environment and resides in between local network and internet. Apart from giving access rights, it filters the unnecessary traffic coming from internet to your local network that might be harmful and cause problems in systems.

Intrusion Detection System (IDS) An IDS can also be a software or hardware installed on the network or host to detect and report intrusion attempts to the network. There are two types of IDS, one is network IDS (NIDS) and other one is Host IDS (HIDS) depending upon where exactly it is installed.

The table given below consists the major differences between firewall and IDS:

Firewall	Intrusion detection system
A firewall cannot detect security breaches for traffic that does not pass through it (E.g. a gateman can watch only at front gate. He is not aware of wall-jumpers)	IDS is fully capable of internal security by collecting information from a variety of system and network resources and analyzing the symptoms of security problems
Firewall doesn't inspect content of permitted traffic. (A gateman will never suspect an employee of the company)	IDS keeps a check of overall network
No man-power is required to manage a firewall.	An administrator (man-power) is required to respond to threats issued by IDS
Firewalls are most visible part of a network to an outsider. Hence, more vulnerable to be attacked first. (A gateman will be the first person attacked by a thief!!)	IDS are very difficult to be spotted in a network (especially stealth mode of IDS).

Table 1 Firewall and IDS

There are various firewalls and IDS available in market. One can choose based on the level of security required in the system/network. A list of firewalls and IDS is given as follows which can be explored further:

Firewall software	Intrusion detection system software
FortiGate NGFW	SolarWinds Security Event manager
Juniper Firewall Check Point Next Generation Firewalls (NGFWs)	Bro
Sophos XG Firewall	OSSEC
Huawei Firewall	Snort
WatchGuard Network Security	Suricata
Palo Alto Networks Next-Generation Firewall	Security onion

GlassWire Firewall	Open WIPS-NG
Cisco Next-Generation Firewall Virtual (NGFWv)	Sagan
CrowdSec	McAfee Network Security Pl
Azure Firewall	Palo Alto Networks

Table 2 List of few Firewall and IDS software available in market

Check you progress 4

- ✓ How does forensic expert find out the hidden information in an image and what is the name of technique to hide such information?
  ……
  ……
  ……
  ……
- ✓ Comparing a firewall and an IDS, which one is more powerful in context of securing your system/network?
  ……
  ……
  ……
  ……

4.13 SUMMARY

In preceding sections of this chapter, best efforts are made to present available knowledge in the domain of network security. Currently, extreme use of computers in distributed manner require networking strategies to make them connected with each other. This implementation further poses a lot of security concerns for the systems involved in such networks. So, this chapter gives a detailed overview of security concerns like threats which can result into many cyber-attacks. After that discussion move towards defining the major reasons due to which one may be exposed to these attacks are known as vulnerabilities. Finally, the preventive measures are discussed through which individuals can protect themselves from these cyber-attacks. In addition to this, few recent cyber-attacks are also discussed which includes the popular attack on

Facebook. At last, readers are suggested to explore topics such as firewall or an IDS for effective solution which may fight against these threats.

The objective of this chapter is to make aware of the importance of cybersecurity area to students and readers of this book so that techniques mentioned in various sections may be implemented to work in secure environment. Although, topics covered are presented in a very general way so that beginners can easily understand and implement in their day to day life. Furthermore, the topics can be explored for more details following the material link provided in Further Reading section.

4.14 SOLUTIONS AND ANSWERS

Check you progress 1

1. Both the terms cyber-attack and cyber threat are correlated with each other. A threat is a possibility of occurrence of an attack whereas an attack is a malicious activity which has already taken place to breach the security of system/network.
2. The necessary counter measures in big organization include the following points:

- Educating the employee of the organization at every level since one weak link may damage the complete network of systems.
- Use of good firewall, antivirus and an IDS to ensure safety from outside, inside and from intermediate networks.

Check you progress 2

1. The main differences between MiTM and MiTB attack are as follows:
    - an attacker need to be nearby the router in MiTM, whereas in MiTB, the attacker need not to be there near the connecting router or switches.
    - In MiTM, an attacker needs to watch the network traffic continuously, whereas in MiTB this is not the case since hacker already get control of browser.
2. In comparison to other type of attacks, ransomware is more dangerous because it completely blocks the system by encrypting all the files that leads to make user feel helpless. Thus, user gets forced to pay the ransom to attackers for releasing the encrypted information.

Check you progress 3

1. Exploiting browser vulnerability is the easiest way for attackers to attack on system. Since it requires least effort from attacker. In this, attacker only creates links consisting malicious code, and once user click on that link it automatically get installed on user's system.
2. OS vulnerabilities in Linux and MacOS are as follows:

- In Linux: Dual boot with Windows

Physical theft is always an option with Linux

Open source Operating system where every line of code is public.

- In MacOS: security holes in software

App Store vulnerability

Inflexibility of hardware and software upgradation.

Check you progress 4

1. Computer forensic experts find out the hidden information behind the image using hash. Since, hash information gets changed when any alteration being performed on an original image. The name of technique to hide such information is stenography.
2. We cannot perform any comparison between firewall and an IDS in terms of power since the functionalities are different for both. Firewall provides security by blocking the incoming traffic whereas an IDS secures the system/network from internally by detecting the intrusion activity.

4.15 FURTHER READINGS

1. Kizza JM. Computer network security. Springer Science & Business Media; 2005 Apr 7.
2. Kizza JM, Kizza, Wheeler. Guide to computer network security. Heidelberg, Germany: Springer; 2013 Jan 3.
3. Pawar MV, Anuradha J. Network security and types of attacks in network. Procedia Computer Science. 2015 Jan 1;48:503-6.
4. Zhu B, Joseph A, Sastry S. A taxonomy of cyber attacks on SCADA systems. In2011 International conference on internet of things and 4th international conference on cyber, physical and social computing 2011 Oct 19 (pp. 380-388). IEEE.
5. Cashell B, Jackson WD, Jickling M, Webel B. The economic impact of cyber-attacks. Congressional research service documents, CRS RL32331 (Washington DC). 2004 Apr 1;2.
6. Kumar V, Srivastava J, Lazarevic A, editors. Managing cyber threats: issues, approaches, and challenges. Springer Science & Business Media; 2006 Mar 30.
7. Buczak AL, Guven E. A survey of data mining and machine learning methods for cyber security intrusion detection. IEEE Communications surveys & tutorials. 2015 Oct 26;18(2):1153-76.

8. Mahmoud MS, Hamdan MM, Baroudi UA. Modeling and control of cyber-physical systems subject to cyber attacks: A survey of recent advances and challenges. Neurocomputing. 2019 Apr 21;338:101-15.
9. Albahar M. Cyber attacks and terrorism: A twenty-first century conundrum. Science and engineering ethics. 2019 Aug;25(4):993-1006.
10. Beavers J, Pournouri S. Recent cyber attacks and vulnerabilities in medical devices and healthcare institutions. InBlockchain and Clinical Trial 2019 (pp. 249-267). Springer, Cham.
11. Rot A, Olszewski B. Advanced Persistent Threats Attacks in Cyberspace. Threats, Vulnerabilities, Methods of Protection. InFedCSIS (Position Papers) 2017 Sep (pp. 113-117).
12. Gupta BB, editor. Computer and cyber security: principles, algorithm, applications, and perspectives. CRC Press; 2018 Nov 19.