CHAPTER – 1

INTRODUCTION

1.1 CRYPTOGRAPHY

Cryptography is an art and science of hiding existing data from the external environment. This is an approach of converting the original message into coded and scrambled message and converting it back to its original form. It can be done by using either a simple algorithm or a complex algorithm. Cryptography involves techniques to hide valuable information from malicious users. It should be remembered that cryptography is a method to alter the information in such a manner that unintended users are not able to access the original content of the information.

Cryptography allows information to be sent in a secure form in such a manner that only authorized person is able to retrieve the original information. It is associated closely with information theory, data security, and communication engineering. It hides the information from unauthorized users over the vulnerable communication channel. However, hiding the information cannot be considered as cryptography. For example, uploading personal files to Google Drive or Drop box whose password is known to you only can’t be considered as an act of cryptography.

Cryptography is used in many applications present in scientifically advanced societies; examples include the security of bank debit cards and credit cards, computer passwords, and electronic transactions, all of which are based on cryptography. It is significant for various applications and tools used in our daily lives such as folder zipping software, PDF reader, web browser, word processor and many more.

The original message being transmitted is known as plaintext. It is then converted into a coded message using a cryptographic algorithm. This process is called encryption. An encrypted or coded message is known as ciphertext, and is converted back into plaintext by the process of decryption. The process of decryption is the same as that for encryption but performed in reverse direction.
Cryptography is broadly classified into two types such as symmetric-key cryptography that uses a single key for encryption and decryption which both the sender and recipient are supposed to know, and public-key cryptography that uses two keys namely public key for encryption and private key for the decryption purpose where the public key is known to everyone but private key is known only to the recipient of the messages.

1.2 PLAINTEXT AND CIPHERTEXT

All original messages and readable data which are yet to be coded or scrambled are called ‘plaintext’. In the past, when the science of scrambling information started, it was usually the plaintext that needed to be transmitted securely from one place to another place. In the current scenario, for secure data communication we are not just limited to text only; images, programs, documents and videos are also to be transferred securely.
All scrambled and coded data which are produced as an output of the conversion process, which is done in plaintext using any encryption algorithm. The algorithm which is used to encrypt data is often known as ‘cipher’. The ciphertext is a result of the encryption process, which has no useful meaning for the intruder or unauthorized user when they access it in an unauthorized manner. The ciphertext is not limited to text only; it also includes images, symbols, audios and videos.

1.3 ENCRYPTION

Encryption is a process to convert the plaintext into ciphertext. Besides other networks it also provides the security for data communication over unreliable wireless network. It works as an interface between plaintext and ciphertext. It converts the original message into a scrambled message using an encryption algorithm. In the present scenerio, this technique is widely used to provide security for communicating confidential information from the source point to the destination point. The encryption process is generally performed by the sender at the source point.

To perform the encryption operation, three main components participate in different forms which are follows as:

a) Plaintext as an input.
b) Encryption Algorithm as an interface.
c) Ciphertext as an output or desired result.

1.4 DECRYPTION

Decryption is a process to convert the ciphertext into plaintext. This process is similar to the encryption process but it is performed in reverse order. It works as an interface between ciphertext and plaintext. It converts the scrambled or coded message into the original message using a decryption algorithm. It always occurs along with the encryption process to complete the successful communication between a sender and a receiver. The decryption process is generally performed by the receiver at the destination end to get the original message.
In the decryption operation, three components participate in different forms which are follows as:

a) Ciphertext as an input.
b) Decryption Algorithm as an interface.
c) Plaintext as an output or desired result.

![Encryption-Decryption Process](image)

**Fig. 1.3: Encryption–Decryption Process**

### 1.5 ENCRYPTION KEY

Encryption requires two things – an encryption algorithm and at least one encryption key. The encryption algorithm converts the plaintext into ciphertext based on the encryption key as well as the algorithm. If we change the algorithm or key, output will be different. For example, in the alphabet rotation system, the algorithm is taken as ‘Forward Rotation’ and the ‘Encryption Key’ is ‘1’, if we change the rotation of alphabet by 2 places, the key will be considered as ‘2’ and we will see that the results are totally different in comparison of previous case.

Here, it is noteworthy that, if we change the algorithm from forward rotation to backward rotation and allow the key to be the same, the resulting output is different. Similarly, when algorithm and key are changed simultaneously, we also get the different results.

### 1.6 STREAM CIPHER AND BLOCK CIPHER

A stream cipher is a symmetric cipher where bits of plaintext are combined with the bit stream of pseudorandom cipher bit stream using an XOR (eXclusive-OR) operation. In the stream cipher, the plaintext is encrypted one at a time, and the conversion of successive texts varies during the
encryption. This is also called as a state cipher, as the encryption of each text is dependent on the current state. In practice, the texts are in binary format typically single bits or bytes.

Block cipher is a symmetric key cipher which operates on fixed length groups of bits, termed as blocks, with an identical transformation. During the encryption, a block cipher might take a 64-bit block of plaintext as input, and corresponding 64-bit block of ciphertext as output. The exact transformation or conversion is controlled using a second input as a secret key. Decryption is performed in a similar manner. During the decryption, a 64-bit block of ciphertext together with the secret key is taken which produces the original 64-bit block of plaintext.

![Stream Cipher vs Block Cipher](image)

**Fig. 1.4: Stream Cipher and Block Cipher**

While both ciphers are symmetric key ciphers, stream ciphers depend on generating an "infinite" cryptographic bit stream to encrypt one bit or byte at a time similar to the one-time pad, whereas block ciphers work on larger chunks of data blocks at a time.

- Stream ciphers are normally faster than block ciphers, but that has its own price.
- Block ciphers generally require more memory, since they work on larger chunks of data blocks and regularly have "carry over" from previous blocks, whereas stream ciphers work on merely a few bits at a time having low memory requirements, and therefore, it is cheaper to implement in restricted scenarios such as embedded systems, firmware, and hardware.
Stream ciphers are more difficult to implement in the correct manner, and prone to weaknesses based on procedure because the principles of stream ciphers are similar to one-time pad where the bit stream has very limited requirements.

As block ciphers encrypt the whole block at a time, they are more vulnerable to noise during transmission, where if we mess up one part of the data, the rest part of the data is probably unrecoverable. But in stream ciphers, bytes are individually encrypted and have no connection with other chunks of data.

Stream ciphers do not provide protection as integrity or authentication, whereas in some cases block ciphers can provide the protection as integrity along with confidentiality.

Thus stream ciphers are generally best for the cases where the size of data is either continuous or unknown. On the other hand, Block ciphers are more useful when we know the amount of data in advance.

**1.7 CRYPTOGRAPHIC ALGORITHM**

The algorithm which is used to encrypt and decrypt the message is called ‘Cryptographic Algorithm’. The cryptographic algorithm always occurs in pair. The first algorithm is responsible for the encryption where the plaintext is converted into ciphertext while another is used for the decryption where the ciphertext is converted back to plaintext. In many cases like symmetric cryptography, the decryption algorithm is the reverse of the encryption algorithm where all operations are performed in reverse order. Due to this reason, in many research works, both algorithms are referred as a single algorithm only.

**1.8 OPERATIONS USED BY THE CRYPTOGRAPHIC ALGORITHM**

The methods of data encryption and decryption have been changed due to the introduction of computer system. There are some basic operations such as substitution, transposition and XOR (eXclusive OR) operation which can be carried out by cryptographic algorithm on the chosen plaintext. Earlier, these operations were carried out with the alphabet, but nowadays they are carried out with binary data.
A. Substitution

Substitution operations replace bits of the plaintext with other bits as per the algorithm specification, to produce the ciphertext. This substitution has to be reversed at the destination side to get the plaintext from the ciphertext. This substitution can be made complicated as per our need and nature of data. For example, one plaintext character is substituted by a character of corresponding position in the running cipher.

Fig. 1.5: Substitution Operation

B. Transposition

The transposition operation does not alter any bit of the plaintext, but changes the bit position around within it. If numbers of transposition operations are performed on the resultant ciphertext, the end result is more secured.

Fig. 1.6: Transposition Operation

C. XOR (eXclusive OR)

XOR stands for exclusive-OR operation. In Boolean algebra, it is a binary operator such that if one of two bits is true then the result is true and if both bits are true or false then the result is false.
In current scenario, XOR function is widely used in many commercial applications to provide adequate security. It plays a vital role in many advanced cryptographic algorithms to make it more complex and secure.

**1.9 STRENGTH OF CRYPTOGRAPHIC ALGORITHM**

The strength of cryptographic algorithm depends on the complexity of the algorithm. If we take strong key and weak algorithm then result will come out as weak cipher. So, it can be concluded that the strength of the cryptographic algorithm is directly proportional to the complexity of the algorithm.

For example, if we are rotate the English alphabet by n number of times where n is treated as key and if we assume n=100 or 1000, it wouldn’t matter much because there is a limit of its effectiveness as 25 times because on 26th times first letter will be repeated and it will come back to the same position. So, this algorithm is not secure.
The strength of the cryptographic algorithm is mainly dependent on the following critical factors.

a) **Computational Power**

If we are aware of the ciphertext and the encryption algorithm we need to know the encryption key to break the ciphertext. To figure out the encryption key, brute force attack is the only way which would use the encryption algorithm with every possible input as an encryption key. If the encryption algorithm is weak then it is easy to implement the brute force attack to break the ciphertext. These days most of the encryption algorithms are computationally exhaustive and require more computational power.

b) **Computational Time**

Apart from using a strong encryption algorithm and huge computational power, it would take time to decrypt the ciphertext. Here, we can consider DES as an example, which was developed in the 1970s and became a standard due to its popularity. Initially, it was said to be a very strong encryption algorithm. Today, many drawbacks of this algorithm have been pointed out and it has been proven susceptible. Computational time, if it is small, is an interesting factor which makes
the algorithm useless and vulnerable. Many algorithms became vulnerable due to further research in the concerned area and increase in computational capabilities.

c) Encryption Key

In modern cryptography, strong encryption key is required to enhance the strength of cryptographic algorithm. RSA is a cryptographic algorithm that depends on the difficulty of finding the factors of the product of two prime numbers as a key. RSA is vulnerable if we find out the factors of the product of small prime numbers. RSA is still widely used because we can increase the strength of an algorithm by using a larger number of the key, which is very difficult to crack. The use of this encryption algorithm exhibits the importance of strong encryption keys for enhancing more security.

1.10 CONVENTIONAL ENCRYPTION TECHNIQUES

A conventional encryption technique has following five ingredients:

- Plaintext: This is the original understandable message or data that is fed into the algorithm as input.
- Encryption Algorithm: The encryption algorithm performs various substitution and transformation operations on the plaintext.
- Encryption Key: An encryption key is also input to the encryption algorithm. The key is independent of the plaintext and of the encryption algorithm. The encryption algorithm will produce different results depending on the key being used. The exact substitution and transformation operations performed by the algorithm depend on the encryption key.
- Ciphertext: This is the coded and scrambled message produced as output. It depends on the plaintext and the encryption key. For any given message, two different keys will generate two different ciphertext. The ciphertext is an actual random stream of data and has no meaning.
- Decryption algorithm: This is basically the encryption algorithm run in reverse order. It takes the ciphertext and the encryption key and produces the original plaintext.
1.10.1 Caesar Cipher Technique

This is an earlier known simplest substitution encryption technique which was developed by Julius Caesar. The Caesar cipher technique involves replacing each letter of the plaintext into another successive letter which is determined by the key.

For example,

Plaintext: I AM A RESEARCH SCHOLAR

Key: 3

Ciphertext: L DP D UHVHDFK VFKRODU

Note that the letter following Z is A and so on. For above mentioned example, we can define the conversion by listing all possibilities as follows:

Ciphertext: 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

Let us assign an equivalent numerical value as a key to each letter of plaintext:

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
<th>g</th>
<th>h</th>
<th>i</th>
<th>j</th>
<th>k</th>
<th>l</th>
<th>m</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
</tr>
</tbody>
</table>
Then for each plaintext letter \( P \), substituting the ciphertext letter \( C \). This algorithm can be expressed as follows:

\[
C = E(K, P) = (P + K) \mod 26
\]

A shift may be of any amount, where \( k \) takes on a value in the range 1 to 25.

The decryption algorithm is simply illustrated as:

\[
P = D(K, C) = (C - K) \mod 26
\]

If it is known that a given ciphertext is produced by Caesar cipher, then a brute-force attack is easily performed by simply trying all the 25 possible keys.

Three important characteristics of such problem enable us to use a brute-force attack:

1. The encryption and decryption algorithms are known.
2. There are only 25 keys to analyse.
3. The language of the plaintext is well known and easily identifiable.

### 1.10.2 Monoalphabetic Technique

A monoalphabetic substitution technique is one where a letter of plaintext always produces the same letter of ciphertext. The operation is similar to the Caesar Cipher technique, with the exception that the cipher alphabet does not have the order.

An example of a monoalphabetic substitution technique is illustrated below:

**PLAINTEXT:**  ABCDEFGHIJKLMNOPQRSTUVWXYZ

**CIPHERTEXT:** QRSKOWEIPLTUAYACZMNVDHFGXJB
So, we can encrypt the message “HELLO” as “IOUUC” by using the above mentioned substitution chart.

These more advanced encryption techniques include looking thoroughly at the position of letters in words in order to identify pattern words, and looking at the letter frequencies, though common pairings (TH, HE etc.) may come up.

1.10.3 Polyalphabetic Technique

One of the main problems with simple substitution cipher technique is that they are very susceptible to frequency analysis. If a sufficiently large ciphertext is given, it can easily be analyzed by mapping the frequency of its letters to the known frequencies of plaintext. Therefore, to make ciphers more secure, researchers have long been concerned with developing encryption techniques that are invulnerable to frequency analysis.

A polyalphabetic substitution cipher technique involves the use of two or more cipher alphabets. Instead of one-to-one relationship between each letter of plaintext and its substitute, there is a one-to-many relationship between each letter of plaintext and its substitutes.

The Vigenere Table

The Vigenere Cipher Table, proposed by Blaise de Vigenere from France in the sixteenth century, is a polyalphabetic substitution technique based on the following table:

|       | 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 |
| 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 |
It should be remembered that each row of the table corresponds to a Caesar Cipher. The first row is a shift of 0; the second is a shift of 1; and the last is a shift of 25.

The Vigenere cipher uses Vigenere table together with a keyword as a key to encrypt a message. For example, suppose we want to encrypt the plaintext message given as:

Plaintext: TO BE OR NOT TO BE THAT IS THE QUESTION

Using the keyword RELATIONS, we begin by writing the keyword, repeated as many times as necessary, above the plaintext message.

To derive the ciphertext using the Vigenere table, for each letter in the plaintext, one finds the intersection of the row given by the corresponding keyword letter and the column given by the plaintext letter to find out the ciphertext letter.

Keyword: RELATI ONSR ELATI ONSR ELATI ONSR ELATI ONS
Plaintext: TOBEO RNOTT OBETH ATIST HEQUE STION
Ciphertext: KSMEH ZBBLK SMEMP OGAJX SEJCS FLZSY
Decryption of ciphertext is equally simple. We write the keyword repeatedly above the message:

Keyword: RELAT IONS R ELATI ONS RELATI ONS REL
Ciphertext: KSMEH ZBBLK SMEMP OGAJX SEJCS FLZSY
Plaintext: TOBEO RNOTT OBETH ATIST HEQUE STION

Here, we use the keyword letter to pick a column of the table and then find out the column to the row containing the ciphertext letter. The index of that row is the letter of plaintext.

1.10.4 Hill Cipher Technique

Hill Cipher technique was proposed by the mathematician Lester Hill in 1929 [FLUH02]. This encryption algorithm takes N successive plaintext letters and substitutes for them N ciphertext letters. The substitution operation is determined by N linear equations in which each character is assigned a numerical value (a = 0, b = 1 ... z = 25). For N = 3, the system can be described as:

\[ c_1 = (k_{11}p_1 + k_{12}p_2 + k_{13}p_3) \mod 26 \]
\[ c_2 = (k_{21}p_1 + k_{22}p_2 + k_{23}p_3) \mod 26 \]
\[ c_3 = (k_{31}p_1 + k_{32}p_2 + k_{33}p_3) \mod 26 \]

This can be expressed in term of column vectors and matrices:

\[
\begin{bmatrix}
  c_1 \\ c_2 \\ c_3
\end{bmatrix} =
\begin{bmatrix}
  k_{11} & k_{12} & k_{13} \\
  k_{21} & k_{22} & k_{23} \\
  k_{31} & k_{32} & k_{33}
\end{bmatrix}
\begin{bmatrix}
  p_1 \\ p_2 \\ p_3
\end{bmatrix} \mod 26
\]

or

\[ C = KP \mod 26 \]

where P and C are column vectors of length 3, representing the plaintext and ciphertext, and K is a 3 x 3 matrix, representing the encryption key. Operations are performed mod 26.
For example, consider the plaintext "paymoremoney" and use the encryption key

\[ K = \begin{pmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{pmatrix} \]

The first three letters of the plaintext are represented in the vector form as

\[ \begin{pmatrix} 15 \\ 0 \\ 24 \end{pmatrix} \]. Then \[ K \begin{pmatrix} 15 \\ 0 \\ 24 \end{pmatrix} = \begin{pmatrix} 375 \\ 819 \\ 486 \end{pmatrix} \mod 26 = \begin{pmatrix} 11 \\ 13 \\ 18 \end{pmatrix} = \text{LNS}. \] Continuing in this fashion, the ciphertext for the entire plaintext is LNSHDLEWMTRW.

Decryption requires using the inverse of the key matrix \( K \). The inverse \( K^{-1} \) of a matrix \( K \) is defined by the equation \( K K^{-1} = K^{-1} K = I \), where \( I \) is the identity matrix that contains all zeros except for ones along the main diagonal from upper left to lower right. The inverse of a matrix does not always exist, but when it exists, it satisfies the previous equation. In this case, the inverse is:

\[ K^{-1} = \begin{pmatrix} 4 & 9 & 15 \\ 15 & 17 & 6 \\ 24 & 0 & 17 \end{pmatrix} \]

This is represented as follows:

\[ \begin{pmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{pmatrix} \begin{pmatrix} 4 & 9 & 15 \\ 15 & 17 & 6 \\ 24 & 0 & 17 \end{pmatrix} = \begin{pmatrix} 443 & 442 & 442 \\ 858 & 495 & 780 \\ 494 & 52 & 365 \end{pmatrix} \mod 26 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \]

It can be easily seen that if the matrix \( K^{-1} \) is applied to the ciphertext, we can recover the plaintext.

Although the Hill cipher is strong encryption technique against a ciphertext-only attack, but it is easily broken with a known plaintext attack. For \( m \times m \) Hill cipher, suppose that we have \( m \) plaintext-ciphertext pairs, each of length \( m \). We label the pairs
unknown key matrix $K$. Now define two $m \times m$ matrices $X = (P_{ij})$ and $Y = (C_{ij})$. Then we can form the equation of matrix as $Y = KX$. If $X$ has an inverse, then we can determine $K = YX^{-1}$. If $X$ has no inverse, then a new version of $X$ can be formed with additional plaintext-ciphertext pairs until an inverse of $X$ is obtained.

Suppose that the plaintext "FRIDAY" is encrypted using a $2 \times 2$ Hill cipher to produce the ciphertext PQCFKU. Thus, we know that

$$K\begin{pmatrix}5 \\ 17\end{pmatrix} \mod 26 = \begin{pmatrix}15 \\ 16\end{pmatrix}; \quad K\begin{pmatrix}8 \\ 3\end{pmatrix} \mod 26 = \begin{pmatrix}2 \\ 5\end{pmatrix}; \quad \text{and} \quad K\begin{pmatrix}0 \\ 24\end{pmatrix} \mod 26 = \begin{pmatrix}10 \\ 20\end{pmatrix}$$

Using the first two plaintext-ciphertext pairs, we have

$$\begin{pmatrix}15 & 2 \\ 16 & 5\end{pmatrix} = K\begin{pmatrix}5 & 8 \\ 17 & 3\end{pmatrix} \mod 26$$

The inverse of $X$ can be computed as:

$$\begin{pmatrix}5 & 8 \\ 17 & 3\end{pmatrix}^{-1} = \begin{pmatrix}9 & 2 \\ 1 & 15\end{pmatrix}$$

So,

$$K = \begin{pmatrix}15 & 2 \\ 16 & 5\end{pmatrix}\begin{pmatrix}9 & 2 \\ 1 & 15\end{pmatrix} = \begin{pmatrix}137 & 60 \\ 149 & 107\end{pmatrix} \mod 26 = \begin{pmatrix}7 & 8 \\ 19 & 3\end{pmatrix}$$

This result may be verified by testing the remaining plaintext-ciphertext pair. In such a manner, we can compute the encryption key easily if plaintext and ciphertext are known and this feature reflects the vulnerability of this encryption technique.

### 1.10.5 Transposition Techniques
In cryptography, a **transposition encryption technique** describes a method by which the positions held by letters of plaintext are shifted in such a manner that the ciphertext constitutes a permutation of the plaintext. That is, the order of the plaintext letter is changed. Mathematically, a Boolean function is used on the character’s positions for the encryption and an inverse function for the decryption.

### 1.10.5.1 Rail Fence Cipher

The Rail Fence cipher is a type of transposition technique that gets its name from the method in which it is encoded. In the rail fence cipher, the letters of the plaintext are written downwards on the successive "rails" of an unreal fence, moving up and down in a sequential manner. The message is then generated in rows.

For example, using three "rails" and a plaintext message 'WE ARE DISCOVERED. FLEE AT ONCE', the ciphertext is written as:

```
```

Then ciphertext is written as:

```
WECRL TEERD SOEEO EAOCA IVDEN
```

(Hence, the ciphertext is broken into blocks of five letters to avoid the errors.)

### 1.10.5.2 Route Cipher

In a route cipher transposition technique, the plaintext is first written in a grid format of given dimensions, and then read in a pattern given in the encryption key. For example, using the same plaintext that we have used for rail fence:

```
W RiOrFeOE
E E S V E L A N J
```
The secret key might specify "spiral inwards, clockwise and starting from the top right". It produces a cipher text as “EJXCTEDECDAEWRIORFEONALEVSE”.

Route ciphers have many more options of keys than a rail fence. Actually, for messages of reasonable length, the number of possible keys is potentially too great to be uncovered even by modern technology. However, all keys are not equally good. If chosen routes are selected in bad manner, then it will leave an excessive block of plaintext, and this will give a clue for cryptanalysts to analyze the routes.

A significant variation of the route cipher was the Union Route Cipher, which was used by Union forces during the American Civil War. This worked in a different way to convert whole words instead of individual letters of plain text. This approach would leave certain highly sensitive information exposed; such words would first be masked by code. The sender may also add complete null words, which were often used to make the humorous ciphertext.

1.10.5.3 Columnar Transposition

In a columnar transposition technique, the message is written in rows of a fixed length, and then the data is read column by column which are chosen in some scrambled or random order. The keyword is generally used to define the width of the rows and the permutation of the columns. For example, the word ZEBRAS is of length 6 (so the length of the rows is 6), and the permutation is described by the alphabetical order of the letters in the keyword. In this case, the alphabetical order would be "6 3 2 4 1 5".

In a regular columnar transposition technique, any spare spaces are filled with nulls (dummy letters); in an irregular columnar transposition technique, the spaces are left blank. Finally, the message is read out in columns, in the order specified by the keyword. For example, suppose we use the keyword as ZEBRAS and the message is ‘WE ARE DISCOVERED. FLEE AT ONCE.’ In a regular columnar transposition, we can write this into the grid as:

\[
\begin{align*}
6 & 3 & 2 & 4 & 1 & 5 \\
W & E & A & R & E & D
\end{align*}
\]
Here, we are providing five nulls (QKJEU) at the end.

The ciphertext is then read out as:

EVLNE ACDTK ESEAQ ROFOJ DEECU WIREE

In the case of irregular columnar transposition, the columns are kept blank and not completed by nulls:

6 3 2 4 1 5
WEA RED
ISCO VE
RED FLE
E A T O N C
E

This results in the following ciphertext as:

EVLNA CDTES EBARF ODEEC WIREE

To decrypt it, the recipient has to calculate the column lengths by dividing the message length by the key length. Then he can write the message in columns again, and then re-order the columns by reforming the key word.

1.10.5.4 Double Transposition

A single columnar transposition technique could be attacked by guessing possible column lengths, writing the message in its columns and then looking for possible patterns. Thus, to make it stronger, a double transposition technique was often used. This is simply a columnar transposition technique which is applied twice. The same or different keys can be used for both transpositions.
For example, one can take the result of the irregular columnar transposition technique in the previous section and perform the second encryption with a different keyword as `STRIPE` which gives the permutation "564231":

```
5 6 4 2 3 1
E V L N A C
D T E S E A
R O F O D E
E C W I R E
E
```

As in previous case, this is read out columnwise to give the ciphertext as:

```
CAEEN SOIAE DRLEF WEDRE EVTOC
```

If multiple messages of the same length are encrypted using the same keys, then it can be anagrammed simultaneously. This can lead to recovery of the messages as well as recovery of the keys.

### 1.11 KEY BASED ALGORITHMS

Cryptographic algorithm can be broadly classified into symmetric-key algorithm that uses a single key that is shared between both the sender and receiver, and public-key algorithm that use two different keys, a public key which is known to everyone and a private key that only the recipient uses for communication purpose.

#### 1.11.1 Symmetric Key Algorithms

**Symmetric Key Algorithms** use a single key to perform the encryption as well as decryption of data. Any party that has the symmetric key can use it to encrypt and decrypt data. They are also referred as block ciphers. Symmetric cryptographic algorithms are typically fast and are suitable for processing large volume of data. The disadvantage of symmetric algorithm is that it presumes two parties, sender and receiver, have agreed on a single key and they have been able to exchange that secret key in a secure manner prior to communication. This is a significant
challenge to maintain the security. Symmetric algorithms are generally mixed with public key algorithms to enhance the security and speed.

With a symmetric key algorithm, the encryption key can be calculated from the decryption key and vice versa and the same key is used for both encryption and decryption processes.

As the symmetric key encryption technique uses the same key to both encrypt and decrypt the data, this encryption technique is valuable because:

- It is relatively cheap to produce a strong encryption key.
- The keys have a tendency to be much smaller for the level of security they afford.
- The algorithms are relatively cheaper to implement.

### 1.11.2 Public Key Algorithms

Public Key Algorithms refer to a cryptographic system requiring two different keys, one of which is secret key and the other is public key. Although separate, the two parts of the key pair are mathematically linked. One key encrypts or locks the plaintext and the other decrypts or unlocks the ciphertext. Any key can’t perform both functions. One of these keys is publicly available or published, while the other is kept confidential or private.

The public - key algorithms use asymmetric key algorithms such as RSA, and can also be referred as "asymmetric key algorithms." The algorithms which are used in public key cryptography are based on mathematical relationships. Although it is computationally easy for the proposed recipient to generate the pair of public and private keys, however, it is easy for the recipient to decrypt the message using the private key and for the sender to encrypt the message.
using the public key. It is extremely difficult for anyone to recover the private key, based only on their familiarity of the public key. This is the reason that unlike symmetric key algorithms, a public key algorithm does not require a secure initial exchange of one or more secret keys between the sender and receiver.

![Public Key Algorithm Diagram]

**Fig. 1.11: Public Key Algorithm**

The two main uses for public-key algorithms are:

- **Public-Key Encryption:** A message encrypted with a recipient's public key cannot be decrypted by anyone except an owner of the matching private key. It is assumed that this will be the owner of that key and the person associated with the public key used. This is used to maintain the security and confidentiality.

- **Digital Signatures:** A message signed with a sender's private key can be verified by anyone who has right to use the sender's public key, therefore by proving that the sender had access to the private key and, therefore, is likely to be the person associated with the public key used. This also ensures the integrity of the message that the message has not been tampered in the midway of communication.

### 1.12 ONE TIME PADS

The concept of one time pad was proposed by Major Joseph Mauborgne and Gilbert Bernam in year 1917 [SVAU02]. This is an unconditionally reliable and secure algorithm. The one time pad
is a non-repeating random data streams in which each letter on the pad is used once only to encrypt the one corresponding plaintext letter. After using the pad, it can’t be used again for further encryption purpose. As long as the pad is secure, the message remains secure. In this method, a random key is added with the non-random plaintext to produce completely random ciphertext. This technique is considered more secure in comparison to other encryption techniques because any guess or cryptanalysis is not possible in this algorithm. If pad is destroyed and not stored anywhere then the original message will never be recovered.

There are two major drawbacks in one time pad such as:

a) It is extremely difficult to generate truly random numbers while a pad has a non-random properties.

b) The pad can never be re-used and length of the pad must be equivalent to the length of the message. It is fine for text but practically not possible for videos.

1.13 STEGANOGRAPHY

The methods of steganography hide the existence of the message, whereas the methods of cryptography provide the message in meaningless shape to the outsiders by performing various transformations to the text. A simple and time consuming example of steganography, is one in which an arrangement of words or letters within an apparently protected text recovers the real message.

Various techniques of steganography have been used historically; some of these are:

- **Character Marking**: Chosen letters of printed text of the typewriter are overwritten in pencil. The marks are generally not visible unless the paper is held at an angle to bright light.

- **Invisible Ink**: A number of chemical substances can be used for writing but leave no visible text until heat or some chemical is applied to the paper.

- **Pin Punctures**: Small pin punctures on selected letters are normally not visible unless the paper is held up in front of a light.
- **The Typewriter Correction Ribbon**: Used between lines typed with a black ribbon, the results of typing with the correction ribbon are visible only under a strong light.

Steganography has a number of drawbacks when compared to the cryptography. It requires a lot of efforts relatively to hide a few bits of the message, although using some popular scheme which is proposed in the preceding paragraph may make it more effective. Alternatively, if a message is encrypted first and then hidden using the steganography, may produce better results.

![Steganography and Steganalysis Diagram](image)

Fig. 1.12: Steganography and Steganalysis

Steganography seeks to be strong against steganalysis, which is a challenge to uncover the hidden message within a stego-object. Figure 1.12 summarizes the process of steganalysis. Steganalysis can fight with steganography in many ways other than detecting the message, but determining how to uncover the original message is the main issue steganalysis seeks to solve.

### 1.14 CRYPTANALYSIS

Cryptanalysis is an art and science to recover the plaintext of a message from the ciphertext without access to the encryption key. Here, it is assumed that the cryptanalyst has full access to the encryption algorithm. An attempted cryptanalysis is called as an attack belonging to any one of four major types as:
a) **Known-ciphertext only:** In this attack, the cryptanalyst is aware about the ciphertext of various messages which are encrypted with the same algorithm.

b) **Known-plaintext:** Here, the cryptanalyst has access not only about the ciphertext of various messages, but also the corresponding plaintext as well.

c) **Chosen-plaintext:** In this case, specific plaintext can be chosen that may yield more information about the encryption key. This attack is more powerful to recover the key.

d) **Chosen-ciphertext:** This option facilitates the cryptanalyst to repeatedly choose ciphertext to be decrypted and access to the resulting plain text. Here, the cryptanalyst can try to discover the encryption key.

The aim of attacking an encryption system is to recover the encryption key in use rather than simply to recover the plaintext of a single ciphertext. There are two general approaches to attacking a conventional encryption scheme:

- **Cryptanalysis:** A cryptanalytic attack depends on the nature of the algorithm and some knowledge of the general characteristics of the plaintext or even some sample plaintext-ciphertext pairs. This type of attack exploits the characteristics of the encryption algorithm to attempt to figure out a specific plaintext or to figure out the encryption key being used.

- **Brute-force attack:** In this attack, the attacker guesses and tries every possible key on a piece of ciphertext until an understandable translation into plaintext is obtained. On average, half of all possible encryption keys must be tried to achieve success.

### 1.15 SECURITY OF CRYPTOGRAPHIC ALGORITHM

In cryptography, one algorithm which is totally secure against any cryptanalysis attack, is “One Time Pad”. All other cryptographic algorithms can be broken at the cost of time and computational efforts.

Modern cryptography relies on making it computationally impractical and completely unrealistic to break an algorithm. If an encryption algorithm is claimed to be perfect, then the only method
to break it is called ‘Brute Force Attack’, in which all possible key combinations will be tried to recover the original message. The field of parallel computing is well suited to perform the task of Brute Force Attack. Nowadays, a technique known as “parallel processing” is becoming increasingly popular in this area, where thousands computers are connected to each other through the internet to give high performance result. This is called ‘Distribute Computing’.

In current scenario, an encryption algorithm used to encrypt the data is secure, if computational time and computational power to break it is completely unrealistic and impractical.

1.16 MODERN CRYPTOGRAPHY

Modern cryptography is a significant discipline, which is an essential part and foundation of secure communication and information security. The study of modern cryptography touches the mathematical concepts that may have been considered mysterious and brings close various areas such as number theory, computational theory, and probability theory.

1.16.1 Data Encryption Standard (DES)

In 1975, National Bureau of Standards (NSB) of the United States made a public request to submit the proposals for a standard cryptographic algorithm. For this request, IBM created DES, which was accepted and published by NSB as a standard cryptographic algorithm. This standard cryptographic algorithm satisfied the following criteria:

a) The algorithm should provide a high level of security

b) The security should be dependent on the keys instead of secrecy of the algorithm

c) The security should be capable of being evaluated

d) The algorithm should be specific and easy to understand

e) The algorithm should be adaptable and efficient to use

f) The algorithm should be available to all users

g) The algorithm should be implementable
DES has been widely used all over the world for over 20 years, and any DES implemented system is able to communicate with any other system due to its defined standard. DES is commonly used in businesses, banks, networks and to protect the files on UNIX Operating Systems.

**Description of the Algorithm**

DES is a symmetric and block-cipher cryptographic algorithm with a block size of 64 bits and key length of 64 bits. As the algorithm is symmetric, the same algorithm and same key are used for encryption and decryption purpose.

First a transposition operation is carried out on the basis of the initial permutation, the plaintext of 64-bit block is then split into two halves of 32-bit, and in each half 16 identical operations called rounds are carried out. The two halves of 32-bit are then joined back together, and the reverse of the initial permutation is carried out in 64-bit block. In each round, only one half of the original 64-bit plaintext block is operated on. The rounds interchange between the two halves of 32-bit.

One round in DES has a number of operations as:

![Fig. 1.13: Description of Each Round in DES](image)

**Key Transformation**
The 64-bit key in DES is reduced to 56-bit by removing every 8\textsuperscript{th} bit, which are sometimes used for error checking. For 16 operations in each round sixteen different 48-bit subkeys are generated. This can be achieved by splitting the 56-bit key into two halves, and then performing left circular shift by 1 or 2 bits, depending on the round. After that, 48 of the bits are selected. Different sets of key bits are used in each subkey by shift operations. This process is called a compression permutation due to the transposition of the bits.

**Expansion Permutation**

After the process of key transformation operated, half of the block undergoes for an expansion permutation. In this operation, the expansion and transposition are carried out simultaneously by allowing the 1\textsuperscript{st} bit and 4\textsuperscript{th} bit in each 4 bit block to appear twice in the output.

![Fig. 1.14: The Expansion Permutation](image)

The expansion permutation performs 3 activities; firstly it increases the size of the half-block from 32 bits to 48 bits which is equal to the number of bits in the compressed subset of key, and is important as the next operation is to perform XOR operation on it. Secondly, it produces a longer data stream for the substitution operation that successively compresses it. Thirdly, and most importantly, because in the successive substitutions the 1\textsuperscript{st} bit and 4\textsuperscript{th} bit appear in two S-boxes, they affect two substitution operations. So, the dependencies of the output bits on the input bits increases rapidly and therefore provide the adequate security of the algorithm.

**XOR**
The XOR operation is performed on 48-bit block and appropriate subset of key for that round.

**Substitution**

The next operation is the substitution which is performed on the expanded block. There are 8 substitution boxes, called S-boxes. The first S-box operates on the first 6 bits of the 48-bit expanded block; the 2nd S-box operates on the next six, and so on. Each S-box operates from a table of 4 rows and 16 columns; a 4-bit number is assigned for each entry in the table. The 6-bit number is taken as input for S-box to look up the correct entry in the table in the following way. The 1st bit and 6th bit are combined to generate a 2-bit number corresponding to a row number, and the 2nd bit to 5th bit is combined to generate a 4-bit number corresponding to a particular column. The final result of the substitution phase is eight 4-bit blocks that are further combined into a 32-bit block.

![Fig. 1.15: The S-box Substitution](image)

It is the non-linear relationship of the S-boxes which provides the enhanced security in DES, while all other processes within the DES algorithm are linear, and are relatively easy to analyze.

**Permutation**

The 32-bit output block of the substitution phase undergoes a basic transposition using a transposition table, which is sometimes known as the P-box. Finally, after all the rounds have been completed, the two 32-bits ‘half-blocks’ are recombined to generate a 64-bit block and final
permutation is performed on it. The resulting 64-bit block is the DES encrypted ciphertext block of the 64-bit plaintext block.

**Decryption**

Decryption of DES encrypted block is very easy, if the key is known. The decryption algorithm is similar to the encryption algorithm and the only difference is that, to decrypt DES encrypted block, the subsets of the key should be used in reverse order in each round, i.e. the 16th subset of key should be used first.

**Security of DES**

Due to advancements in the area of cryptanalysis and the increased computing power, DES is no longer considered to be very safe and secure. There are a number of algorithms that can be used to reduce the number of possible keys that need to be checked, but by using highly efficient computers DES can be cracked in a matter of minutes.

If a time limit of 1 hour is given to crack a DES encrypted file, then we have to check all $2^{56}$ possible keys in one hour, which is roughly 10 trillion keys per second, which is easily manageable by today’s faster system. So, this is the conclusion that DES cannot be considered a sufficiently secure algorithm.

If a DES encrypted message can be broken in minutes by supercomputers in the current scenario, then the rapidly increasing computing power means that it will be a small matter to break DES algorithm in the future.

**1.16.2 Triple DES**

DES is susceptible to brute force attacks and therefore using Data Encryption Standard (DES) for encryption does not make sure complete security. Hence to improve the security of the algorithm, the plain text is encrypted multiple times using same algorithm but with different keys. In triple DES, the plain text is encrypted three times using the DES encryption algorithm.

**Triple DES using Two Keys:**
Chapter-1: Introduction

2014

Designing Some Multiple and Multiphase Encryption Techniques for the Enhancement of Data Security

Fig. 1.16: Triple DES with Two Keys

\[ C = E_{K1} [D_{K2} (E_{K1} (P))] \]

\[ P = D_{K1} [E_{K2} (E_{K1} (C))] \]

**Triple DES using Three Keys:**

Fig. 1.17: Triple DES with Three Keys

\[ C = E_{K3} [D_{K2} (E_{K1} (P))] \]

\[ P = D_{K3} [E_{K2} (D_{K1} (C))] \]

1.16.3 Advanced Encryption Standard (AES)

The Triple DES algorithm was recommended in year 1999 by Federal Information Processing Standard FIPS PUB 46-3 as the new standard with 168-bit key [ZIKA10]. The principal drawback of this algorithm is that this is relatively slow in software. A secondary drawback is the use of small block size of 64 bits. So, a larger block size is required due to reasons of efficiency and security.
In 1997, National Institute of Standards and Technology NIST issued a call for proposals for a new Advanced Encryption Standard (AES), which should have security strength equal to or better than 3DES, and significantly improved efficiency. In addition, NIST also specified that AES must be a symmetric block cipher with a block length of 128 bits and support for key lengths of 128, 192, and 256 bits.

In a first round of evaluation, 15 proposed algorithms were accepted. A 2\textsuperscript{nd} round narrowed to 5 algorithms. NIST completed its evaluation process and published a final standard (FIPS PUB 197) in November, 2001. NIST selected Rijndael as the proposed AES algorithm. The 2 researchers of AES are Dr. Joan Daemon and Dr. Vincent Rijmen from Belgium [USDE01].

**Working Principle of AES**

A number of AES parameters depend on the key length. In the description of this section, we assume the key length of 128 bits. The input to the encryption and decryption algorithm is a single 128-bit block; which is depicted as a square matrix of bytes. This block is copied into the State array, which is modified at each stage of encryption or decryption. After the final stage, State is copied to an output matrix.

AES is based on a design principle known as a substitution-permutation network and is fast in both software and hardware. Unlike its predecessor DES, AES does not use a Feistel network. AES is a variant of Rijndael which has a fixed block size of 128 bits, and a key size of 128, 192, or 256 bits. By contrast, the Rijndael specification *per se* is specified with a block and key sizes that may be any multiple of 32 bits, both with a minimum of 128 and a maximum of 256 bits.

AES operates on a 4\times4 column-major order matrix of bytes, termed as the state, although some versions of Rijndael have a larger block size and have additional columns in the state. Most AES calculations are done in a special finite field. The key size used for an AES cipher specifies the number of repetitions of transformation rounds that converts the input, called the plaintext, into the final output, called the ciphertext. The number of cycles of repetition are as follows:

- 10 cycles of repetition for 128 bit keys.
- 12 cycles of repetition for 192 bit keys.
- 14 cycles of repetition for 256 bit keys.
Each round consists of several processing steps, including one that depends on the encryption key itself. A set of reverse rounds is applied to transform ciphertext back into the original plaintext using the same encryption key.

![Diagram of AES Algorithm](image.png)

**Fig. 1.18: Working Architecture of AES Algorithm**
High-level description of the AES Algorithm

1. KeyExpansion—round keys are derived from the cipher key using the Rijndael's key schedule
2. Initial Round
   1. AddRoundKey—each byte of the state is combined with the round key using bitwise xor
3. Rounds
   1. SubBytes—a non-linear substitution step where each byte is replaced with another according to a lookup table.
   2. ShiftRows—a transposition step where each row of the state is shifted cyclically a certain number of steps.
   3. MixColumns—a mixing operation which operates on the columns of the state, combining the four bytes in each column.
   4. AddRoundKey
4. Final Round (no MixColumns)
   1. SubBytes
   2. ShiftRows
   3. AddRoundKey

(a) In the SubBytes step, each byte in the state is replaced with its entry in a fixed 8-bit lookup table, \( S; b_{ij} = S(a_{ij}). \)
Fig. 1.20: ShiftRows Operation

(b) In the ShiftRows step, bytes in each row of the state are shifted cyclically to the left. The number of places each byte is shifted differs for each row.

Fig. 1.21: MixColumn Operation

(c) In the MixColumns step, each column of the state is multiplied with a fixed polynomial $c(x)$.

Fig. 1.22: AddRoundKey Operation
In the AddRoundKey step, each byte of the state is combined with a byte of the round subkey using the XOR operation ($\oplus$).

### 1.16.4 IDEA

The cryptographic algorithm IDEA was proposed by Xuejia Lai and James Massey in year 1990 [ELAI11], which was also known as Proposed Encryption Standard (PES). In year 1991, Lai and Massey protected the algorithm against differential cryptanalysis and result was called as Improved PES (IPES). The name of Improved PES was changed to International Data Encryption Algorithm (IDEA) in year 1992. IDEA is a symmetric block-cipher algorithm with a key size of 128 bits and block size of 64 bits using DES algorithm to perform the encryption and decryption operations.

![Fig. 1.23: IDEA Operational Architecture](image_url)
In IDEA, 52 subkeys are used in 8 rounds. Each round uses 6 subkeys and remaining 4 subkeys are used for the output transformation. The subkeys are created as follows:

Firstly, the 128-bit key is divided into eight 16-bit keys to provide the first eight subkeys. The bits of the original key are then shifted 25 bits to the left, and then it is again split into 8 subkeys. This process of shifting and then splitting is repeated until all 52 subkeys (SK1 to SK52) have been generated.

![Fig. 1.24: IDEA Encryption and Decryption](image)

The 64-bit plaintext block is firstly split into four (B1-B4), a round then consists of the following steps:

- OB1 = B1 * SK1 (multiply 1st sub-block with 1st subkey), where OB is Output Block.
- OB2 = B2 + SK2 (add 2nd sub-block to 2nd subkey)
- OB3 = B3 + SK3
- OB4 = B4 * SK4 (multiply 3rd sub-block with 3rd subkey)
OB5 = OB1 XOR OB3 (XOR results of steps 1 and 3)

OB6 = OB2 XOR OB4

OB7 = OB5 * SK5 (multiply result of step 5 with 5th subkey)

OB8 = OB6 + OB7 (add results of steps 5 and 7)

OB9 = OB8 * SK6 (multiply result of step 8 with 6th subkey)

OB10 = OB7 + OB9

OB11 = OB1 XOR OB9 (XOR results of steps 1 and 9)

OB12 = OB3 XOR OB9

OB13 = OB2 XOR OB10

OB14 = OB4 XOR OB10

The input to the next round is the 4 sub-blocks as OB11, OB13, OB12, and OB14. After the eighth round, the 4 final output blocks (F1-F4) are used in a final transformation to produce 4 sub-blocks of ciphertext (C1-C4) that are then reconnected to form the final 64-bit block of ciphertext.

C1 = F1 * SK49

C2 = F2 + SK50

C3 = F3 + SK51

C4 = F4 * SK52

Ciphertext = C1 & C2 & C3 & C4. (where, ‘&’ operator is used for bit’s concatenation)

Security of IDEA

IDEA is approximately double as fast as DES and it is also significantly more secure. As per brute-force approach, there are $2^{128}$ possible keys. If a billion processor chips that could each test 1 billion keys a second were used to analyze and crack an IDEA-encrypted message, it would
take $10^{13}$ years which is considerably longer than the age of the universe. In the current scenario, it is possibly a better attack than brute-force. In future, with the design of much more powerful machines, it may be possible to crack the original message. However, for the longer time to come, IDEA seems to be an extremely secure algorithm.

1.16.5 RSA Algorithm

RSA is a public-key cryptographic algorithm that is based on the factoring problem as the factorization of large prime integers. RSA algorithm was developed by three scientists Ron Rivest, Adi Shamir and Leonard Adleman. This algorithm was first publicly described in the year 1977 [GEMM97]. It consists of following modules.

1. **Key Generation:**

RSA algorithm involves a **public key** and a **private key**. The public key can be known to the public and is used for encrypting the messages. Messages encrypted with the public key can only be decrypted using the private key. The public key and private key for the RSA algorithm are generated in the following way:

1. Choose two distinct prime numbers as $p$ and $q$.
   - For security purposes, the integers $p$ and $q$ should be chosen as large random numbers, and should be of similar bit-length.
2. Compute $n$ as $n = pq$.
   - $n$ is used as the modulus for both the public and private keys
3. Compute $\phi (n) = (p - 1) (q - 1)$, where $\phi$ is Euler's totient function, which counts the totatives of $n$, that is, the positive integers less than or equal to $n$ that are relatively prime to $n$.
4. Choose an integer $e$ such that $1 < e < \phi (n)$ and greatest common divisor of $(e, \phi (n)) = 1$; i.e., $e$ and $\phi (n)$ are co-prime.
   - $e$ is released as the public key exponent.
   - $e$ is having a short bit-length and small weight resulting more competent encryption. However, small values of $e$ have been exposed to be less secure in some cases.
5. Determine $d$ as:

$$e*d \equiv 1 \mod \Phi$$
i.e., \( d \) is the multiplicative inverse of \( e \mod \phi(n) \).

- This is clearly stated as solve for \( d \) given \((d^e) = 1 \mod \phi(n)\)
- This is frequently computed using the extended Euclidean algorithm.
- \( d \) is kept as the private key exponent.

By creation, \( d^e = 1 \mod \phi(n) \). The **public key** consists of the modulus \( n \) and the public (or encryption) exponent \( e \). The **private key** consists of the modulus \( n \) and the private (or decryption) exponent \( d \) which must be kept secret.

### II. Encryption:

The sender encrypts the message \( M \) as:

1. Obtain the public key of the intended receiver.
2. Convert the message \( M \) in integer in the interval \( 0 \) to \( n - 1 \).
3. Compute the cipher as \( C = M^e \mod n \) and transmit it to the intended receiver.

### III. Decryption:

The receiver decrypts the plain text from the cipher text as:

\[
P = C^d \mod n = M^{ed} \mod n
\]

Although we have to select the values of \( p \) and \( q \) which are having similar bit-length, we should not take very nearby values because if \( p \approx q \) then \( p^e \approx q^e \).

---

**Fig. 1.25:** RSA Implementation for Encryption and Decryption
1.16.6 Hybrid Cryptographic System

This is well known that even if we are not using a large size keys, RSA algorithms is very slow for encryption and decryption purposes in comparison of DES algorithm, so as a result RSA is not being widely used as an individual cryptographic system. However, this algorithm is used in many hybrid cryptographic systems such as Pretty Good Privacy (PGP). The basic concept of hybrid cryptographic systems is to encrypt plaintext with a symmetric key algorithm (usually DES or IDEA); the symmetric algorithm’s key is then itself encrypted with a public-key algorithm such as RSA. The symmetric algorithm encrypted message and the RSA-encrypted key is then tied together and then sent to the recipient who uses his private RSA algorithm’s key to decrypt the symmetric algorithm’s key, and then that key is used to decrypt the message. This is significantly faster than using RSA algorithm throughout, and allows a different symmetric key to be used each time, significantly enhancing the security of the symmetric key algorithm.

1.16.6.1 Digital Signing

A serious problem with public-key cryptography is that anyone can send a message to the user using the user’s public key, it is then necessary to authenticate the intended sender who transmitted this message. A message which is encrypted by someone’s private key, can be decrypted by any person with their public key. It means that if the sender encrypted a message with his private key, and then further encrypted the resulting ciphertext with the recipient’s public key, the recipient would be able to decrypt the first message with their private key, and then the sender’s public key, as a result recovering the original message and proving that it came from the authorized sender.

This process is very time-consuming, hence is not used widely. A much more common technique of digitally signing a message is one-way hashing which is very popular and widely used.

1.16.6.2 One-way Hash Function

A one-way hash function is a mathematical function that takes a string of message of any length and returns a smaller fixed-length string as a hash value. This functions is designed in such a way that not only it is very difficult to figure out the message from its hashed translation, but also that it is extremely hard to find out two messages that hash to the same value, even if all hash values
of certain length are known. In fact, to find out two messages with the same hash from a hash function of 128-bit, \(2^{64}\) hashes would have to be tried. In other words, the hash value of a document is a small unique ‘fingerprint’.

A hash function can be described as:

\[ H = f(M), \]

where \(H\) = Hash Value, \(f\) = Hash Function and \(M\) = Original Message String

If we know original message \(M\), we can compute \(H\) easily. However, if we know the hash value \(H\) and hash function \(f\), it is very hard to compute original message \(M\), and is confidently computationally impractical. Given two messages with same hash value then there is very less chances of collision, and it is very hard to reverse the hash value, so one-way hash function is considered as an extremely useful technique in the field of cryptography.

The use of one-way hash technique for a message, results into much smaller in length but still a unique number. This method can be used as an evidence of ownership of a message without having to disclose the contents of the original message. For example, rather than keeping a database of copyright documents, we can store only the hash values of each document which would save a lot of space as well as it would provide a great security.

In most cases, the hash functions are commonly used to digitally sign the messages. In this method, the sender implements the one-way hash on the original plaintext message, encrypts it with his private key and then encrypts both with the recipient’s public key and sends it in the usual way. On the receiving end, the recipient can use the sender’s public key to decrypt the hash value, he can then perform a one-way hash himself on the original plaintext message, and compare this with the one he has received. If the hash values are identical, the recipient confirms not only that the message is transmitted from the correct sender, as it is used their private key to encrypt the hash, but also that the original plaintext message is completely authentic as it hashes to the same value.

This method is significantly preferable to encrypting the original message with a private key, as the hash value of a message will normally be significantly smaller than the original message itself. This means that it will not considerably slow down the decryption process in the way as it
would have done when decrypting the whole message with the sender’s public key, and then decrypting it again with the recipient’s private key. The PGP system uses the message digest (MD5) hash function for exactly this purpose.

1.16.6.3 PGP: An Example of Hybrid Cryptographic System

One of the most successful achievements in the field of cryptography is a system called Pretty Good Privacy (PGP). PGP was developed by Phil Zimmerman in year 1990 [NIST10], who developed this cryptographic system for charitable reasons. In year 1991, he published this encryption system on the Internet. For the development of PGP, he stated his objectives to protect privacy and conserve confidentiality of human data assets from external environment. Now, PGP has become a genuine international standard for e-mail security.

PGP is widely used for e-mail encryption over the wireless network as the internet. PGP combines the best features of both symmetric and public key cryptography. PGP is a hybrid cryptographic system, which uses both symmetrical and asymmetrical cryptographic systems as a part of its process. Following figure describes an overview of how the various components of a PGP work together to provide e-mail security.

![Encryption and Decryption Process in PGP](image-url)
As shown in the above Fig. 1.26, PGP creates a session key, which is a unique secret key and acts as a one-time pad. This key is generated as a random number from the random movements of our mouse and the keystrokes. This session key works in a very secure manner. The plaintext message is encrypted by fast conventional encryption algorithm and the result is ciphertext. Once the message is encrypted, the session key is then encrypted by the recipient's public key. This encrypted session key is then transmitted along with the ciphertext to the recipient. At the receiving end, decryption process is performed in the reverse order. The receiver uses his or her private key to decrypt the session key and recover the original session key, which is used to decrypt the traditionally encrypted ciphertext to get the original message.

1.16.7 Elliptic Curve Cryptography (ECC)

Elliptic Curve Cryptography (ECC) is a public-key cryptography based on the mathematical structure of elliptical curve over the finite fields. The concept of Elliptic Curve Cryptography was suggested by Neal Koblitz (University of Washington) and Victor S. Miller (IBM) in year 1985 [NIST10]. The U.S. National Security Agency has approved ECC by including it in the set of recommended cryptographic algorithms and allowed its use for protecting top secret information with 384-bit keys. In year 2005, at the RSA conference, the National Security Agency (NSA) announced Suite B which exclusively uses ECC for the generation of digital signatures and key exchange. This Suite is designed to protect classified and unclassified top secret national security information and system.

In ECC, the size of the elliptic curve determines the difficulty of the problem. The primary benefit of ECC is a small key size, which reduces storage capacity and transmission requirements. An elliptic curve cryptography can provide the same level of security provided by an RSA based cryptographic system with the correspondingly larger key. For example, an elliptic curve cryptographic system with 256-bit public key can provide the same security that is provided by the RSA algorithm with 3072-bit public key.

An elliptic curve is a plane curve which consists number of points satisfying the following algebraic equation

\[ y^2 = x^3 + ax + b \]
Along with a well known point at infinity, denoted by $\infty$.

\[ y^2 = x^3 + ax + b \]

Fig. 1.27: A Simple Elliptic Curve

**Key Generation in ECC**

This is an important part where we have to generate both public key and private key. The sender will be encrypting the message with receiver’s public key and the receiver will decrypt its private key.

Using the following equation we can generate the public key

\[ Q = d \times P \]

Where, \( d \) = A random number that we can select within the range of 1 to \( n-1 \) and \( P \) is the point on the curve.

So, ‘Q’ is the public key and ‘d’ is the private key.

**Encryption**

Let ‘m’ be the message which we want to transmit. Now, we have to represent this message on the curve. Consider ‘m’ having the point ‘M’ on the curve ‘E’. Key ‘k’ is to be selected randomly from 1 to \( n-1 \).

As a result, two cipher texts will be generated as shown below:

\[ C1 = k \times P \]
C2 = M + k*Q

Now, C1 and C2 will be transmitted to the receiver.

**Decryption**

Through decryption, we can get back the original message ‘m’ from ciphertexts as,

\[ M = C2 - d * C1 \]

M is the original message which was transmitted from sender to receiver.

The security of elliptic curve cryptography (ECC) depends on the capacity to compute a point multiplication and the incapacity to compute the multiplicand to give the product and original points. The elliptic curve cryptography is considered as a secure cryptographic algorithm due to its complexity and high technical aspects. It is based on the properties of an algebraic equation produced from the mathematical group (a set of values for which arithmetic operations can be performed on any two members of the group to generate a third member), derived from points where the line intersects the axes. Multiplying a point on the curve by a number will produce another point on the curve, but it is very difficult to find what number was used, even if you know the original point and the result.