



# CRYPTOGRAPHY

*just for beginners*

**tutorialspoint**  
SIMPLY EASY LEARNING

[www.tutorialspoint.com](http://www.tutorialspoint.com)



<https://www.facebook.com/tutorialspointindia>



<https://twitter.com/tutorialspoint>

## **About the Tutorial**

---

This tutorial covers the basics of the science of cryptography. It explains how programmers and network professionals can use cryptography to maintain the privacy of computer data. Starting with the origins of cryptography, it moves on to explain cryptosystems, various traditional and modern ciphers, public key encryption, data integration, message authentication, and digital signatures.

## **Audience**

---

This tutorial is meant for students of computer science who aspire to learn the basics of cryptography. It will be useful for networking professionals as well who would like to incorporate various cryptographic algorithms to ensure secure data communication over their networks.

## **Prerequisites**

---

This tutorial has been prepared with the view to make it useful for almost anyone who is curious about cryptography. A basic knowledge of computer science and a secondary level of mathematics knowledge is sufficient to make the most of this tutorial.

## **Disclaimer & Copyright**

---

© Copyright 2015 by Tutorials Point (I) Pvt. Ltd.

All the content and graphics published in this e-book are the property of Tutorials Point (I) Pvt. Ltd. The user of this e-book is prohibited to reuse, retain, copy, distribute or republish any contents or a part of contents of this e-book in any manner without written consent of the publisher. We strive to update the contents of our website and tutorials as timely and as precisely as possible, however, the contents may contain inaccuracies or errors. Tutorials Point (I) Pvt. Ltd. provides no guarantee regarding the accuracy, timeliness or completeness of our website or its contents including this tutorial. If you discover any errors on our website or in this tutorial, please notify us at [contact@tutorialspoint.com](mailto:contact@tutorialspoint.com).

# Table of Contents

---

About the Tutorial.....	i
Audience .....	i
Prerequisites .....	i
Disclaimer & Copyright.....	i
Table of Contents .....	ii
 1. CRYPTOGRAPHY – ORIGIN .....	 1
History of Cryptography .....	1
Evolution of Cryptography .....	3
 2. MODERN CRYPTOGRAPHY .....	 4
Characteristics of Modern Cryptography .....	4
Context of Cryptography .....	4
Security Services of Cryptography .....	5
Cryptography Primitives.....	6
 3. CRYPTOSYSTEMS .....	 8
Components of a Cryptosystem .....	8
Types of Cryptosystems .....	9
Relation between Encryption Schemes .....	13
Kerckhoff's Principle for Cryptosystem.....	13
 4. ATTACKS ON CRYPTOSYSTEMS .....	 15
Passive Attacks.....	15
Active Attacks .....	16
Assumptions of Attacker .....	16
Cryptographic Attacks .....	18
Practicality of Attacks.....	20

5.	CRYPTOGRAPHY – TRADITIONAL CIPHERS .....	21
	Earlier Cryptographic Systems .....	21
	Caesar Cipher .....	21
	Simple Substitution Cipher .....	22
	Monoalphabetic and Polyalphabetic Cipher .....	23
	Playfair Cipher .....	24
	Vigenere Cipher .....	26
	One-Time Pad .....	27
	Transposition Cipher .....	27
6.	MODERN SYMMETRIC KEY ENCRYPTION .....	29
	Block Ciphers .....	29
	Stream Ciphers .....	29
7.	BLOCK CIPHER .....	30
	Block Size .....	30
	Padding in Block Cipher .....	31
	Block Cipher Schemes .....	31
8.	FEISTEL BLOCK CIPHER .....	32
	Encryption Process .....	32
	Decryption Process .....	33
	Number of Rounds .....	33
9.	DATA ENCRYPTION STANDARD .....	35
	Initial and Final Permutation .....	36
	Round Function .....	36
	Key Generation .....	40
	DES Analysis .....	40

10. TRIPLE DES.....	42
3-KEY Triple DES .....	42
11. ADVANCED ENCRYPTION STANDARD.....	44
Operation of AES .....	44
Encryption Process .....	45
Decryption Process.....	46
AES Analysis .....	47
12. MODES OF OPERATION .....	48
Electronic Code Book (ECB) Mode .....	48
Cipher Block Chaining (CBC) Mode .....	49
Cipher Feedback (CFB) Mode .....	50
Output Feedback (OFB) Mode .....	51
Counter (CTR) Mode.....	52
13. PUBLIC KEY ENCRYPTION .....	54
Public Key Cryptography .....	54
RSA Cryptosystem .....	55
ElGamal Cryptosystem .....	58
Elliptic Curve Cryptography (ECC) .....	61
RSA and ElGamal Schemes – A Comparison.....	62
14. DATA INTEGRITY .....	63
Threats to Data Integrity .....	63
15. HASH FUNCTIONS.....	64
Features of Hash Functions .....	64
Properties of Hash Functions.....	65
Design of Hashing Algorithms .....	66
Popular Hash Functions.....	67

Applications of Hash Functions .....	69
16. MESSAGE AUTHENTICATION .....	71
Message Authentication Code (MAC).....	71
Limitations of MAC.....	72
17. DIGITAL SIGNATURE .....	73
Model of Digital Signature.....	73
Importance of Digital Signature .....	74
Encryption with Digital Signature .....	75
18. PUBLIC KEY INFRASTRUCTURE .....	77
Key Management .....	77
Public Key Infrastructure (PKI).....	78
Digital Certificate .....	78
Certifying Authority (CA) .....	79
Hierarchy of CA .....	81
19. CRYPTOGRAPHY – BENEFITS AND DRAWBACKS.....	83
Cryptography – Benefits.....	83
Cryptography – Drawbacks .....	83
Future of Cryptography .....	84



# 1. CRYPTOGRAPHY – ORIGIN

Human being from ages had two inherent needs: (a) to communicate and share information and (b) to communicate selectively. These two needs gave rise to the art of coding the messages in such a way that only the intended people could have access to the information. Unauthorized people could not extract any information, even if the scrambled messages fell in their hand.

*The art and science of concealing the messages to introduce secrecy in information security is recognized as cryptography.*

The word 'cryptography' was coined by combining two Greek words, 'Krypto' meaning hidden and 'graphene' meaning writing.

## History of Cryptography

---

The art of cryptography is considered to be born along with the art of writing. As civilizations evolved, human beings got organized in tribes, groups, and kingdoms. This led to the emergence of ideas such as power, battles, supremacy, and politics. These ideas further fueled the natural need of people to communicate secretly with selective recipient which in turn ensured the continuous evolution of cryptography as well.

The roots of cryptography are found in Roman and Egyptian civilizations.

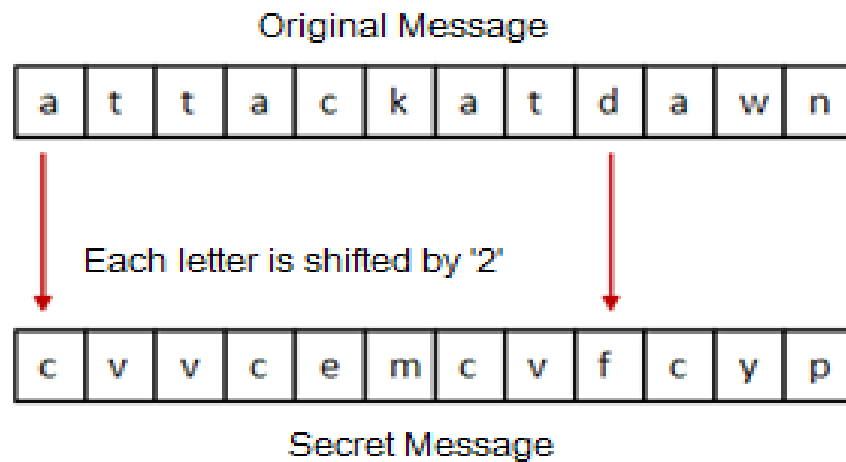
### Hieroglyph – The Oldest Cryptographic Technique

The first known evidence of cryptography can be traced to the use of 'hieroglyph'. Some 4000 years ago, the Egyptians used to communicate by messages written in hieroglyph. This code was the secret known only to the scribes who used to transmit messages on behalf of the kings. One such hieroglyph is shown below.



Later, the scholars moved on to using simple mono-alphabetic substitution ciphers during 500 to 600 BC. This involved replacing alphabets of message with other alphabets with some secret rule. This **rule** became a **key** to retrieve the message back from the garbled message.

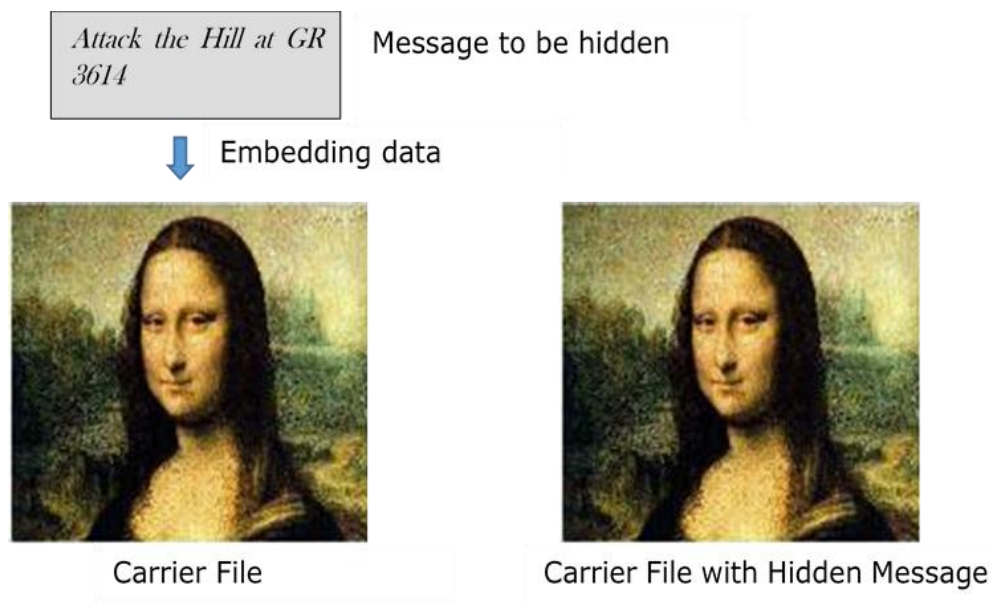
The earlier Roman method of cryptography, popularly known as the **Caesar Shift Cipher**, relies on shifting the letters of a message by an agreed number (three was a common choice), the recipient of this message would then shift the letters back by the same number and obtain the original message.



### Steganography

Steganography is similar but adds another dimension to Cryptography. In this method, people not only want to protect the secrecy of an information by concealing it, but they also want to make sure any unauthorized person gets no evidence that the information even exists. For example, **invisible watermarking**.

In steganography, an unintended recipient or an intruder is unaware of the fact that observed data contains hidden information. In cryptography, an intruder is normally aware that data is being communicated, because they can see the coded/scrambled message.





## Evolution of Cryptography

---

It is during and after the European Renaissance, various Italian and Papal states led the rapid proliferation of cryptographic techniques. Various analysis and attack techniques were researched in this era to break the secret codes.

- Improved coding techniques such as **Vigenere Coding** came into existence in the 15<sup>th</sup> century, which offered moving letters in the message with a number of variable places instead of moving them the same number of places.
- Only after the 19<sup>th</sup> century, cryptography evolved from the ad hoc approaches to encryption to the more sophisticated art and science of information security.
- In the early 20<sup>th</sup> century, the invention of mechanical and electromechanical machines, such as the **Enigma rotor machine**, provided more advanced and efficient means of coding the information.
- During the period of World War II, both **cryptography** and **cryptanalysis** became excessively mathematical.

With the advances taking place in this field, government organizations, military units, and some corporate houses started adopting the applications of cryptography. They used cryptography to guard their secrets from others. Now, the arrival of computers and the Internet has brought effective cryptography within the reach of common people.

## 2. MODERN CRYPTOGRAPHY

Modern cryptography is the cornerstone of computer and communications security. Its foundation is based on various concepts of mathematics such as number theory, computational-complexity theory, and probability theory.

### Characteristics of Modern Cryptography

---

There are three major characteristics that separate modern cryptography from the classical approach.

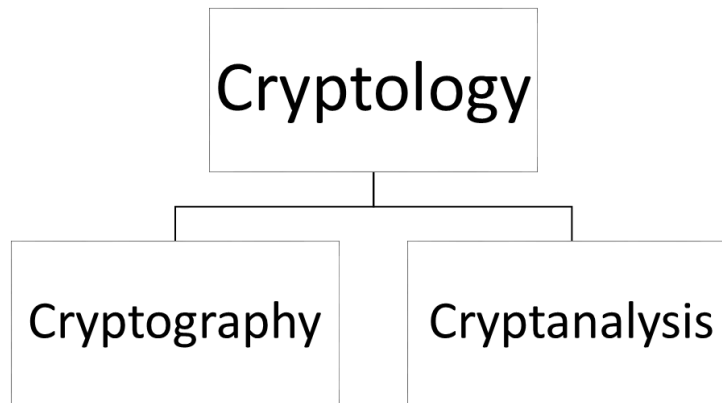
Classic Cryptography	Modern Cryptography
It manipulates traditional characters, i.e., letters and digits directly.	It operates on binary bit sequences.
It is mainly based on 'security through obscurity'. The techniques employed for coding were kept secret and only the parties involved in communication knew about them.	It relies on publicly known mathematical algorithms for coding the information. Secrecy is obtained through a secret key which is used as the seed for the algorithms. The computational difficulty of algorithms, absence of secret key, etc., make it impossible for an attacker to obtain the original information even if he knows the algorithm used for coding.
It requires the entire cryptosystem for communicating confidentially.	Modern cryptography requires parties interested in secure communication to possess the secret key only.

### Context of Cryptography

---

Cryptology, the study of cryptosystems, can be subdivided into two branches:

- Cryptography
- Cryptanalysis



### What is Cryptography?

*Cryptography is the art and science of making a cryptosystem that is capable of providing information security.*

Cryptography deals with the actual securing of digital data. It refers to the design of mechanisms based on mathematical algorithms that provide fundamental information security services. You can think of cryptography as the establishment of a large toolkit containing different techniques in security applications.

### What is Cryptanalysis?

*The art and science of breaking the cipher text is known as cryptanalysis.*

Cryptanalysis is the sister branch of cryptography and they both co-exist. The cryptographic process results in the cipher text for transmission or storage. It involves the study of cryptographic mechanism with the intention to break them. Cryptanalysis is also used during the design of the new cryptographic techniques to test their security strengths.

**Note:** Cryptography concerns with the design of cryptosystems, while cryptanalysis studies the breaking of cryptosystems.

## Security Services of Cryptography

---

The primary objective of using cryptography is to provide the following four fundamental information security services. Let us now see the possible goals intended to be fulfilled by cryptography.

### Confidentiality

Confidentiality is the fundamental security service provided by cryptography. It is a security service that keeps the information from an unauthorized person. It is sometimes referred to as **privacy** or **secrecy**.

Confidentiality can be achieved through numerous means starting from physical securing to the use of mathematical algorithms for data encryption.

### Data Integrity

It is security service that deals with identifying any alteration to the data. The data may get modified by an unauthorized entity intentionally or accidentally. Integrity service confirms that whether data is intact or not since it was last created, transmitted, or stored by an authorized user.

Data integrity cannot prevent the alteration of data, but provides a means for detecting whether data has been manipulated in an unauthorized manner.

### Authentication

Authentication provides the identification of the originator. It confirms to the receiver that the data received has been sent only by an identified and verified sender.

Authentication service has two variants:

- **Message authentication** identifies the originator of the message without any regard router or system that has sent the message.
- **Entity authentication** is assurance that data has been received from a specific entity, say a particular website.

Apart from the originator, authentication may also provide assurance about other parameters related to data such as the date and time of creation/transmission.

### Non-repudiation

It is a security service that ensures that an entity cannot refuse the ownership of a previous commitment or an action. It is an assurance that the original creator of the data cannot deny the creation or transmission of the said data to a recipient or third party.

Non-repudiation is a property that is most desirable in situations where there are chances of a dispute over the exchange of data. For example, once an order is placed electronically, a purchaser cannot deny the purchase order, if non-repudiation service was enabled in this transaction.

## Cryptography Primitives



---

Cryptography primitives are nothing but the tools and techniques in Cryptography that can be selectively used to provide a set of desired security services:

- Encryption
- Hash functions

- Message Authentication codes (MAC)
- Digital Signatures

The following table shows the primitives that can achieve a particular security service on their own.

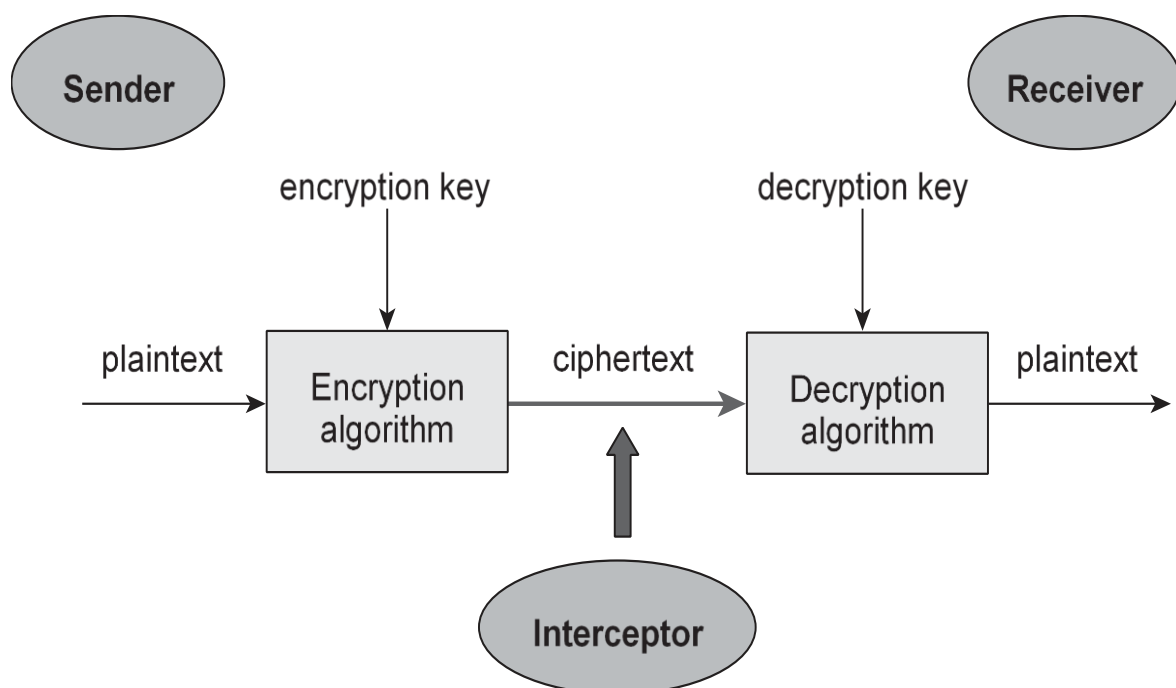
<b>Primitives Service</b>  	<b>Encryption</b>	<b>Hash Function</b>	<b>MAC</b>	<b>Digital Signature</b>
Confidentiality	Yes	No	No	No
Integrity	No	Sometimes	Yes	Yes
Authentication	No	No	Yes	Yes
Non Reputation	No	No	Sometimes	Yes

**Note:** Cryptographic primitives are intricately related and they are often combined to achieve a set of desired security services from a cryptosystem.

### 3. CRYPTOSYSTEMS

A cryptosystem is an implementation of cryptographic techniques and their accompanying infrastructure to provide information security services. A cryptosystem is also referred to as a **cipher system**.

Let us discuss a simple model of a cryptosystem that provides confidentiality to the information being transmitted. This basic model is depicted in the illustration below:



The illustration shows a sender who wants to transfer some sensitive data to a receiver in such a way that any party intercepting or eavesdropping on the communication channel cannot extract the data.

The objective of this simple cryptosystem is that at the end of the process, only the sender and the receiver will know the plaintext.

#### **Components of a Cryptosystem**

The various components of a basic cryptosystem are as follows:

- **Plaintext.** It is the data to be protected during transmission.
- **Encryption Algorithm.** It is a mathematical process that produces a ciphertext for any given plaintext and encryption key. It is a cryptographic



algorithm that takes plaintext and an encryption key as input and produces a ciphertext.

- **Ciphertext.** It is the scrambled version of the plaintext produced by the encryption algorithm using a specific the encryption key. The ciphertext is not guarded. It flows on public channel. It can be intercepted or compromised by anyone who has access to the communication channel.
- **Decryption Algorithm,** It is a mathematical process, that produces a unique plaintext for any given ciphertext and decryption key. It is a cryptographic algorithm that takes a ciphertext and a decryption key as input, and outputs a plaintext. The decryption algorithm essentially reverses the encryption algorithm and is thus closely related to it.
- **Encryption Key.** It is a value that is known to the sender. The sender inputs the encryption key into the encryption algorithm along with the plaintext in order to compute the ciphertext.
- **Decryption Key.** It is a value that is known to the receiver. The decryption key is related to the encryption key, but is not always identical to it. The receiver inputs the decryption key into the decryption algorithm along with the ciphertext in order to compute the plaintext.

For a given cryptosystem, a collection of all possible decryption keys is called a **key space**.

An **interceptor** (an attacker) is an unauthorized entity who attempts to determine the plaintext. He can see the ciphertext and may know the decryption algorithm. He, however, must never know the decryption key.

## Types of Cryptosystems

---

Fundamentally, there are two types of cryptosystems based on the manner in which encryption-decryption is carried out in the system:

- Symmetric Key Encryption
- Asymmetric Key Encryption

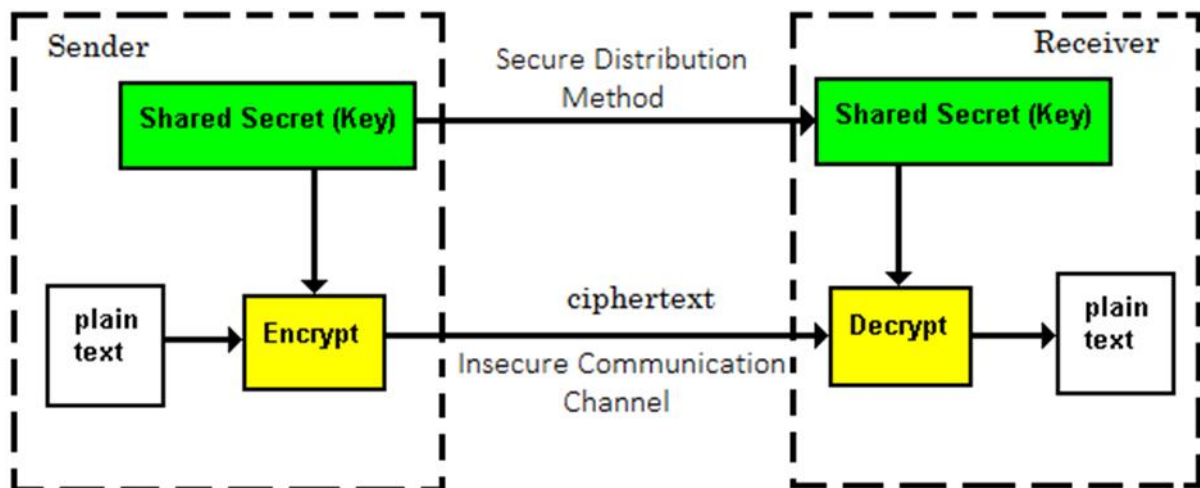
The main difference between these cryptosystems is the relationship between the encryption and the decryption key. Logically, in any cryptosystem, both the keys are closely associated. It is practically impossible to decrypt the ciphertext with the key that is unrelated to the encryption key.

## Symmetric Key Encryption

The encryption process where **same keys are used for encrypting and decrypting** the information is known as Symmetric Key Encryption.

The study of symmetric cryptosystems is referred to as **symmetric cryptography**. Symmetric cryptosystems are also sometimes referred to as **secret key cryptosystems**.

A few well-known examples of symmetric key encryption methods are: Digital Encryption Standard (DES), Triple-DES (3DES), IDEA, and BLOWFISH.



Prior to 1970, all cryptosystems employed symmetric key encryption. Even today, its relevance is very high and it is being used extensively in many cryptosystems. It is very unlikely that this encryption will fade away, as it has certain advantages over asymmetric key encryption.

The salient features of cryptosystem based on symmetric key encryption are:

- Persons using symmetric key encryption must share a common key prior to exchange of information.
- Keys are recommended to be changed regularly to prevent any attack on the system.
- A robust mechanism needs to exist to exchange the key between the communicating parties. As keys are required to be changed regularly, this mechanism becomes expensive and cumbersome.
- In a group of  $n$  people, to enable two-party communication between any two persons, the number of keys required for group is  $n \times (n - 1)/2$ .
- Length of Key (number of bits) in this encryption is smaller and hence, process of encryption-decryption is faster than asymmetric key encryption.
- Processing power of computer system required to run symmetric algorithm is less.

### Challenge of Symmetric Key Cryptosystem

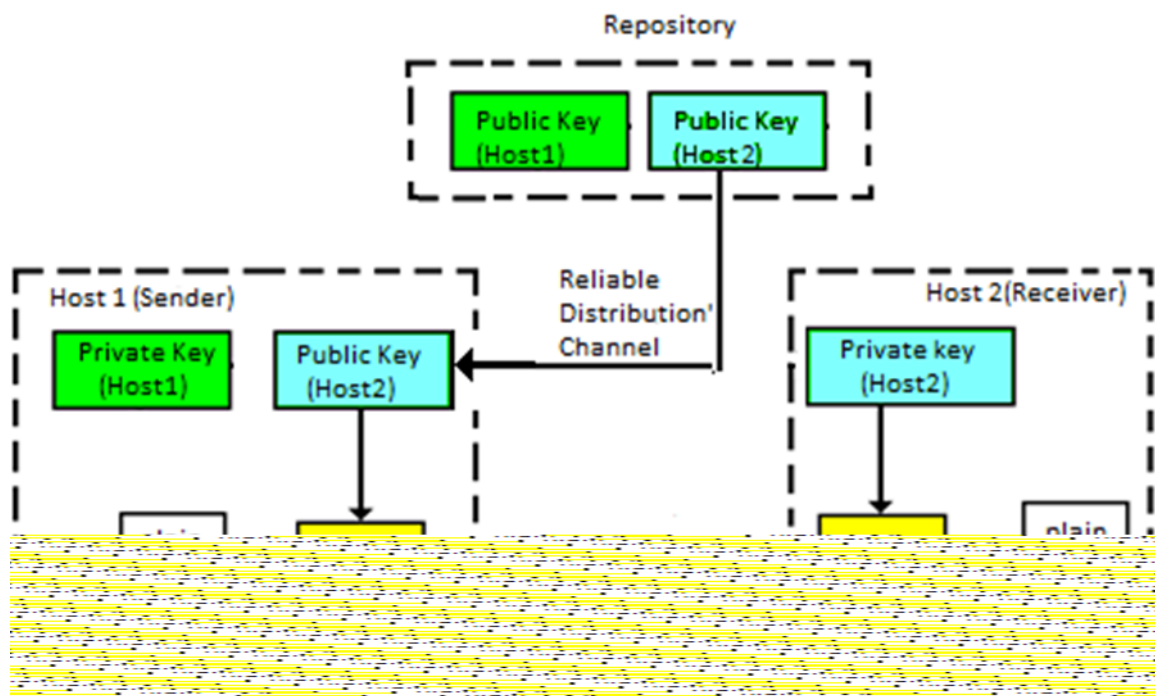
There are two restrictive challenges of employing symmetric key cryptography.

- **Key establishment** – Before any communication, both the sender and the receiver need to agree on a secret symmetric key. It requires a secure key establishment mechanism in place.
- **Trust Issue** – Since the sender and the receiver use the same symmetric key, there is an implicit requirement that the sender and the receiver 'trust' each other. For example, it may happen that the receiver has lost the key to an attacker and the sender is not informed.

These two challenges are highly restraining for modern day communication. Today, people need to exchange information with non-familiar and non-trusted parties. For example, a communication between online seller and customer. These limitations of symmetric key encryption gave rise to asymmetric key encryption schemes.

### Asymmetric Key Encryption

The encryption process where **different keys are used for encrypting and decrypting the information** is known as Asymmetric Key Encryption. Though the keys are different, they are mathematically related and hence, retrieving the plaintext by decrypting ciphertext is feasible. The process is depicted in the following illustration:



Asymmetric Key Encryption was invented in the 20<sup>th</sup> century to come over the necessity of pre-shared secret key between communicating persons. The salient features of this encryption scheme are as follows:























































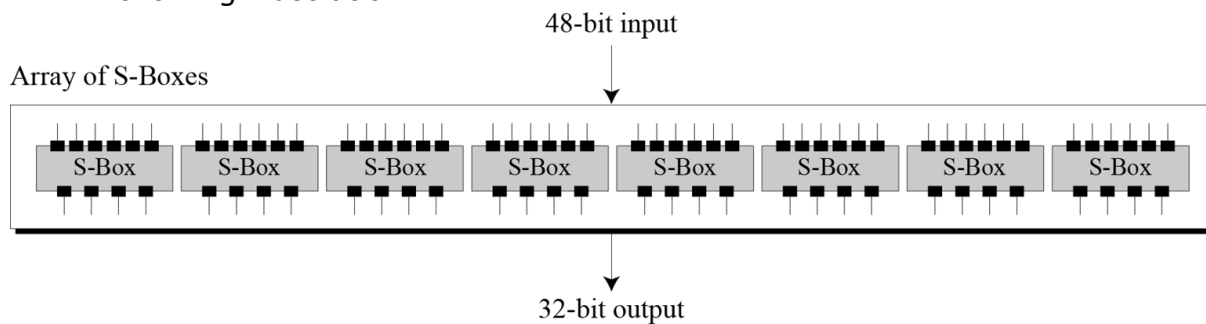




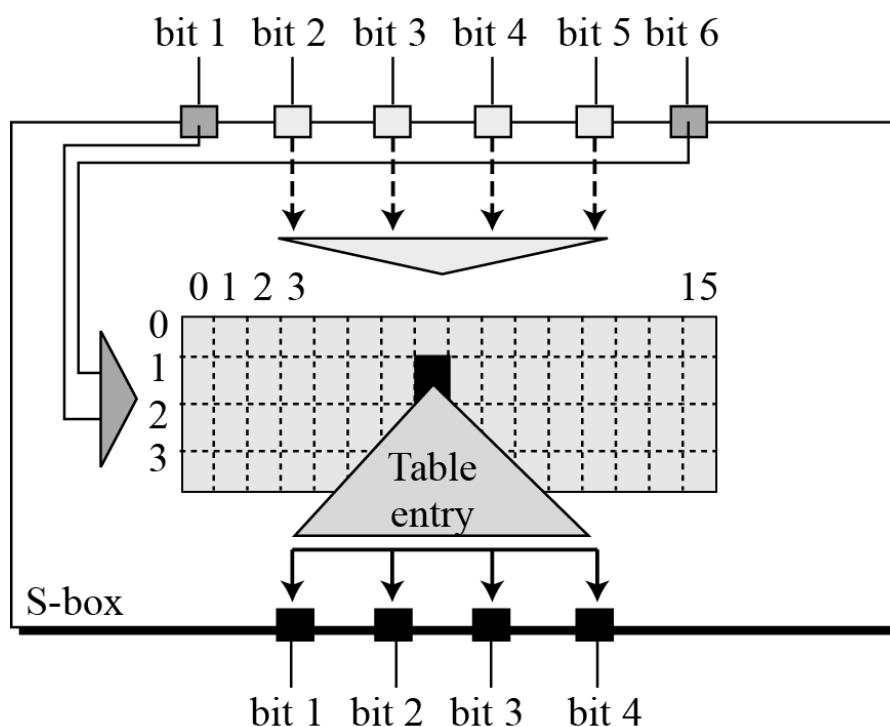


32	01	02	03	04	05
04	05	06	07	08	09
08	09	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	31	31	32	01

- **XOR (Whitener).** After the expansion permutation, DES does XOR operation on the expanded right section and the round key. The round key is used only in this operation.
- **Substitution Boxes.** The S-boxes carry out the real mixing (confusion). DES uses 8 S-boxes, each with a 6-bit input and a 4-bit output. Refer the following illustration:



The S-box rule is illustrated below:



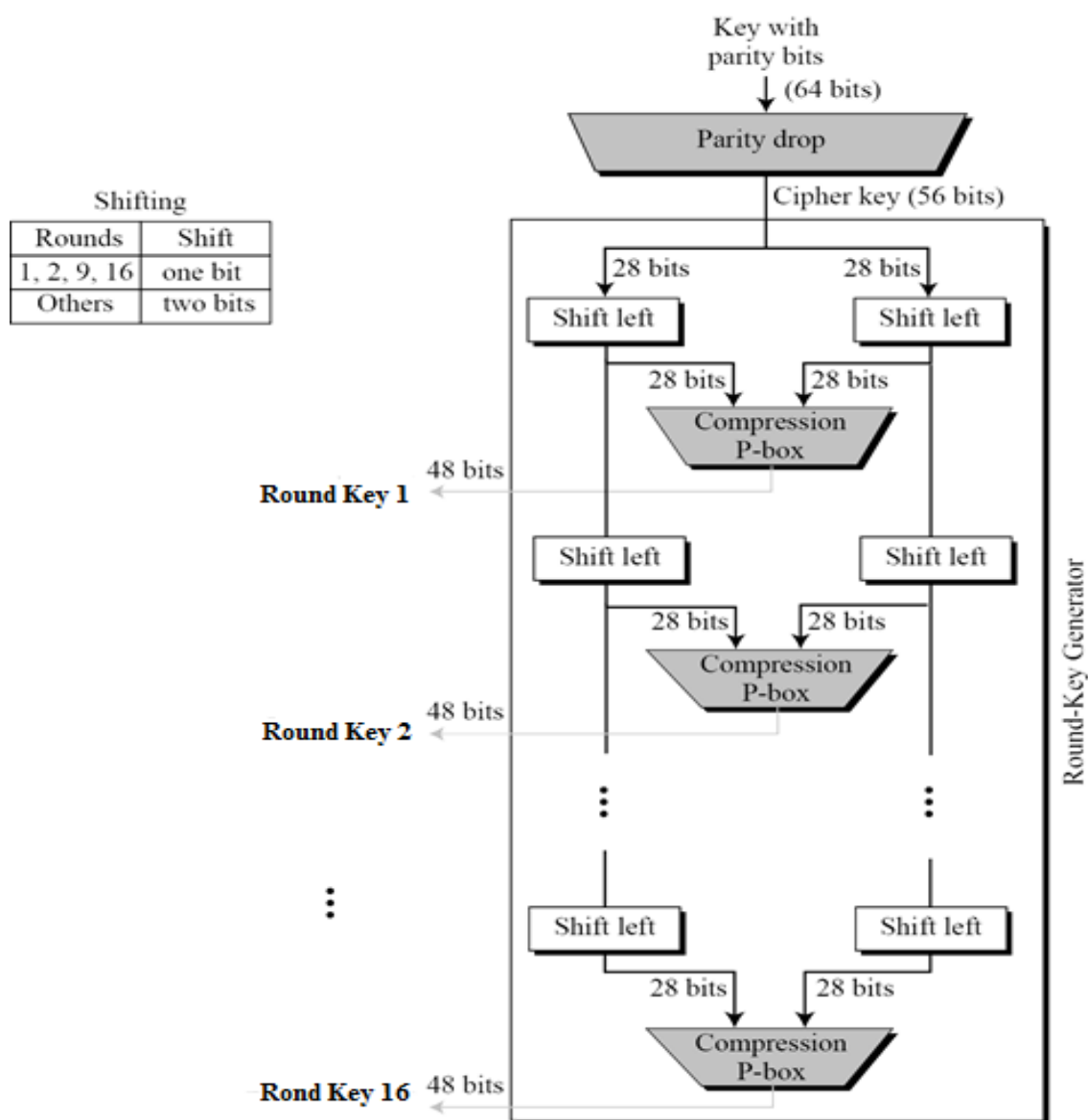
There are a total of eight S-box tables. The output of all eight s-boxes is then combined in to 32 bit section.

- **Straight Permutation** – The 32 bit output of S-boxes is then subjected to the straight permutation with rule shown in the following illustration:

16	07	20	21	29	12	28	17
01	15	23	26	05	18	31	10
02	08	24	14	32	27	03	09
19	13	30	06	22	11	04	25

## Key Generation

The round-key generator creates sixteen 48-bit keys out of a 56-bit cipher key. The process of key generation is depicted in the following illustration:



The logic for Parity drop, shifting, and Compression P-box is given in the DES description.

## DES Analysis

The DES satisfies both the desired properties of block cipher. These two properties make cipher very strong.

- **Avalanche effect:** A small change in plaintext results in the very grate change in the ciphertext.

- **Completeness:** Each bit of ciphertext depends on many bits of plaintext.

During the last few years, cryptanalysis have found some weaknesses in DES when key selected are weak keys. These keys shall be avoided.

DES has proved to be a very well designed block cipher. There have been no significant cryptanalytic attacks on DES other than exhaustive key search.

# 10. TRIPLE DES

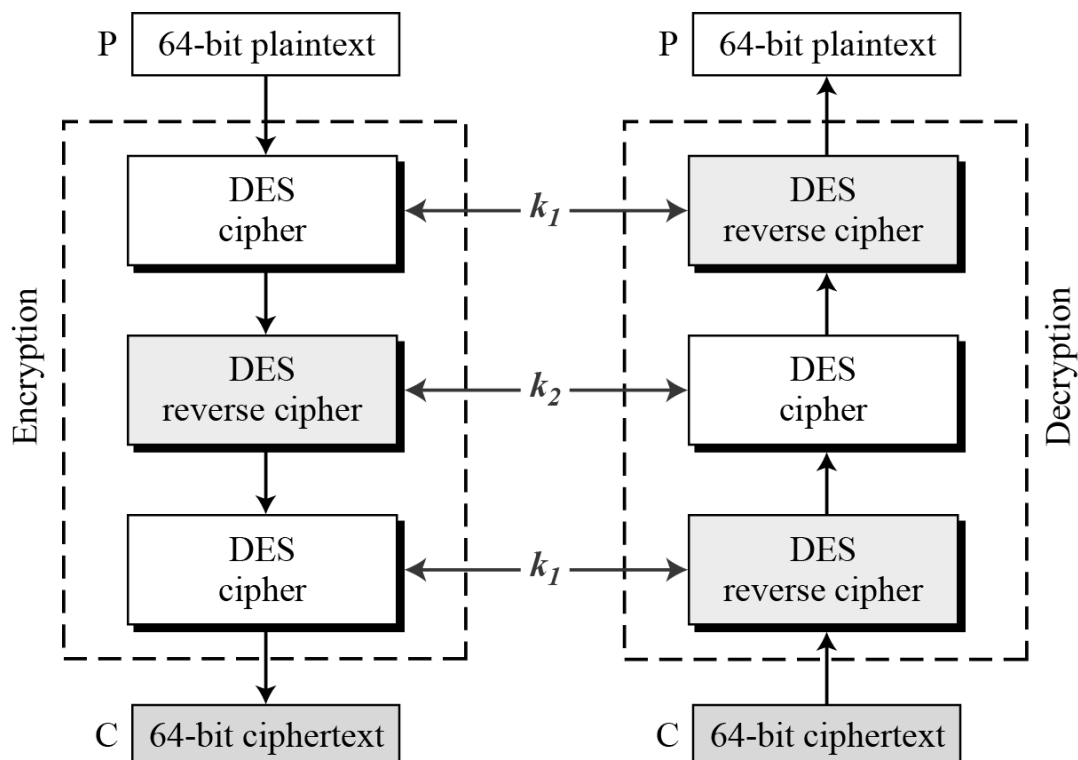
The speed of exhaustive key searches against DES after 1990 began to cause discomfort amongst users of DES. However, users did not want to replace DES as it takes an enormous amount of time and money to change encryption algorithms that are widely adopted and embedded in large security architectures.

The pragmatic approach was not to abandon the DES completely, but to change the manner in which DES is used. This led to the modified schemes of Triple DES (sometimes known as 3DES).

Incidentally, there are two variants of Triple DES known as 3-key Triple DES (3TDES) and 2-key Triple DES (2TDES).

## 3-KEY Triple DES

Before using 3TDES, user first generate and distribute a 3TDES key  $K$ , which consists of three different DES keys  $K_1$ ,  $K_2$  and  $K_3$ . This means that the actual 3TDES key has length  $3 \times 56 = 168$  bits. The encryption scheme is illustrated as follows:





The encryption-decryption process is as follows:

- Encrypt the plaintext blocks using single DES with key  $K_1$ .
- Now decrypt the output of step 1 using single DES with key  $K_2$ .
- Finally, encrypt the output of step 2 using single DES with key  $K_3$ .
- The output of step 3 is the ciphertext.
- Decryption of a ciphertext is a reverse process. User first decrypt using  $K_3$ , then encrypt with  $K_2$ , and finally decrypt with  $K_1$ .

Due to this design of Triple DES as an encrypt–decrypt–encrypt process, it is possible to use a 3TDES (hardware) implementation for single DES by setting  $K_1$ ,  $K_2$ , and  $K_3$  to be the same value. This provides backwards compatibility with DES.

Second variant of Triple DES (2TDES) is identical to 3TDES except that  $K_3$  is replaced by  $K_1$ . In other words, user encrypt plaintext blocks with key  $K_1$ , then decrypt with key  $K_2$ , and finally encrypt with  $K_1$  again. Therefore, 2TDES has a key length of 112 bits.

Triple DES systems are significantly more secure than single DES, but these are clearly a much slower process than encryption using single DES.

# 11. ADVANCED ENCRYPTION STANDARD

The more popular and widely adopted symmetric encryption algorithm likely to be encountered nowadays is the Advanced Encryption Standard (AES). It is found at least six times faster than triple DES.

A replacement for DES was needed as its key size was too small. With increasing computing power, it was considered vulnerable against exhaustive key search attack. Triple DES was designed to overcome this drawback but it was found slow.

The features of AES are as follows:

- Symmetric key symmetric block cipher
- 128-bit data, 128/192/256-bit keys
- Stronger and faster than Triple-DES
- Provide full specification and design details
- Software implementable in C and Java

## Operation of AES

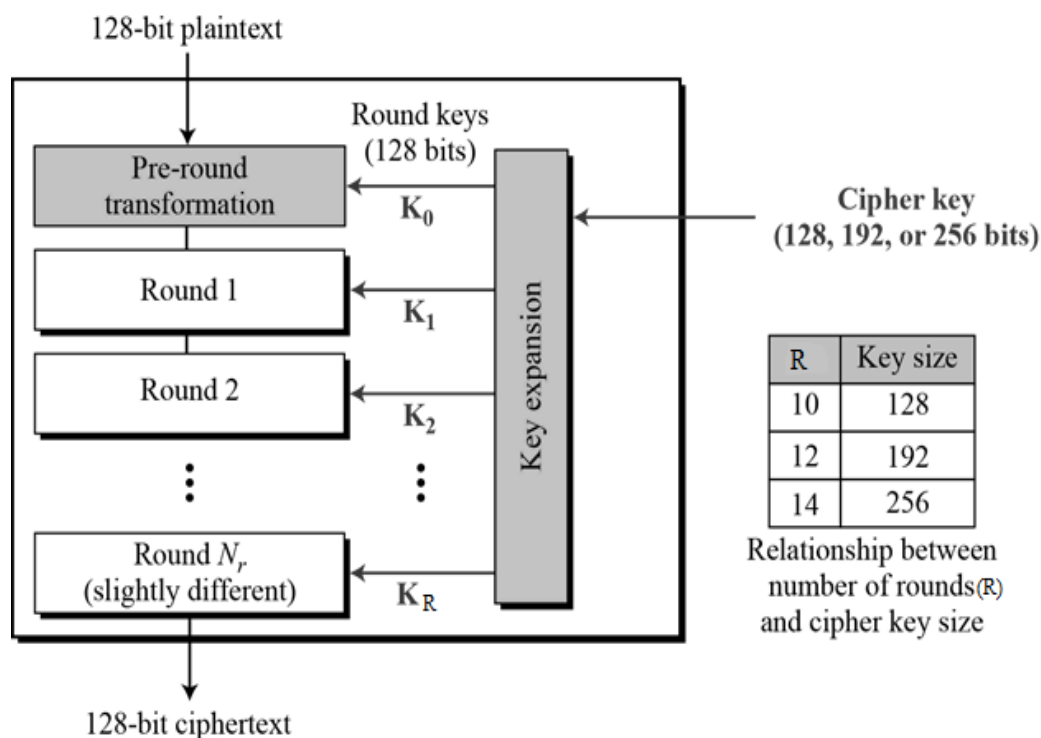
---

AES is an iterative rather than Feistel cipher. It is based on 'substitution-permutation network'. It comprises of a series of linked operations, some of which involve replacing inputs by specific outputs (substitutions) and others involve shuffling bits around (permutations).

Interestingly, AES performs all its computations on bytes rather than bits. Hence, AES treats the 128 bits of a plaintext block as 16 bytes. These 16 bytes are arranged in four columns and four rows for processing as a matrix:

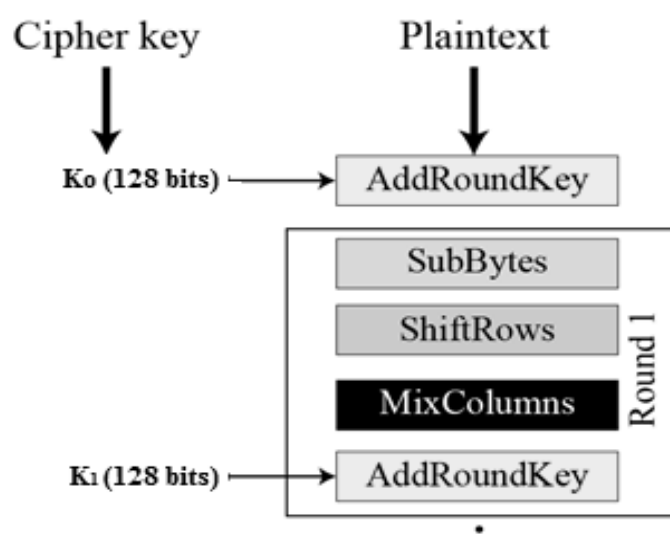
Unlike DES, the number of rounds in AES is variable and depends on the length of the key. AES uses 10 rounds for 128-bit keys, 12 rounds for 192-bit keys and 14 rounds for 256-bit keys. Each of these rounds uses a different 128-bit round key, which is calculated from the original AES key.

The schematic of AES structure is given in the following illustration:



## Encryption Process

Here, we restrict to description of a typical round of AES encryption. Each round comprise of four sub-processes. The first round process is depicted below:



### Byte Substitution (SubBytes)

The 16 input bytes are substituted by looking up a fixed table (S-box) given in design. The result is in a matrix of four rows and four columns.

### Shiftrows

Each of the four rows of the matrix is shifted to the left. Any entries that 'fall off' are re-inserted on the right side of row. Shift is carried out as follows:

- First row is not shifted.
- Second row is shifted one (byte) position to the left.
- Third row is shifted two positions to the left.
- Fourth row is shifted three positions to the left.
- The result is a new matrix consisting of the same 16 bytes but shifted with respect to each other.

### MixColumns

Each column of four bytes is now transformed using a special mathematical function. This function takes as input the four bytes of one column and outputs four completely new bytes, which replace the original column. The result is another new matrix consisting of 16 new bytes. It should be noted that this step is not performed in the last round.

### Addroundkey

The 16 bytes of the matrix are now considered as 128 bits and are XORed to the 128 bits of the round key. If this is the last round then the output is the ciphertext. Otherwise, the resulting 128 bits are interpreted as 16 bytes and we begin another similar round.

## Decryption Process

---

The process of decryption of an AES ciphertext is similar to the encryption process in the reverse order. Each round consists of the four processes conducted in the reverse order:

- Add round key
- Mix columns
- Shift rows
- Byte substitution

Since sub-processes in each round are in reverse manner, unlike for a Feistel Cipher, the encryption and decryption algorithms need to be separately implemented, although they are very closely related.

## **AES Analysis**

---

In present day cryptography, AES is widely adopted and supported in both hardware and software. Till date, no practical cryptanalytic attacks against AES has been discovered. Additionally, AES has built-in flexibility of key length, which allows a degree of 'future-proofing' against progress in the ability to perform exhaustive key searches.

However, just as for DES, the AES security is assured only if it is correctly implemented and good key management is employed.

# 12. MODES OF OPERATION

In this chapter, we will discuss the different modes of operation of a block cipher. These are procedural rules for a generic block cipher. Interestingly, the different modes result in different properties being achieved which add to the security of the underlying block cipher.

A block cipher processes the data blocks of fixed size. Usually, the size of a message is larger than the block size. Hence, the long message is divided into a series of sequential message blocks, and the cipher operates on these blocks one at a time.

## Electronic Code Book (ECB) Mode

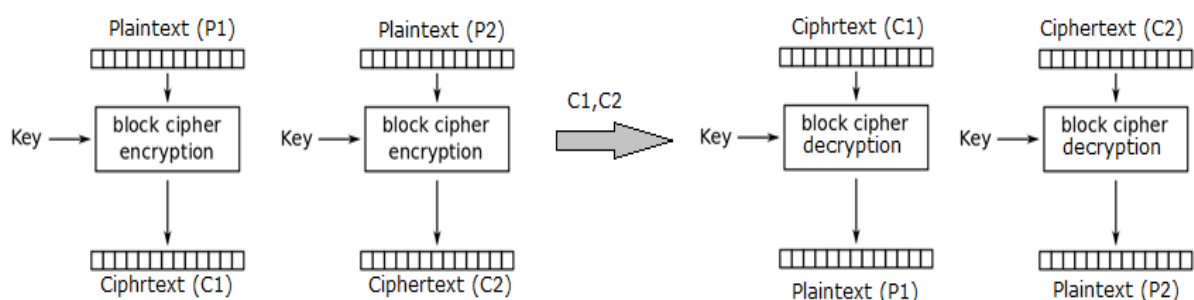
This mode is a most straightforward way of processing a series of sequentially listed message blocks.

### Operation

- The user takes the first block of plaintext and encrypts it with the key to produce the first block of ciphertext.
- He then takes the second block of plaintext and follows the same process with same key and so on so forth.

The ECB mode is **deterministic**, that is, if plaintext block  $P_1, P_2, \dots, P_m$  are encrypted twice under the same key, the output ciphertext blocks will be the same.

In fact, for a given key technically we can create a codebook of ciphertexts for all possible plaintext blocks. Encryption would then entail only looking up for required plaintext and select the corresponding ciphertext. Thus, the operation is analogous to the assignment of code words in a codebook, and hence gets an official name: Electronic Codebook mode of operation (ECB). It is illustrated as follows:



## Analysis of ECB Mode

In reality, any application data usually have partial information which can be guessed. For example, the range of salary can be guessed. A ciphertext from ECB can allow an attacker to guess the plaintext by trial-and-error if the plaintext message is within predictable.

For example, if a ciphertext from the ECB mode is known to encrypt a salary figure, then a small number of trials will allow an attacker to recover the figure. In general, we do not wish to use a deterministic cipher, and hence the ECB mode should not be used in most applications.

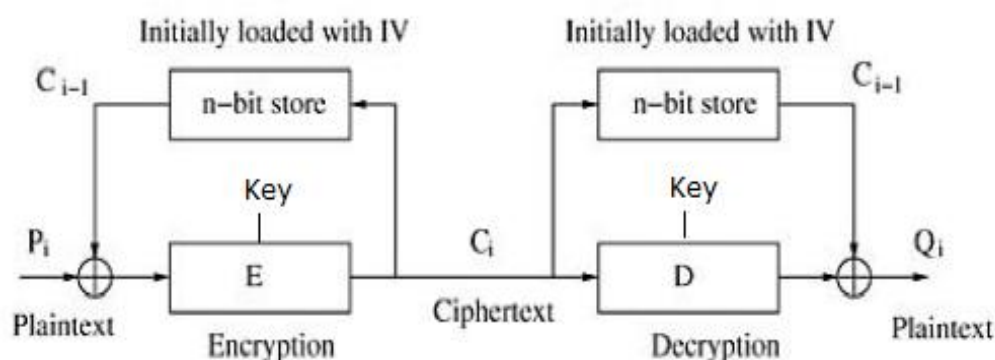
## Cipher Block Chaining (CBC) Mode

CBC mode of operation provides message dependence for generating ciphertext and makes the system non-deterministic.

### Operation

The operation of CBC mode is depicted in the following illustration. The steps are as follows:

- Load the n-bit Initialization Vector (IV) in the top register.
- XOR the n-bit plaintext block with data value in top register.
- Encrypt the result of XOR operation with underlying block cipher with key K.
- Feed ciphertext block into top register and continue the operation till all plaintext blocks are processed.
- For decryption, IV data is XORed with first ciphertext block decrypted. The first ciphertext block is also fed into to register replacing IV for decrypting next ciphertext block.



### Analysis of CBC Mode

In CBC mode, the current plaintext block is added to the previous ciphertext block, and then the result is encrypted with the key. Decryption is thus the reverse process, which involves decrypting the current ciphertext and then adding the previous ciphertext block to the result.

Advantage of CBC over ECB is that changing IV results in different ciphertext for identical message. On the drawback side, the error in transmission gets propagated to few further block during decryption due to chaining effect.

It is worth mentioning that CBC mode forms the basis for a well-known data origin authentication mechanism. Thus, it has an advantage for those applications that require both symmetric encryption and data origin authentication.

### Cipher Feedback (CFB) Mode

---

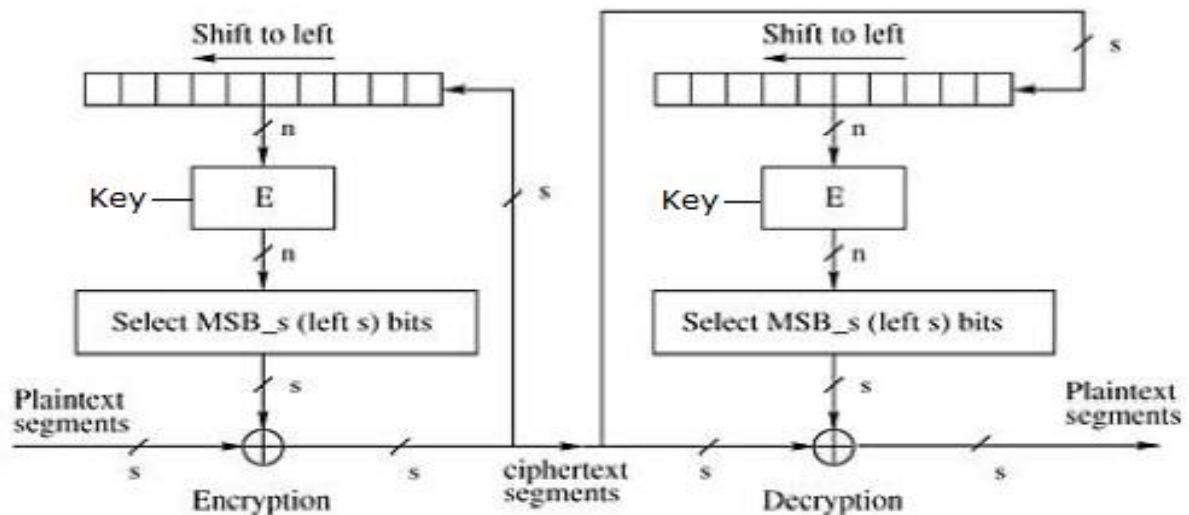
In this mode, each ciphertext block gets 'fed back' into the encryption process in order to encrypt the next plaintext block.

### Operation

The operation of CFB mode is depicted in the following illustration. For example, in the present system, a message block has a size 's' bits where  $1 < s < n$ . The CFB mode requires an initialization vector (IV) as the initial random n-bit input block. The IV need not be secret. Steps of operation are:

- Load the IV in the top register.
- Encrypt the data value in top register with underlying block cipher with key K.
- Take only 's' number of most significant bits (left bits) of output of encryption process and XOR them with 's' bit plaintext message block to generate ciphertext block.
- Feed ciphertext block into top register by shifting already present data to the left and continue the operation till all plaintext blocks are processed.
- Essentially, the previous ciphertext block is encrypted with the key, and then the result is XORed to the current plaintext block.
- Similar steps are followed for decryption. Pre-decided IV is initially loaded at the start of decryption.





## Analysis of CFB Mode

CFB mode differs significantly from ECB mode, the ciphertext corresponding to a given plaintext block depends not just on that plaintext block and the key, but also on the previous ciphertext block. In other words, the ciphertext block is dependent of message.

CFB has a very strange feature. In this mode, user decrypts the ciphertext using only the encryption process of the block cipher. The decryption algorithm of the underlying block cipher is never used.

Apparently, CFB mode is converting a block cipher into a type of stream cipher. The encryption algorithm is used as a key-stream generator to produce key-stream that is placed in the bottom register. This key stream is then XORed with the plaintext as in case of stream cipher.

By converting a block cipher into a stream cipher, CFB mode provides some of the advantageous properties of a stream cipher while retaining the advantageous properties of a block cipher.

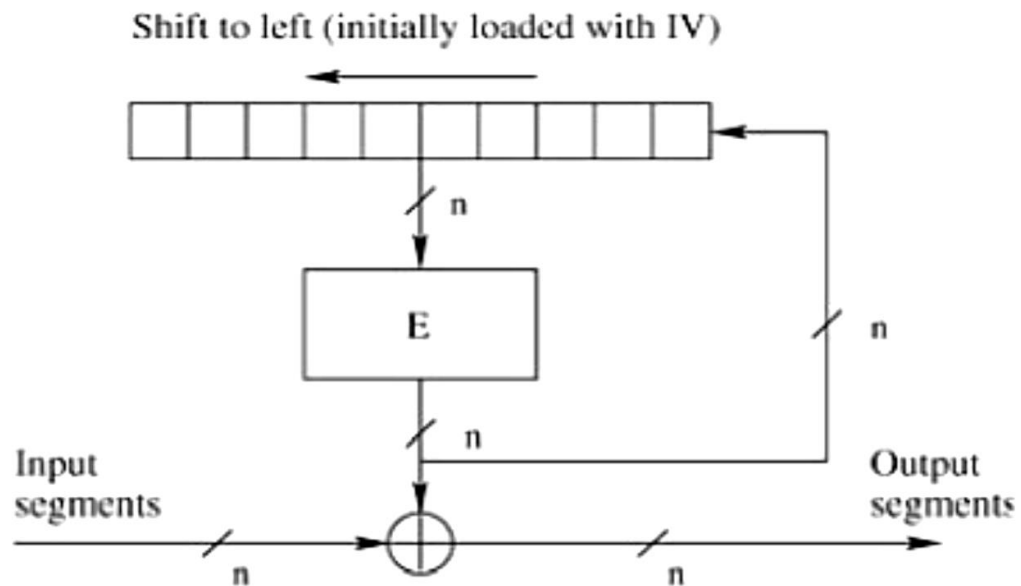
On the flip side, the error of transmission gets propagated due to changing of blocks.

## Output Feedback (OFB) Mode

It involves feeding the successive output blocks from the underlying block cipher back to it. These feedback blocks provide string of bits to feed the encryption algorithm which act as the key-stream generator as in case of CFB mode.

The key stream generated is XOR-ed with the plaintext blocks. The OFB mode requires an IV as the initial random n-bit input block. The IV need not be secret.

The operation is depicted in the following illustration:



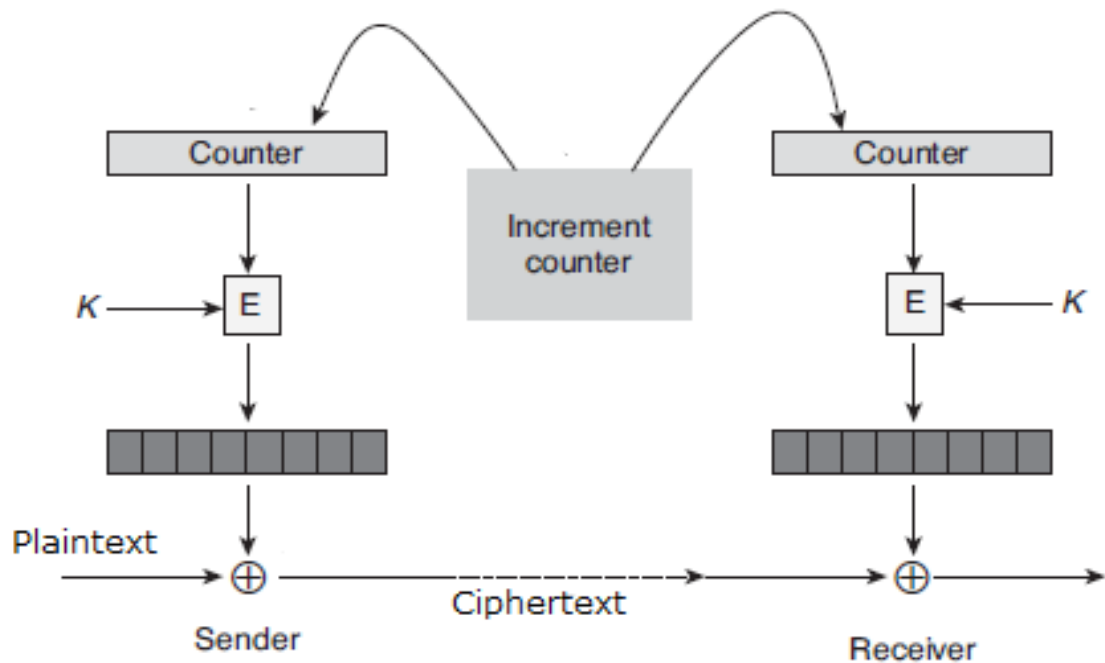
## Counter (CTR) Mode

It can be considered as a counter-based version of CFB mode without the feedback. In this mode, both the sender and receiver need to access to a reliable counter, which computes a new shared value each time a ciphertext block is exchanged. This shared counter is not necessarily a secret value, but challenge is that both sides must keep the counter synchronized.

### Operation

Both encryption and decryption in CTR mode are depicted in the following illustration. Steps in operation are:

- Load the initial counter value in the top register is the same for both the sender and the receiver. It plays the same role as the IV in CFB (and CBC) mode.
- Encrypt the contents of the counter with the key and place the result in the bottom register.
- Take the first plaintext block  $P_1$  and XOR this to the contents of the bottom register. The result of this is  $C_1$ . Send  $C_1$  to the receiver and update the counter. The counter update replaces the ciphertext feedback in CFB mode.
- Continue in this manner until the last plaintext block has been encrypted.
- The decryption is the reverse process. The ciphertext block is XORed with the output of encrypted contents of counter value. After decryption of each ciphertext block counter is updated as in case of encryption.



## Analysis of Counter Mode

It does not have message dependency and hence a ciphertext block does not depend on the previous plaintext blocks.

Like CFB mode, CTR mode does not involve the decryption process of the block cipher. This is because the CTR mode is really using the block cipher to generate a key-stream, which is encrypted using the XOR function. In other words, CTR mode also converts a block cipher to a stream cipher.

The serious disadvantage of CTR mode is that it requires a synchronous counter at sender and receiver. Loss of synchronization leads to incorrect recovery of plaintext.

However, CTR mode has almost all advantages of CFB mode. In addition, it does not propagate error of transmission at all.

# 13. PUBLIC KEY ENCRYPTION

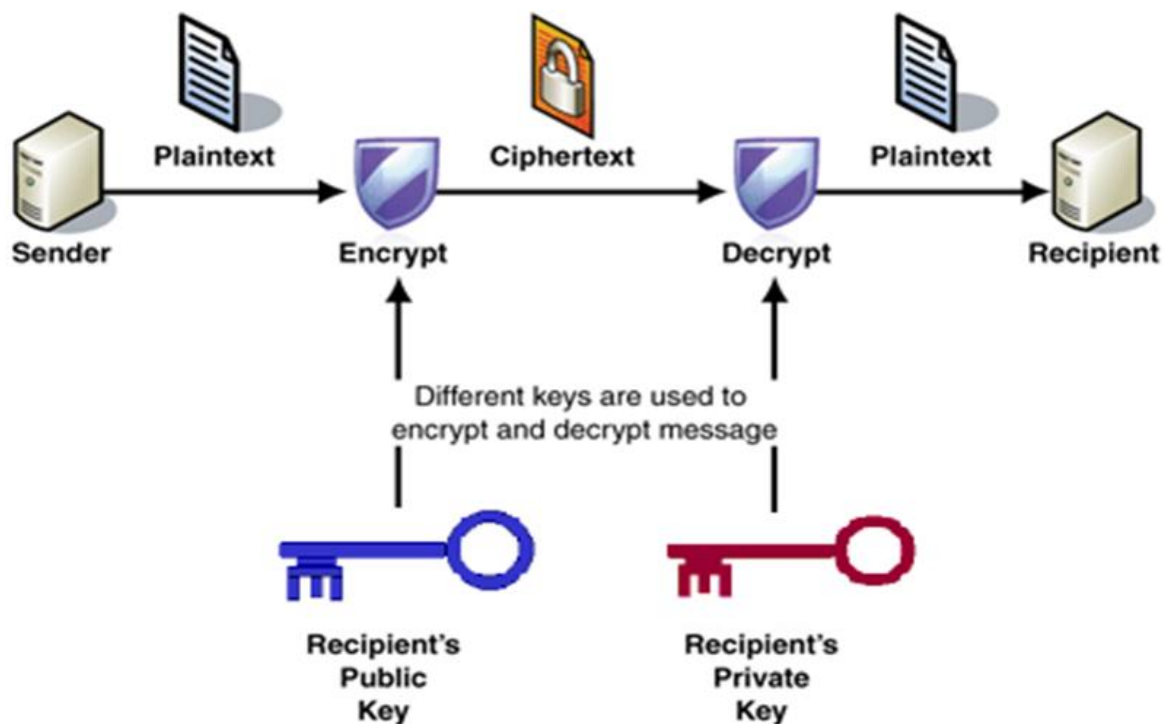
## Public Key Cryptography

Unlike symmetric key cryptography, we do not find historical use of public-key cryptography. It is a relatively new concept.

Symmetric cryptography was well suited for organizations such as governments, military, and big financial corporations were involved in the classified communication.

With the spread of more unsecure computer networks in last few decades, a genuine need was felt to use cryptography at larger scale. The symmetric key was found to be non-practical due to challenges it faced for key management. This gave rise to the public key cryptosystems.

The process of encryption and decryption is depicted in the following illustration:



The most important properties of public key encryption scheme are:

- Different keys are used for encryption and decryption. This is a property which set this scheme different than symmetric encryption scheme.
- Each receiver possesses a unique decryption key, generally referred to as his private key.

- Receiver needs to publish an encryption key, referred to as his public key.
- Some assurance of the authenticity of a public key is needed in this scheme to avoid spoofing by adversary as the receiver. Generally, this type of cryptosystem involves trusted third party which certifies that a particular public key belongs to a specific person or entity only.
- Encryption algorithm is complex enough to prohibit attacker from deducing the plaintext from the ciphertext and the encryption (public) key.
- Though private and public keys are related mathematically, it is not be feasible to calculate the private key from the public key. In fact, intelligent part of any public-key cryptosystem is in designing a relationship between two keys.

There are three types of Public Key Encryption schemes. We discuss them in following sections:

### RSA Cryptosystem

---

This cryptosystem is one the initial system. It remains most employed cryptosystem even today. The system was invented by three scholars **Ron Rivest**, **Adi Shamir**, and **Len Adleman** and hence, it is termed as RSA cryptosystem.

We will see two aspects of the RSA cryptosystem, firstly generation of key pair and secondly encryption-decryption algorithms.

### Generation of RSA Key Pair

Each person or a party who desires to participate in communication using encryption needs to generate a pair of keys, namely public key and private key. The process followed in the generation of keys is described below:

- **Generate the RSA modulus (n)**
  - o Select two large primes,  $p$  and  $q$ .
  - o Calculate  $n=p*q$ . For strong unbreakable encryption, let  $n$  be a large number, typically a minimum of 512 bits.
- **Find Derived Number (e)**
  - Number  $e$  must be greater than 1 and less than  $(p - 1)(q - 1)$ .
  - There must be no common factor for  $e$  and  $(p - 1)(q - 1)$  except for 1. In other words two numbers  $e$  and  $(p - 1)(q - 1)$  are coprime.

- **Form the public key**

- o The pair of numbers  $(n, e)$  form the RSA public key and is made public.
- o Interestingly, though  $n$  is part of the public key, difficulty in factorizing a large prime number ensures that attacker cannot find in finite time the two primes  $(p \& q)$  used to obtain  $n$ . This is strength of RSA.

- **Generate the private key**

- o Private Key  $d$  is calculated from  $p$ ,  $q$ , and  $e$ . For given  $n$  and  $e$ , there is unique number  $d$ .
- o Number  $d$  is the inverse of  $e$  modulo  $(p - 1)(q - 1)$ . This means that  $d$  is the number less than  $(p - 1)(q - 1)$  such that when multiplied by  $e$ , it is equal to 1 modulo  $(p - 1)(q - 1)$ .
- o This relationship is written mathematically as follows:

$$ed = 1 \bmod (p - 1)(q - 1)$$

The Extended Euclidean Algorithm takes  $p$ ,  $q$ , and  $e$  as input and gives  $d$  as output.

### Example

An example of generating RSA Key pair is given below. (For ease of understanding, the primes  $p$  &  $q$  taken here are small values. Practically, these values are very high).

- Let two primes be  $p = 7$  and  $q = 13$ . Thus, modulus  $n = pq = 7 \times 13 = 91$ .
- Select  $e = 5$ , which is a valid choice since there is no number that is common factor of 5 and  $(p - 1)(q - 1) = 6 \times 12 = 72$ , except for 1.
- The pair of numbers  $(n, e) = (91, 5)$  forms the public key and can be made available to anyone whom we wish to be able to send us encrypted messages.
- Input  $p = 7$ ,  $q = 13$ , and  $e = 5$  to the Extended Euclidean Algorithm. The output will be  $d = 29$ .
- Check that the  $d$  calculated is correct by computing:

$$de = 29 \times 5 = 145 = 1 \bmod 72$$

- Hence, public key is  $(91, 5)$  and private keys is  $(91, 29)$ .

## Encryption and Decryption

Once the key pair has been generated, the process of encryption and decryption are relatively straightforward and computationally easy.

Interestingly, RSA does not directly operate on strings of bits as in case of symmetric key encryption. It operates on numbers modulo  $n$ . Hence, it is necessary to represent the plaintext as a series of numbers less than  $n$ .

### RSA Encryption

- Suppose the sender wish to send some text message to someone whose public key is  $(n, e)$ .
- The sender then represents the plaintext as a series of numbers less than  $n$ .
- To encrypt the first plaintext  $P$ , which is a number modulo  $n$ . The encryption process is simple mathematical step as:

$$C = P^e \bmod n$$

- In other words, the ciphertext  $C$  is equal to the plaintext  $P$  multiplied by itself  $e$  times and then reduced modulo  $n$ . This means that  $C$  is also a number less than  $n$ .
- Returning to our Key Generation example with plaintext  $P = 10$ , we get ciphertext  $C$ :

$$C = 10^5 \bmod 91$$

### RSA Decryption

- The decryption process for RSA is also very straightforward. Suppose that the receiver of public-key pair  $(n, e)$  has received a ciphertext  $C$ .
- Receiver raises  $C$  to the power of his private key  $d$ . The result modulo  $n$  will be the plaintext  $P$ .

$$\text{Plaintext} = C^d \bmod n$$

- Returning again to our numerical example, the ciphertext  $C = 82$  would get decrypted to number 10 using private key 29:

$$\text{Plaintext} = 82^{29} \bmod 91 = 10$$

## RSA Analysis

The security of RSA depends on the strengths of two separate functions. The RSA cryptosystem is most popular public-key cryptosystem strength of which is based on the practical difficulty of factoring the very large numbers.

- **Encryption Function:** It is considered as a one-way function of converting plaintext into ciphertext and it can be reversed only with the knowledge of private key  $d$ .
- **Key Generation:** The difficulty of determining a private key from an RSA public key is equivalent to factoring the modulus  $n$ . An attacker thus cannot use knowledge of an RSA public key to determine an RSA private key unless he can factor  $n$ . It is also a one way function, going from  $p$  &  $q$  values to modulus  $n$  is easy but reverse is not possible.

If either of these two functions are proved non one-way, then RSA will be broken. In fact, if a technique for factoring efficiently is developed then RSA will no longer be safe.

The strength of RSA encryption drastically goes down against attacks if the number  $p$  and  $q$  are not large primes and/ or chosen public key  $e$  is a small number.

## ElGamal Cryptosystem

---

Along with RSA, there are other public-key cryptosystems proposed. Many of them are based on different versions of the Discrete Logarithm Problem.

ElGamal cryptosystem, called Elliptic Curve Variant, is based on the Discrete Logarithm Problem. It derives the strength from the assumption that the discrete logarithms cannot be found in practical time frame for a given number, while the inverse operation of the power can be computed efficiently.

Let us go through a simple version of ElGamal that works with numbers modulo  $p$ . In the case of elliptic curve variants, it is based on quite different number systems.

### Generation of ElGamal Key Pair

Each user of ElGamal cryptosystem generates the key pair through as follows:

- **Choosing a large prime  $p$ .** Generally a prime number of 1024 to 2048 bits length is chosen.
- **Choosing a generator element  $g$ .**
  - o This number must be between 1 and  $p - 1$ , but cannot be any number.



- o It is a generator of the multiplicative group of integers modulo  $p$ . This means for every integer  $m$  co-prime to  $p$ , there is an integer  $k$  such that  $g^k = a \pmod{p}$ .

For example, 3 is generator of group 5 ( $Z_5 = \{1, 2, 3, 4\}$ ).

N	$3^n$	$3^n \pmod{5}$
1	3	3
2	9	4
3	27	2
4	81	1

- **Choosing the private key.** The private key  $x$  is any number bigger than 1 and smaller than  $p-1$ .
- **Computing part of the public key.** The value  $y$  is computed from the parameters  $p$ ,  $g$  and the private key  $x$  as follows:

$$y = g^x \pmod{p}$$

- **Obtaining Public key.** The ElGamal public key consists of the three parameters  $(p, g, y)$ .

For example, suppose that  $p = 17$  and that  $g = 6$  (It can be confirmed that 6 is a generator of group  $Z_{17}$ ). The private key  $x$  can be any number bigger than 1 and smaller than 16, so we choose  $x = 5$ . The value  $y$  is then computed as follows:

$$y = 6^5 \pmod{17} = 7$$

- Thus the private key is 5 and the public key is  $(17, 6, 7)$ .

## Encryption and Decryption

The generation of an ElGamal key pair is comparatively simpler than the equivalent process for RSA. But the encryption and decryption are slightly more complex than RSA.

### ElGamal Encryption

Suppose sender wishes to send a plaintext to someone whose ElGamal public key is  $(p, g, y)$ , then:

- Sender represents the plaintext as a series of numbers modulo  $p$ .

- To encrypt the first plaintext P, which is represented as a number modulo p. The encryption process to obtain the ciphertext C is as follows:

- o Randomly generate a number k;
- o Compute two values C1 and C2, where:

$$\begin{aligned}C1 &= g^k \text{ mod } p \\C2 &= (P * y^k) \text{ mod } p\end{aligned}$$

- o Send the ciphertext C, consisting of the two separate values (C1, C2), sent together.
- o Referring to our ElGamal key generation example given above, the plaintext P = 13 is encrypted as follows:

- Randomly generate a number, say k = 10
- Compute the two values C1 and C2, where:

$$\begin{aligned}C1 &= 6^{10} \text{ mod } 17 \\C2 &= (13 * 7^{10}) \text{ mod } 17 = 9\end{aligned}$$

- Send the ciphertext C = (C1, C2) = (15, 9).

### ElGamal Decryption

- To decrypt the ciphertext (C1, C2) using private key x, the following two steps are taken:

- o Compute the modular inverse of (C1)<sup>x</sup> modulo p, which is (C1)<sup>-x</sup>, generally referred to as decryption factor.
- o Obtain the plaintext by using the following formula:

$$C2 \times (C1)^{-x} \text{ mod } p = \text{Plaintext}$$

- In our example, to decrypt the ciphertext C = (C1, C2) = (15, 9) using private key x = 5, the decryption factor is

$$15^{-5} \text{ mod } 17 = 9$$

- Extract plaintext P = (9 × 9) mod 17 = 13.

## ElGamal Analysis

In ElGamal system, each user has a private key  $x$ . and has **three components** of public key: **prime modulus  $p$** , **generator  $g$** , and **public  $Y = g^x \bmod p$** . The strength of the ElGamal is based on the difficulty of discrete logarithm problem.

The secure key size is generally  $> 1024$  bits. Today even 2048 bits long key are used. On the processing speed front, Elgamal is quite slow, it is used mainly for key authentication protocols. Due to higher processing efficiency, Elliptic Curve variants of ElGamal are becoming increasingly popular.

## Elliptic Curve Cryptography (ECC)

---

Elliptic Curve Cryptography (ECC) is a term used to describe a suite of cryptographic tools and protocols whose security is based on special versions of the discrete logarithm problem. It does not use numbers modulo  $p$ .

ECC is based on sets of numbers that are associated with mathematical objects called elliptic curves. There are rules for adding and computing multiples of these numbers, just as there are for numbers modulo  $p$ .

ECC includes a variants of many cryptographic schemes that were initially designed for modular numbers such as ElGamal encryption and Digital Signature Algorithm.

It is believed that the discrete logarithm problem is much harder when applied to points on an elliptic curve. This prompts switching from numbers modulo  $p$  to points on an elliptic curve. Also an equivalent security level can be obtained with shorter keys if we use elliptic curve-based variants.

The shorter keys result in two benefits:

- Ease of key management
- Efficient computation

These benefits make elliptic-curve-based variants of encryption scheme highly attractive for application where computing resources are constrained.

## RSA and ElGamal Schemes – A Comparison

---

Let us briefly compare the RSA and ElGamal schemes on the various aspects.

<b>RSA</b>	<b>ElGamal</b>
It is more efficient for encryption.	It is more efficient for decryption.
It is less efficient for decryption.	It is more efficient for decryption.
For a particular security level, lengthy keys are required in RSA.	For the same level of security, very short keys are required.
It is widely accepted and used.	It is new and not very popular in market.

# 14. DATA INTEGRITY

Until now, we discussed the use of symmetric and public key schemes to achieve the confidentiality of information. With this chapter, we begin our discussion on different cryptographic techniques designed to provide other security services.

The focus of this chapter is on data integrity and cryptographic tools used to achieve the same.

## Threats to Data Integrity

---

When sensitive information is exchanged, the receiver must have the assurance that the message has come intact from the intended sender and is not modified inadvertently or otherwise. There are two different types of data integrity threats, namely **passive** and **active**.

### Passive Threats

This type of threats exists due to accidental changes in data.

- These data errors are likely to occur due to noise in a communication channel. Also, the data may get corrupted while the file is stored on a disk.
- Error-correcting codes and simple checksums like Cyclic Redundancy Checks (CRCs) are used to detect the loss of data integrity. In these techniques, a digest of data is computed mathematically and appended to the data.

### Active Threats

In this type of threats, an attacker can manipulate the data with malicious intent.

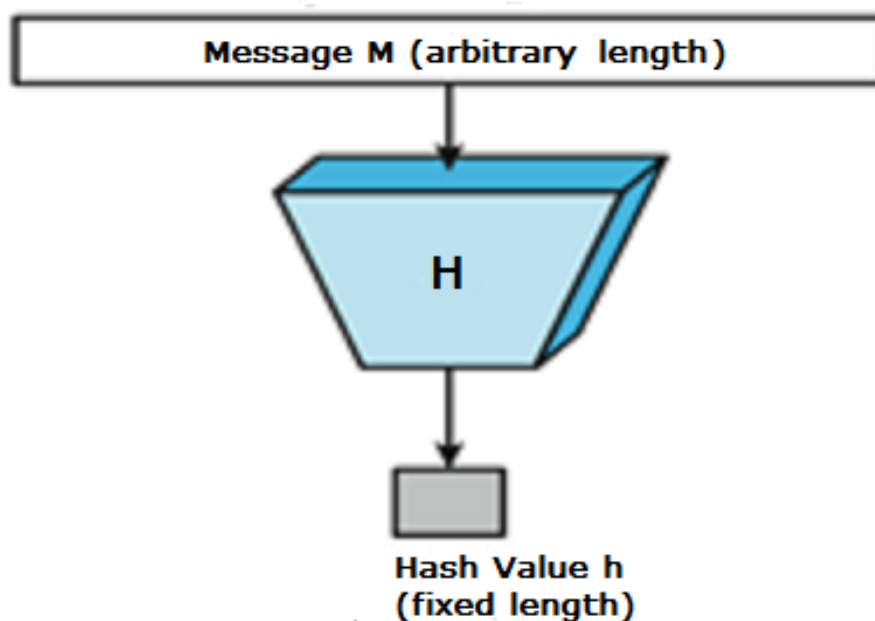
- At simplest level, if data is without digest, it can be modified without detection. The system can use techniques of appending CRC to data for detecting any active modification.
- At higher level of threat, attacker may modify data and try to derive new digest for modified data from existing digest. This is possible if the digest is computed using simple mechanisms such as CRC.
- Security mechanism such as Hash functions are used to tackle the active modification threats.

# 15. HASH FUNCTIONS

Hash functions are extremely useful and appear in almost all information security applications.

A hash function is a mathematical function that converts a numerical input value into another compressed numerical value. The input to the hash function is of arbitrary length but output is always of fixed length.

Values returned by a hash function are called **message digest** or simply **hash values**. The following picture illustrated hash function:



## Features of Hash Functions

---

The typical features of hash functions are:

- **Fixed Length Output (Hash Value)**
  - o Hash function converts data of arbitrary length to a fixed length. This process is often referred to as **hashing the data**.
  - o In general, the hash is much smaller than the input data, hence hash functions are sometimes called **compression functions**.
  - o Since a hash is a smaller representation of a larger data, it is also referred to as a **digest**.

- o Hash function with  $n$  bit output is referred to as an  **$n$ -bit hash function**. Popular hash functions generate values between 160 and 512 bits.
- **Efficiency of Operation**
  - o Generally for any hash function  $h$  with input  $x$ , computation of  $h(x)$  is a fast operation.
  - o Computationally hash functions are much faster than a symmetric encryption.

## Properties of Hash Functions

---

In order to be an effective cryptographic tool, the hash function is desired to possess following properties:

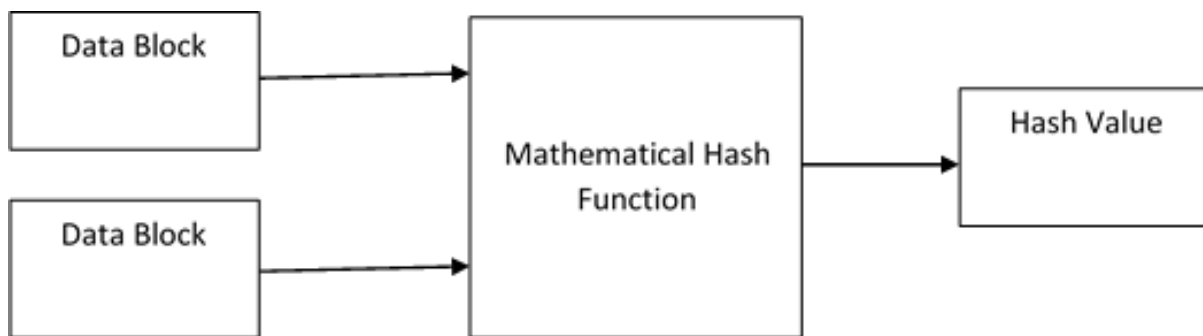
- **Pre-Image Resistance**
  - o This property means that it should be computationally hard to reverse a hash function.
  - o In other words, if a hash function  $h$  produced a hash value  $z$ , then it should be a difficult process to find any input value  $x$  that hashes to  $z$ .
  - o This property protects against an attacker who only has a hash value and is trying to find the input.
- **Second Pre-Image Resistance**
  - o This property means given an input and its hash, it should be hard to find a different input with the same hash.
  - o In other words, if a hash function  $h$  for an input  $x$  produces hash value  $h(x)$ , then it should be difficult to find any other input value  $y$  such that  $h(y) = h(x)$ .
  - o This property of hash function protects against an attacker who has an input value and its hash, and wants to substitute different value as legitimate value in place of original input value.
- **Collision Resistance**
  - o This property means it should be hard to find two different inputs of any length that result in the same hash. This property is also referred to as collision free hash function.
  - o In other words, for a hash function  $h$ , it is hard to find any two different inputs  $x$  and  $y$  such that  $h(x) = h(y)$ .

- o Since, hash function is compressing function with fixed hash length, it is impossible for a hash function not to have collisions. This property of collision free only confirms that these collisions should be hard to find.
- o This property makes it very difficult for an attacker to find two input values with the same hash.
- o Also, if a hash function is collision-resistant then it is second pre-image resistant.

## Design of Hashing Algorithms

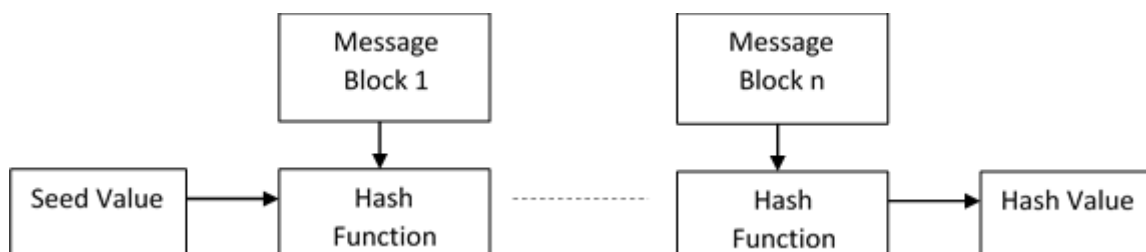
At the heart of a hashing is a mathematical function that operates on two fixed-size blocks of data to create a hash code. This hash function forms the part of the hashing algorithm.

The size of each data block varies depending on the algorithm. Typically the block sizes are from 128 bits to 512 bits. The following illustration demonstrates hash function:



Hashing algorithm involves rounds of above hash function like a block cipher. Each round takes an input of a fixed size, typically a combination of the most recent message block and the output of the last round.

This process is repeated for as many rounds as are required to hash the entire message. Schematic of hashing algorithm is depicted in the following illustration:





Since, the hash value of first message block becomes an input to the second hash operation, output of which alters the result of the third operation, and so on. This effect, known as an **avalanche** effect of hashing.

Avalanche effect results in substantially different hash values for two messages that differ by even a single bit of data.

Understand the difference between hash function and algorithm correctly. The hash function generates a hash code by operating on two blocks of fixed-length binary data.

Hashing algorithm is a process for using the hash function, specifying how the message will be broken up and how the results from previous message blocks are chained together.

## Popular Hash Functions

---

Let us briefly see some popular hash functions:

### Message Digest (MD)

MD5 was most popular and widely used hash function for quite some years.

- The MD family comprises of hash functions MD2, MD4, MD5 and MD6. It was adopted as Internet Standard RFC 1321. It is a 128-bit hash function.
- MD5 digests have been widely used in the software world to provide assurance about integrity of transferred file. For example, file servers often provide a pre-computed MD5 checksum for the files, so that a user can compare the checksum of the downloaded file to it.
- In 2004, collisions were found in MD5. An analytical attack was reported to be successful only in an hour by using computer cluster. This collision attack resulted in compromised MD5 and hence it is no longer recommended for use.

### Secure Hash Function (SHA)

Family of SHA comprise of four SHA algorithms; SHA-0, SHA-1, SHA-2, and SHA-3. Though from same family, there are structurally different.

- The original version is SHA-0, a 160-bit hash function, was published by the National Institute of Standards and Technology (NIST) in 1993. It had few weaknesses and did not become very popular. Later in 1995, SHA-1 was designed to correct alleged weaknesses of SHA-0.
- SHA-1 is the most widely used of the existing SHA hash functions. It is employed in several widely used applications and protocols including Secure Socket Layer (SSL) security.

- In 2005, a method was found for uncovering collisions for SHA-1 within practical time frame making long-term employability of SHA-1 doubtful.
- SHA-2 family has four further SHA variants, SHA-224, SHA-256, SHA-384, and SHA-512 depending up on number of bits in their hash value. No successful attacks have yet been reported on SHA-2 hash function.
- Though SHA-2 is a strong hash function. Though significantly different, its basic design is still follows design of SHA-1. Hence, NIST called for new competitive hash function designs.
- In October 2012, the NIST chose the Keccak algorithm as the new SHA-3 standard. Keccak offers many benefits, such as efficient performance and good resistance for attacks.

## RIPEMD

The RIPEMD is an acronym for RACE Integrity Primitives Evaluation Message Digest. This set of hash functions was designed by open research community and generally known as a family of European hash functions.

- The set includes RIPEMD, RIPEMD-128, and RIPEMD-160. There also exist 256, and 320-bit versions of this algorithm.
- Original RIPEMD (128 bit) is based upon the design principles used in MD4 and found to provide questionable security. RIPEMD 128-bit version came as a quick fix replacement to overcome vulnerabilities on the original RIPEMD.
- RIPEMD-160 is an improved version and the most widely used version in the family. The 256 and 320-bit versions reduce the chance of accidental collision, but do not have higher levels of security as compared to RIPEMD-128 and RIPEMD-160 respectively.

## Whirlpool

This is a 512-bit hash function.

- It is derived from the modified version of Advanced Encryption Standard (AES). One of the designer was Vincent Rijmen, a co-creator of the AES.
- Three versions of Whirlpool have been released; namely WHIRLPOOL-0, WHIRLPOOL-T, and WHIRLPOOL.

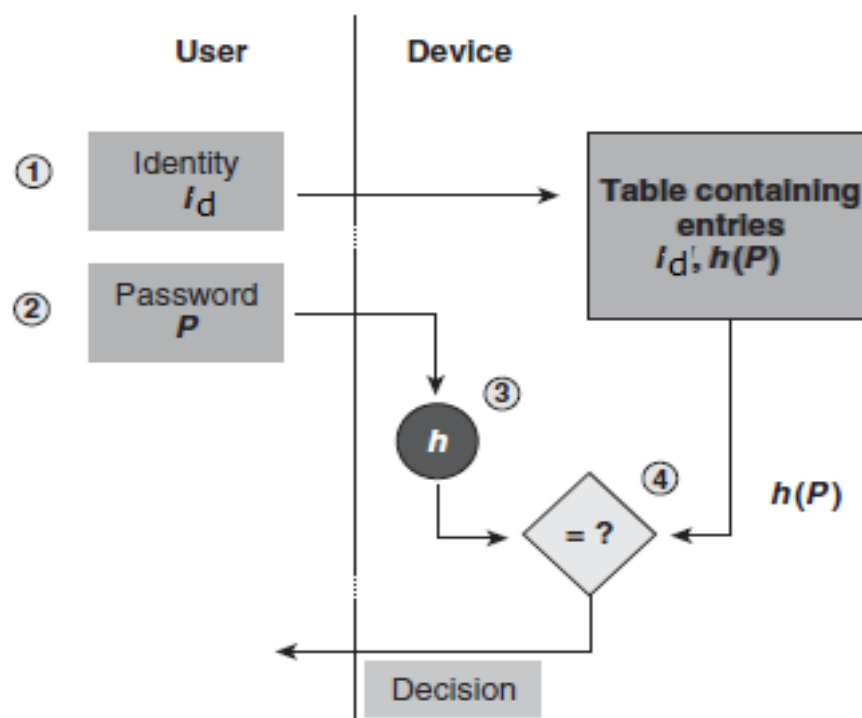
## Applications of Hash Functions

There are two direct applications of hash function based on its cryptographic properties.

### Password Storage

Hash functions provide protection to password storage.

- Instead of storing password in clear, mostly all logon processes store the hash values of passwords in the file.
- The Password file consists of a table of pairs which are in the form (user id,  $h(P)$ ).
- The process of logon is depicted in the following illustration:

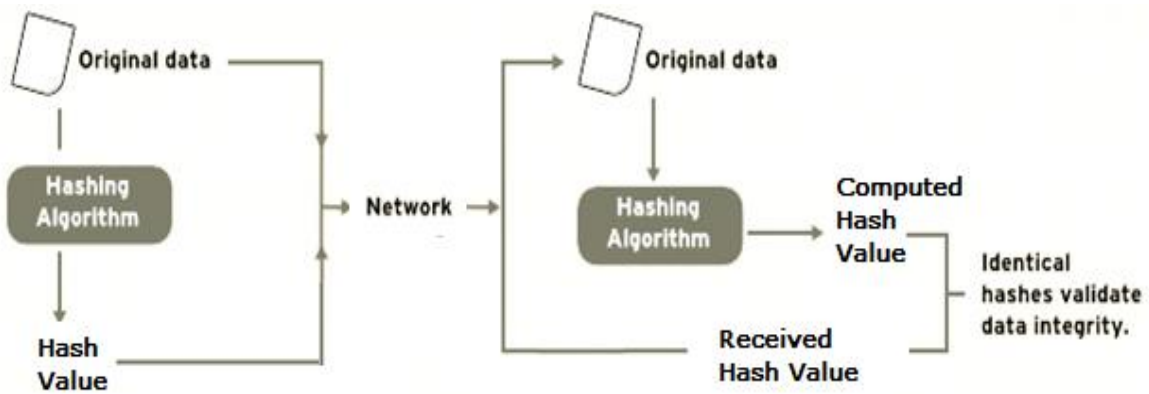


- An intruder can only see the hashes of passwords, even if he accessed the password. He can neither logon using hash nor can he derive the password from hash value since hash function possesses the property of pre-image resistance.

### Data Integrity Check

Data integrity check is a most common application of the hash functions. It is used to generate the checksums on data files. This application provides assurance to the user about correctness of the data.

The process is depicted in the following illustration:



The integrity check helps the user to detect any changes made to original file. It however, does not provide any assurance about originality. The attacker, instead of modifying file data, can change the entire file and compute all together new hash and send to the receiver. This integrity check application is useful only if the user is sure about the originality of file.

# 16. MESSAGE AUTHENTICATION

In the last chapter, we discussed the data integrity threats and the use of hashing technique to detect if any modification attacks have taken place on the data.

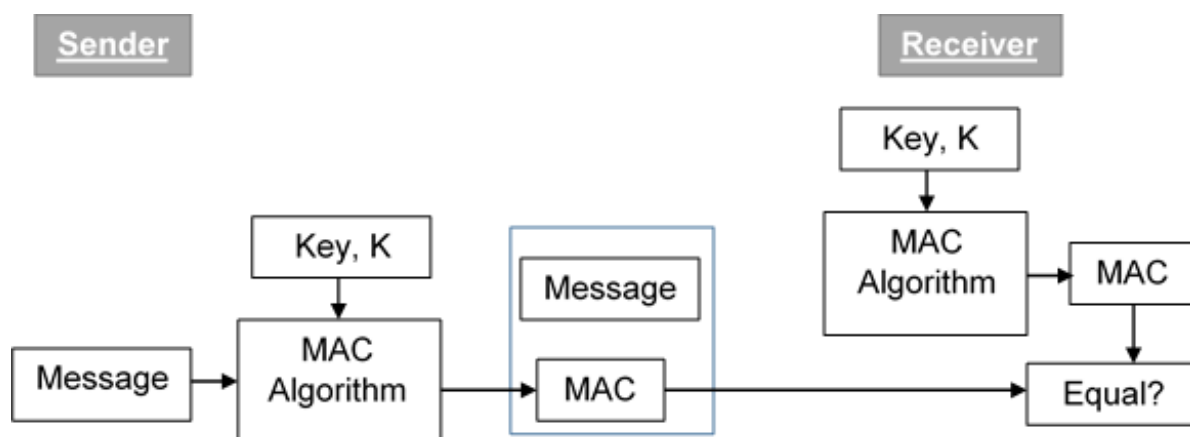
Another type of threat that exist for data is the lack of **message authentication**. In this threat, the user is not sure about the originator of the message. Message authentication can be provided using the cryptographic techniques that use secret keys as done in case of encryption.

## Message Authentication Code (MAC)

MAC algorithm is a symmetric key cryptographic technique to provide message authentication. For establishing MAC process, the sender and receiver share a symmetric key  $K$ .

Essentially, a MAC is an encrypted checksum generated on the underlying message that is sent along with a message to ensure message authentication.

The process of using MAC for authentication is depicted in the following illustration:



Let us now try to understand the entire process in detail:

- The sender uses some publicly known MAC algorithm, inputs the message and the secret key  $K$  and produces a MAC value.
- Similar to hash, MAC function also compresses an arbitrary long input into a fixed length output. The major difference between hash and MAC is that MAC uses secret key during the compression.

- The sender forwards the message along with the MAC. Here, we assume that the message is sent in the clear, as we are concerned of providing message origin authentication, not confidentiality. If confidentiality is required then the message needs encryption.
- On receipt of the message and the MAC, the receiver feeds the received message and the shared secret key  $K$  into the MAC algorithm and re-computes the MAC value.
- The receiver now checks equality of freshly computed MAC with the MAC received from the sender. If they match, then the receiver accepts the message and assures himself that the message has been sent by the intended sender.
- If the computed MAC does not match the MAC sent by the sender, the receiver cannot determine whether it is the message that has been altered or it is the origin that has been falsified. As a bottom-line, a receiver safely assumes that the message is not the genuine.

### Limitations of MAC

---

There are two major limitations of MAC, both due to its symmetric nature of operation:

- **Establishment of Shared Secret.**
  - o It can provide message authentication among pre-decided legitimate users who have shared key.
  - o This requires establishment of shared secret prior to use of MAC.
- **Inability to Provide Non-Repudiation**
  - o Non-repudiation is the assurance that a message originator cannot deny any previously sent messages and commitments or actions.
  - o MAC technique does not provide a non-repudiation service. If the sender and receiver get involved in a dispute over message origination, MACs cannot provide a proof that a message was indeed sent by the sender.
  - o Though no third party can compute the MAC, still sender could deny having sent the message and claim that the receiver forged it, as it is impossible to determine which of the two parties computed the MAC.

Both these limitations can be overcome by using the public key based digital signatures discussed in following section.

# 17. DIGITAL SIGNATURE

Digital signatures are the public-key primitives of message authentication. In the physical world, it is common to use handwritten signatures on handwritten or typed messages. They are used to bind signatory to the message.

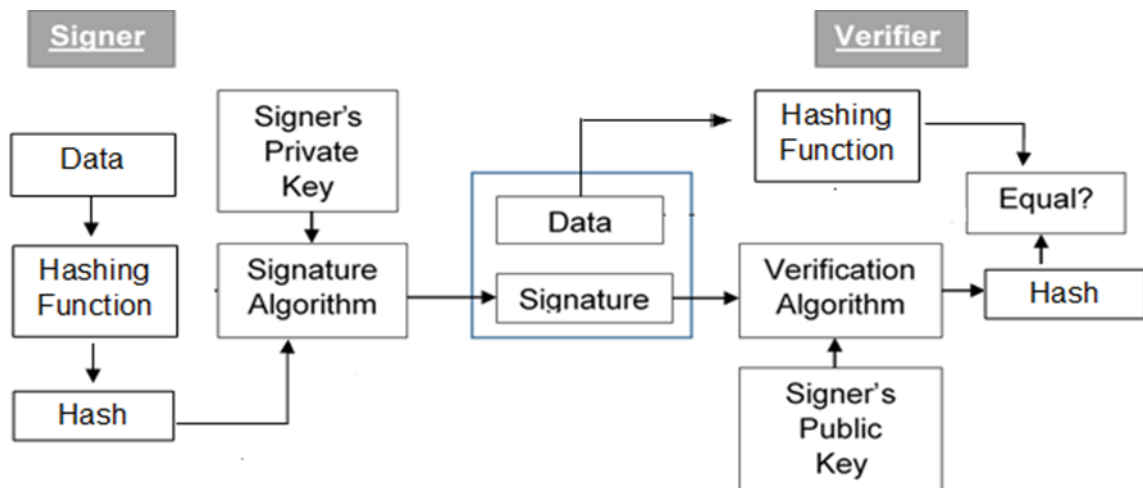
Similarly, a digital signature is a technique that binds a person/entity to the digital data. This binding can be independently verified by receiver as well as any third party.

Digital signature is a cryptographic value that is calculated from the data and a secret key known only by the signer.

In real world, the receiver of message needs assurance that the message belongs to the sender and he should not be able to repudiate the origination of that message. This requirement is very crucial in business applications, since likelihood of a dispute over exchanged data is very high.

## Model of Digital Signature

As mentioned earlier, the digital signature scheme is based on public key cryptography. The model of digital signature scheme is depicted in the following illustration:



The following points explain the entire process in detail:

- Each person adopting this scheme has a public-private key pair.
- Generally, the key pairs used for encryption/decryption and signing/verifying are different. The private key used for signing is referred to as the signature key and the public key as the verification key.

- Signer feeds data to the hash function and generates hash of data.
- Hash value and signature key are then fed to the signature algorithm which produces the digital signature on given hash. Signature is appended to the data and then both are sent to the verifier.
- Verifier feeds the digital signature and the verification key into the verification algorithm. The verification algorithm gives some value as output.
- Verifier also runs same hash function on received data to generate hash value.
- For verification, this hash value and output of verification algorithm are compared. Based on the comparison result, verifier decides whether the digital signature is valid.
- Since digital signature is created by 'private' key of signer and no one else can have this key; the signer cannot repudiate signing the data in future.

It should be noticed that instead of signing data directly by signing algorithm, usually a hash of data is created. Since the hash of data is a unique representation of data, it is sufficient to sign the hash in place of data. The most important reason of using hash instead of data directly for signing is efficiency of the scheme.

Let us assume RSA is used as the signing algorithm. As discussed in public key encryption chapter, the encryption/signing process using RSA involves modular exponentiation.

Signing large data through modular exponentiation is computationally expensive and time consuming. The hash of the data is a relatively small digest of the data, hence **signing a hash is more efficient than signing the entire data**.

## Importance of Digital Signature

---

Out of all cryptographic primitives, the digital signature using public key cryptography is considered as very important and useful tool to achieve information security.

Apart from ability to provide non-repudiation of message, the digital signature also provides message authentication and data integrity. Let us briefly see how this is achieved by the digital signature:

- **Message authentication** – When the verifier validates the digital signature using public key of a sender, he is assured that signature has been created only by sender who possess the corresponding secret private key and no one else.



- **Data Integrity** – In case an attacker has access to the data and modifies it, the digital signature verification at receiver end fails. The hash of modified data and the output provided by the verification algorithm will not match. Hence, receiver can safely deny the message assuming that data integrity has been breached.
- **Non-repudiation** – Since it is assumed that only the signer has the knowledge of the signature key, he can only create unique signature on a given data. Thus the receiver can present data and the digital signature to a third party as evidence if any dispute arises in the future.

By adding public-key encryption to digital signature scheme, we can create a cryptosystem that can provide the four essential elements of security namely: Privacy, Authentication, Integrity, and Non-repudiation.

### Encryption with Digital Signature

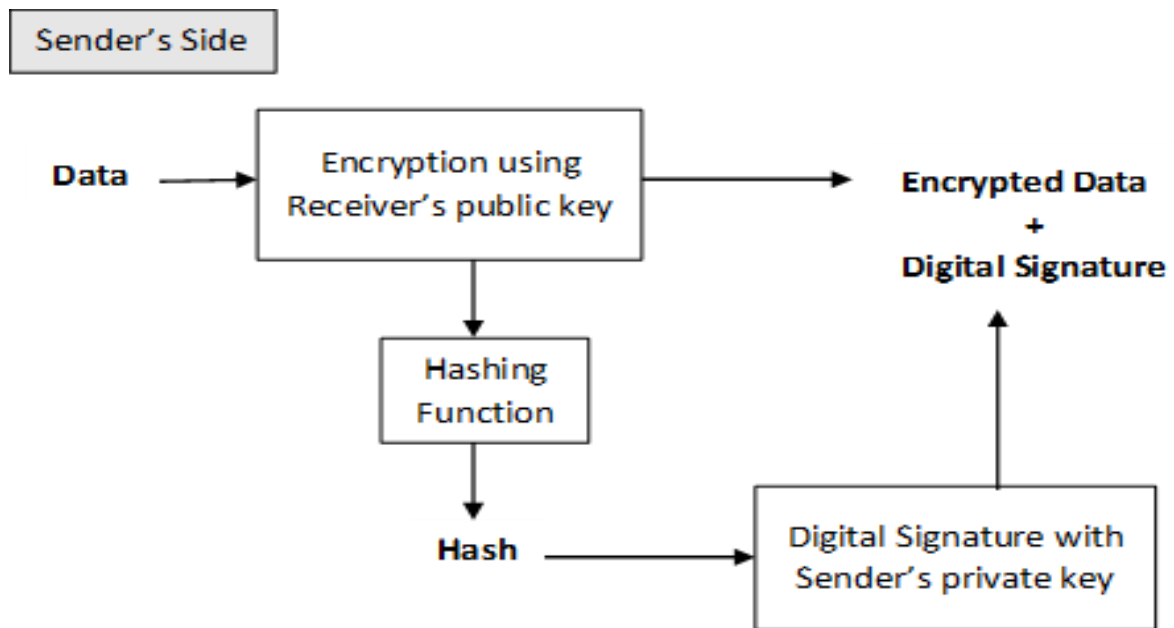
---

In many digital communications, it is desirable to exchange an encrypted messages than plaintext to achieve confidentiality. In public key encryption scheme, a public (encryption) key of sender is available in open domain, and hence anyone can spoof his identity and send any encrypted message to the receiver.

This makes it essential for users employing PKC for encryption to seek digital signatures along with encrypted data to be assured of message authentication and non-repudiation.

This can archived by combining digital signatures with encryption scheme. Let us briefly discuss how to achieve this requirement. There are **two possibilities, sign-then-encrypt** and **encrypt-then-sign**.

However, the crypto system based on sign-then-encrypt can be exploited by receiver to spoof identity of sender and sent that data to third party. Hence, this method is not preferred. The process of encrypt-then-sign is more reliable and widely adopted. This is depicted in the following illustration:



The receiver after receiving the encrypted data and signature on it, first verifies the signature using sender's public key. After ensuring the validity of the signature, he then retrieves the data through decryption using his private key.

# 18. PUBLIC KEY INFRASTRUCTURE

The most distinct feature of Public Key Infrastructure (PKI) is that it uses a pair of keys to achieve the underlying security service. The key pair comprises of private key and public key.

Since the public keys are in open domain, they are likely to be abused. It is, thus, necessary to establish and maintain some kind of trusted infrastructure to manage these keys.

## Key Management

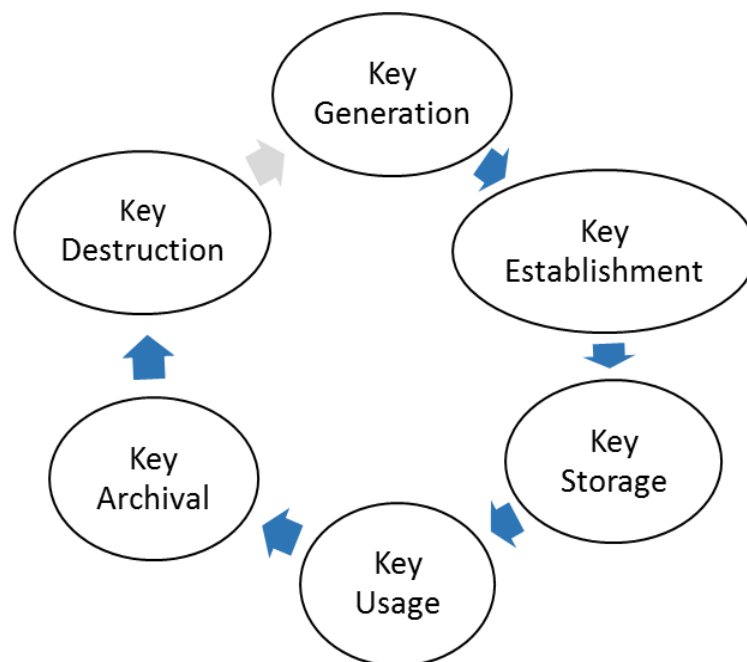
---

It goes without saying that the security of any cryptosystem depends upon how securely its keys are managed. Without secure procedures for the handling of cryptographic keys, the benefits of the use of strong cryptographic schemes are potentially lost.

It is observed that cryptographic schemes are rarely compromised through weaknesses in their design. However, they are often compromised through poor key management.

There are some important aspects of key management which are as follows:

- Cryptographic keys are nothing but special pieces of data. Key management refers to the secure administration of cryptographic keys.
- Key management deals with entire key lifecycle as depicted in the following illustration:



- There are two specific requirements of key management for public key cryptography.
  - o **Secrecy of private keys.** Throughout the key lifecycle, secret keys must remain secret from all parties except those who are owner and are authorized to use them.
  - o **Assurance of public keys.** In public key cryptography, the public keys are in open domain and seen as public pieces of data. By default there are no assurances of whether a public key is correct, with whom it can be associated, or what it can be used for. Thus key management of public keys needs to focus much more explicitly on assurance of purpose of public keys.

The most crucial requirement of 'assurance of public key' can be achieved through the public-key infrastructure (PKI), a key management systems for supporting public-key cryptography.

### Public Key Infrastructure (PKI)

---

PKI provides assurance of public key. It provides the identification of public keys and their distribution. An anatomy of PKI comprises of the following components.

- Public Key Certificate, commonly referred to as 'digital certificate'.
- Private Key tokens.
- Certification Authority.
- Registration Authority.
- Certificate Management System.

### Digital Certificate

---

For analogy, a certificate can be considered as the ID card issued to the person. People use ID cards such as a driver's license, passport to prove their identity. A digital certificate does the same basic thing in the electronic world, but with one difference.

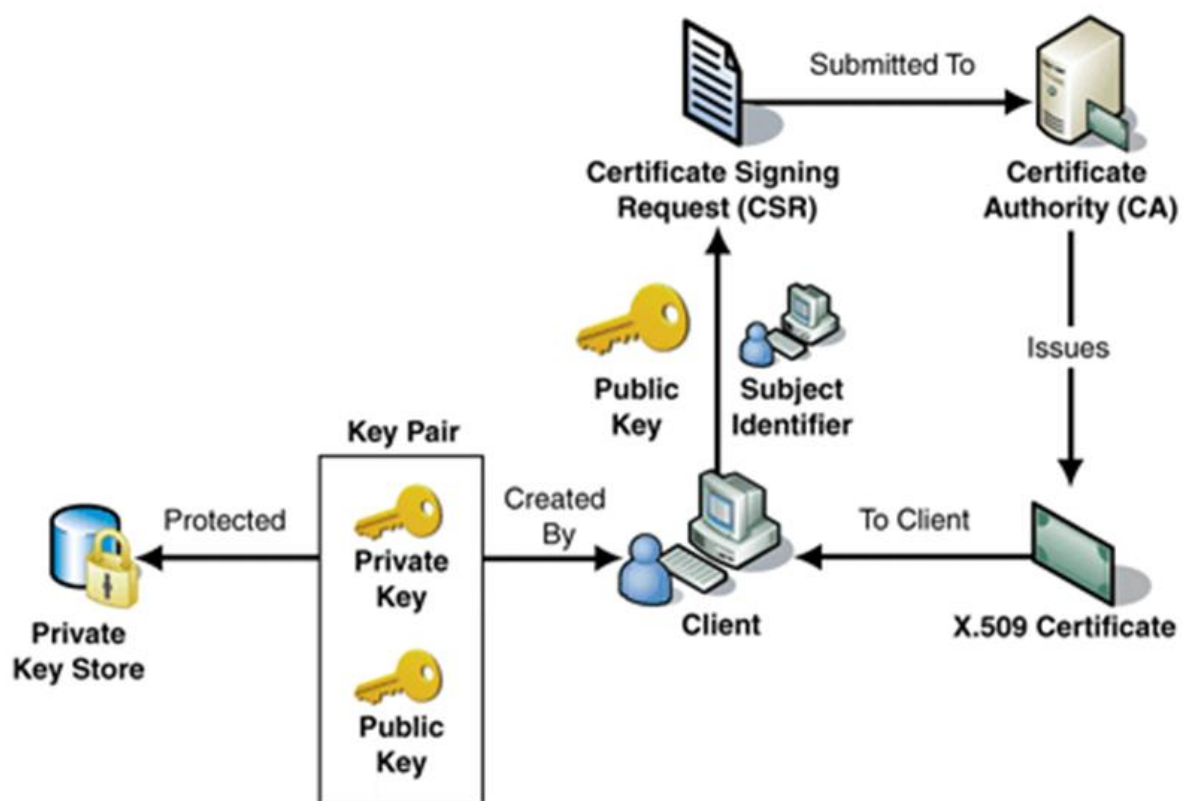
Digital Certificates are not only issued to people but they can be issued to computers, software packages or anything else that need to prove the identity in the electronic world.

- Digital certificates are based on the ITU standard X.509 which defines a standard certificate format for public key certificates and certification validation. Hence digital certificates are sometimes also referred to as X.509 certificates.

Public key pertaining to the user client is stored in digital certificates by The Certification Authority (CA) along with other relevant information such as client information, expiration date, usage, issuer etc.

- CA digitally signs this entire information and includes digital signature in the certificate.
- Anyone who needs the assurance about the public key and associated information of client, he carries out the signature validation process using CA's public key. Successful validation assures that the public key given in the certificate belongs to the person whose details are given in the certificate.

The process of obtaining Digital Certificate by a person/entity is depicted in the following illustration.



As shown in the illustration, the CA accepts the application from a client to certify his public key. The CA, after duly verifying identity of client, issues a digital certificate to that client.

## Certifying Authority (CA)

As discussed above, the CA issues certificate to a client and assist other users to verify the certificate. The CA takes responsibility for identifying correctly the identity of the client asking for a certificate to be issued, and ensures that the information contained within the certificate is correct and digitally signs it.

## Key Functions of CA

The key functions of a CA are as follows:

- **Generating key pairs** – The CA may generate a key pair independently or jointly with the client.
- **Issuing digital certificates** – The CA could be thought of as the PKI equivalent of a passport agency - the CA issues a certificate after client provides the credentials to confirm his identity. The CA then signs the certificate to prevent modification of the details contained in the certificate.
- **Publishing Certificates** – The CA need to publish certificates so that users can find them. There are two ways of achieving this. One is to publish certificates in the equivalent of an electronic telephone directory. The other is to send your certificate out to those people you think might need it by one means or another.
- **Verifying Certificates** – The CA makes its public key available in environment to assist verification of his signature on clients' digital certificate.
- **Revocation of Certificates** – At times, CA revokes the certificate issued due to some reason such as compromise of private key by user or loss of trust in the client. After revocation, CA maintains the list of all revoked certificate that is available to the environment.

## Classes of Certificates

There are four typical classes of certificate:

- **Class 1:** These certificates can be easily acquired by supplying an email address.
- **Class 2:** These certificates require additional personal information to be supplied.
- **Class 3:** These certificates can only be purchased after checks have been made about the requestor's identity.
- **Class 4:** They may be used by governments and financial organizations needing very high levels of trust.

## Registration Authority (RA)

CA may use a third-party Registration Authority (RA) to perform the necessary checks on the person or company requesting the certificate to confirm their identity. The RA may appear to the client as a CA, but they do not actually sign the certificate that is issued.

## **Certificate Management System (CMS)**

It is the management system through which certificates are published, temporarily or permanently suspended, renewed, or revoked. Certificate management systems do not normally delete certificates because it may be necessary to prove their status at a point in time, perhaps for legal reasons. A CA along with associated RA runs certificate management systems to be able to track their responsibilities and liabilities.

## **Private Key Tokens**

While the public key of a client is stored on the certificate, the associated secret private key can be stored on the key owner's computer. This method is generally not adopted. If an attacker gains access to the computer, he can easily gain access to private key. For this reason, a private key is stored on secure removable storage token access to which is protected through a password.

Different vendors often use different and sometimes proprietary storage formats for storing keys. For example, Entrust uses the proprietary .epf format, while Verisign, GlobalSign, and Baltimore use the standard .p12 format.

## **Hierarchy of CA**

---

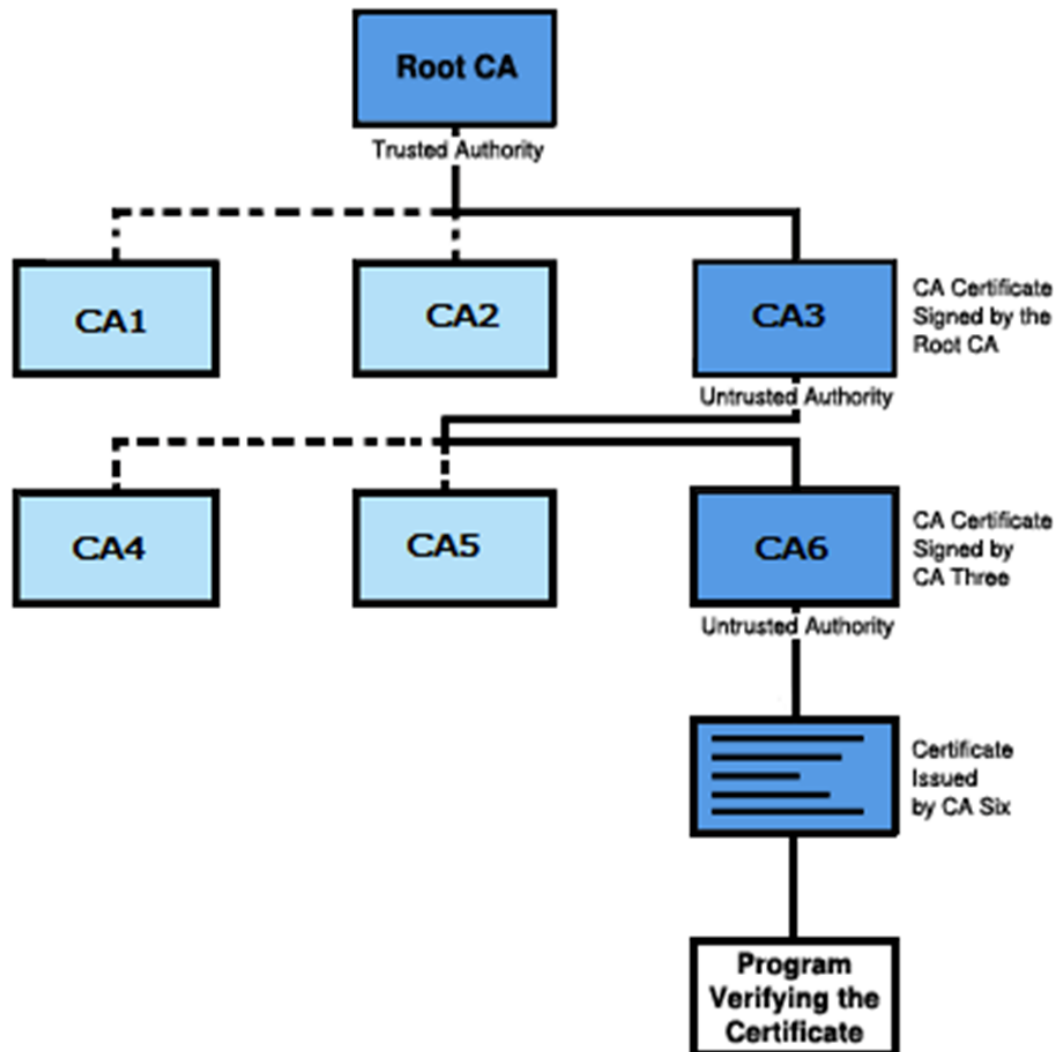
With vast networks and requirements of global communications, it is practically not feasible to have only one trusted CA from whom all users obtain their certificates. Secondly, availability of only one CA may lead to difficulties if CA is compromised.

In such case, the hierarchical certification model is of interest since it allows public key certificates to be used in environments where two communicating parties do not have trust relationships with the same CA.

- The root CA is at the top of the CA hierarchy and the root CA's certificate is a self-signed certificate.
- The CAs, which are directly subordinate to the root CA (For example, CA1 and CA2) have CA certificates that are signed by the root CA.
- The CAs under the subordinate CAs in the hierarchy (For example, CA5 and CA6) have their CA certificates signed by the higher-level subordinate CAs.

Certificate authority (CA) hierarchies are reflected in certificate chains. A certificate chain traces a path of certificates from a branch in the hierarchy to the root of the hierarchy.

The following illustration shows a CA hierarchy with a certificate chain leading from an entity certificate through two subordinate CA certificates (CA6 and CA3) to the CA certificate for the root CA.



Verifying a certificate chain is the process of ensuring that a specific certificate chain is valid, correctly signed, and trustworthy. The following procedure verifies a certificate chain, beginning with the certificate that is presented for authentication:

- A client whose authenticity is being verified supplies his certificate, generally along with the chain of certificates up to Root CA.
- Verifier takes the certificate and validates by using public key of issuer. The issuer's public key is found in the issuer's certificate which is in the chain next to client's certificate.
- Now if the higher CA who has signed the issuer's certificate, is trusted by the verifier, verification is successful and stops here.
- Else, the issuer's certificate is verified in a similar manner as done for client in above steps. This process continues till either trusted CA is found in between or else it continues till Root CA.



# 19. CRYPTOGRAPHY – BENEFITS AND DRAWBACKS

Nowadays, the networks have gone global and information has taken the digital form of bits and bytes. Critical information now gets stored, processed and transmitted in digital form on computer systems and open communication channels.

Since information plays such a vital role, adversaries are targeting the computer systems and open communication channels to either steal the sensitive information or to disrupt the critical information system.

Modern cryptography provides a robust set of techniques to ensure that the malevolent intentions of the adversary are thwarted while ensuring the legitimate users get access to information. Here in this chapter, we will discuss the benefits that we draw from cryptography, its limitations, as well as the future of cryptography.

## Cryptography – Benefits

---

Cryptography is an essential information security tool. It provides the four most basic services of information security:

- **Confidentiality** – Encryption technique can guard the information and communication from unauthorized revelation and access of information.
- **Authentication** – The cryptographic techniques such as MAC and digital signatures can protect information against spoofing and forgeries.
- **Data Integrity** – The cryptographic hash functions are playing vital role in assuring the users about the data integrity.
- **Non-repudiation** – The digital signature provides the non-repudiation service to guard against the dispute that may arise due to denial of passing message by the sender.

All these fundamental services offered by cryptography has enabled the conduct of business over the networks using the computer systems in extremely efficient and effective manner.

## Cryptography – Drawbacks

---

Apart from the four fundamental elements of information security, there are other issues that affect the effective use of information:

- A strongly encrypted, authentic, and digitally signed information can be **difficult to access even for a legitimate user** at a crucial time of

decision-making. The network or the computer system can be attacked and rendered non-functional by an intruder.

- **High availability**, one of the fundamental aspects of information security, cannot be ensured through the use of cryptography. Other methods are needed to guard against the threats such as denial of service or complete breakdown of information system.
- Another fundamental need of information security of **selective access control** also cannot be realized through the use of cryptography. Administrative controls and procedures are required to be exercised for the same.
- Cryptography does not guard against the vulnerabilities and **threats that emerge from the poor design of systems**, protocols, and procedures. These need to be fixed through proper design and setting up of a defensive infrastructure.
- Cryptography comes at cost. The cost is in terms of time and money:
  - o Addition of cryptographic techniques in the information processing leads to delay.
  - o The use of public key cryptography requires setting up and maintenance of public key infrastructure requiring the handsome financial budget.
- The security of cryptographic technique is based on the computational difficulty of mathematical problems. Any breakthrough in solving such mathematical problems or increasing the computing power can render a cryptographic technique vulnerable.

## Future of Cryptography

---

**Elliptic Curve Cryptography** (ECC) has already been invented but its advantages and disadvantages are not yet fully understood. ECC allows to perform encryption and decryption in a drastically lesser time, thus allowing a higher amount of data to be passed with equal security. However, as other methods of encryption, ECC must also be tested and proven secure before it is accepted for governmental, commercial, and private use.

**Quantum computation** is the new phenomenon. While modern computers store data using a binary format called a "bit" in which a "1" or a "0" can be stored; a quantum computer stores data using a quantum superposition of multiple states. These multiple valued states are stored in "quantum bits" or "qubits". This allows the computation of numbers to be several orders of magnitude faster than traditional transistor processors.

To comprehend the power of quantum computer, consider RSA-640, a number with 193 digits, which can be factored by eighty 2.2GHz computers over the span

of 5 months, one quantum computer would factor in less than 17 seconds. Numbers that would typically take billions of years to compute could only take a matter of hours or even minutes with a fully developed quantum computer.

In view of these facts, modern cryptography will have to look for computationally harder problems or devise completely new techniques of achieving the goals presently served by modern cryptography.