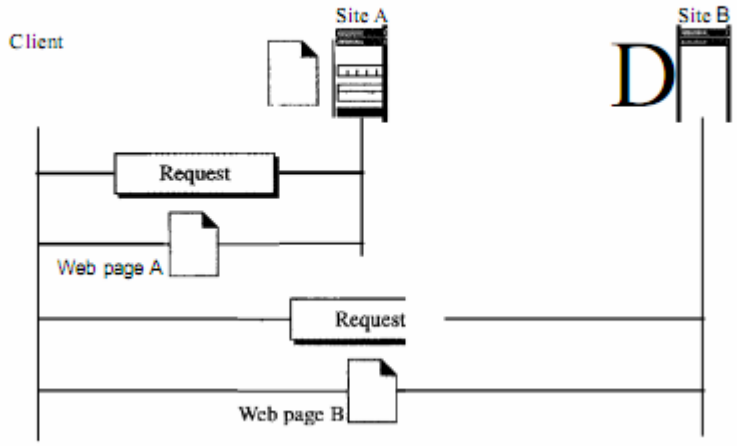
World Wide Web (WWW)

The World Wide Web (WWW) is a repository of information linked together from points all over the world. TheWWW has a unique combination of flexibility, portability, and user-friendly features that distinguish it from other services provided by the Internet. The WWW project was initiated by CERN (European Laboratory for Particle Physics) to create a system to handle distributed resources necessary for scientific research.

ARCHITECTURE

TheWWW today is a distributed clientJserver service, in which a client using a browser can access a service using a server. However, the service provided is distributed over many locations called sites, as shown in Figure 27.1.

Figure 27.1 *Architecture ofWWW*



Each site holds one or more documents, referred to as Web pages. EachWeb page can contain a link to other pages in the same site or at other sites. The pages can be retrieved and viewed by using browsers. Let us go through the scenario shown in Figure 27.1. The client needs to see some information that it knows belongs to site A. It sends a request through its browser, a program that is designed to fetch Web documents. The request, among other information, includes the address of the site and the Web page, called the URL, which we will discuss shortly. The server at site A finds the document and sends it to the client. When the user views the document, she finds some references to other documents, including a Web page at site B. The reference has the URL for the new site. The user is also interested in seeing this document. The client sends another request to the new site, and the new page is retrieved.

HTTP

The Hypertext Transfer Protocol (HTTP) is a protocol used mainly to access data on the WorldWideWeb. HTTP functions as a combination of FTP and SMTP. It is similar to FfP because it transfers files and uses the services of TCP. However, it is much simpler than FfP because it uses only one TCP connection. There is no separate control connection; only data are transferred between the client and the server. HTTP is like SMTP because the data transferred between the client and the server look like SMTP messages. In addition, the format of the messages is controlled by MIME-like headers. Unlike SMTP, the HTTP messages are not destined to be read by humans; they are read and interpreted by the HTTP server and HTTP client (browser). SMTP messages are stored and forwarded, but HTTP messages are delivered immediately. The commands from the client to the server are embedded in a request message. The contents of the requested file or other information are embedded in a response message.

HTTP uses the services of TCP on well-known port 80.

HTTP Transaction

Figure 27.12 illustrates the HTTP transaction between the client and server. Although HTTP uses the services ofTCP, HTTP itself is a stateless protocol. The client initializes the transaction by sending a request message. The server replies by sending a response. Messages The formats of the request and response messages are similar; both are shown in Fig- ure 27.13. A request message consists of a request line, a header, and sometimes a body. A response message consists of a status line, a header, and sometimes a body.

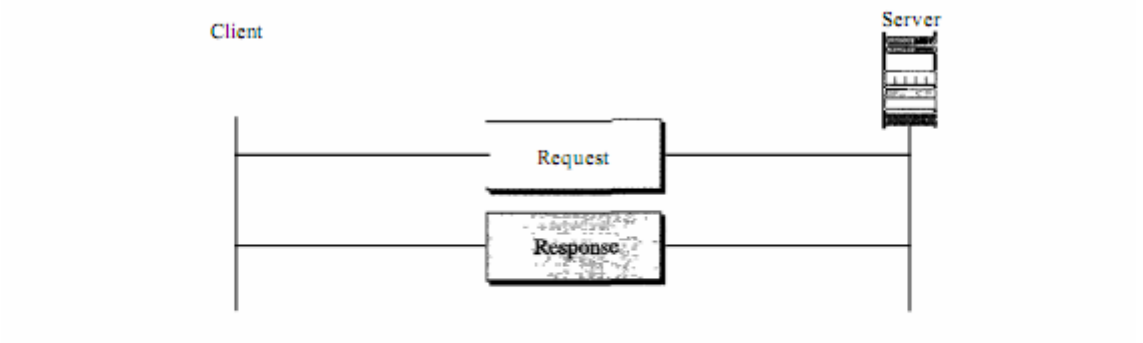
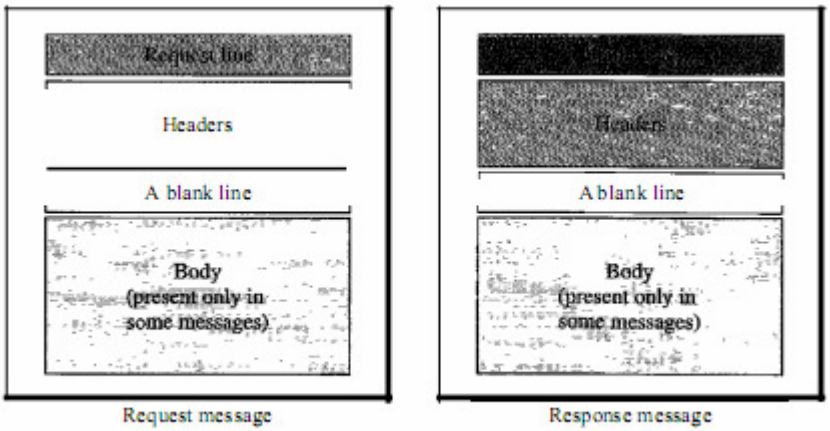
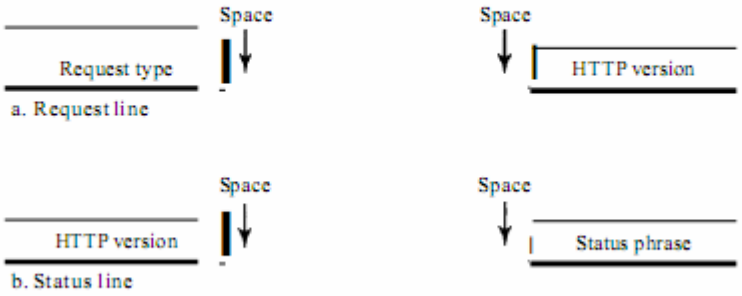


Figure 27.13 *Request and response messages*



Request and Status Lines The first line in a request message is called a request line; the first line in the response message is called the status line. There is one common field, as shown in Figure 27.14.

Figure 27.14 *Request and status lines*



Request type. This field is used in the request message. In version 1.1 of HTTP, several request types are defined. The request type is categorized into methods as defined in Table 27.1.

Table 27.1 *Methods*

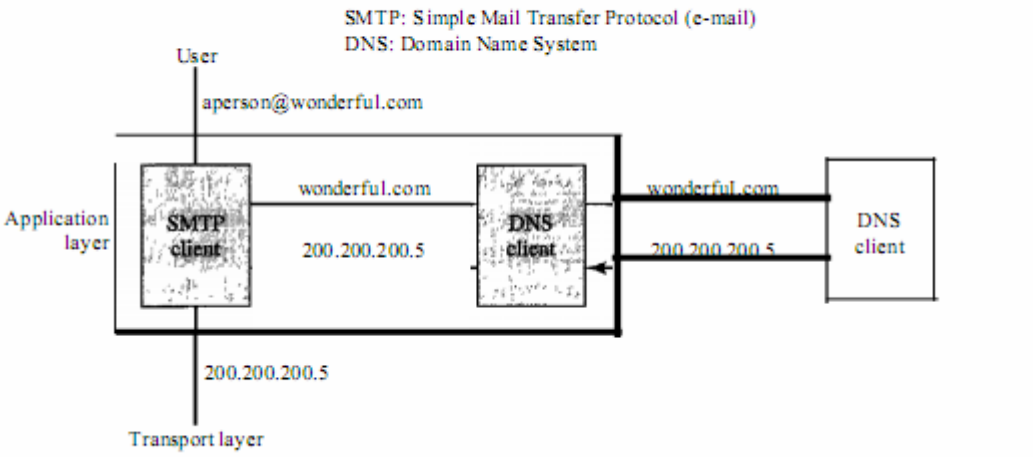
Method	*Action*
GET	Requests a document from the server
HEAD	Requests information about a document but not the document itself
POST	Sends some information from the client to the server
PUT	Sends a document from the server to the client
TRACE	Echoes the incoming request
CONNECT	Reserved
OPTION	Inquires about available options

DNS

The Domain Name System (DNS) is a supporting program that is used by other programs such as e-mail.

Figure 25.1 shows an example of how a DNS client/server program can support an e-mail program to find the IP address of an e-mail recipient. A user of an e-mail program may know the e-mail address of the recipient; however, the IP protocol needs the IP address. The DNS client program sends a request to a DNS server to map the e-mail address to the corresponding IP address.

Figure 25.1 *Example of using the DNS service*



To identify an entity, TCPIIP protocols use the IP address, which uniquely identifies the connection of a host to the Internet. However, people prefer to use names instead of numeric addresses. Therefore, we need a system that can map a name to an address or an address to a name.

When the Internet was small, mapping was done by using a host file. The host file had only two columns: name and address. Every host could store the host file on its disk and update it periodically from a master host file. When a program or a user wanted to map a name to an address, the host consulted the host file and found the mapping.

Today, however, it is impossible to have one single host file to relate every address with a name and vice versa. The host file would be too large to store in every host. In addition, it would be impossible to update all the host files every time there was a change.

One solution would be to store the entire host file in a single computer and allow access to this centralized information to every computer that needs mapping. But we know that this would create a huge amount of traffic on the Internet.

Another solution, the one used today, is to divide this huge amount of information into smaller parts and store each part on a different computer. In this method, the host that needs mapping can contact the closest computer holding the needed information.

This method is used by the Domain Name System (DNS). In this chapter, we first discuss the concepts and ideas behind the DNS. We then describe the DNS protocol itself.

SNMP

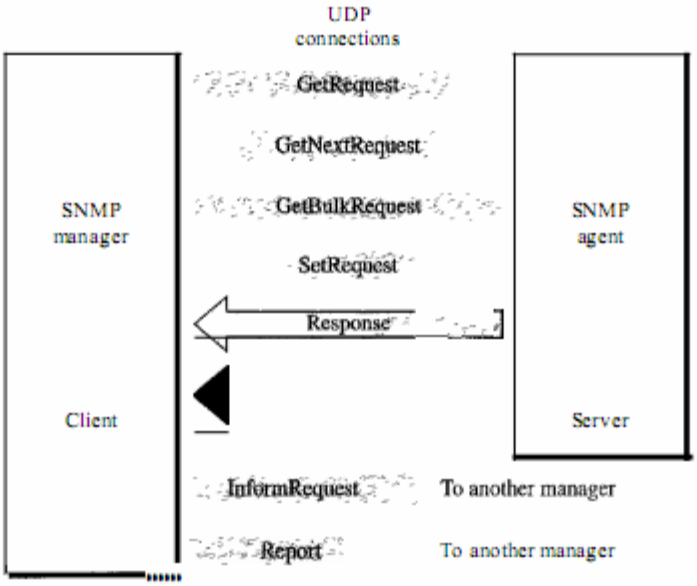
SNMP uses both SMI and MIB in Internet network management. It is an application program that allows

1. A manager to retrieve the value of an object defined in an agent
2. A manager to store a value in an object defined in an agent
3. An agent to send an alarm message about an abnormal situation to the manager

PDUs

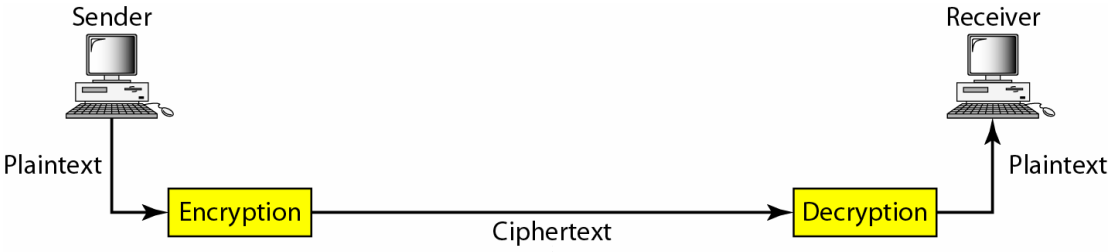
SNMPv3 defines eight types of packets (or PDUs): GetRequest, GetNextRequest, GetBulkRequest, SetRequest, Response, Trap, InformRequest, and Report (see Figure 28.20).

Figure 28.20 *SNMP PDUs*



5.1 Introduction to Basic Encryption and Decryption:

The term 'Cryptography' means the concept of encryption and decryption together. Cryptography is the technique in which the original 'plain text' message is 'encrypted' i.e. converted into a coded form called 'cipher text' at the sender's end, which is then transmitted to the receiver. The receiver then 'decrypts' i.e. converts the 'cipher text' back into the 'plain text' to get the original message back.



Cryptography is also called as an art or technique to achieve secure communication between the communicating parties by encoding the messages between them such that no third party can gain anything useful out of interception.

Various techniques are utilized for this purpose of cryptography. Broadly these techniques fall into two categories.

1) Symmetric key cryptography: - in which the 'key' element used, is the 'same' for both encryption as well as decryption and
2) Asymmetric key cryptography - in which the 'key' element used, is different for both encryption as well as decryption.

    - Symmetric key cryptography is also known as 'private or secret key cryptography' (*please refer to section 5.5 of these notes for details*) whereas

    - Asymmetric key cryptography is also known as 'public key cryptography', (*please refer to section 5.6 of these notes for details*)

The techniques used in symmetric key cryptography are as below.

Substitution technique - the very basic technique, which makes use of simple letter substitution to generate cipher text.

Specific methods used in this type include

1. Caesar cipher (used by Julius Caesar),
2. Modified Caesar Cipher,
3. Mono-alphabetic cipher,
4. Homophonic substitution cipher,
5. Polygram substitution cipher
6. Polyalphabetic cipher etc.

Now let us study them (Substitution Technique) one by one:

1. Caesar Cipher

A cryptographic scheme proposed by Julius Caesar is one special case of substitutional cipher where each alphabet is the message is replaced by an alphabet, three places down the line, in the alphabetical order.

Thus “A” becomes “D” and “B” becomes “E”

Plain text	A	B	C	D	E	F	G	H	I	J	K	L	M	N
Cipher Text	D	E	F	G	H	I	J	K	L	M	N	O	P	Q

Plain text	O	P	Q	R	S	T	U	V	W	X	Y	Z
Cipher Text	R	S	T	U	V	W	X	Y	Z	A	B	C

Caesar Cipher is very simple. But this simplicity comes with a cost. Obviously it is a very weak scheme.

Algorithm to break Caesar cipher

1. Read each alphabet in the cipher text message, and search for it in the second row of the figure above
2. When a match is found, replace that alphabet in the cipher text message with the corresponding alphabet in the same column but the first row of the table (e.g. if the alphabet in cipher text is J, replace it with G).
3. Repeat the process for all alphabets in the cipher text message.

The process shown above will reveal the original plain text. Thus, given a cipher text message L ORYH BRX, it is easy to work backwards and obtain the plain text I LOVE YOU as shown below.

Cipher text		L	O	R	Y	H	B	R	X
Plain text		I	L	O	V	E	Y	O	U

Caesar Cipher is good in theory, but not so good in practice.

Let Ke be the encryption key and Kd be the decryption key. Here we have assumed that the value of Ke = 3 and thus Kd would also be 3,

Let us now try and complicate the Caesar Cipher to make an attacker's life difficult.

2. Modified Version of Caesar Cipher

How can we generalize Caesar Cipher a bit more? Let us assume that the cipher text alphabets corresponding to the original plain text alphabets may not necessarily be three places down the order, but instead, can be any places down the order. This can complicate matters a bit.

Thus, we are now saying that an alphabet A in plain text would not necessarily be replaced by D. It can be replaced by any valid alphabet, i.e. by E or by F or by G, and so on. Once the replacement scheme is decided, it would be constant and will be used for all other alphabets in that message. As we know, the English language contains 26 alphabets. Thus, an alphabet A can be replaced by any other alphaber in the English alphabet set, (i.e. B through Z).

Of course, it does not make sense to replace an alphabet by itself (i.e. replacing A with A). Thus, for each alphabet, we have 25 possibilities of replacement. Hence, to break a message in the modified version of Caesar Cipher, our earlier algorithm would not work.

Let us write a new algorithm to break this version of Caesar Cipher, as shown:

1. Let k be a number equal to 1.
2. Read the complete cipher text message.
3. Replace each alphabet in the cipher text message with an alphabet that is k positions down the order.
4. Increment k by 1.
5. If k is less than 26, then go to step 2. Otherwise, stop the process.
   6. The original text message corresponding to the cipher text message is one of the 25 possibilities produced by the above steps.

We write down all the 25 possibilities and try to make sense. Whichever makes some sense we keep and the other 24 are rejected. Trying out all possibilities is called Brute-Force Attack.

3. Mono-alphabetic Cipher

The major weakness of the Caesar Cipher is its predictability. Once we

decide to replace an alphabet in a plain text message with an alphabet that is k positions up or down the order, we replace all other alphabets in the plain text message with the same technique. Thus, the cryptanalyst has to tryout a maximum of 25 possible attacks, and she is assured of a success.

Now imagine that rather than using a uniform scheme for all the alphabets in a given plain text message, we decide to use random substitution. This means that in a given plain text message, each A can be replaced by any other alphabet (B through Z), each B can also be replaced by any other random alphabet (A or C through Z), and so on. The crucial difference being, there is no relation between the replacement of B and replacement of A. That is, if we have decided to replace each A with D, we need not necessarily replace each B with E-we can replace each B with any other character I

To put it mathematically, we can now have any permutation or combination of the 26 alphabets, which means (26 x 25 x 24 x 23 x ... 2) or 4 x 1026 possibilities I This is extremely hard to crack. It might actually take years to tryout these many combinations even with the most modern computers.

4. Homophonic Substitution Cipher:

The Homophonic Substitution Cipher is very similar to Mono Alphabetic Cipher. In a plain substitution cipher technique, we replace one alphabet with another, but in this scheme, the difference is that instead of having a fixed substitution, We can, choose the alphabet from a set. So in this technique, A can be replaced by D,H,P,R; B can be replaced by E,I,Q,S etc.

Homophonic Substitution Cipher also involved substitution of one plain text character with a Cipher Text character at a time. However the cipher text character can be any one of the chosen set.

5. Polygram Substitution Cipher.

In Polygram Substitution Cipher technique, rather than replacing one plain text alphabet with one cipher text alphabet at a time, a block of alphabets is replaced with another block. For instance, HELLO could be replaced with YUQQW, but HELL could be replaced by a totally different cipher text block TEUL

6. Poly-alphabetic Substitution Cipher.

This cipher uses multiple one character keys. Each of the keys encrypts

one plain text character. The first key encrypts the first plain text character; the second key encrypts the second plain text character, and so on. After al the keys are used, they are recycled. Thus if we have 30 one letter keys, every 30th character in the plain text would be replaced with the same key. This number is called as the period of the cipher.

In some cases, the mono alphabetic cipher technique is used round after round over already converted plain text and its cipher text. The more number of rounds, the more complex the cipher becomes.

Transposition technique - Modified version of substitution technique because this not only substitutes letters but also makes some sort of permutation over the plain text in order to generate cipher text. Specific examples include

1. Rail fence technique
2. Simple columnar transposition
3. Simple columnar transposition with multiple rounds
4. Vemam cipher,
5. Book cipher etc.

Now let us study them (Transposition Technique) one by one:

1. Rail Fence Technique:

It uses a simple algorithm as:

1. Write down the plain text message as a sequence of diagonals.
2. Read the plain text written in step 1 as a sequence of rows.

Example: Original Plain text message: " Come home tomorrow"

1. After we arrange the plain text diagonally, it would like as follows:

C		M		H		M		T		M		R		O	
	O		E		O		E		O		O		R		W

2. Now read the text row by row, write it sequentially. Thus we have:

C-M-H-M-T-M-R-O-O-E-O-E-O-O-R-W

2. Simple Columnar Transposition Technique:

Basic Technique:

The idea is to:

a. Write the plain text message row by row in a rectangle of a pre-defined size.
b. Read the message column-by column, however, it need not be in the order of columns 1,2,3 etc. It can be any random order such as 2,1,3 etc.
c. The message thus obtained is the cipher text message.

Original Plain text Message: Secrets have to be kept.

1. Let us consider a rectangle with S columns. Therefore, when we write the message into the rectangle row by row it would look as follows:

Column 1	Column 2	Column 3	Column 4	Column 5	Column 6
C	O	M	E	H	O
M	E	T	O	M	O
R	R	O	W		

2. Now read the text in the order of the columns. 4,6,1,2,5,3
3. The cipher text thus obtained is:

E-O-W-O-O-C-M-R-O-E-R-H-M-M-T-O

3. Simple columnar transposition technique with multiple rounds:

Here, the basic Simple columnar technique is repeated for multiple rounds. The more number of rounds, the more complex the cipher becomes. Hence, it is more difficult to crack.

The Basic algorithm:

1. Write the plain text message row-by-row in a rectangle of a pre-determined size
2. Read the message column by column in a random sequence
3. The message thus obtained as the cipher text message of round 1
4. Use this output as a plain text for the next step

5. Vemam Cipher (One-Time Pad)

The Vemam Cipher, also called as One-Time Pad, is implemented using a random set of non-repeating characters as the input cipher text. The most significant point her is that once an input cipher text for transposition is used; it is never used again for any other message (hence the name one-time). The length of the cipher text is equal to the length of the original plain text.

Since, it is used as one-time pad and is discarded after a single use, this technique is highly secure and suitable for small plain text message, but is impractical for large messages.

6. Book Cipher / Running Block Key Cipher:

The idea used is quite simple and similar in principle to Vernam Cipher. For producing cipher text, some portion of text from a book is used, which

serves the purpose of a one-time pad. This, the characters from a book are used as one time pad, and they are added to the input plain text messages.

Every process of encryption and decryption is necessarily associated with a 'key'- the combination used for encryption and/or decryption, and an algorithm i.e. the rules or steps used for both encryption and decryption. The requirement of 'same' key as in case of 'symmetric' key cryptography leads to a common problem called 'problem of key distribution', i.e. how the two parties should agree upon a 'common' key that has to be used for the process. This is as described below.

Problem of Key distribution in Symmetric Key cryptography:

As in case of symmetric key cryptography, the key that has to be used for both encryption and decryption should be the 'same' this leads to a problem that how the two parties requiring secure communication can 'agree' or 'decide' upon a common key, without letting any third person know about it? There can be many ways in which the two parties will try to communicate assuming it is secure, but it may not be so. e.g. even if they exchange letters, seal envelopes into locked boxes, talk over open media for the common key, or send the key along with the locked boxes, whatever may be the means used, it turns out to be practically non-viable or difficult to implement.

That is to say, there are very much chances of intercepting the communication between two parties if any of these methods are used. This is called the 'problem of key distribution'.

In order to come out of this problem, one good solution was given by two scientists jointly known as 'Diffie-Hellman key exchange algorithm'.

5.3 The concept of Public key and Private key:

The Asymmetric key cryptography is also known as a 'public key cryptography', which uses a key-pair rather than a single key. The importance of this scheme is that only one key-pair is required to securely communicate between any number of other parties. (unlike the huge no. of keys that we've seen with earlier method.) Hence, one problem is overcome right away. One of these two keys is called public key (which can be announced to the world) and another is private key (obviously to be kept with oneself). This is to be followed by everyone who wants to communicate securely.

The working of public and private kevs:

Asymmetric key cryptography (using public and private keys) works as under:
Suppose, X wants to send a message to Y without having to worry about its security.

1. Then X and Y should each have a private key and a public key.
   - **X should keep its private key secret.**
   - **Y should keep its private key secret**.
   - X should inform Y about its public key.
   - Y should inform X about its public key

   (Both now have their own set of keys ready.)

2. When X wants to send message to Y, X encrypts with Y's public key (as it is known to everyone)
3. X then sends this message to Y.
4. Then, Y decrypts this message using his own private key (known only to Y)

   [This ensures in this case, that the message can be encrypted & sent by anyone, but can only be decrypted by Y. Hence, any interception will not result in knowing the sensitive information as key is only with Y.]
   Similarly, on the other side, if Y wants to send the message to X, reverse method is performed.

5. Y encrypts the message using X's public key and sends this to X
6. On receiving the message, X can further decrypt it using his own private key.

The basis of this working lies in the assumption of large prime number with only two factors. If one of the factors is used for encryption process, only the other factor shall be used for decryption. The **best example** of an asymmetric key cryptography algorithm is the famous **RSA algorithm** (developed by **Rivest, Shamir and Adleman** at MIT in 1978, based on the framework setup by Diffie & Hellman earlier).

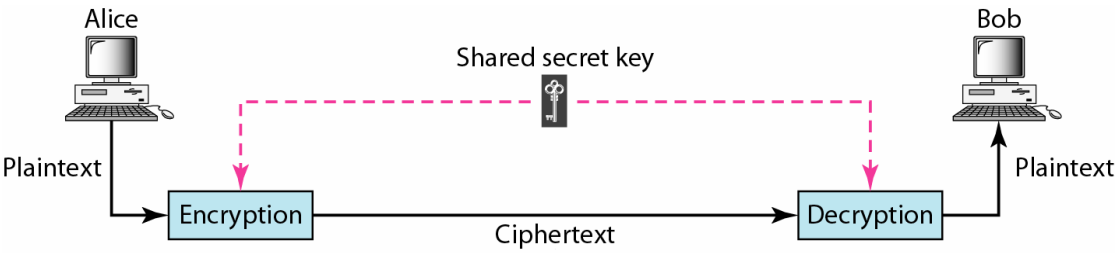
e public????

aw all that you have but also
keep paying throughout your

5. 5 SYMMETRIC- KEY CRYPTOGRAPHY

We can divide all the cryptography algorithms in the world into two groups: symmetric-key (sometimes called secret-key) cryptography algorithms and public-key (sometimes called asymmetric) cryptography algorithms.

In symmetric-key cryptography, the same key is used by both parties. The sender uses this key and an encryption algorithm to encrypt data; the receiver uses the same key and the corresponding decryption algorithm to decrypt the data



In symmetric-key cryptography, the same key is used by the sender (for encryption) and the receiver (for decryption). The key is shared.

In symmetric-key cryptography, the algorithm used for decryption is the inverse of the algorithm used for encryption. This means that if the encryption algorithm uses a combination of addition and multiplication, the decryption algorithm uses a combination of division and subtraction.

Note that the symmetric-key cryptography algorithms are so named because the same key can be used in both directions.

In symmetric-key cryptography, the same key is used in both directions.

Symmetric-key algorithms are efficient; it takes less time to encrypt a message using a symmetric-key algorithm than it takes to encrypt using a public-key algorithm. The reason is that the key is usually smaller. For this reason, symmetric-key algorithms are used to encrypt and decrypt long messages.

Symmetric-key cryptography is often used for long messages.

Disadvantages of symmetric key:

A symmetric-key algorithm has two major disadvantages.

1. Each pair of users must have a unique symmetric key.
This means that if N people in the world want to use this method, there needs to be $N(N - 1)/2$ symmetric keys.

For example, for 1 thousand people to communicate, 1000 * 999 /2 = 4,99,500 (4 lakhs 99 thousand and five hundred symmetric keys are needed. The distribution of the keys between two parties can be difficult.

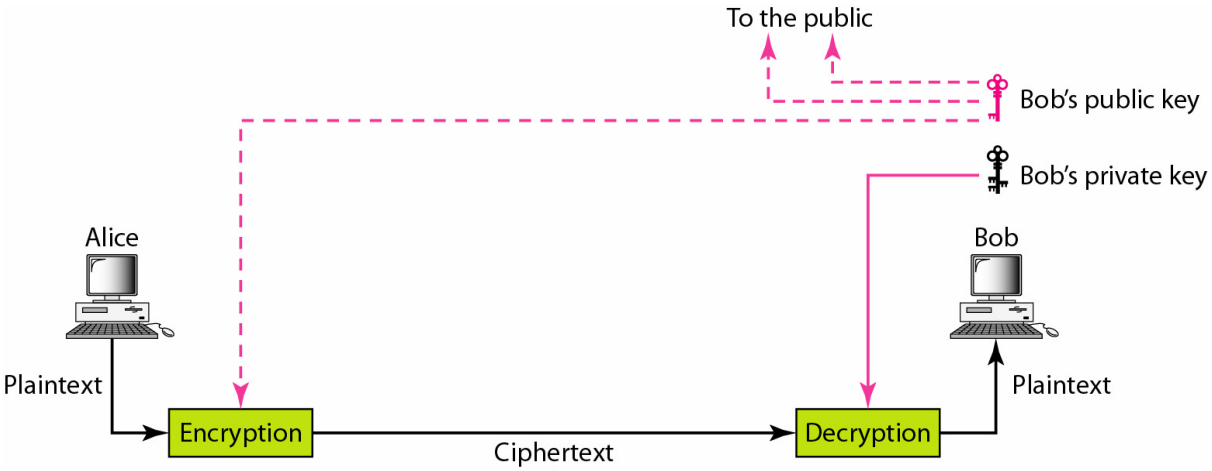
2. The sender needs to exchange the key to the receiver. It may be hijacked in between!

---

5. 6 Asymmetric Key Cryptography:

In public-key cryptography, there are two keys: a private key and a public key. The private key is kept by the receiver. The public key is announced to the public.

Imagine Alice, as shown in Figure 29.20, wants to send a message to Bob. Alice uses the public key to encrypt the message. When the message is received by Bob, the private key is used to decrypt the message.



In public-key encryption/decryption, the public key that is used for encryption is different from the private key that is used for decryption.

The public key is available to the public; the private key is available only to an individual.

Public-key encryption/decryption has two advantages.

First, it removes the restriction of a shared symmetric key between two entities (e.g., persons) who need to communicate with each other. A shared symmetric key is shared by the two parties and cannot be used when one of them wants to communicate with a third party. In public-key encryption! decryption, each entity creates a pair of keys; the private one is kept, and the public one is distributed. Each entity is independent, and the pair of keys created can be used to communicate with any other entity.

The second advantage is that the number of keys needed is reduced tremendously.

In this system, for I thousand users to communicate, only 1 thousand pairs of keys ie 2000 keys are needed, not 4,99,500, as was the case in symmetric-key cryptography.

Public-key cryptography also has two disadvantages.

The big disadvantage is the complexity of the algorithm. If we want the

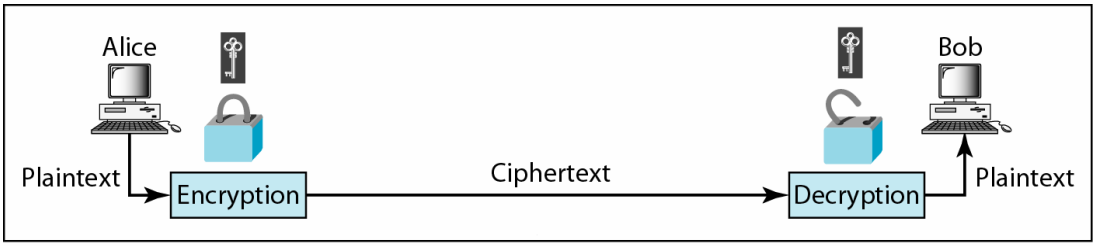
method to be effective, the algorithm needs large numbers. Calculating the ciphertext from plaintext using the long keys takes a lot of time. That is the main reason that public-key cryptography is not recommended for large amounts of text.

Public-key algorithms are more efficient for short messages.

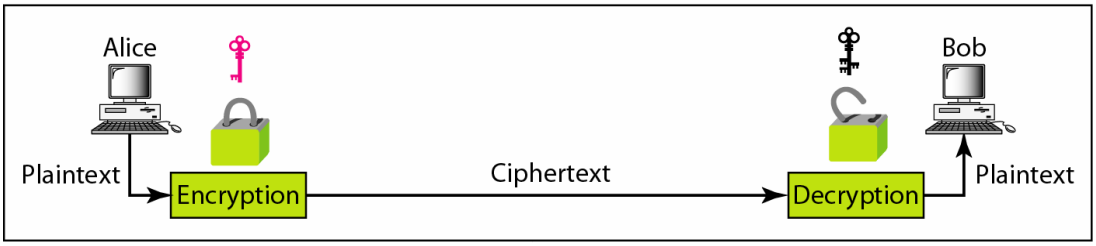
The second disadvantage of the public-key method is that the association between an entity and its public key must be verified. If Alice sends her public key via an email to Bob, then Bob must be sure that the public key really belongs to Alice and nobody else.

One point needs to re-mentioned that if your private key were made public you would Get Bankrupted in no time!

5.7 Compare and contrast between Symmetric Key Cryptography and Asymmetric Key Cryptography:



a. Symmetric-key cryptography



b. Asymmetric-key cryptography

S. No.	Characteristic	Symmetric Key Cryptography	Asymmetric Key Cryptography
1	Key used for encryption/decryption	Same key is used for encryption and decryption	One key used for encryption and another, different key is used for decryption
2		Ke = Kd	Kd ≠ Kd
3	Speed of encryption/decryption	Very fast	Slower
4	Size of resulting encrypted text	Usually same as or less than the original clear text size	More than the original clear text size
5	Key agreement / exchange	A big problem	No problem at all
6	Number of keys required as compared to the number of participants in the message exchange	Equals about the square of the number of participants, so scalability is an issue	Same as the number of participants, so scales up quite well

7	Usage	Mainly for encryption and decryption (confidentiality), cannot be used for digital signatures (integrity and non-repudiation checks)	Can be used for encryption and decryption (confidentiality) as well as for digital signatures (integrity and non-repudiation checks)
8	Efficiency in usage	Symmetric key cryptography is often used for long messages	Public key algorithm are more efficient for short messages

The above table shows that both symmetric key cryptography and asymmetric key cryptography have nice features.

Also, both have some areas where better alternatives are generally desired. Asymmetric key cryptography solves the major problem of key agreement / key exchange as well as scalability.

However, it is far slower and produces huge chunks of cipher text as compared to symmetric key Cryptography (essentially because it uses large keys and complex algorithms as compared to symmetric key cryptography).

How nice it would be, if we can combine the two cryptography mechanisms, so as to achieve the better of the two, and yet do not compromise on any of the features? More specifically, we need to ensure that the following objectives are met.

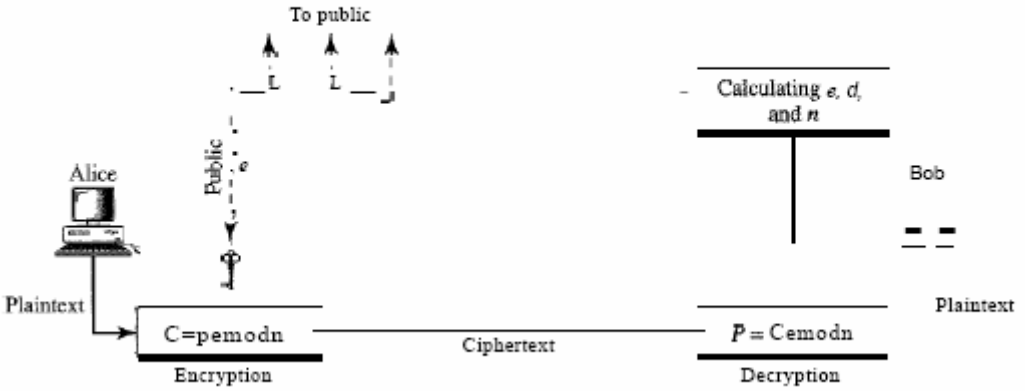
1. The solution should be completely secure.
2. The encryption and decryption processes must not take a long time.
3. The generated cipher text should be compact in size.
4. The solution should scale to a large number of users easily, without introducing any additional complications.
5. The key distribution problem must be solved by the solution.

In practice symmetric key cryptography and asymmetric key cryptography are combined to have a very efficient security solutions.

RSA

The most common public key algorithm is RSA, named for its inventors Rivest, Shamir, and Adleman (RSA). lt uses two numbers, e and d, as the public and private keys,

Figure 30.24 *RSA*



The two keys, *e* and *d,* have a special relationship to each other, a discussion of this relationship is beyond the scope of this book. We just show how to calculate the keys without proof.

Selecting Keys

Bob use the following steps to select the private and public keys:

1. Bob chooses two very large prime numbers *p* and *q.* Remember that a prime number is one that can be divided evenly only by 1 and itself.
2. Bob multiplies the above two primes to find *n,* the modulus for encryption and decryption. In other words, $n = p \times q$.
3. Bob calculates another number $\phi = (p - 1) \times (q - 1)$.
4. Bob chooses a random integer e. He then calculates *d* so that $d \times e = 1 \bmod \phi$.
5. Bob announces *e* and *n* to the public; he keeps ϕ and *d* secret.

In RSA, *e* and *n* are announced to the public; *d* and ϕ are kept secret.

Encryption

Anyone who needs to send a message to Bob can use nand *e.* For example, if Alice needs to send a message to Bob, she can change the message, usually a short one, to an integer. This is the plaintext. She then calculates the ciphertext, using *e* and *n.*

$$C = P^e \pmod{n}$$

Alice sends C, the ciphertext, to Bob.

Decryption

Bob keeps ϕ and d private. When he receives the ciphertext, he uses his private key d to decrypt the message:

$$P = C^d \pmod{n}$$

Restriction

For RSA to work, the value of P must be less than the value of n. If P is a large number, the plaintext needs to be divided into blocks to make P less than n.

Example 30.7

Bob chooses 7 and 11 as p and q and calculates $n = 7 \cdot 11 = 77$. The value of $\phi = (7 - 1)(11 - 1)$ or 60. Now he chooses two keys, e and d. If he chooses e to be 13, then d is 37. Now imagine Alice sends the plaintext 5 to Bob. She uses the public key 13 to encrypt 5.

Plaintext: 5
$C = 5^{13} = 26 \bmod 77$
Ciphertext: 26

Bob receives the ciphertext 26 and uses the private key 37 to decipher the ciphertext:

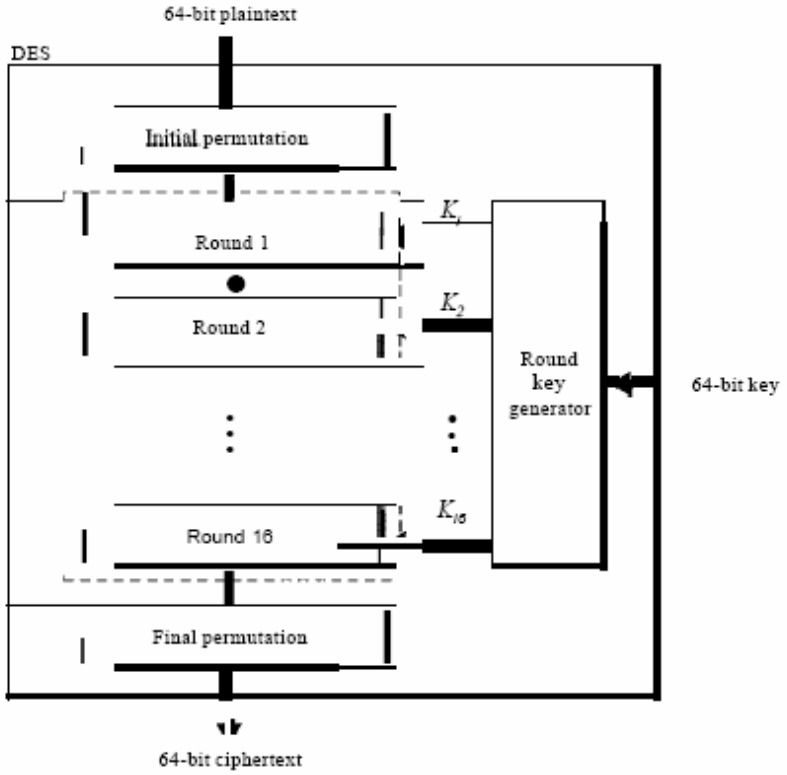
Ciphertext: 26
$P = 26^{37} = 5 \bmod 77$
Plaintext: 5 Intended message sent by Alice

The plaintext 5 sent by Alice is received as plaintext 5 by Bob.

DES

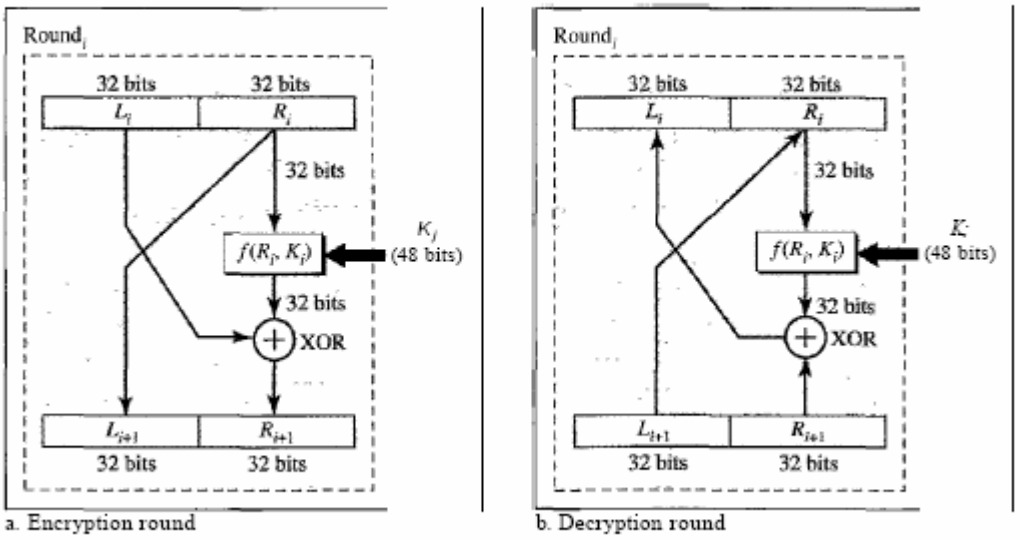
Data Encryption Standard (DES)
One example of a complex block cipher is the Data Encryption Standard (DES). DES was designed by IBM and adopted by the U.S. government as the standard encryption method for nonmilitary and nonclassified use. The algorithm encrypts a 64-bit plaintext block using a 64-bit key, as shown in Figure 30.13.

Figure 30.13 *DES*



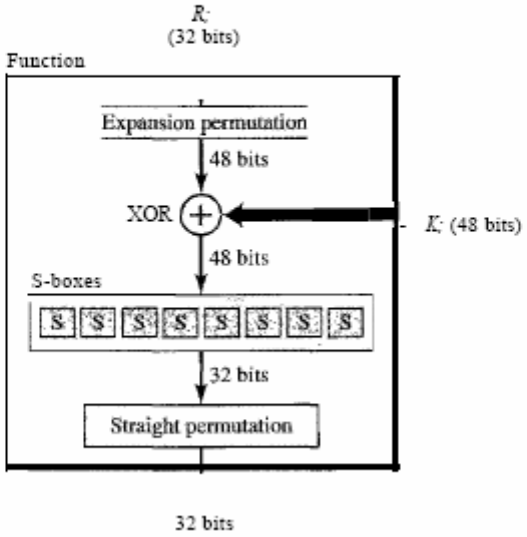
DES has two transposition blocks (P-boxes) and 16 complex round ciphers (they are repeated). Although the 16 iteration round ciphers are conceptually the same, each uses a different key derived from the original key.
The initial and final permutations are keyless straight permutations that are the inverse of each other. The permutation takes a 64-bit input and permutes them according to predefined values.
Each round of DES is a complex round cipher, as shown in Figure 30.14. Note that the structure of the encryption round ciphers is different from that of the decryption one.

Figure 30.14 *One round in DES ciphers*



DES Function The heart of DES is the **DES function.** The DES function applies a 48-bit key to the rightmost 32 bits *Ri* to produce a 32-bit output. This function is made up of four operations: an XOR, an expansion permutation, a group of S-boxes, and a straight permutation, as shown in Figure 30.15.

Figure 30.15 *DESfunction*



Triple DES

Critics of DES contend that the key is too short. To lengthen the key, Triple DES or 3DES has been proposed and implemented. This uses three DES blocks, as shown in Figure 30.16. Note that the encrypting block uses an encryption-decryption-encryption combination of DESs, while the decryption block uses a decryption-encryption-decryption combination.

Two different versions of 3DES are in use: 3DES with two keys and 3DES with three keys. To make the key size 112 bits and at the same time protect DES from attacks such as the man-in-the-middle attack, 3DES with two keys was designed. In this version, the first and the third keys are the same (KeYl = KeY3)' This has the advantage in that a text encrypted by a single DES block can be decrypted by the new 3DES. We just set all keys equal to KeYl' Many algorithms use a 3DES cipher with three keys. This increases the size of the key to 168 bits.

Figure 30.16 *Triple DES*