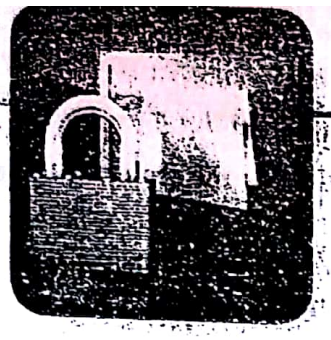


Unit :- 3

CRYPTOGRAPHY TECHNIQUES

2.1 INTRODUCTION

This chapter introduces the basic concepts in **cryptography**. Although this word sounds fearful, we shall realize that it is very simple to understand. In fact, most terms in computer security have a very straightforward meaning. Many terms, for no reason, sound complicated. Our aim will be to demystify all such terms in relation to cryptography in this chapter. After we are through with this chapter, we shall be equipped to understand computer-based security solutions and issues that follow in later chapters.

Cryptography is the art of achieving security by encoding messages to make them non-readable.

Figure 2.1 shows the conceptual view of cryptography.

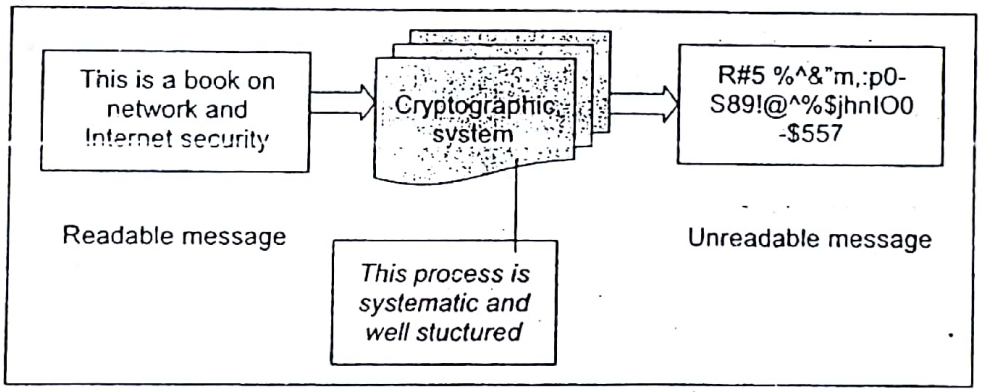


Fig. 2.1 Cryptographic system

Some more terms need to be introduced in this context.

***Cryptanalysis** is the technique of decoding messages from a non-readable format back to a readable format without knowing how they were initially converted from readable format to non-readable format.*

In other words, it is like *breaking a code*. This concept is shown in Fig. 2.2.

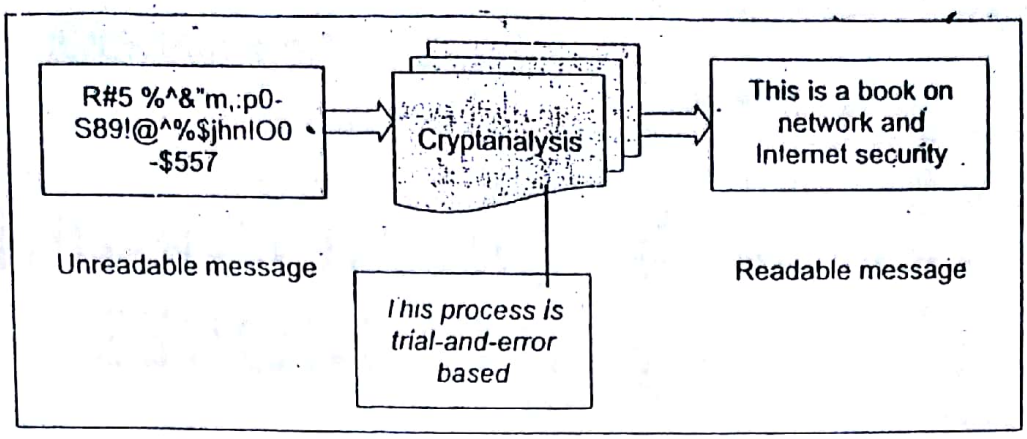


Fig. 2.2 Cryptanalysis

***Cryptology** is a combination of cryptography and cryptanalysis.*

This concept is shown in Fig. 2.3.

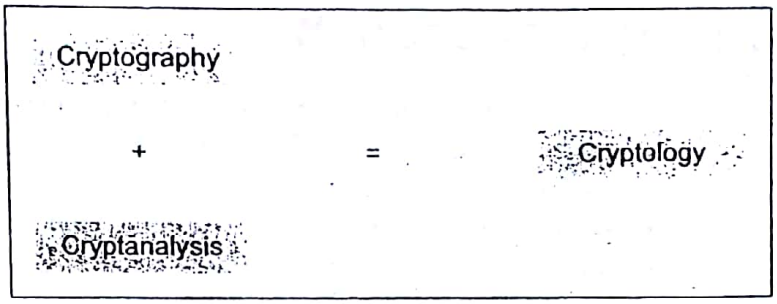


Fig. 2.3 Cryptography + Cryptanalysis = Cryptology

In the early days, cryptography was performed by using manual techniques. The basic framework of performing cryptography has remained more or less the same, of course, with a lot of improvements in the actual implementation. More importantly, computers now perform these cryptographic functions/ algorithms, thus making the process a lot faster and secure. This chapter, however, discusses the basic methods of achieving cryptography without referring to computers.

The basic concepts in cryptography are introduced first. We then proceed to discuss how we can make messages illegible, and thus secure. This can be done in many ways. We discuss all these approaches in this chapter. Modern computer-based cryptography solutions have actually evolved based on these premises. This chapter touches upon all these cryptography algorithms. We also discuss the relative advantages and disadvantages of the various algorithms, as and when applicable.

Some cryptography algorithms are very trivial to understand, replicate, and therefore, crack. Some other cryptography algorithms are highly complicated, and therefore difficult to crack. The rest are somewhere in the middle. A detailed discussion of these is highly essential in cementing our concepts that we shall keep referring to when we actually discuss computer-based cryptography solutions in later chapters.

2.2 PLAIN TEXT AND CIPHER TEXT

Any communication in the language that you and I speak—that is the human language—takes the form of **plain text** or **clear text**. That is, a message in plain text can be understood by anybody knowing the language as long as the message is not codified in any manner. For instance, when we speak with

our family members, friends or colleagues, we use plain text because we do not want to hide anything from them. Suppose I say "Hi Anita", it is plain text because both Anita and I know its meaning and intention. More significantly, anybody in the same room would also get to hear these words, and would know that I am greeting Anita.

Notably, we also use plain text during electronic conversations. For instance, when we send an email to someone, we compose the email message using English,or these days another language. For instance, I can compose the email message as shown in Fig. 2.4.

Hi Amit

Hope you are doing fine. How about meeting at the train station this Friday at 5 p.m.? Please let me know if it is OK with you.

Regards.

Atul

Fig. 2.4 Example of a plain-text message

Now, not only Amit, but also any other person who reads this email would know what I have written. As before, this is simply because I am not using any codified language here. I have composed my email message using plain English. This is another example of plain text, albeit in written form.

Clear text, or plain text, signifies a message that can be understood by the sender, the recipient, and also by anyone else who gets access to that message.

In normal life, we do not bother much about the fact that someone could be overhearing us. In most cases, that makes little difference to us because the person overhearing us can do little damage by using the overheard information. After all, we do not reveal many secrets in our day-to-day lives!

However, there are situations where we are concerned about the secrecy of our conversations. For instance, suppose I wish to know my bank account's balance and hence I call up my phone banker from my office. The phone banker would generally ask a secret question (e.g. What is your mother's maiden name?) whose answer only I know. This is to ascertain that someone else is not posing as me. Now, when I give the answer to the secret question (e.g. Leela), I would generally speak in low voice, or better yet, initially call up from a phone that is isolated. This ensures that only the intended recipient (the phone banker) gets to know the correct answer.

On the same lines, suppose that my email to my friend Amit shown earlier is confidential for some reason. Therefore, I do not want anyone else to understand what I have written, even if he/she is able to access the email by using some means, before it reaches Amit. How do I ensure this? This is exactly the problem that small children face. Many times, they want to communicate in such a manner that their little secrets are hidden from the elderly. What do they do in order to achieve this? Usually, the simplest trick that they use is a code language. For instance, they replace each alphabet in their conversation with another character. As an example, they replace each alphabet with the alphabet that is actually three alphabets down the order. So, each A will be replaced by D, B will be replaced by E, C will be replaced by F, and so on. To complete the cycle, each W will be replaced by Z, each X will be replaced by A, each Y will be replaced by B and each Z will be replaced by C. We can summarize this scheme as shown in Fig. 2.5. The first row shows the original alphabets, and the second row shows what each original alphabet will be replaced with.

ori	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Cipher	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Fig. 2.5 A scheme for codifying messages (replacing each alphabet with an alphabet three places down the line)

Thus, using the scheme of replacing each alphabet with the one that is three places down the line, a message *I love you* shall become *L ORYH BRX* as shown in Fig. 2.6.

I		L	O	V	E		Y	O	U
L		O	R	Y	H		B	R	X

Fig. 2.6 Codification using the alphabet-replacement scheme

Of course, there can be many variants of such a scheme. It is not necessary to replace each alphabet with the one that is three places down the order. It can be the one that is four, five or more places down the order. The point is, however, that each alphabet in the original message can be replaced by another to hide the original contents of the message. The codified message is called **cipher text**. Cipher means a code or a secret message.

When a plain-text message is codified using any suitable scheme, the resulting message is called cipher text.

Based on these concepts, let us put these terms into a diagrammatic representation, as shown in Fig. 2.7.

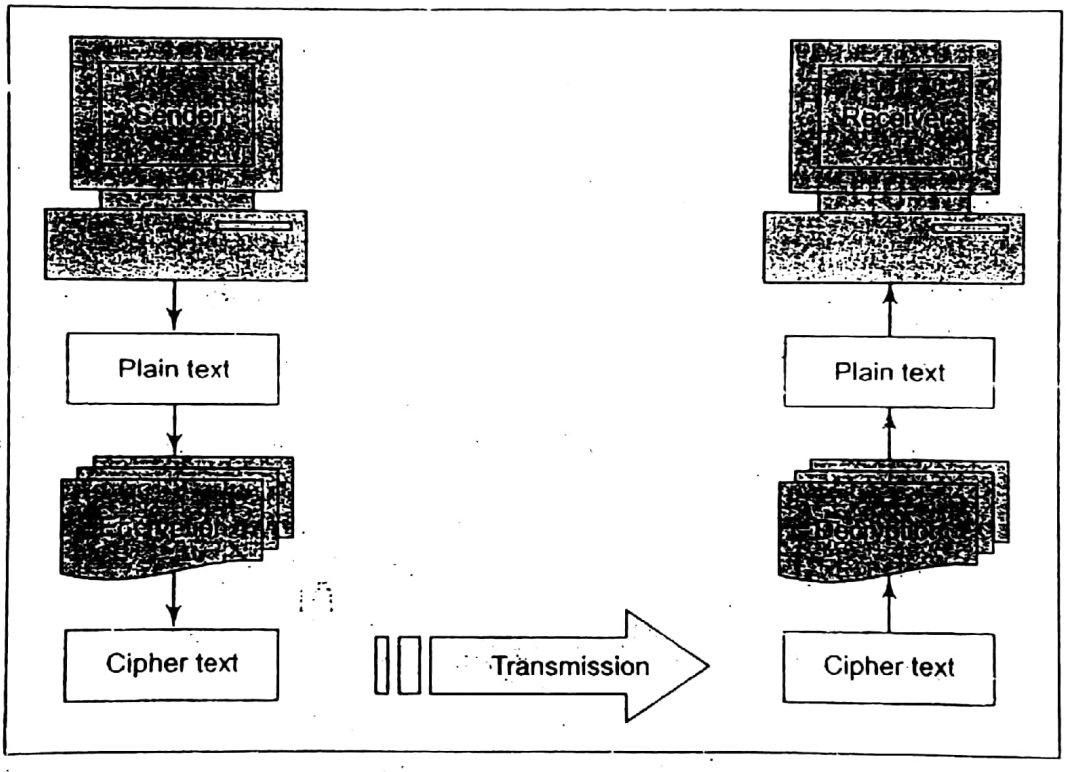


Fig. 2.7 Elements of a cryptographic operation

Let us now write our original email message and the resulting cipher text by using the alphabet-replacing scheme, as shown in Fig. 2.8. This will clarify the idea further.

As shown in Fig. 2.9, there are two primary ways in which a plain-text message can be codified to obtain the corresponding cipher text: **substitution** and **transposition**.

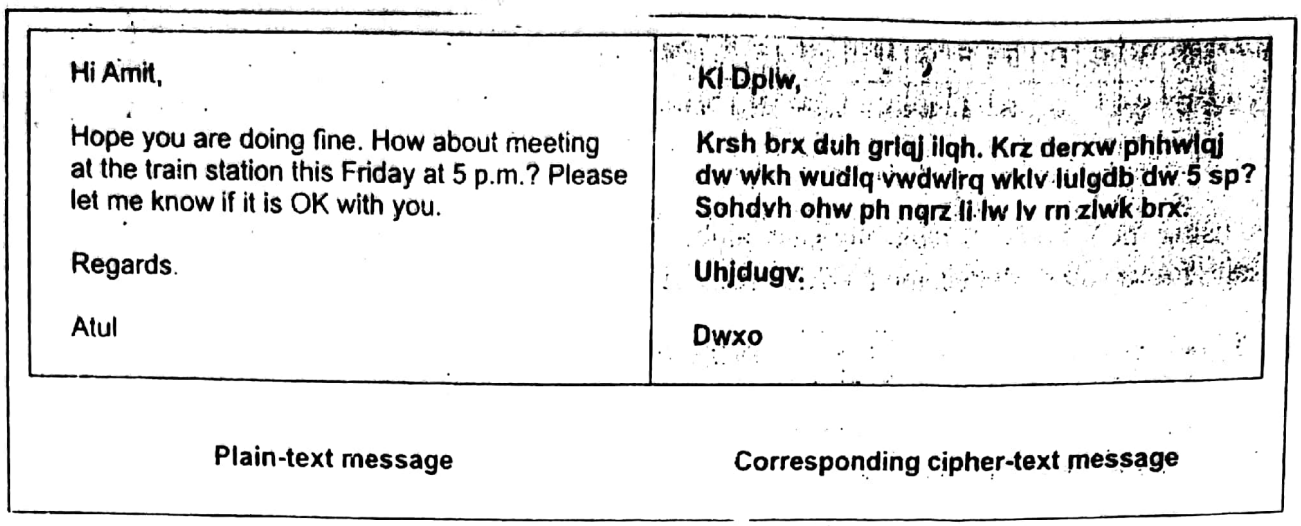


Fig. 2.8 Example of a plain-text message being transformed into cipher text

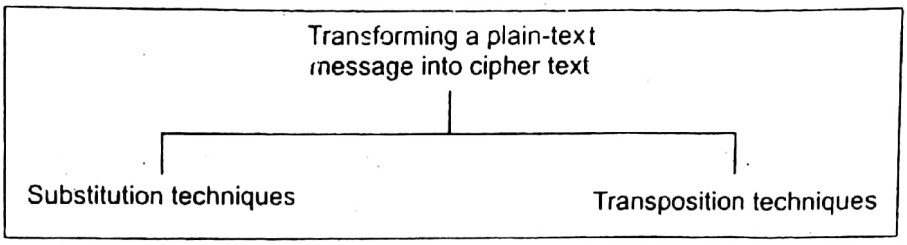


Fig. 2.9 Techniques for transforming plain text to cipher text

Let us discuss these two approaches now. Note that when the two approaches are used together, we call the technique **product cipher**.

2.3 SUBSTITUTION TECHNIQUES

2.3.1 Caesar Cipher

The scheme explained earlier (of replacing an alphabet with the one three places down the order) was first proposed by Julius Caesar, and is termed **Caesar cipher**. It was the first example of substitution cipher. In the substitution-cipher technique, the characters of a plain-text message are replaced by other characters, numbers or symbols. The Caesar cipher is a special case of substitution technique wherein each alphabet in a message is replaced by an alphabet three places down the line. For instance, using the Caesar cipher, the plain-text ATUL will become cipher-text DWXO.

In the substitution-cipher technique, the characters of a plain-text message are replaced by other characters, numbers or symbols.

Clearly, the Caesar cipher is a very weak scheme of hiding plain-text messages. All that is required to break the Caesar cipher is to do the reverse of the Caesar cipher process—i.e. replace each alphabet in a cipher-text message produced by Caesar cipher with the alphabet that is three places up the line. Thus, to work backwards, take a cipher text produced by Caesar cipher, and replace each A with X, B with Y, C with Z, D with A, E with B and so-on. The simple algorithm required to break the Caesar cipher can be summarized as shown in Fig. 2.10.

1. Read each alphabet in the cipher-text message, and search for it in the second row of the *replacement* table (i.e. the second row of the table).
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.

Fig. 2.10 Algorithm to break Caesar cipher

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 in Fig. 2.11.

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

Fig. 2.11 Example of breaking Caesar cipher

2.3.2 Modified Version of Caesar Cipher

The Caesar cipher is good in theory, but not so good in practice. Let us now try and complicate the Caesar cipher to make an attacker's task difficult. 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* alphabet 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 the Caesar cipher, as shown in Fig. 2.12.

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.

Fig. 2.12 Algorithm to break the modified Caesar cipher

Let us take a cipher-text message produced by the modified Caesar cipher, and try breaking it to obtain the original plain-text message by applying the algorithm shown earlier. Since each alphabet in the plain-text can be potentially replaced by any other of the 25 alphabets, we have 25 possible plain-text messages to choose from. Thus, the output produced by the above algorithm to break a cipher-text message *KWUM PMZN* is shown in Fig. 2.13.

We can see that the cipher text shown in the first row of the figure needs 25 different attempts to break in, as depicted by the algorithm shown earlier. As it turns out, the 18th attempt reveals the correct

Cipher text	K	W	U	M		P	M	Z	M
Attempt Number (Value of *k*)									
1	L	X	V	N		Q	N	A	N
2	M	Y	W	O		R	O	B	O
3	N	Z	X	P		S	P	C	P
4	O	A	Y	Q		T	Q	D	Q
5	P	B	Z	R		U	R	E	R
6	Q	C	A	S		V	S	F	S
7	R	D	B	T		W	T	G	T
8	S	E	C	U		X	U	H	U
9	T	F	D	V		Y	V	I	V
10	U	G	E	W		Z	W	J	W
11	V	H	F	X		A	X	K	X
12	W	I	G	Y		B	Y	L	Y
13	X	J	H	Z		C	Z	M	Z
14	Y	K	I	A		D	A	N	A
15	Z	L	J	B		E	B	O	B
16	A	M	K	C		F	C	P	C
17	B	N	L	D		G	D	Q	D
18	C	O	M	E		H	E	R	E
19	D	P	N	F		I	F	S	F
20	E	Q	O	G		J	G	T	G
21	F	R	P	H		K	H	U	H
22	G	S	Q	I		L	I	V	I
23	H	T	R	J		M	J	W	J
24	I	U	S	K		N	K	X	K
25	J	V	T	L		O	L	Y	L

Fig. 2.13 Attempts to break modified Caesar-cipher text using multiple possibilities

plain text corresponding to the cipher text. Therefore, we can actually stop at this juncture. For the sake of completeness, however, we have shown all the 25 steps, which is, of course, the worst possible case.

A mechanism of encoding messages so that they can be sent securely is called cryptography. Let us take this opportunity to introduce a few terms used in cryptography. An attack on a cipher-text message, wherein the attacker attempts to use all possible permutations and combinations, is called a **brute-force attack**. The process of trying to break any cipher-text message to obtain the original plain-text message itself is called **cryptanalysis**, and the person attempting a cryptanalysis is called a **cryptanalyst**.

A cryptanalyst is a person who attempts to break a cipher-text message to obtain the original plain-text message. The process itself is called cryptanalysis.

As we have noticed, even the modified version of the Caesar cipher is not very secure. After all, the cryptanalyst needs to be aware of only the following points to break a cipher-text message using the brute-force attack, in this scheme:

1. Substitution technique was used to derive the cipher text from the original plain text.
2. There are only 25 possibilities to try out.
3. The language of the plain text was English.

A cryptanalyst attempting a brute-force attack tries all possibilities to derive the original plain-text message from a given cipher-text message.

Anyone armed with this knowledge can easily break a cipher text produced by the modified version of Caesar cipher. How can we make the modified Caesar cipher even tougher to crack?

2.3.3 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 try out a maximum of 25 possible attacks, and he/she is assured of 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!

To put it mathematically, we can now have any permutation or combination of the 26 alphabets, which means $(26 \times 25 \times 24 \times 23 \times \ldots 2)$ or 4×10^{26} possibilities! This is extremely hard to crack. It might actually take years to try out these many combinations even with the most modern computers.

Mono-alphabetic ciphers pose a difficult problem for a cryptanalyst because it can be very difficult to crack, thanks to the high number of possible permutations and combinations.

There is only one hitch. If the cipher text created with this technique is short, the cryptanalyst can try different attacks based on his/her knowledge of the English language. As we know, some alphabets in the English language occur more frequently than others. Language analysts have found that given a single alphabet in cipher text, the probability that it is a P is 13.33%—the highest. After P comes Z, which is likely to occur 11.67%. The probability that the alphabet is C, K, L, N or R is almost 0—the lowest.

A cryptanalyst looks for patterns of alphabets in a cipher text, substitutes the various available alphabets in place of cipher-text alphabets, and then tries his/her attacks.

Apart from single-alphabet replacements, the cryptanalyst also looks for repeated patterns of words to try the attacks. For example, the cryptanalyst might look for two-alphabet cipher text patterns since the word *to* occurs very frequently in English. If the cryptanalyst finds that two alphabet combinations are found frequently in a cipher-text message, he/she might try and replace all of them with *to*, and then try and deduce the remaining alphabets/words. Next, the cryptanalyst might try to find repeating three-alphabet patterns and try and replace them with the word *the*, and so on.

2.3.4 Homophonic Substitution Cipher

The **homophonic substitution cipher** is very similar to mono-alphabetic cipher. Like a plain substitution cipher technique, we replace one alphabet with another in this scheme. However, the difference between the two techniques is that whereas the replacement alphabet set in case of the simple substitution techniques is fixed (e.g. replace A with D, B with E, etc.), in the case of homophonic substitution cipher, one plain-text alphabet can map to more than one cipher-text alphabet. For instance, A can be replaced by D, H, P, R; B can be replaced by E, I, Q, S, etc.

Homophonic substitution cipher also involves 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.

2.3.5 Polygram Substitution Cipher

In the **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 by YUQQW, but HELL could be replaced by a totally different cipher text block TEUI, as shown in Fig. 2.14. This is true in spite of the first four characters of the two blocks of text (HELL) being the same. This shows that in the polygram substitution cipher, the replacement of plain text happens block by block, rather than character by character.

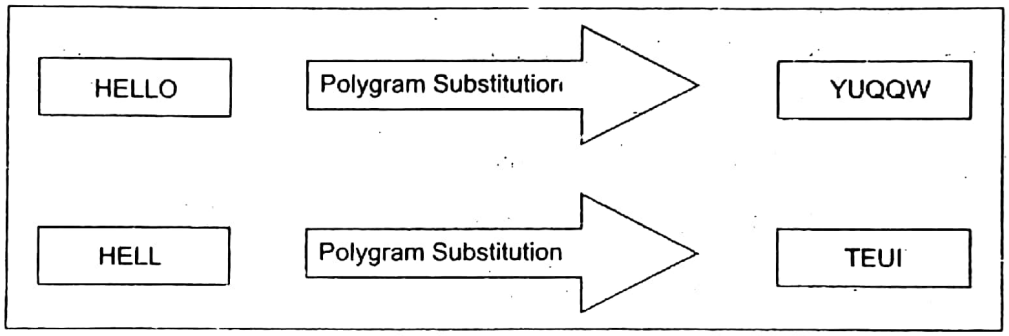


Fig. 2.14 Polygram substitution

Polygram substitution cipher technique replaces one block of plain text with another block of cipher text—it does not work on a character-by-character basis.

2.3.6 Polyalphabetic Substitution Cipher

Leon Battista invented the **polyalphabetic substitution cipher** in 1568. This cipher has been broken many times, and yet it has been used extensively. The **Vigenére cipher** and the **Beaufort cipher** are examples of polyalphabetic 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 all 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 (in this case, 30) is called the **period** of the cipher.

The main features of polyalphabetic substitution cipher are the following:

(a) It uses a set of related monoalphabetic substitution rules.

(b) It uses a key that determines which rule is used for which transformation.

For example, let us discuss the Vigenére cipher, which is an example of this cipher. In this algorithm, 26 Caesar ciphers make up the mono-alphabetic substitution rules. There is a shifting mechanism, from a count of 0 to 25. For each plain-text letter, we have a corresponding substitution, which we call the *key letter*. For instance, the key value is *e* for a letter with shift as 3.

To understand this technique, we need to take a look at a table, which is formally known as Vigenére tableau. This table is shown in Fig. 2.15.

	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
b	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
c	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
d	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
e	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
f	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
g	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
h	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
i	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
j	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
k	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
l	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
m	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
n	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
o	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
p	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
r	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
s	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
t	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
u	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
v	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
w	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
x	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Fig. 2.15 Vigenére tableau

The logic for encryption is quite simple. For key letter *p* and plain-text letter *q*, the corresponding cipher-text letter is at the intersection of row titled *p* and column titled *q*. For this very particular case, the cipher text, therefore, would be *F*, based on the above table.

By now, it should be clear that for encrypting a plain-text message, we need a key whose length is equal to that of the plain-text message. Usually, a key that repeats itself is used.

2.3.7 Playfair Cipher

The **Playfair cipher**, also called **Playfair square**, is a cryptographic technique used for manual encryption of data. This scheme was invented by Charles Wheatstone in 1854. However, eventually the scheme came to be known by the name of Lord Playfair, who was Wheatstone's friend. Playfair made this scheme popular, and hence his name was used.

The Playfair cipher was used by the British army in World War I and by the Australians in World War II. This was possible because the Playfair cipher is quite fast to use and does not demand any special equipment to be used. It was used to protect important but not very critical information, so that by the time the cryptanalysts could break it, the value of the information was nullified anyway! In today's world, of course, Playfair cipher would be deemed as an outdated cryptographic algorithm, and rightly so. Playfair cipher now has only academic purpose, except in its usage in some crosswords that appear in several newspapers.

The Playfair encryption scheme uses two main processes, as shown in Fig. 2.16.

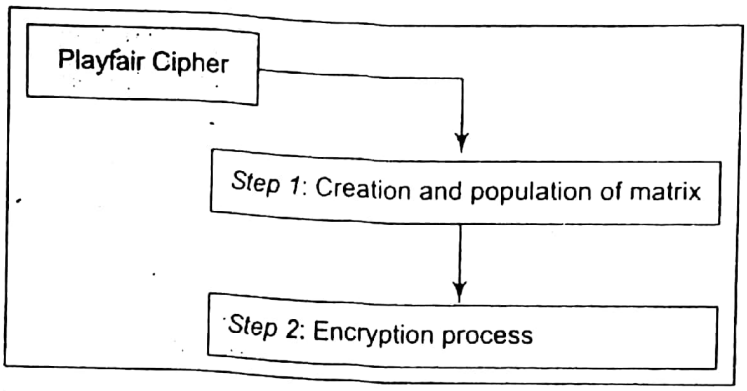


Fig. 2.16 Playfair Cipher steps

Step 1: Creation and Population of Matrix

The Playfair cipher makes use of a 5×5 matrix (table), which is used to store a *keyword* or *phrase* that becomes the *key* for encryption and decryption. The way this is entered into the 5×5 matrix is based on some simple rules, as shown in Fig. 2.17.

1. Enter the keyword in the matrix row-wise: left-to-right, and then top-to-bottom.
2. Drop duplicate letters.
3. Fill the remaining spaces in the matrix with the rest of the English alphabets (A-Z) that were not a part of our keyword. While doing so, combine I and J in the same cell of the table. In other words, if I or J is a part of the keyword, disregard both I and J while filling the remaining slots.

Fig. 2.17 Matrix creation and population

For example, suppose that our keyword is PLAYFAIR EXAMPLE. Then, the 5×5 matrix containing our keyword will look as shown in Fig. 2.18.

Explanation of this row-wise is as follows.

P	L	A	Y	F
I	R	E	X	M
B	C	D	G	H
K	N	O	Q	S
T	U	V	W	Z

Fig. 2.18 Keyword matrix for our example

Row 1 The first row of the matrix is as shown in Fig. 2.19.

P	L	A	Y	F

Fig. 2.19 Keyword matrix (*first* row of mentioned example)

As we can see, this is simply the first five letters of our keyword (*PLAYF*). None of the alphabets is a duplicate so far. Therefore, we write these one after the other in the same row, as per our rule #1 defined earlier.

Row 2 The second row of the matrix is as shown in Fig. 2.20.

I	R	E	X	M

Fig. 2.20 Keyword matrix (*second* row of mentioned example)

The first four cells are just the continuation of keyword (*IREX*) from where we had left in the previous row. This is as per our rule #1 defined earlier. However, after this, we have a repetition of alphabet A, as shown in Fig. 2.21.

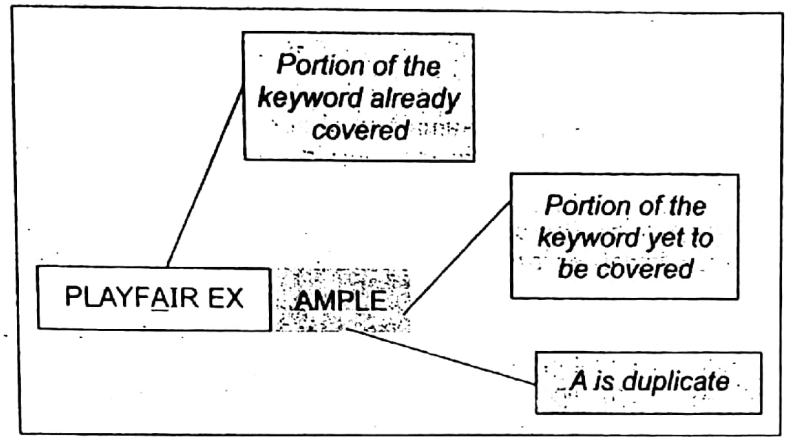


Fig. 2.21 Situation when a duplicate alphabet (*A*) is discovered

Therefore, we disregard the repeating A as per our rule #2 defined earlier, and choose the next alphabet, which is M. This gets loaded into the fifth cell of this row.

Row 3 The third row of the matrix is as shown in Fig. 2.22.

B	C	D	G	H

Fig. 2.22 Keyword matrix (*third* row of mentioned example)

Let us now review what part of our keyword was covered in the first two rows.

Our keyword was PLAYFAIR EXAMPLE.

Our first two rows were P L A Y F and I R E X M.

Let us now change the font of the alphabets in our keyword which are covered anywhere in these two rows, in italics. Let us also indicate duplicates with an underscore. This would make our keyword look like this:

PLAYFAIR EXAMPLE

As we can see, all of our keyword alphabets are covered now (because every alphabet is (a) either in italics, indicating that it is a part of our matrix, or is (b) underlined, indicating that it is a duplicate).

Therefore, there is nothing left to populate in our matrix from the third row onwards. Hence, we would now consider our rule #3 defined earlier. This rule suggests that we should fill the remaining part of the matrix with alphabets from A-Z that are yet unused. Based on this criterion, we see that B, C, D, G, and H will fit into the third row of our matrix.

Row 4 The fourth row of the matrix is as shown in Fig. 2.23

K	N	O	Q	S

Fig. 2.23 Keyword matrix (*fourth* row of mentioned example)

Here, we simply apply our rule #3 straightaway to get the te of K N O Q S.

Row 5 The fifth row of the matrix is as shown in Fig. 2.24.

T	U	V	W	Z

Fig. 2.24 Keyword matrix (fifth row of mentioned example)

Here, we simply apply our rule #3 straightaway to get the text of T U V W Z.

Step 2: Encryption Process

The encryption process consists of five steps, as outlined in Fig. 2.25.

1. Before executing these steps, the plain-text message that we want to encrypt needs to be broken down into groups of two alphabets. For example, if our message is MY NAME IS ATUL, it becomes MY NA ME IS AT UL. The encryption process works on this *broken-down* message.
2. If both alphabets are the same (or only one is left), add an X after the first alphabet. Encrypt the new pair and continue.
3. If both the alphabets in the pair appear in the same row of our matrix, replace them with alphabets to their immediate right respectively. If the original pair is on the right side of the row, then wrapping around to the left side of the row happens.
4. If both the alphabets in the pair appear in the same column of our matrix, replace them with alphabets immediately below them respectively. If the original pair is on the bottom side of the row, then wrapping around to the top side of the row happens.
5. If the alphabets are not in the same row or column, replace them with the alphabets in the same row respectively, but at the other pair of corners of the rectangle defined by the original pair. The order is quite significant here. The first encrypted alphabet of the pair is the one that is present on the same row as the first plain-text alphabet.

Fig. 2.25 Encryption process in Playfair cipher

Decryption process works in the opposite direction. We also need to remove the extra X alphabets that we had added in step #1 above, if any.

Let us now take a concrete example to illustrate the process of encrypting some text using a keyword. Our keyword is PLAYFAIR EXAMPLE and the original text is MY NAME IS ATUL. We know that the matrix for our keyword is as shown in Fig. 2.26. We have discussed this in detail earlier and need not repeat it.

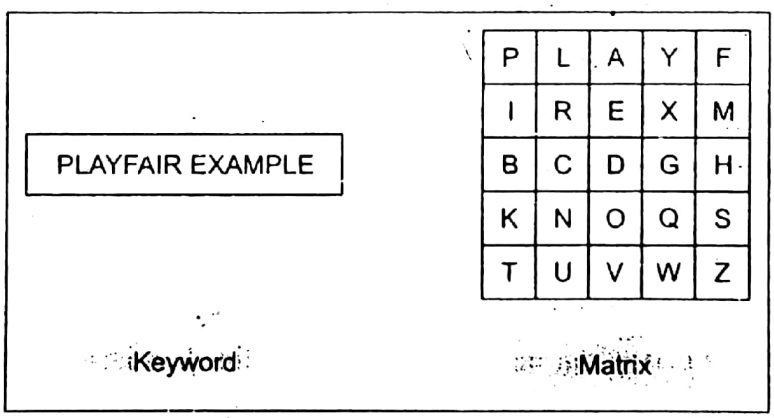


Fig. 2.26 Keyword matrix for our example

Here is an explanation of the encryption process.

1. First we break the original text into pairs of two alphabets each. This means that our original text would now look like this:

   MY NA ME IS AT UL

2. Now, we apply our Playfair cipher algorithm to this text. The first pair of alphabets is MY. Looking at the matrix, we see that the alphabets M and Y do not occur in the same row or column. Therefore, we need to apply Step #5 of our Playfair cipher encryption process. This means that we need to replace this text with the text diagonally opposite to it. In this case, this text is XF, which is our first cipher text block. This is shown in Fig. 2.27.

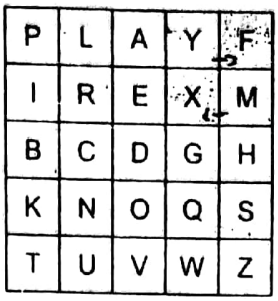


Fig. 2.27 Alphabet Pair 1

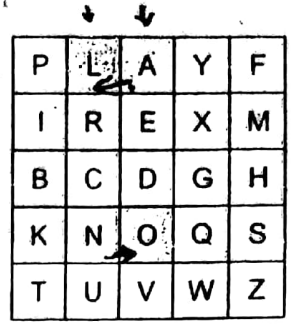


Fig. 2.28 Alphabet Pair 2

3. Our next text block to be encrypted is NA. Again, Step #5 will apply as depicted in Fig. 2.28.

   As we can see, our second block of cipher text is OL.

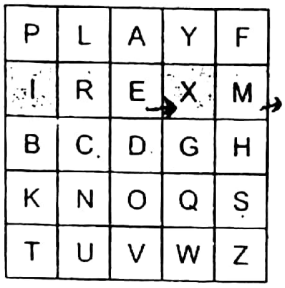


Fig. 2.29 Alphabet Pair 3

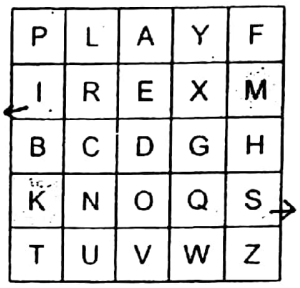


Fig. 2.30 Alphabet Pair 4

4. We will now take a look at the third block of plain text, which is ME. This is shown in Fig. 2.29. We can see that the alphabets E and M making up this block are in the same (second) row. Therefore, based on our logic of Step #3, the cipher-text block would be IX.

5. We will now take a look at the fourth block of plain text, which is IS. This is shown in Fig. 2.30. We can see that we need to apply the logic of Step #5 to get the diagonal alphabets. Based on this, the cipher-text block would be MK.

6. We will now take a look at the fifth block of plain text, which is AT. This is shown in Fig. 2.31. We can see that we need to apply the logic of Step #5 to get the diagonal alphabets. Based on this, the cipher-text block would be PV.

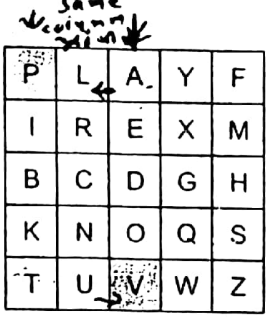


Fig. 2.31 Alphabet Pair 5

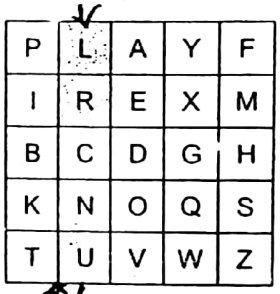


Fig. 2.32 Alphabet Pair 6

7. We will now take a look at the sixth and last block of plain text, which is UL. This is shown in Fig. 2.32. We can see that the two alphabets U and L are in the same column. Therefore, we need to apply the logic of Step #4 to get the alphabets LR.

Thus, our plain-text blocks MY NA ME IS AT UL becomes XF OL IX MK PV LR.

We will not illustrate the decryption process. It is pretty straightforward. It would work in exactly the opposite steps. We leave it to the reader to verify this.

Just to get ourselves more familiar with the process further, we illustrate another example. For brevity, we have taken out the description of the various steps.

2.3.8 Hill Cipher

The **Hill cipher** works on multiple letters at the same time. Hence, it is a type of polygraphic substitution cipher. Lester Hill invented this in 1929. The Hill cipher has its roots in the matrix theory of mathematics. More specifically, we need to know how to compute the inverse of a matrix. This mathematics is explained in *Appendix A*. Interested readers are encouraged to refer to the mathematical theory there.

The way the Hill cipher works is as shown in Fig. 2.34.

Our keyword here is *Harsh*. The plain text to be encrypted is *My name is Jui Kahate."I am Harshu's sister.*

Based on this information, the keyword matrix is as follows:

H	A	R	S	B
C	D	E	F	G
I	K	L	M	N
O	P	Q	T	U
V	W	X	Y	Z

Our plain-text message broken down into pairs of alphabets is:
MY NA ME IS IU IK AH AT EI AM HA RS HU 'S XS IS TE RX

Using Playfair cipher based on the above matrix, the resulting cipher text would be:
TS KB LF MH NO KL RA SP CL SK AR SB BO AB YR MH QF ER

The reader is encouraged to work out this example step by step.

Fig. 2.33 Practice example for Playfair cipher

1. Treat every letter in the plain-text message as a number, so that A = 0, B = 1, ..., Z = 25.
2. The plain-text message is organized as a matrix of numbers, based on the above conversion. For example, if our plain text is CAT. Based on the above step, we know that C = 2, A = 0, and T = 19. Therefore, our plain-text matrix would look as follows:

$$\begin{pmatrix} 2 \\ 0 \\ 19 \end{pmatrix}$$

3. Now, our plain-text matrix is multiplied by a matrix of randomly chosen keys. The key matrix consists of size $n \times n$, where n is the number of rows in our plain-text matrix. For example, we take the following key matrix:

$$\begin{pmatrix} 6 & 24 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{pmatrix}$$

4. Now multiply the two matrices, as shown below:

$$\begin{pmatrix} 2 \\ 0 \\ 19 \end{pmatrix} \quad X \quad \begin{pmatrix} 6 & 24 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{pmatrix} \quad = \quad \begin{pmatrix} 31 \\ 216 \\ 325 \end{pmatrix}$$

5. Now compute a *mod 26* value of the above matrix. That is, take the remainder after dividing the above matrix values by 26. That is

$$\begin{pmatrix} 31 \\ 216 \\ 325 \end{pmatrix} \quad \text{mod } 26 = \quad \begin{pmatrix} 5 \\ 8 \\ 13 \end{pmatrix}$$

6. (This is because: 31 / 26 = 1 with a remainder of 5: which goes in the above matrix, and so on).
7. Now, translating the numbers to alphabets, 5 = F, 8 = I, and 13 = N. Therefore, our cipher text is *FIN*.
8. For decryption, take the cipher-text matrix and multiply it by the inverse of our original key matrix (explained later). The inverse of our original key matrix is

$$\begin{pmatrix} 8 & 5 & 10 \\ 21 & 8 & 21 \\ 21 & 12 & 8 \end{pmatrix}$$

Fig. 2.34(a) Hill cipher example—Part 1/2

1. For decryption, take the cipher-text matrix and multiply it by the inverse of our original key matrix. The inverse of our original key matrix is

$$\begin{pmatrix} 8 & 5 & 10 \\ 21 & 8 & 21 \\ 21 & 12 & 8 \end{pmatrix} \times \begin{pmatrix} 5 \\ 8 \\ 13 \end{pmatrix} = \begin{pmatrix} 210 \\ 442 \\ 305 \end{pmatrix}$$

2. Now we need to take modulo 26 of this matrix, as follows.

$$\begin{pmatrix} 210 \\ 442 \\ 305 \end{pmatrix} \quad \text{mod } 26 = \quad \begin{pmatrix} 2 \\ 0 \\ 19 \end{pmatrix}$$

3. Thus, our plain-text matrix contains 2, 0, 19; which corresponds to 2 = C, 0 = A, and 19 = T. This gives is the original plain text back successfully.

Fig. 2.34(b) Hill cipher example—Part 2/2

The Hill cipher is vulnerable to the *known-plain-text attack*, which we are going to discuss later. This is because it is linear (i.e. it is possible to compute smaller factors of the matrices, work on them individually, and then join them back as and when they are ready).

■ 2.4 TRANSPOSITION TECHNIQUES ■

As we discussed, substitution techniques focus on substituting a plain-text alphabet with a cipher-text alphabet. Transposition techniques differ from substitution techniques in the way that they do not simply replace one alphabet with another, but they also perform some permutation over the plain text

2.4.1 Rail-Fence Technique

The **rail-fence technique** is an example of transposition. It uses a simple algorithm as shown in Fig. 2.35.

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.

Fig. 2.35 Rail-fence technique

Let us illustrate the rail-fence technique with a simple example. Suppose that we have a plain-text message :*Come home tomorrow*. How would we transform that into a cipher-text message using the rail-fence technique? This is shown in Fig. 2.36.

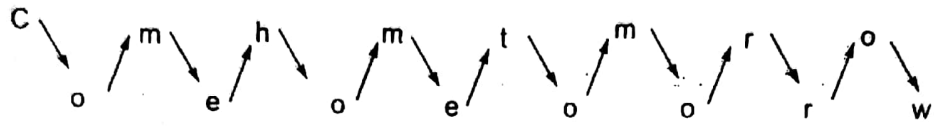
As the figure shows, the plain-text message '*Come home tomorrow*' transforms into '*Cmhmtm-rooeoeoorw*' with the help of rail-fence technique.

✓ *Rail-fence technique involves writing plain text as a sequence of diagonals and then reading it row by row to produce cipher text.*

It should be quite clear that the rail-fence technique is quite simple for a cryptanalyst to break into. It has very little sophistication built in.

Original plain-text message: ***Come home tomorrow***

1. After we arrange the plain-text message as a sequence of diagonals, it would look as follows (write the first character on the first line i.e. *C*, then second character on the second line, i.e. *o*, then the third character on the first line, i.e. *m*, then the fourth character on the second line, i.e. e, and so on). This creates a zigzag sequence, as shown below.



2. Now read the text row by row, and write it sequentially. Thus, we have: ***Cmhmtmrooeoeoorw*** as the cipher text.

Fig. 2.36 Example of rail-fence technique

2.4.2 Simple Columnar Transposition Technique

1. Basic Technique

Variations of the basic transposition technique such as rail-fence technique exist. Such a scheme is shown in Fig. 2.37, which we shall call **simple columnar transposition technique**.

1. Write the plain-text message row by row in a rectangle of a pre-defined size.
2. 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, 3, 1, etc.
3. The message thus obtained is the cipher-text message.

Fig. 2.37 Simple columnar transposition technique

Let us examine the simple columnar transposition technique with an example. Consider the same plain-text message *'Come home tomorrow'*. Let us understand how it can be transformed into cipher text using this technique. This is illustrated in Fig. 2.38.

Original plain-text message: ***Come home tomorrow***

1. Let us consider a rectangle with six columns. Therefore, when we write the message in the rectangle row by row (suppressing spaces), 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, let us decide the order of columns as some random order, say 4, 6, 1, 2, 5 and 3. Then read the text in the order of these columns.
3. The cipher text thus obtained would be ***eowoocmroerhmmto***.

Fig. 2.38 Example of simple columnar transposition technique

✓ *The simple columnar transposition technique simply arranges the plain text as a sequence of rows of a rectangle that are read in columns randomly.*

Like the rail-fence technique, the simple columnar transposition Technique is also quite simple to break into. It is a matter of trying out a few permutations and combinations of column orders to get hold of

the original plain text. To make matters complex for a cryptanalyst, we can modify the simple columnar transposition technique to add another twist: perform more than one round of transposition using the same technique.

2. Simple Columnar Transposition Technique with Multiple Rounds

To improve the basic simple columnar transposition technique, we can introduce more complexity. The idea is to use the same basic procedure as used by the simple columnar transposition technique, but to do it more than once. That adds considerably more complexity for the cryptanalyst.

The basic algorithm used in this technique is shown in Fig. 2.39.

1. Write the plain text message row by row in a rectangle of a pre-defined size.
2. 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, 3, 1, etc.
3. The message thus obtained is the cipher text message of round 1.
4. Repeat steps 1 to 3 as many times as desired.

Fig. 2.39 Simple columnar transposition technique with multiple rounds

As we can see, the only addition in this technique to the basic simple columnar transposition technique is step 4, which results in the execution of the basic algorithm on more than one occasion. Although this sounds trivial, in reality, it makes the cipher text far more complex as compared to the basic simple columnar transposition technique. Let us extend our earlier example to now have multiple rounds of transposition, as shown in Fig. 2.40.

Original plain-text message: ***Come home tomorrow***

1. Let us consider a rectangle with six columns. Therefore, when we write the message in 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, let us decide the order of columns as some random order, say 4, 6, 1, 2, 5, 3. Then read the text in the order of these columns.
3. The cipher text thus obtained would be ***eowoocmroerhmmto*** in round 1.
4. Let us perform Steps 1 through 3 once more. So, the tabular representation of the cipher text after round 1 is as follows:

Column 1	Column 2	Column 3	Column 4	Column 5	Column 6
e	o	w	o	o	c
m	r	o	e	r	h
m	m	t	o		

5. Now, let us use the same order of columns, as before, that is 4, 6, 1, 2, 5, and 3. Then read the text in the order of these columns.
6. The cipher text thus obtained would be ***oeochemmormorwot*** in round 2.
7. Continue like this if more number of iterations is desired, otherwise stop.

Fig. 2.40 Example of simple columnar transposition technique with multiple rounds

As the figure shows, multiple rounds or iterations add more complexity to the cipher text produced by the basic simple columnar transposition technique. The more the number of iterations, the more complex is the cipher text thus produced.

Cipher text produced by the simple columnar transposition technique with multiple rounds is much more complex to crack as compared to the basic technique.

2.4.3 Vernam Cipher (One-Time Pad)

The **Vernam cipher**, whose specific subset is called **one-time pad**, is implemented using a random set of non-repeating characters as the input cipher text. The most significant point here 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 input cipher text is equal to the length of the original plain text. The algorithm used in the Vernam cipher is described in Fig. 2.41.

1. Treat each plain-text alphabet as a number in an increasing sequence, i.e. A = 0, B = 1, ... Z = 25.
2. Do the same for each character of the input cipher text.
3. *Add* each number corresponding to the plain-text alphabet to the corresponding input cipher-text alphabet number.
4. If the sum thus produced is greater than 26, subtract 26 from it.
5. Translate each number of the sum back to the corresponding alphabet. This gives the output cipher text.

Fig. 2.41 Algorithm for Vernam cipher

Let us apply the Vernam cipher algorithm to a plain-text message *HOW ARE YOU* using a one-time pad *NCBTZQARX* to produce a cipher-text message *UQXTRUYFR* as shown in Fig. 2.42.

1. Plain text	H 7	O 14	W 22	A 0	R 17	E 4	Y 24	O 14	U 20
+									
2. One-time pad	13 N	2 C	1 B	19 T	25 Z	16 Q	0 A	17 R	23 X
3. Initial Total	20	16	23	19	42	20	24	31	43
4. Subtract 26, if > 25	20	16	23	19	16	20	24	5	17
5. Cipher text	U	Q	X	T	Q	U	Y	F	R

Fig. 2.42 Example of Vernam cipher

It should be clear that since the one-time pad is discarded after a single use, this technique is highly secure and suitable for small plain-text message, but is clearly impractical for large messages. The Vernam Cipher was first implemented at AT&T with the help of a device called the **Vernam machine**.

Vernam cipher uses a one-time pad, which is discarded after a single use, and therefore, is suitable only for short messages.

8/19/

Book Cipher/Running-Key Cipher

The iden used in **book cipher**, also incorrectly called **running-key cipher**, is quite simple, and is similar in principle to the Vernam cipher. For producing cipher text, some portion of text from a book is used, which serves the purpose of a one-time pad. Thus, the characters from a book are used as one-time pad, and they are *added* to the input plain-text message similar to the way a one-time pad works.

Running Key Cipher

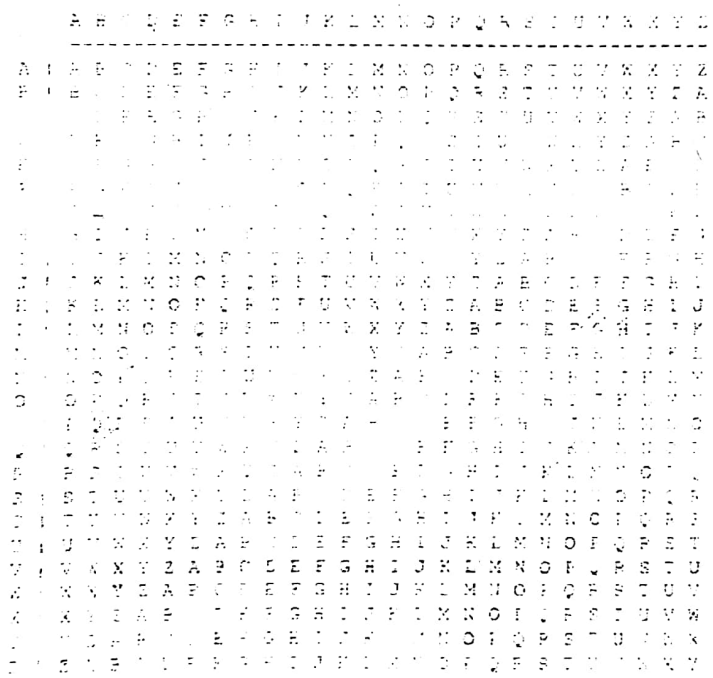
Introduction

The Running Key cipher has the same internal workings as the . The difference lies in how the key is chosen: the Vigenere cipher uses a short key that repeats, whereas the running key cipher uses a long key such as an excerpt from a book. This means the key does not repeat, making cryptanalysis more difficult. The cipher can still be broken though, as there are statistical patterns in both the key and the plaintext which can be exploited.

If the key for the running key cipher comes from a statistically random source, then it becomes a 'one time pad' cipher. One time pads are theoretically unbreakable ciphers, because every possible decryption is equally likely.

The Algorithm

The 'key' for a running key cipher is a long piece of text, e.g. an excerpt from a book. The running key cipher uses the following tableau (the 'tabula recta') to encipher the plaintext



To encipher a message, write the key stream above the plaintext. In this case our key is from a Terry Pratchett book 'How does the duck know that? said Victor'. If we needed to encipher a longer plaintext, we could just continue reading from the book.

```
HOWDOESTHEDUCKKNOWTHATSAIDVI
DEFENDTHEEASTWALLOFTHECASTLE
```

Now we take the letter we will be encoding, 'D', and find it on the first column on the tableau. Then, we move along the 'D' row of the tableau until we come to the column with the 'H' at the top (The 'H' is the keyword letter for the first 'D'). the intersection is our ciphertext character, 'K'

So, the ciphertext for the above plaintext is:

```
HOWDOESTHEDUCKKNOWTHATSAIDVI
```

Contents

Further reading

We recommend these books if you're interested in finding out more.

2.5 ENCRYPTION AND DECRYPTION

We have discussed the concepts of plain text and how it can be transformed into cipher text so that only the sender and the recipient can make any sense out of it. There are technical terms to describe these concepts, which we shall learn now. In technical terms, the process of encoding plain-text messages into cipher text messages is called **encryption**. Figure 2.43 illustrates the idea.

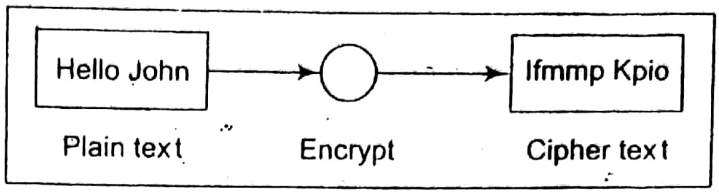


Fig. 2.43 Encryption

The reverse process of transforming cipher-text messages back to plain text messages is called **decryption**. Figure 2.44 illustrates the idea.

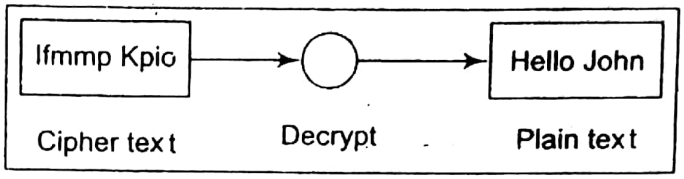


Fig. 2.44 Decryption

Decryption is exactly the opposite of encryption. Encryption transforms a plain-text message into cipher text, whereas decryption transforms a cipher text message back into plain text.

In computer-to-computer communications, the computer at the sender's end usually transforms a plain-text message into ciphertext by performing encryption. The encrypted cipher-text message is then sent to the receiver over a network (such as the Internet, although it can be any other network). The receiver's computer then takes the encrypted message, and performs the reverse of encryption, i.e. it performs the decryption process to obtain the original plain-text message. This is shown in Figure 2.45.

To encrypt a plain-text message, the sender (we shall henceforth treat the term *sender* to mean the *sender's computer*) performs encryption, i.e. applies the encryption algorithm. To decrypt a received encrypted message, the recipient performs decryption, i.e. applies the **decryption algorithm**. The algorithm is similar in concept to the algorithms we discussed earlier.

Clearly, the decryption algorithm must be the same as the **encryption algorithm**. Otherwise, decryption would be unable to retrieve the original message. For instance, if the sender uses the rail-fence technique for encryption and the receiver uses the simple columnar technique for decryption, the decryption would yield a totally incorrect plain text. Thus, the sender and the receiver must agree on a common algorithm for any meaningful communication can take place. The algorithm basically takes one text as input and produces another as the output.

The second aspect of performing encryption and encryption of messages is the **key**. What is a key? A key is something similar to the one-time pad used in the Vernam cipher. Anyone can use the Vernam cipher. However, as long as only the sender and the receiver know the one-time pad, no one except the sender and the receiver can do anything with the message.

Every encryption and decryption process has two aspects: the **algorithm** *and the* **key** *used for encryption and decryption.*

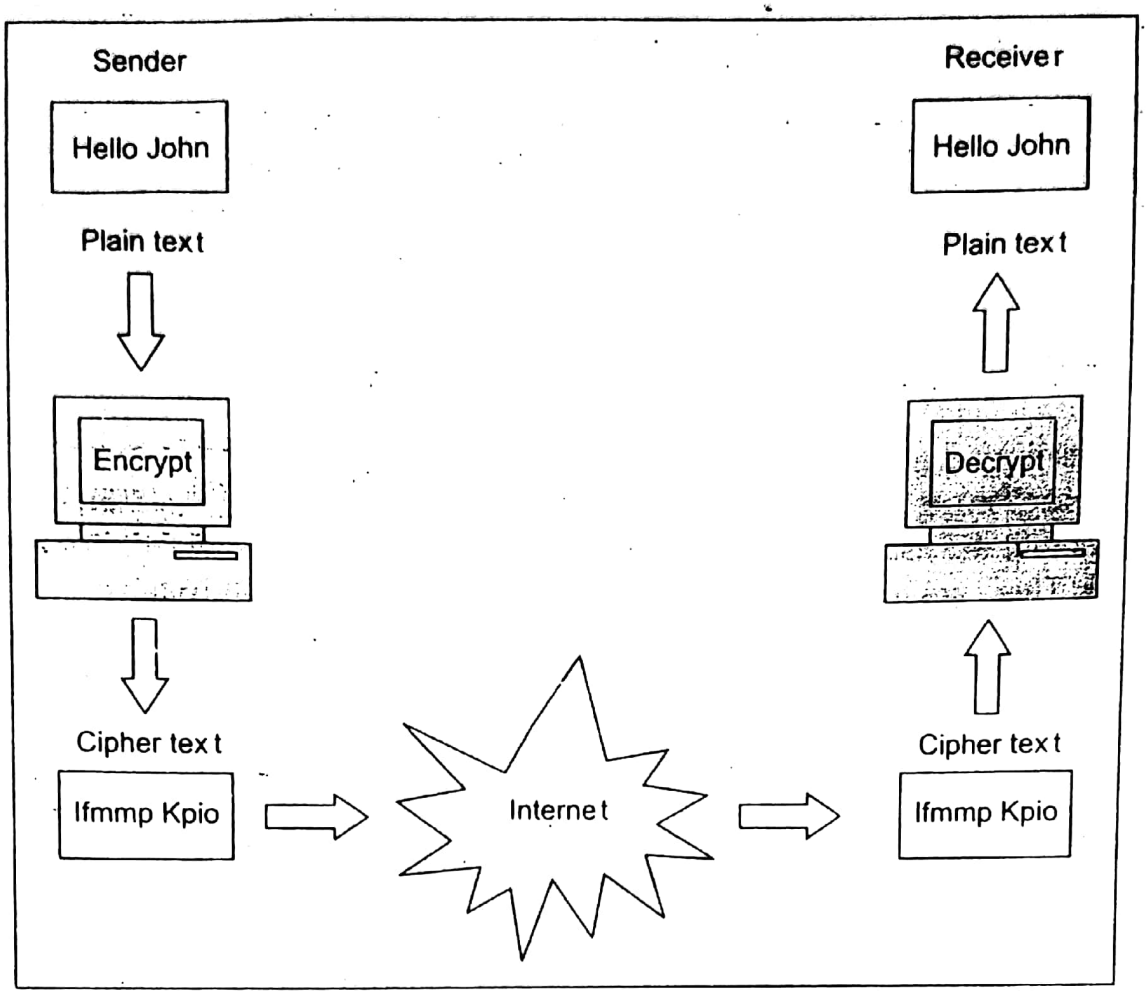


Fig. 2.45 Encryption and decryption in the real world

This is shown in Fig. 2.46.

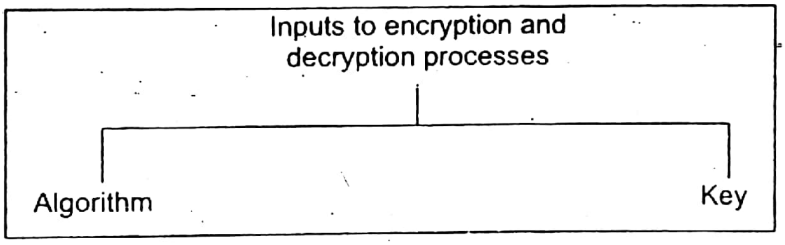


Fig. 2.46 Aspects of encryption and decryption

To understand this better, let us take the example of a combination lock which we use in real life. We need to remember the combination (which is a number, such as 871) needed to open up the lock. The facts that it is a combination lock and how to open it (*algorithm*) are pieces of public knowledge. However, the actual value of the key required for opening a specific lock (*key*), which is 871 in this case, is kept secret. The idea is illustrated in Fig. 2.47.

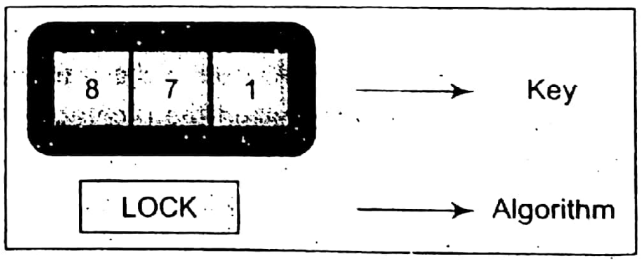


Fig. 2.47 Combination lock

Thus, as an example, the sender and receiver can safely agree to use Vernam cipher as the algorithm, and XYZ as the key, and be assured that no one else is able to get any access to their conversation. Others might know that the Vernam cipher is in use. However, they do not know that XYZ is the encryption/decryption key.

In general, the algorithm used for encryption and decryption processes is usually known to everybody. However, it is the key used for encryption and decryption that makes the process of cryptography secure.

Broadly, there are two cryptographic mechanisms, depending on what keys are used. If the *same* key is used for encryption and decryption, we call the mechanism **symmetric key cryptography**. However, if two *different* keys are used in a cryptographic mechanism, wherein one key is used for encryption, and *another, different* key is used for decryption; we call the mechanism **asymmetric key cryptography**. This is shown in Fig. 2.48.

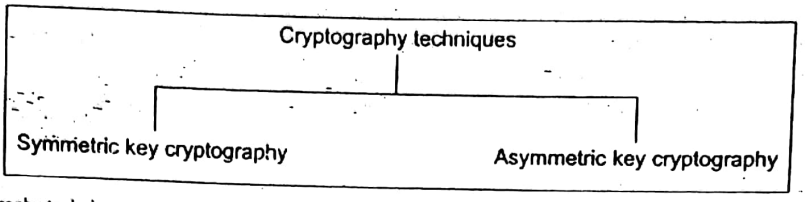


Fig. 2.48 Cryptography techniques

We shall study the basic concepts behind these two mechanisms now. We shall study the various computer-based cryptographic algorithms in each of these categories in great detail in subsequent chapters.

Symmetric key cryptography involves the usage of the same key for encryption and decryption. Asymmetric key cryptography involves the usage of one key for encryption, and another, different key for decryption.

Topic : SYMMETRIC AND ASYMMETRIC CRYPTOGRAPHY

→ There are two kinds of Cryptography as above

SYMMETRIC CRYPTOGRAPHY

→ A type of encryption where the same key is used to encrypt and decrypt the message. This differs from asymmetric (or public-key) encryption, which uses one key to encrypt a message and another to decrypt the message

→An encryption system in which the sender and receiver of a message share a single, common key that is used to encrypt and decrypt the message. Contrast this with public-key cryptology, which utilizes two keys - a public key to encrypt messages and a private key to decrypt them.

→ Symmetric-key systems are simpler and faster, but their main drawback is that the two parties must somehow exchange the key in a secure way. Public-key encryption avoids this problem because the public key can be distributed in a non-secure way, and the private key is never transmitted.

→Symmetric-key cryptography is sometimes called *secret-key cryptography*. The most popular symmetric-key system is the *Data Encryption Standard (DES)*.

→ Symmetric key encryption is also known as shared-key, single-key, secret-key, and private-key or one-key encryption. In this type of message encryption, both sender and receiver share the same key which is used to both encrypt and decrypt messages. Sender and receiver only have to specify the shared key in the beginning and then they can begin to encrypt and decrypt messages between them using that key. Examples include AES (Advanced Encryption Standard) and TripleDES (Data Encryption Standard).

Advantages

13

- **Simple:** This type of encryption is easy to carry out. All users have to do is specify and share the secret key and then begin to encrypt and decrypt messages.

- **Encrypt and decrypt your own files:** If you use encryption for messages or files which you alone intend to access, there is no need to create different keys. Single-key encryption is best for this.

- **Fast:** Symmetric key encryption is much faster than asymmetric key encryption.

- **Uses less computer resources:** Single-key encryption does not require a lot of computer resources when compared to public key encryption.

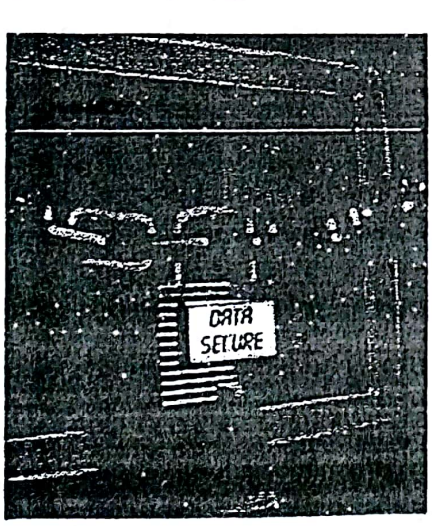
- **Prevents widespread message security compromise:** A different secret key is used for communication with every different party. If a key is compromised, only the messages between a particular pair of sender and receiver are affected. Communications with other people are still secu

Disadvantages

- **Need for secure channel for secret key exchange:** Sharing the secret key in the beginning is a problem in symmetric key encryption. It has to be exchanged in a way that ensures it remains secret

- **Too many keys:** A new shared key has to be generated for communication with every different party. This creates a problem with managing and ensuring the security of all these keys.

- **Origin and authenticity of message cannot be guaranteed:** Since both sender and receiver use th same key, messages cannot be verified to have come from a particular user. This may be a problem i there is a dispute.



→ Symmetric key encryption is an excellent way to protect your hard drive.

ASYMMETRIC CRYPTOGRAPHY

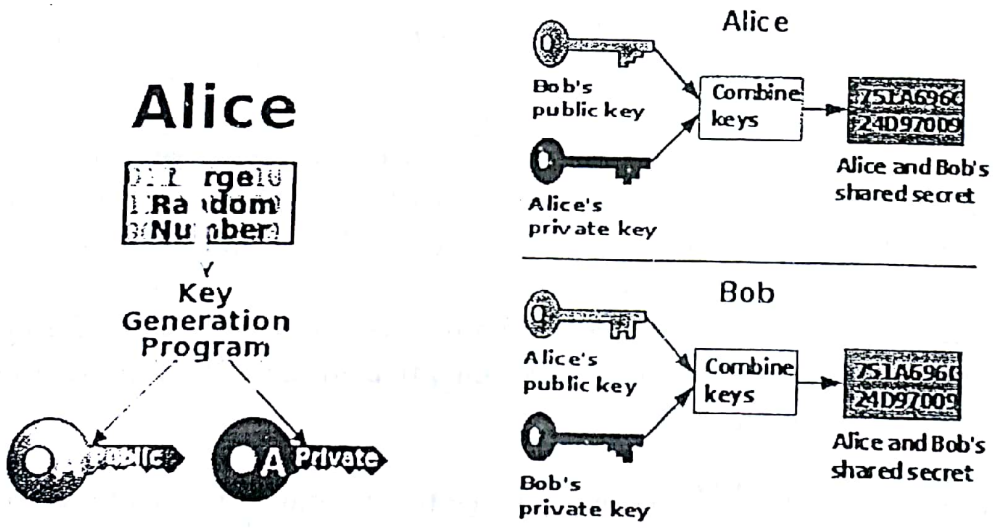
→ Asymmetric cryptography or public-key cryptography is cryptography in which a pair of keys is us to encrypt and decrypt a message so that it arrives securely. Initially, a network user receives a pub and private key pair from a certificate authority.

→ Any other user who wants to send an encrypted message can get the intended recipient's public k from a public directory. They use this key to encrypt the message, and they send it to the recipient. When the recipient gets the message, they decrypt it with their private key, which no one else shou have access to.

→ The problem with secret keys is exchanging them over the Internet or a large network while preventing them from falling into the wrong hands. Anyone who knows the secret key can decrypt th message. One answer is asymmetric encryption, in which there are two related keys--a key pair. A public key is made freely available to anyone who might want to send you a message. A second, privat key is kept secret, so that only you know it.

→Any message (text, binary files, or documents) that are encrypted by using the public key can only be decrypted by applying the same algorithm, but by using the matching private key. Any message th is encrypted by using the private key can only be decrypted by using the matching public key.

→This means that you do not have to worry about passing public keys over the Internet (the keys are supposed to be public). A problem with asymmetric encryption, however, is that it is slower than symmetric encryption. It requires far more processing power to both encrypt and decrypt the conter of the message.



19

❖ **What are some Advantages and Disadvantages of Asymmetric Encryption?**

→ Asymmetric encryption is also known as "public key encryption" because for every encryption and decryption process there are two separate keys: One that encrypts and one that decrypts. Usually, these are known as the public and private keys. You can encrypt a message with someone else's public key and only that person will be able to decrypt it because only they are in possession of their private key.

→ The two main **advantages** of asymmetric encryption are that the two parties don't need to have already shared their secret in order to communicate using encryption and that both authentication and non-repudiation are possible. (Authentication means that you can encrypt the message with my public key and only I can decrypt it with my private key. Non-repudiation means that you can "sign" the message with your private key and I can verify that it came from you with your public key.)

→ The two main **disadvantages** are that the asymmetric algorithms are comparatively complex when compared to symmetric encryption which means that messages take longer to encrypt and decrypt and that you still need to verify the authenticity of the public key, usually via some form of out-of-band communication.

Most implementations use asymmetric encryption to encode a symmetric key and transfer it to the other party. They then transmit the actual message using the symmetric key which is much more efficient in CPU time.

TOPIC : DIGITAL SIGNATURE

→ A digital code that can be attached to an electronically transmitted message that uniquely identifies the sender. Like a written signature, the purpose of a digital signature is to guarantee that the individual sending the message really is who he or she claims to be.

→ A **digital signature** is a mathematical scheme for demonstrating the authenticity of a digital message or document.

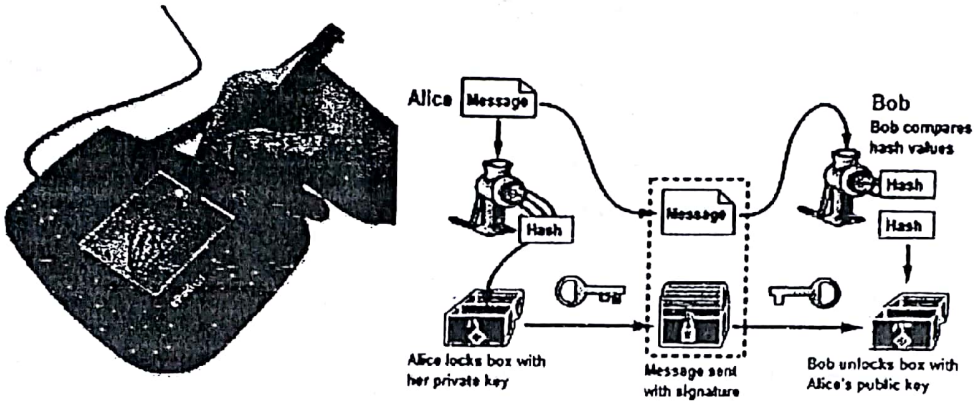
→ A **digital signature** is basically a way to ensure that an electronic document (e-mail, spreadsheet, text file, etc.) is **authentic**. Authentic means that you know who created the document and you know that it has not been altered in any way since that person created it.

→ Digital signatures rely on certain types of **encryption** to ensure authentication. Encryption is the process of taking all the data that one computer is sending to another and encoding it into a form that only the other computer will be able to decode.

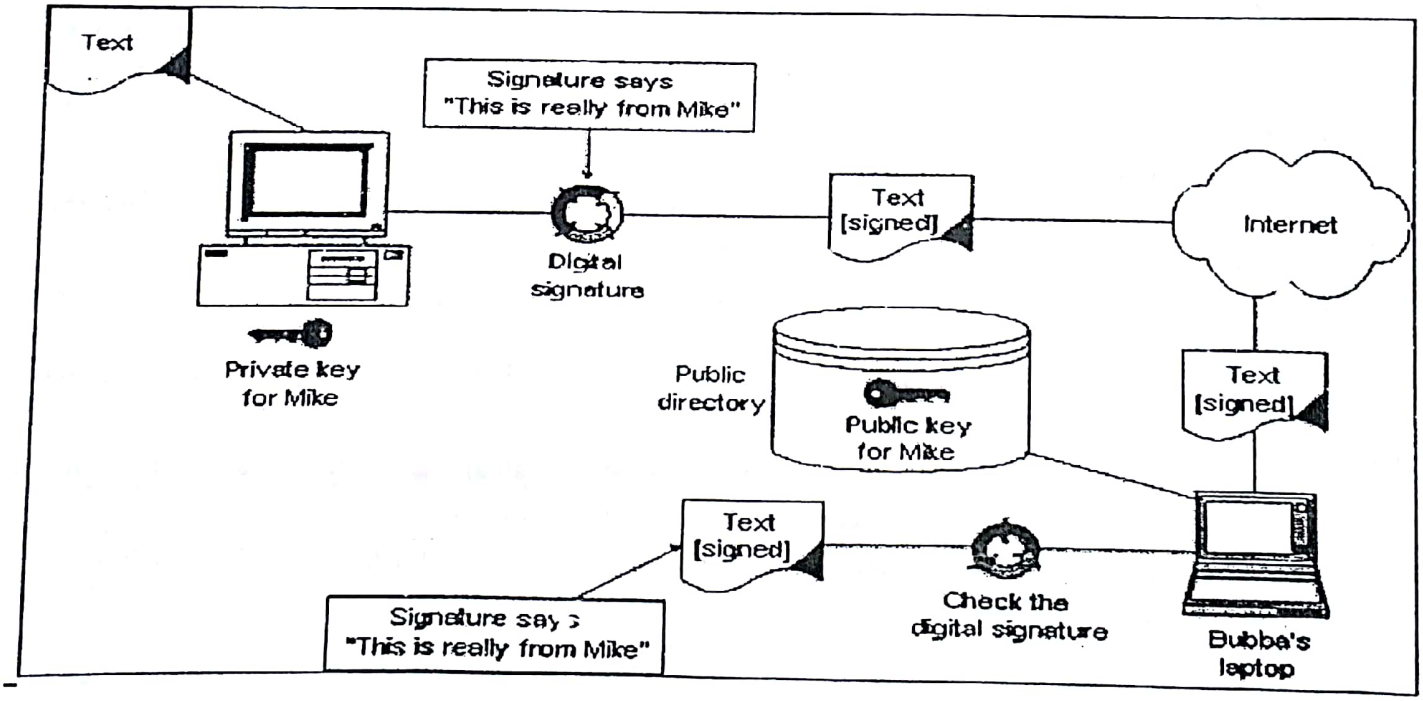
→ Authentication is the process of verifying that information is coming from a trusted source. These two processes work hand in hand for digital signatures.

→ A valid digital signature gives a recipient reason to believe that the message was created by a known sender, such that the sender cannot deny having sent the message (authentication and non-repudiation) and that the message was not altered in transit (integrity). Digital signatures are commonly used for software distribution, financial transactions, and in other cases where it is important to detect forgery or tampering.

→ Digital signatures are especially important for electronic commerce and are a key component of most authentication schemes. To be effective, digital signatures must be unforgeable. There are a number of different encryption techniques to guarantee this level of security.



ADVANTAGES AND DISADVANTAGES OF DIGITAL SIGNATURER



15

How a Digital Signature Works

If you are sending a sensitive document, you would want the recipient of the document to know that i was from you and you would also want to ensure that the document gets to the recipient in the very same state you sent it in, without any alterations. The process of digitally signing your document woul go something like this:

- First, you should copy the document and paste it into an e-mail note.
- Second, you use a special software to obtain a mathematical summary (commonly known as a message hash) of the contract.
- Thirdly, you will use a private key that you purchased from a trusted public-private key authority for encrypting the message hash.
- Lastly, you send your document with the message hash as your digital signature.

The digital signature can be used for signing any form of electronic document whether or not the message is encrypted. The digital signature is protected with a digital certificate that authenticates it. Your digital certificate will contain the certification-issuing authority's digital signature which makes it possible for anyone to verify that your certificate is real.

Advantages of Digital Signatures

The following are the main benefits of using digital signatures:

- Speed: Businesses no longer have to wait for paper documents to be sent by courier. Contracts are easily written, completed, and signed by all concerned parties in a little amount of time no matter how far the parties are geographically.
- Costs: Using postal or courier services for paper documents is much more expensive compared to using digital signatures on electronic documents.
- Security: The use of digital signatures and electronic documents reduces risks of documents being intercepted, read, destroyed, or altered while in transit.
- Authenticity: An electronic document signed with a digital signature can stand up in court just as well as any other signed paper document.
- Tracking: A digitally signed document can easily be tracked and located in a short amount of time.
- Non-Repudiation: Signing an electronic document digitally identifies you as the signatory and that cannot be later denied.
- Imposter prevention: No one else can forge your digital signature or submit an electronic document falsely claiming it was signed by you.
- Time-Stamp: By time-stamping your digital signatures, you will clearly know when the docume was signed.

Disadvantages of Digital Signatures

Just like all other electronic products, digital signatures have some disadvantages that go with them These include:

- Expiry: Digital signatures, like all technological products, are highly dependent on the technology it is based on. In this era of fast technological advancements, many of these tech products have a short shelf life.
- Certificates: In order to effectively use digital signatures, both senders and recipients may have to buy digital certificates at a cost from trusted certification authorities.
- Software: To work with digital certificates, senders and recipients have to buy verification software at a cost.
- Law: In some states and countries, laws regarding cyber and technology-based issues are weak or even non-existent. Trading in such jurisdictions becomes very risky for those who use digitally signed electronic documents.
- Compatibility: There are many different digital signature standards and most of them are incompatible with each other and this complicates the sharing of digitally signed documents.

TOPIC : PUBLIC KEY

- DEFINATION

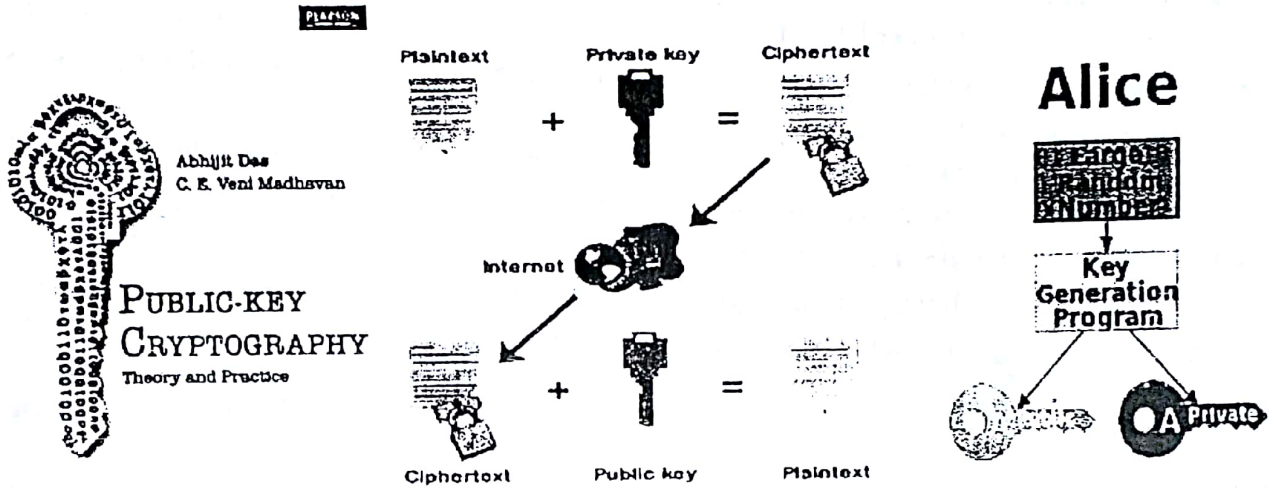
→ A cryptographic system that uses two keys -- a *public key* known to everyone and a *private* or *secret key* known only to the recipient of the message.

→ EXAMPLE : When John wants to send a secure message to Jane, he uses Jane's public key to encrypt the message. Jane then uses her private key to decrypt it.

→ An importa t element to the public key system is that the public and private keys are related in such a way thc only the public key can be used to encrypt messages and only the corresponding private key can be used to decrypt them. Moreover, it is virtually impossible to deduce the private k if you know the public key.

→ Public-key systems, such as Pretty Good Privacy (PGP), are becoming popular for transmitting information via the Internet. They are extremely secure and relatively simple to use. The only difficulty with public-key systems is that you need to know the recipient's public key to encrypt a message for him or her. What's needed, therefore, is a global registry of public keys, which is one the promises of the new LDAP technology.

Public key cryptography was invented in 1976 by Whitfield Diffie and Martin Hellman. For this reason it is sometime called *Diffie-Hellman encryption.* It is also called *asymmetric encryption* because it uses two keys instead of one key (*symmetric encryption*).



EXTRA :What are the advantages and disadvantages of public-key cryptography over secret-key cryptography?

→ The primary <u>advantage</u> of public-key cryptography is increased security: the private keys do not ever need to be transmitted or revealed to anyone. In a secret-key system, by contrast, there is always a chance that an enemy could discover the secret key while it is being transmitted.

→ Another major advantage of public-key systems is that they can provide a method for digital signatures. Authentication via secret-key systems requires the sharing of some secret and sometimes requires trust of a third party as well.

→A sender can then repudiate a previously signed message by claiming that the shared secret was somehow compromised by one of the parties sharing the secret.

→ For example, the Kerberos secret-key authentication system involves a central database that keeps copies of the secret keys of all users; a Kerberos-authenticated message would most likely not be legally binding, since an attack on the database would allow widespread forgery.

→ Public-key authentication, on the other hand, prevents this type of repudiation; each user has sol responsibility for protecting his or her private key. This property of public-key authentication is oft called non-repudiation.

→ Furthermore, digitally signed messages can be proved authentic to a third party, such as a judge, thus allowing such messages to be legally binding. Secret-key authentication systems such as Kerber were designed to authenticate access to network resources, rather than to authenticate documents task which is better achieved via digital signatures.

→ A disadvantage of using public-key cryptography for encryption is speed: there are popular secre key encryption methods which are significantly faster than any currently available public-key encryption method. But public-key cryptography can share the burden with secret-key cryptography to get the best of both worlds.

→ For encryption, the best solution is to combine public- and secret-key systems in order to get bot the security advantages of public-key systems and the speed advantages of secret-key systems. Th public-key system can be used to encrypt a secret key which is then used to encrypt the bulk of a fi or message.

TOPIC : PRIVATE KEY (or secret-key)

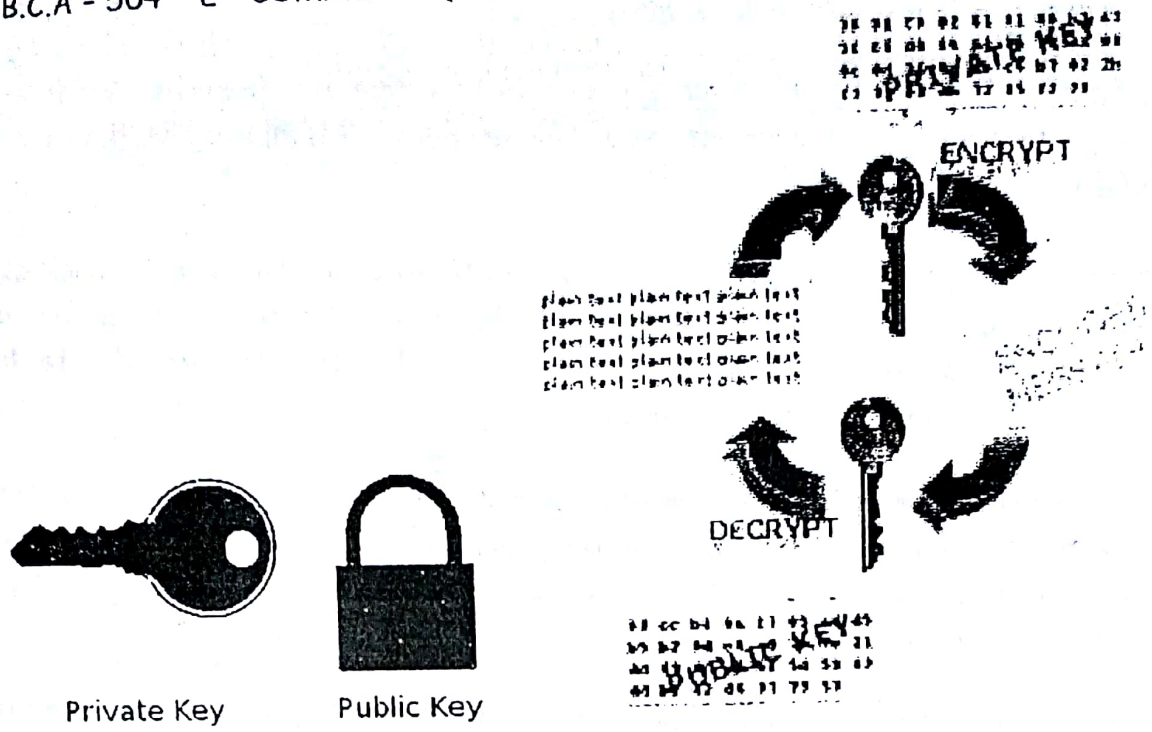
DEFINATION:

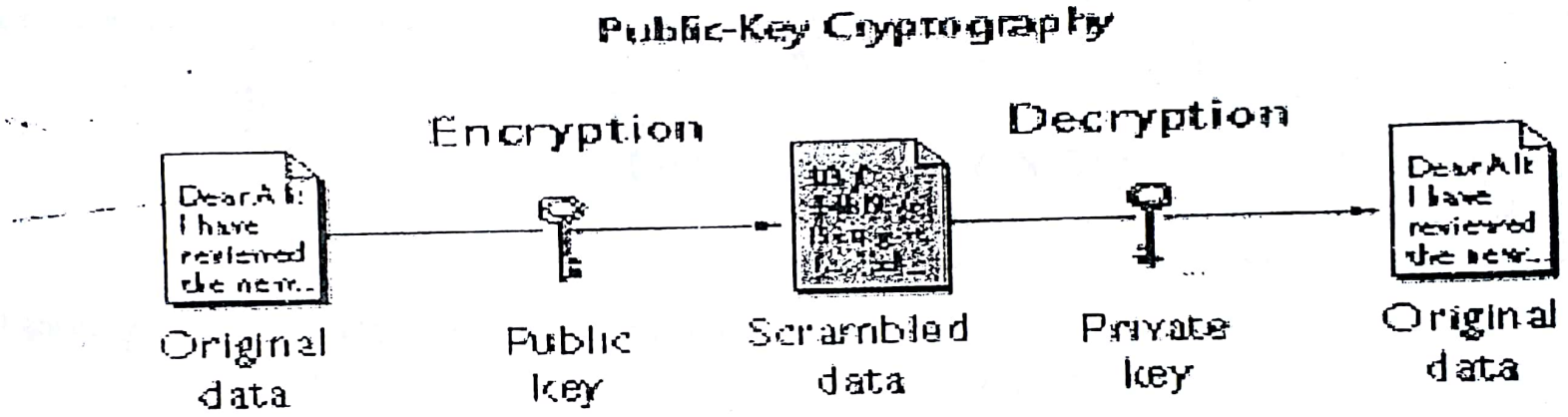
→ A way of keeping Internet messages secret in which a single key (=letter or number) changes the message into code and back again.

→In cryptography, a private or secret key is an encryption/decryption key known only to the party c parties that exchange secret messages. In traditional secret key cryptography, a key would be shar by the communicators so that each could encrypt and decrypt messages. The risk in this system is that if either party loses the key or it is stolen, the system is broken.

→ A more recent alternative is to use a combination of public and private keys. In this system, a pub key is used together with a private key. See public key infrastructure (PKI) for more information.

17





Privacy with secret key

- Data encryption standard (DNS)
  - Advantage
    - Efficiency
    - They are very good candidates for long messages.
  - Disadvantages
    - Each pair of user must have a secret key.

      N people ◊ N(N-1)/2 secrete keys

    - The distribution of the keys between two parties can be difficult.