Dr. D. Y. Patil Pratishthan's

Institute for Advanced Computing &Software Development

IACSD

Public Key Infrastructure

INDEX

1. Origin of Cryptography....1
2. Modern Cryptography....4
3. Context of Cryptography....5
4. Security Services of Cryptography....6
5. Cryptosystems....8
6. Attacks on Cryptosystems....16
7. Traditional Ciphers....23
8. Modern Symmetric Key Encryption....33
9. Data Encryption Standard....39
10. Advanced Encryption Standard....46
11. Block Cipher Modes of Operation....50
12. Public Key Encryption....57
13. Data Integrity in Cryptography....67
14. Cryptography Hash functions....68
15. Message Authentication....76
16. Cryptography Digital signatures....79
17. Public Key Infrastructure....83
18. Cryptography - Benefits & Drawbacks....90
19. Diffie- Hellman key exchange....92
20. Attacks against encryption....99
21. HMAC....102
22. Public key cryptography standards....104
23. Strong authentication....105
24. Single factor authentication....106
25. Single sign on solutions....108
26. OpenID....110
27. OAUTH....111
28. Authentication protocols....113
29. FIDO....118
30. Zero Trust Architecture....119
31. Securing Websites & Email....122
32. Introduction to Block chain technology....127

Origin of Cryptography

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 informationsecurity 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 cryptographyas 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 totransmit 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 wasa common choice), the recipient of this message would then shift the letters back bythe 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 concealingit, 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 ledthe 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 15th 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 19th century, cryptography evolved from the ad hoc approaches to encryption to the more sophisticated art and science of information security.
- In the early 20th 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.

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

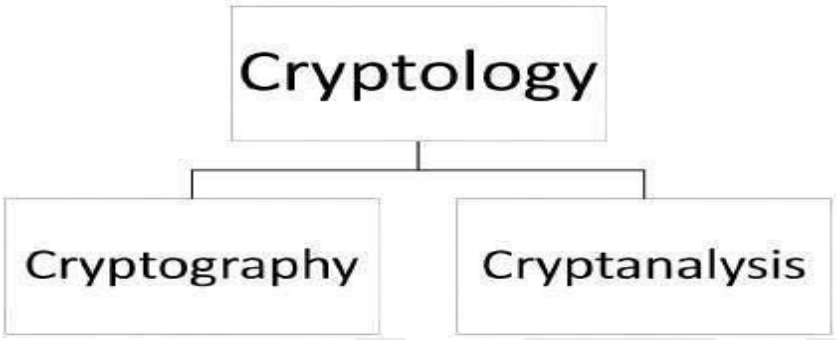
There are three major characteristics that separate modern cryptography from the classical approach.

Classic Cryptography	Modern Cryptography
It manipulates traditional characters, i.e., letters anddigits 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 secrete 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 ofa 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 accidently. 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 fordetecting whether data has been manipulated in an unauthorized manner.

Authentication

Authentication provides the identification of the originator. It confirms to the receiverthat 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.

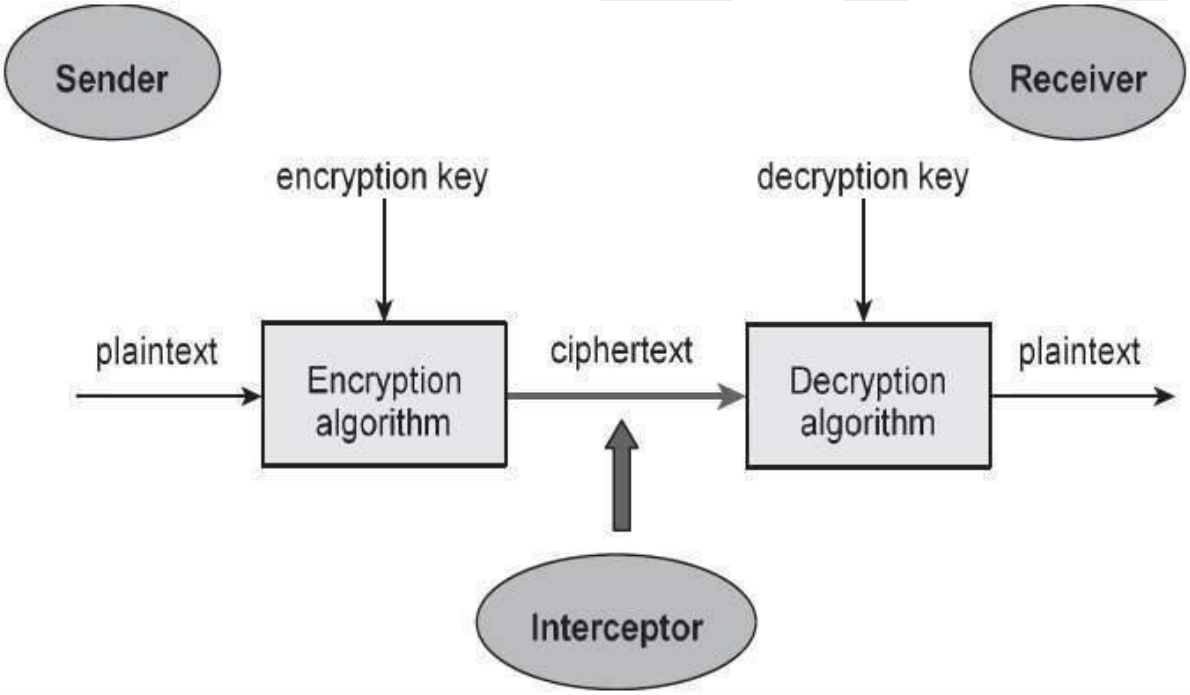
Primitives ⇨ Service ⇩	Encryption	Hash Function	MAC	Digital Signature
Confidentiality	Yes	No	No	No
Integrity	No	Sometimes	Yes	Yes
Authentication	No	No	Yes	Yes
Non Reputation	No	No	Sometimes	Yes

Note − Cryptographic primitives are intricately related and they are often combinedto achieve a set of desired security services from a cryptosystem.

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 notguarded. 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 aplaintext. 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 decryptionkey 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 theciphertext 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

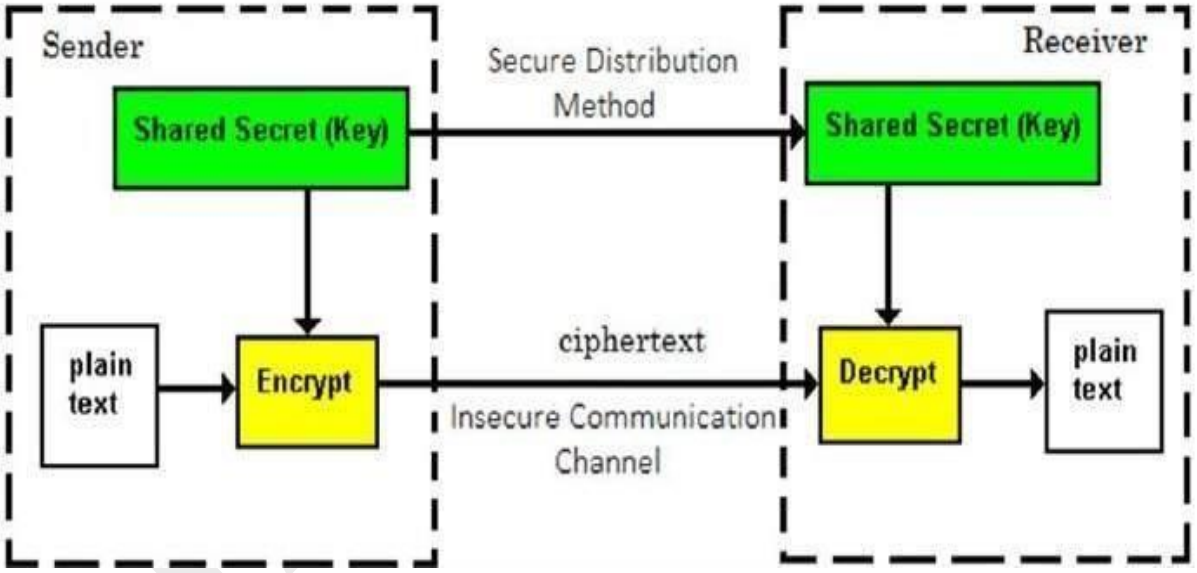
The main difference between these cryptosystems is the relationship between the encryption and the decryption key. Logically, in any cryptosystem, both the keys areclosely associated. It is practically impossible to decrypt the ciphertext with the key that is unrelated to the encryption key.

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 overasymmetric 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 × (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

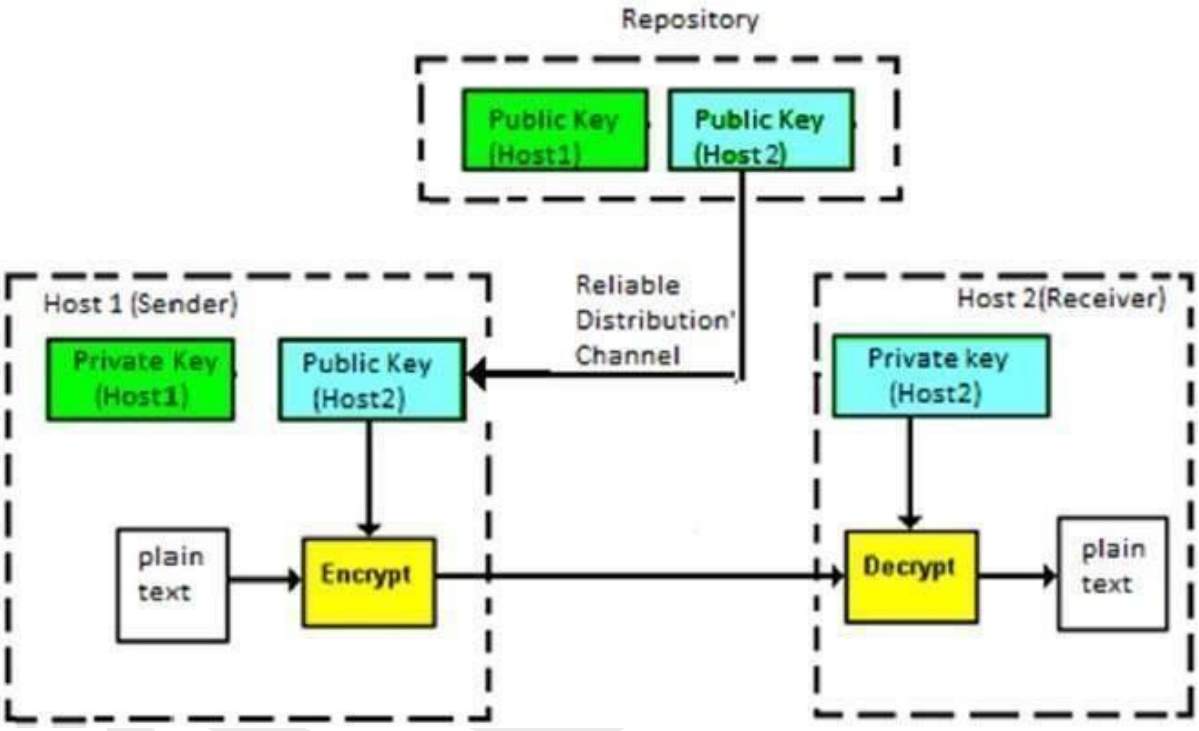
There are two restrictive challenges of employing symmetric key cryptography.

- **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 20th century to come over the necessity of pre-shared secret key between communicating persons. The salient features of this encryption scheme are as follows −

- Every user in this system needs to have a pair of dissimilar keys, **private key** and **public key**. These keys are mathematically related − when one keyis used for encryption, the other can decrypt the ciphertext back to the original plaintext.
- It requires to put the public key in public repository and the private key as a well-guarded secret. Hence, this scheme of encryption is also called **Public Key Encryption**.

- Though public and private keys of the user are related, it is computationallynot feasible to find one from another. This is a strength of this scheme.
- When Host1 needs to send data to Host2, he obtains the public keyof Host2 from repository, encrypts the data, and transmits.
- Host2 uses his private key to extract the plaintext.
- Length of Keys (number of bits) in this encryption is large and hence, the process of encryption-decryption is slower than symmetric key encryption.
- Processing power of computer system required to run asymmetric algorithm is higher.

Symmetric cryptosystems are a natural concept. In contrast, public-key cryptosystems are quite difficult to comprehend.

You may think, how can the encryption key and the decryption key are 'related', andyet it is impossible to determine the decryption key from the encryption key? The answer lies in the mathematical concepts. It is possible to design a cryptosystem whose keys have this property. The concept of public-key cryptography is relatively new. There are fewer public-key algorithms known than symmetric algorithms.

Challenge of Public Key Cryptosystem

Public-key cryptosystems have one significant challenge − the user needs to trust that the public key that he is using in communications with a person really is the public key of that person and has not been spoofed by a malicious third party.

This is usually accomplished through a Public Key Infrastructure (PKI) consisting a trusted third party. The third party securely manages and attests to the authenticity of public keys. When the third party is requested to provide the public key for any communicating person X, they are trusted to provide the correct public key.

The third party satisfies itself about user identity by the process of attestation, notarization, or some other process − that X is the one and only, or globally unique,
X. The most common method of making the verified public keys available is to embed them in a certificate which is digitally signed by the trusted third party.

Relation between Encryption Schemes

A summary of basic key properties of two types of cryptosystems is given below −

	Symmetric Cryptosystems	Public Key Cryptosystems
Relation between Keys	Same	Different, but mathematically related
Encryption Key	Symmetric	Public
Decryption Key	Symmetric	Private

Due to the advantages and disadvantage of both the systems, symmetric key and public-key cryptosystems are often used together in the practical information security systems.

Kerckhoff's Principle for Cryptosystem

In the 19th century, a Dutch cryptographer A. Kerckhoff furnished the requirements of a good cryptosystem. Kerckhoff stated that a cryptographic system should be secure even if everything about the system, except the key, is public knowledge. The six design principles defined by Kerckhoff for cryptosystem are −

- The cryptosystem should be unbreakable practically, if not mathematically.
- Falling of the cryptosystem in the hands of an intruder should not lead to any compromise of the system, preventing any inconvenience to the user.
- The key should be easily communicable, memorable, and changeable.
- The ciphertext should be transmissible by telegraph, an unsecure channel.
- The encryption apparatus and documents should be portable and operableby a single person.

- Finally, it is necessary that the system be easy to use, requiring neither mental strain nor the knowledge of a long series of rules to observe.

The second rule is currently known as **Kerckhoff principle**. It is applied in virtually all the contemporary encryption algorithms such as DES, AES, etc. These public algorithms are considered to be thoroughly secure. The security of the encrypted message depends solely on the security of the secret encryption key.

Keeping the algorithms secret may act as a significant barrier to cryptanalysis. However, keeping the algorithms secret is possible only when they are used in a strictly limited circle.

In modern era, cryptography needs to cater to users who are connected to the Internet. In such cases, using a secret algorithm is not feasible, hence Kerckhoff principles became essential guidelines for designing algorithms in moderncryptography.

Attacks On Cryptosystems

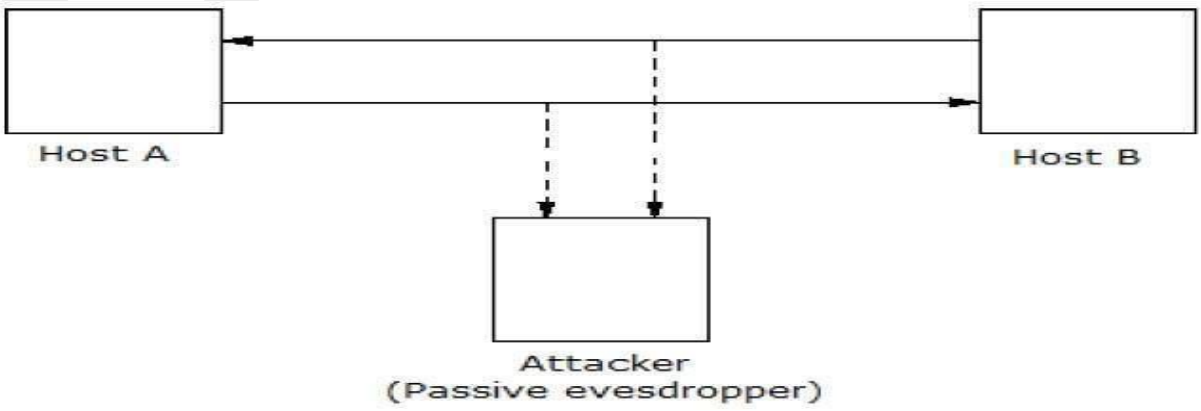
In the present era, not only business but almost all the aspects of human life are driven by information. Hence, it has become imperative to protect useful informationfrom malicious activities such as attacks. Let us consider the types of attacks to which information is typically subjected to.

Attacks are typically categorized based on the action performed by the attacker. An attack, thus, can be **passive** or **active**.

Passive Attacks

The main goal of a passive attack is to obtain **unauthorized access to the information**. For example, actions such as intercepting and eavesdropping on the communication channel can be regarded as passive attack.

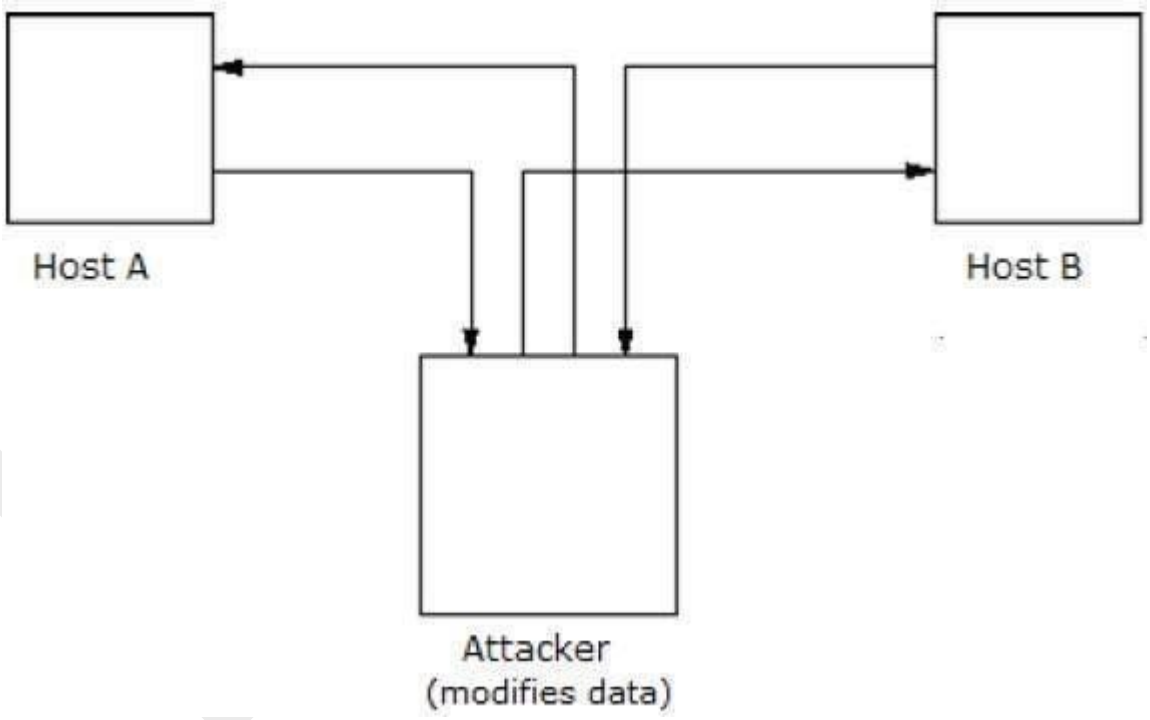
These actions are passive in nature, as they neither affect information nor disrupt the communication channel. A passive attack is often seen as stealing information. The only difference in stealing physical goods and stealing information is that theft of data still leaves the owner in possession of that data. Passive information attackis thus more dangerous than stealing of goods, as information theft may go unnoticed by the owner.



Active Attacks

An active attack involves changing the information in some way by conducting some process on the information. For example,

- Modifying the information in an unauthorized manner.
- Initiating unintended or unauthorized transmission of information.
- Alteration of authentication data such as originator name or timestamp associated with information
- Unauthorized deletion of data.
- Denial of access to information for legitimate users (denial of service).



Cryptography provides many tools and techniques for implementing cryptosystemscapable of preventing most of the attacks described above.

Assumptions of Attacker

Let us see the prevailing environment around cryptosystems followed by the typesof attacks employed to break these systems −

Environment around Cryptosystem

While considering possible attacks on the cryptosystem, it is necessary to know the cryptosystems environment. The attacker's assumptions and knowledge about the environment decides his capabilities.

In cryptography, the following three assumptions are made about the security environment and attacker's capabilities.

Details of the Encryption Scheme

The design of a cryptosystem is based on the following two cryptography algorithms

-

- **Public Algorithms** − With this option, all the details of the algorithm are in the public domain, known to everyone.
- **Proprietary algorithms** − The details of the algorithm are only known by thesystem designers and users.

In case of proprietary algorithms, security is ensured through obscurity. Private algorithms may not be the strongest algorithms as they are developed in-house and may not be extensively investigated for weakness.

Secondly, they allow communication among closed group only. Hence they are not suitable for modern communication where people communicate with large number of known or unknown entities. Also, according to Kerckhoff's principle, the algorithm is preferred to be public with strength of encryption lying in the key.

Thus, the first assumption about security environment is that the **encryption algorithm is known to the attacker**.

Availability of Ciphertext

We know that once the plaintext is encrypted into ciphertext, it is put on unsecure public channel (say email) for transmission. Thus, the attacker can obviously assume that it has **access to the ciphertext generated by the cryptosystem**.

Availability of Plaintext and Ciphertext

This assumption is not as obvious as other. However, there may be situations where an attacker can have **access to plaintext and corresponding ciphertext**. Some such possible circumstances are −

- The attacker influences the sender to convert plaintext of his choice and obtains the ciphertext.
- The receiver may divulge the plaintext to the attacker inadvertently. The attacker has access to corresponding ciphertext gathered from open channel.
- In a public-key cryptosystem, the encryption key is in open domain and is known to any potential attacker. Using this key, he can generate pairs of corresponding plaintexts and ciphertexts.

Cryptographic Attacks

The basic intention of an attacker is to break a cryptosystem and to find the plaintext from the ciphertext. To obtain the plaintext, the attacker only needs to find out the secret decryption key, as the algorithm is already in public domain.

Hence, he applies maximum effort towards finding out the secret key used in the cryptosystem. Once the attacker is able to determine the key, the attacked systemis considered as broken or compromised.

Based on the methodology used, attacks on cryptosystems are categorized as follows −

- **Ciphertext Only Attacks (COA)** − In this method, the attacker has access toa set of ciphertext(s). He does not have access to corresponding plaintext.

COA is said to be successful when the corresponding plaintext can bedetermined from a given set of ciphertext. Occasionally, the encryption key can be determined from this attack. Modern cryptosystems are guarded against ciphertext-only attacks.

- **Known Plaintext Attack (KPA)** − In this method, the attacker knows the plaintext for some parts of the ciphertext. The task is to decrypt the rest of the ciphertext using this information. This may be done by determining the key or via some other method. The best example of this attack is linear cryptanalysis against block ciphers.
- **Chosen Plaintext Attack (CPA)** − In this method, the attacker has the text ofhis choice encrypted. So he has the ciphertext-plaintext pair of his choice. This simplifies his task of determining the encryption key. An example of this attack is differential cryptanalysis applied against block ciphers as well as hash functions. A popular public key cryptosystem, RSA is also vulnerable to chosen-plaintext attacks.
- **Dictionary Attack** − This attack has many variants, all of which involve compiling a 'dictionary'. In simplest method of this attack, attacker builds a dictionary of ciphertexts and corresponding plaintexts that he has learnt over a period of time. In future, when an attacker gets the ciphertext, he refers the dictionary to find the corresponding plaintext.
- **Brute Force Attack (BFA)** − In this method, the attacker tries to determine the key by attempting all possible keys. If the key is 8 bits long, then the number of possible keys is $2^8 = 256$. The attacker knows the ciphertext and the algorithm, now he attempts all the 256 keys one by one for decryption. The time to complete the attack would be very high if the key is long.
- **Birthday Attack** − This attack is a variant of brute-force technique. It is usedagainst the cryptographic hash function. When students in a class are asked about their birthdays, the answer is one of the possible 365 dates. Let us assume the first student's birthdate is 3rd Aug. Then to find the next student whose birthdate is 3rd Aug, we need to enquire $1.25^{*} \cdot \sqrt{365} \approx 25$ students.

Similarly, if the hash function produces 64 bit hash values, the possible hash values are $1.8x10^{19}$. By repeatedly evaluating the function for different inputs, the same output is expected to be obtained after about $5.1x10^{9}$ random inputs.

If the attacker is able to find two different inputs that give the same hash value, it is a **collision** and that hash function is said to be broken.

- **Man in Middle Attack (MIM)** − The targets of this attack are mostly public key cryptosystems where key exchange is involved before communication takes place.
    - Host A wants to communicate to host B, hence requests public key of B.
    - An attacker intercepts this request and sends his public key instead.
    - Thus, whatever host A sends to host B, the attacker is able to read.
    - In order to maintain communication, the attacker re-encrypts the data after reading with his public key and sends to B.
    - The attacker sends his public key as A's public key so that B takes it as if it is taking it from A.
- **Side Channel Attack (SCA)** − This type of attack is not against any particular type of cryptosystem or algorithm. Instead, it is launched to exploit the weakness in physical implementation of the cryptosystem.
- **Timing Attacks** − They exploit the fact that different computations take different times to compute on processor. By measuring such timings, it is be possible to know about a particular computation the processor is carrying out. For example, if the encryption takes a longer time, it indicates that the secret key is long.
- **Power Analysis Attacks** − These attacks are similar to timing attacks exceptthat the amount of power consumption is used to obtain information about the nature of the underlying computations.

- **Fault analysis Attacks** − In these attacks, errors are induced in the cryptosystem and the attacker studies the resulting output for useful information.

Practicality of Attacks

The attacks on cryptosystems described here are highly academic, as majority of them come from the academic community. In fact, many academic attacks involve quite unrealistic assumptions about environment as well as the capabilities of the attacker. For example, in chosen-ciphertext attack, the attacker requires an impractical number of deliberately chosen plaintext-ciphertext pairs. It may not be practical altogether.

Nonetheless, the fact that any attack exists should be a cause of concern, particularly if the attack technique has the potential for improvement.

Traditional Ciphers

In the second chapter, we discussed the fundamentals of modern cryptography. Weequated cryptography with a toolkit where various cryptographic techniques are considered as the basic tools. One of these tools is the Symmetric Key Encryption where the key used for encryption and decryption is the same.

In this chapter, we discuss this technique further and its applications to develop various cryptosystems.

Earlier Cryptographic Systems

Before proceeding further, you need to know some facts about historical cryptosystems −

- All of these systems are **based on symmetric key encryption** scheme.
- The only security service these systems provide is confidentiality of information.
- Unlike modern systems which are digital and treat data as binary numbers, the earlier systems worked on alphabets as basic element.

These earlier cryptographic systems are also referred to as Ciphers. In general, a cipher is simply just a set of steps (an algorithm) for performing both an encryption, and the corresponding decryption.

Caesar Cipher

It is a mono-alphabetic cipher wherein each letter of the plaintext is substituted by another letter to form the ciphertext. It is a simplest form of substitution cipher scheme.

This cryptosystem is generally referred to as the **Shift Cipher**. The concept is to replace each alphabet by another alphabet which is ‘shifted’ by some fixed number between 0 and 25.

For this type of scheme, both sender and receiver agree on a 'secret shift number' for shifting the alphabet. This number which is between 0 and 25 becomes the key of encryption.

The name 'Caesar Cipher' is occasionally used to describe the Shift Cipher when the 'shift of three' is used.

Process of Shift Cipher

- In order to encrypt a plaintext letter, the sender positions the sliding ruler underneath the first set of plaintext letters and slides it to LEFT by the number of positions of the secret shift.
- The plaintext letter is then encrypted to the ciphertext letter on the sliding ruler underneath. The result of this process is depicted in the followingillustration for an agreed shift of three positions. In this case, the plaintext 'tutorial' is encrypted to the ciphertext 'WXWRULDO'. Here is the ciphertext alphabet for a Shift of 3 −

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

- On receiving the ciphertext, the receiver who also knows the secret shift, positions his sliding ruler underneath the ciphertext alphabet and slides it to RIGHT by the agreed shift number, 3 in this case.
- He then replaces the ciphertext letter by the plaintext letter on the sliding ruler underneath. Hence the ciphertext 'WXWRULDO' is decrypted to 'tutorial'. To decrypt a message encoded with a Shift of 3, generate the plaintext alphabet using a shift of '-3' as shown below −

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

Security Value

Caesar Cipher is **not a secure** cryptosystem because there are only 26 possible keys to try out. An attacker can carry out an exhaustive key search with available limited computing resources.

Simple Substitution Cipher

It is an improvement to the Caesar Cipher. Instead of shifting the alphabets by some number, this scheme uses some permutation of the letters in alphabet.

For example, A.B…..Y.Z and Z.Y……B.A are two obvious permutation of all the letters in alphabet. Permutation is nothing but a jumbled up set of alphabets.

With 26 letters in alphabet, the possible permutations are 26! (Factorial of 26) whichis equal to $4x10^{26}$. The sender and the receiver may choose any one of these possible permutation as a ciphertext alphabet. This permutation is the secret key of the scheme.

Process of Simple Substitution Cipher

- Write the alphabets A, B, C,...,Z in the natural order.
- The sender and the receiver decide on a randomly selected permutation of the letters of the alphabet.
- Underneath the natural order alphabets, write out the chosen permutation of the letters of the alphabet. For encryption, sender replaces each plaintext letters by substituting the permutation letter that is directly beneath it in the table. This process is shown in the following illustration. In this example, the chosen permutation is K,D, G, ..., O. The plaintext ‘point’ is encrypted to ‘MJBXZ’.

Here is a jumbled Ciphertext alphabet, where the order of the ciphertext letters is a key.

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

- On receiving the ciphertext, the receiver, who also knows the randomly chosen permutation, replaces each ciphertext letter on the bottom row with the corresponding plaintext letter in the top row. The ciphertext 'MJBXZ' is decrypted to 'point'.

Security Value

Simple Substitution Cipher is a considerable improvement over the Caesar Cipher. The possible number of keys is large (26!) and even the modern computing systemsare not yet powerful enough to comfortably launch a brute force attack to break the system. However, the Simple Substitution Cipher has a simple design and it is prone to design flaws, say choosing obvious permutation, this cryptosystem can be easily broken.

Monoalphabetic and Polyalphabetic Cipher

Monoalphabetic cipher is a substitution cipher in which for a given key, the cipher alphabet for each plain alphabet is fixed throughout the encryption process. For example, if 'A' is encrypted as 'D', for any number of occurrence in that plaintext, 'A'will always get encrypted to 'D'.

All of the substitution ciphers we have discussed earlier in this chapter are monoalphabetic; these ciphers are highly susceptible to cryptanalysis.

Polyalphabetic Cipher is a substitution cipher in which the cipher alphabet for the plain alphabet may be different at different places during the encryption process.The next two examples, **playfair and Vigenere Cipher are polyalphabetic ciphers**.

Playfair Cipher

In this scheme, pairs of letters are encrypted, instead of single letters as in the case of simple substitution cipher.

In playfair cipher, initially a key table is created. The key table is a 5×5 grid of alphabets that acts as the key for encrypting the plaintext. Each of the 25 alphabets must be unique and one letter of the alphabet (usually J) is omitted from the table

as we need only 25 alphabets instead of 26. If the plaintext contains J, then it is replaced by I.

The sender and the receiver deicide on a particular key, say ‘tutorials’. In a key table, the first characters (going left to right) in the table is the phrase, excluding the duplicate letters. The rest of the table will be filled with the remaining letters of the alphabet, in natural order. The key table works out to be −

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

Process of Playfair Cipher

- First, a plaintext message is split into pairs of two letters (digraphs). If there isan odd number of letters, a Z is added to the last letter. Let us say we want toencrypt the message “hide money”. It will be written as −

  HI DE MO NE YZ

- The rules of encryption are −
  - If both the letters are in the same column, take the letter below eachone (going back to the top if at the bottom)

T	U	O	R	I	‘H’ and ‘I’ are in same column, hence take letter below them to replace. HI → QC
A	L	S	B	C	

D	E	F	G	H
K	M	N	P	Q

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

'D' and 'E' are in same row, hence take letter to the right of them to replace. DE → EF

V	W	X	Y	Z

- If both letters are in the same row, take the letter to the right of each one(going back to the left if at the farthest right)
- If neither of the preceding two rules are true, form a rectangle with the twoletters and take the letters on the horizontal opposite corner of the rectangle.

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

'M' and 'O' nor on same column or same row, hence form rectangle as shown, and replace letter by picking up opposite corner letter on same row
MO -> NU

Using these rules, the result of the encryption of 'hide money' with the key of 'tutorials' would be −

QC EF NU MF ZV

Decrypting the Playfair cipher is as simple as doing the same process in reverse. Receiver has the same key and can create the same key table, and then decryptany messages made using that key.

Security Value

It is also a substitution cipher and is difficult to break compared to the simple substitution cipher. As in case of substitution cipher, cryptanalysis is possible on thePlayfair cipher as well, however it would be against 625 possible pairs of letters (25x25 alphabets) instead of 26 different possible alphabets.

The Playfair cipher was used mainly to protect important, yet non-critical secrets, asit is quick to use and requires no special equipment.

Vigenere Cipher

This scheme of cipher uses a text string (say, a word) as a key, which is then usedfor doing a number of shifts on the plaintext.

For example, let's assume the key is 'point'. Each alphabet of the key is convertedto its respective numeric value: In this case,

p → 16, o → 15, i → 9, n → 14, and t → 20.

Thus, the key is: 16 15 9 14 20.

Process of Vigenere Cipher

- The sender and the receiver decide on a key. Say 'point' is the key. Numeric representation of this key is '16 15 9 14 20'.
- The sender wants to encrypt the message, say 'attack from south east'. Hewill arrange plaintext and numeric key as follows −

a	t	t	a	c	k	f	r	o	m	s	o	u	t	h	e	a	s	t
16	15	9	14	20	16	15	9	14	20	16	15	9	14	20	16	15	9	14

- He now shifts each plaintext alphabet by the number written below it to create ciphertext as shown below −

a	t	t	a	c	k	f	r	o	m	s	o	u	t	h	e	a	s	t
16	15	9	14	20	16	15	9	14	20	16	15	9	14	20	16	15	9	14
Q	I	C	O	W	A	U	A	C	G	I	D	D	H	B	U	P	B	H

- Here, each plaintext character has been shifted by a different amount – and that amount is determined by the key. The key must be less than or equal to the size of the message.
- For decryption, the receiver uses the same key and shifts received ciphertext in reverse order to obtain the plaintext.

Q	I	C	O	W	A	U	A	C	G	I	D	D	H	B	U	P	B	H
16	15	9	14	20	16	15	9	14	20	16	15	9	14	20	16	15	9	14
a	t	t	a	c	k	f	r	o	m	s	o	u	t	h	e	a	s	t

Security Value

Vigenere Cipher was designed by tweaking the standard Caesar cipher to reduce the effectiveness of cryptanalysis on the ciphertext and make a cryptosystem more robust. It is significantly **more secure than a regular Caesar Cipher**.

In the history, it was regularly used for protecting sensitive political and military information. It was referred to as the **unbreakable cipher** due to the difficulty it posed to the cryptanalysis.

Variants of Vigenere Cipher

There are two special cases of Vigenere cipher −

- The keyword length is same as plaintect message. This case is called **Vernam Cipher**. It is more secure than typical Vigenere cipher.
- Vigenere cipher becomes a cryptosystem with perfect secrecy, which iscalled **One-time pad**.

One-Time Pad

The circumstances are −

- The length of the keyword is same as the length of the plaintext.
- The keyword is a randomly generated string of alphabets.
- The keyword is used only once.

Security Value

Let us compare Shift cipher with one-time pad.

Shift Cipher − Easy to Break

In case of Shift cipher, the entire message could have had a shift between 1 and 25.This is a very small size, and very easy to brute force. However, with each character now having its own individual shift between 1 and 26, the possible keys grow exponentially for the message.

One-time Pad − Impossible to Break

Let us say, we encrypt the name "point" with a one-time pad. It is a 5 letter text. To break the ciphertext by brute force, you need to try all possibilities of keys and conduct computation for (26 x 26 x 26 x 26 x 26) = 26^5= 11881376 times. That's for a message with 5 alphabets. Thus, for a longer message, the computation grows exponentially with every additional alphabet. This makes it computationally impossible to break the ciphertext by brute force.

Transposition Cipher

It is another type of cipher where the order of the alphabets in the plaintext is rearranged to create the ciphertext. The actual plaintext alphabets are not replaced.

An example is a 'simple columnar transposition' cipher where the plaintext is written horizontally with a certain alphabet width. Then the ciphertext is read vertically as shown.

For example, the plaintext is “golden statue is in eleventh cave” and the secret random key chosen is “five”. We arrange this text horizontally in table with numberof column equal to key value. The resulting text is shown below.

g	o	l	d	e
n	s	t	a	t
u	e	i	s	i
n	e	l	e	v
e	n	t	h	c
a	v	e		

The ciphertext is obtained by reading column vertically downward from first to last column. The ciphertext is ‘gnuneaoseenvltiltedasehetivc’.

To decrypt, the receiver prepares similar table. The number of columns is equal to key number. The number of rows is obtained by dividing number of total ciphertext alphabets by key value and rounding of the quotient to next integer value.

The receiver then writes the received ciphertext vertically down and from left to rightcolumn. To obtain the text, he reads horizontally left to right and from top to bottom row.

Modern Symmetric Key Encryption

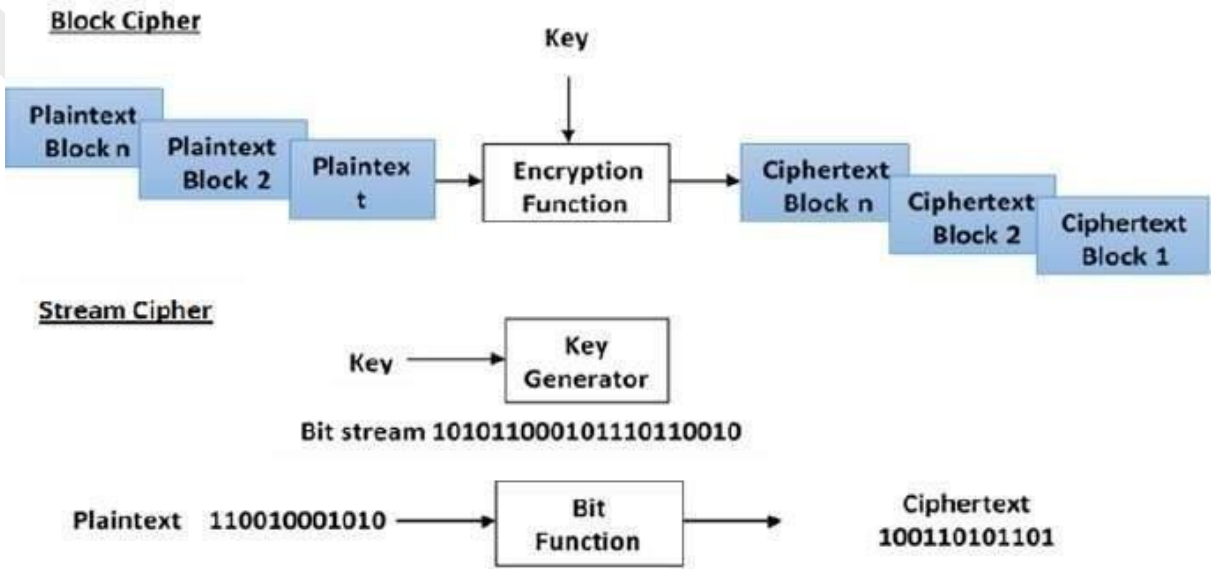
Digital data is represented in strings of binary digits (bits) unlike alphabets. Modern cryptosystems need to process this binary strings to convert in to another binary string. Based on how these binary strings are processed, a symmetric encryption schemes can be classified in to −

Block Ciphers

In this scheme, the plain binary text is processed in blocks (groups) of bits at a time;
i.e. a block of plaintext bits is selected, a series of operations is performed on this block to generate a block of ciphertext bits. The number of bits in a block is fixed.For example, the schemes DES and AES have block sizes of 64 and 128, respectively.

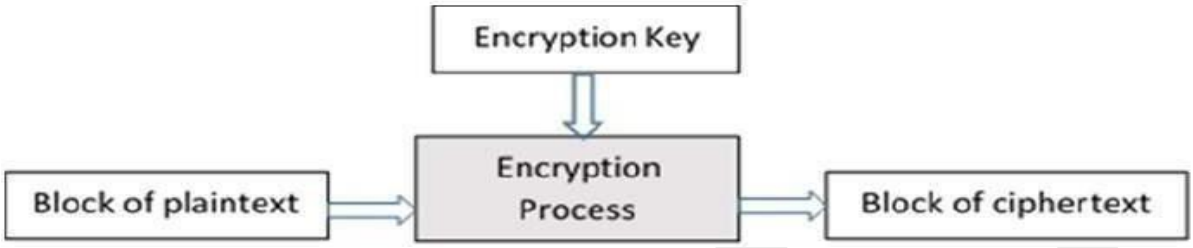
Stream Ciphers

In this scheme, the plaintext is processed one bit at a time i.e. one bit of plaintext is taken, and a series of operations is performed on it to generate one bit of ciphertext.Technically, stream ciphers are block ciphers with a block size of one bit.



Block Cipher

The basic scheme of a block cipher is depicted as follows −



A block cipher takes a block of plaintext bits and generates a block of ciphertext bits, generally of same size. The size of block is fixed in the given scheme. The choice of block size does not directly affect to the strength of encryption scheme. The strength of cipher depends up on the key length.

Block Size

Though any size of block is acceptable, following aspects are borne in mind while selecting a size of a block.

- **Avoid very small block size** − Say a block size is m bits. Then the possibleplaintext bits combinations are then 2^m. If the attacker discovers the plain text blocks corresponding to some previously sent ciphertext blocks, then the attacker can launch a type of 'dictionary attack' by building up a dictionary of plaintext/ciphertext pairs sent using that encryption key. A larger block size makes attack harder as the dictionary needs to be larger.
- **Do not have very large block size** − With very large block size, the cipher becomes inefficient to operate. Such plaintexts will need to be padded beforebeing encrypted.
- **Multiples of 8 bit** − A preferred block size is a multiple of 8 as it is easy for implementation as most computer processor handle data in multiple of 8 bits.

Padding in Block Cipher

Block ciphers process blocks of fixed sizes (say 64 bits). The length of plaintexts is mostly not a multiple of the block size. For example, a 150-bit plaintext provides two

blocks of 64 bits each with third block of balance 22 bits. The last block of bitsneeds to be padded up with redundant information so that the length of the final block equal to block size of the scheme. In our example, the remaining 22 bits need to have additional 42 redundant bits added to provide a complete block. The process of adding bits to the last block is referred to as **padding**.

Too much padding makes the system inefficient. Also, padding may render the system insecure at times, if the padding is done with same bits always.

Block Cipher Schemes

There is a vast number of block ciphers schemes that are in use. Many of them are publically known. Most popular and prominent block ciphers are listed below.

- **Digital Encryption Standard (DES)** − The popular block cipher of the 1990s.It is now considered as a 'broken' block cipher, due primarily to its small key size.
- **Triple DES** − It is a variant scheme based on repeated DES applications. It isstill a respected block ciphers but inefficient compared to the new faster block ciphers available.
- **Advanced Encryption Standard (AES)** − It is a relatively new block cipher based on the encryption algorithm **Rijndael** that won the AES design competition.
- **IDEA** − It is a sufficiently strong block cipher with a block size of 64 and a keysize of 128 bits. A number of applications use IDEA encryption, including early versions of Pretty Good Privacy (PGP) protocol. The use of IDEAscheme has a restricted adoption due to patent issues.
- **Twofish** − This scheme of block cipher uses block size of 128 bits and a keyof variable length. It was one of the AES finalists. It is based on the earlier block cipher Blowfish with a block size of 64 bits.
- **Serpent** − A block cipher with a block size of 128 bits and key lengths of 128,192, or 256 bits, which was also an AES competition finalist. It is a slower buthas more secure design than other block cipher.

In the next sections, we will first discuss the model of block cipher followed by DES and AES, two of the most influential modern block ciphers.

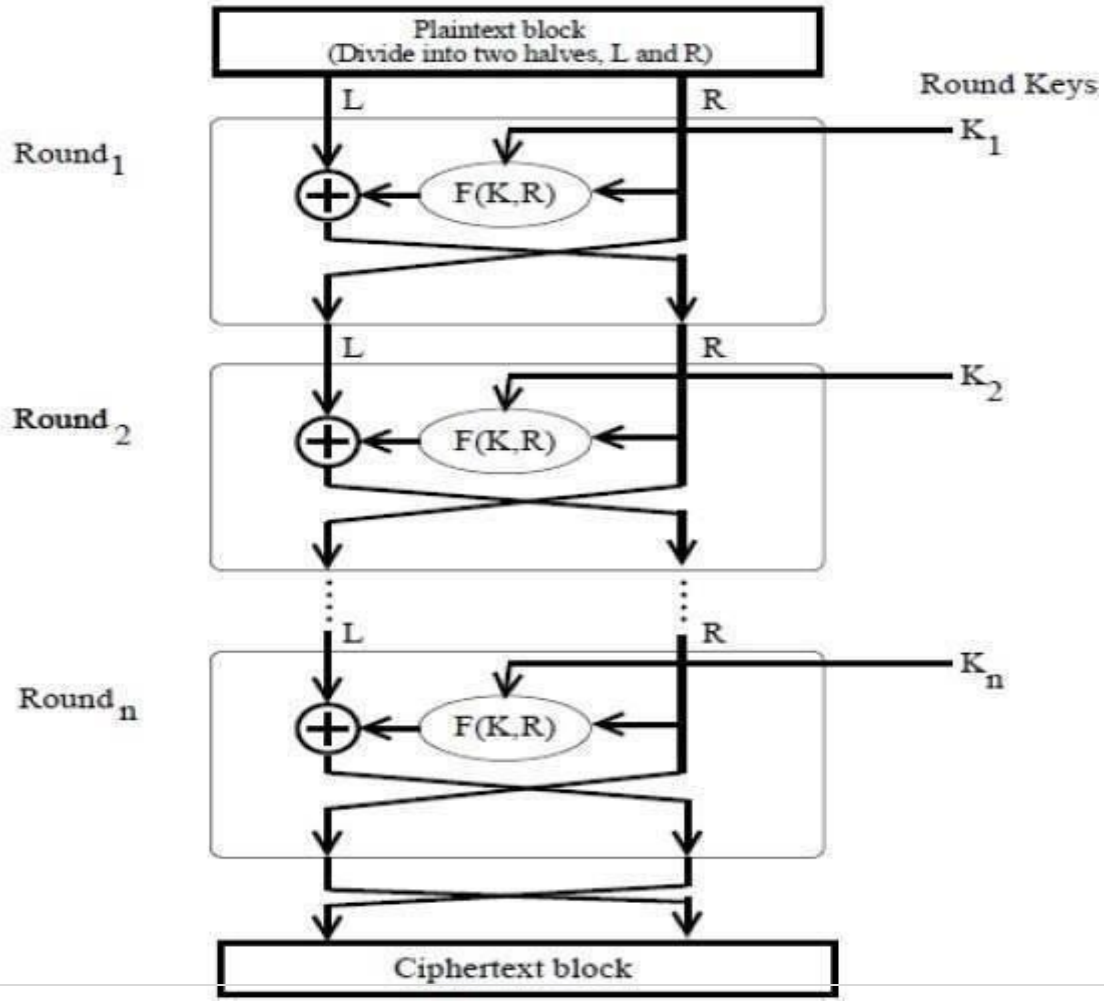
Feistel Block Cipher

Feistel Cipher is not a specific scheme of block cipher. It is a design model from which many different block ciphers are derived. DES is just one example of a Feistel Cipher. A cryptographic system based on Feistel cipher structure uses the same algorithm for both encryption and decryption.

Encryption Process

The encryption process uses the Feistel structure consisting multiple rounds of processing of the plaintext, each round consisting of a "substitution" step followedby a permutation step.

Feistel Structure is shown in the following illustration −



- The input block to each round is divided into two halves that can be denoted as L and R for the left half and the right half.
- In each round, the right half of the block, R, goes through unchanged. But the left half, L, goes through an operation that depends on R and the encryption key. First, we apply an encrypting function 'f' that takes two input − the key K and R. The function produces the output f(R,K). Then, we XOR the output of the mathematical function with L.
- In real implementation of the Feistel Cipher, such as DES, instead of using the whole encryption key during each round, a round-dependent key (asubkey) is derived from the encryption key. This means that each round usesa different key, although all these subkeys are related to the original key.
- The permutation step at the end of each round swaps the modified L and unmodified R. Therefore, the L for the next round would be R of the current round. And R for the next round be the output L of the current round.
- Above substitution and permutation steps form a 'round'. The number of rounds are specified by the algorithm design.
- Once the last round is completed then the two sub blocks, 'R' and 'L' are concatenated in this order to form the ciphertext block.

The difficult part of designing a Feistel Cipher is selection of round function 'f'. In order to be unbreakable scheme, this function needs to have several important properties that are beyond the scope of our discussion.

Decryption Process

The process of decryption in Feistel cipher is almost similar. Instead of starting with a block of plaintext, the ciphertext block is fed into the start of the Feistel structure and then the process thereafter is exactly the same as described in the given illustration.

The process is said to be almost similar and not exactly same. In the case of decryption, the only difference is that the subkeys used in encryption are used in thereverse order.

The final swapping of 'L' and 'R' in last step of the Feistel Cipher is essential. If these are not swapped then the resulting ciphertext could not be decrypted using the same algorithm.

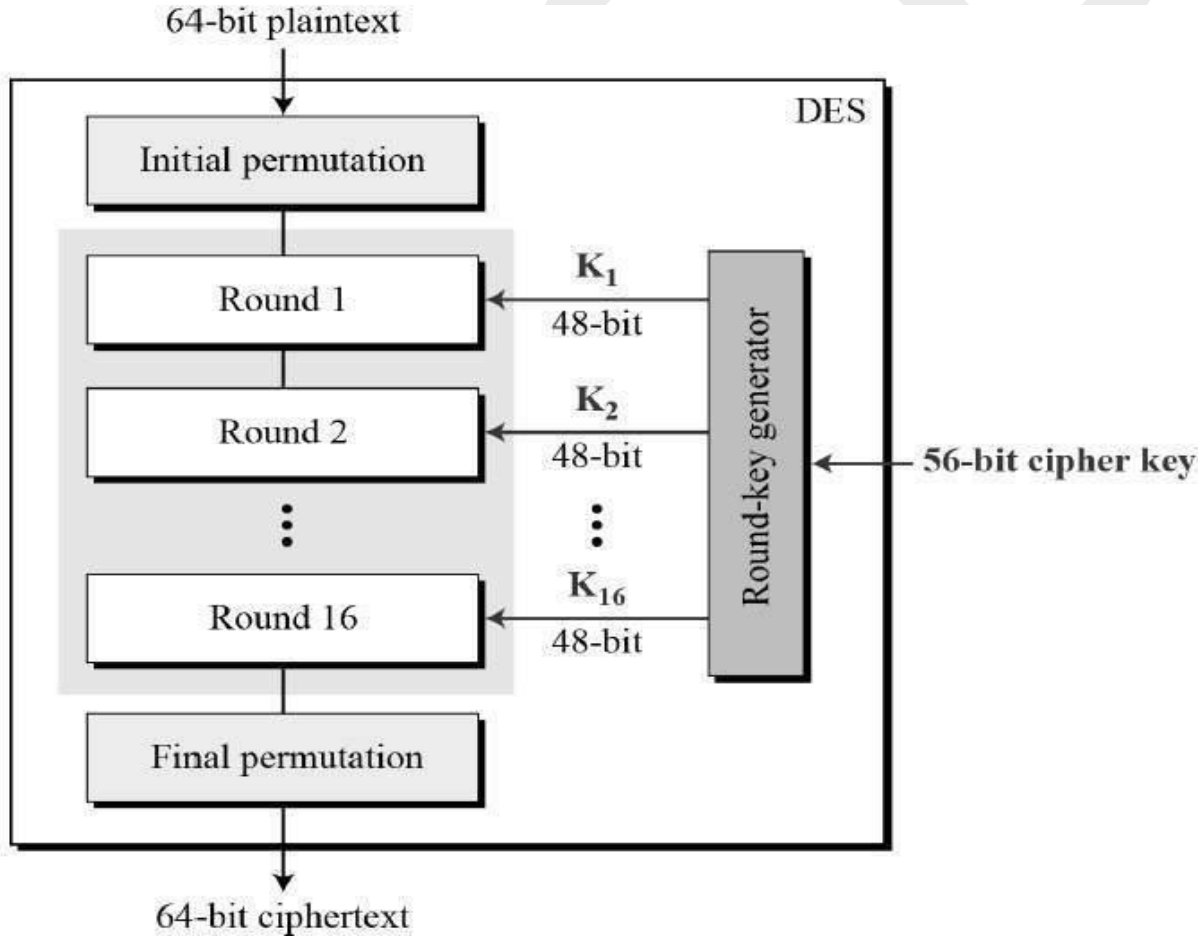
Number of Rounds

The number of rounds used in a Feistel Cipher depends on desired security from the system. More number of rounds provide more secure system. But at the same time, more rounds mean the inefficient slow encryption and decryption processes. Number of rounds in the systems thus depend upon efficiency–security tradeoff.

Data Encryption Standard

The Data Encryption Standard (DES) is a symmetric-key block cipher published bythe National Institute of Standards and Technology (NIST).

DES is an implementation of a Feistel Cipher. It uses 16 round Feistel structure. The block size is 64-bit. Though, key length is 64-bit, DES has an effective key length of 56 bits, since 8 of the 64 bits of the key are not used by the encryption algorithm (function as check bits only). General Structure of DES is depicted in the following illustration −

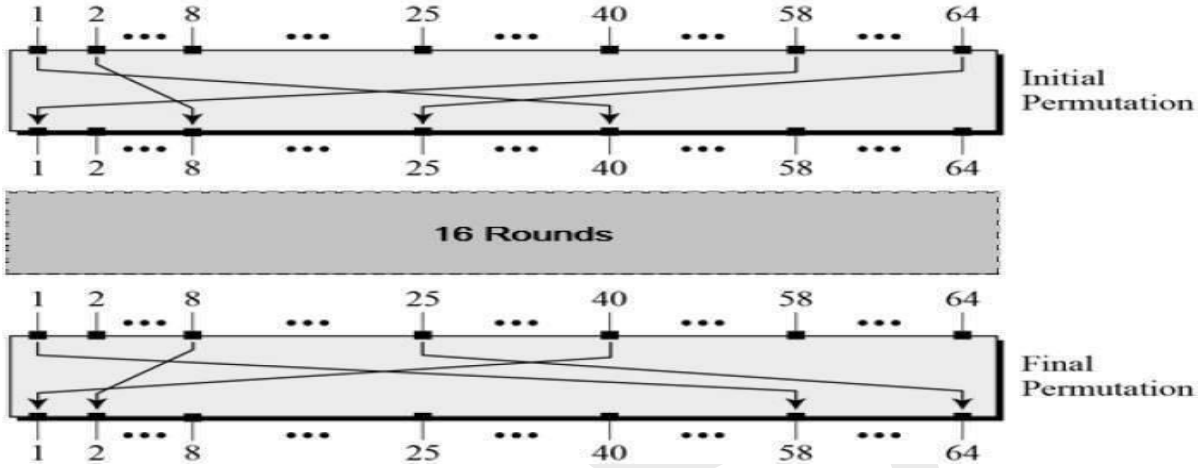


Since DES is based on the Feistel Cipher, all that is required to specify DES is −

- Round function
- Key schedule
- Any additional processing − Initial and final permutation

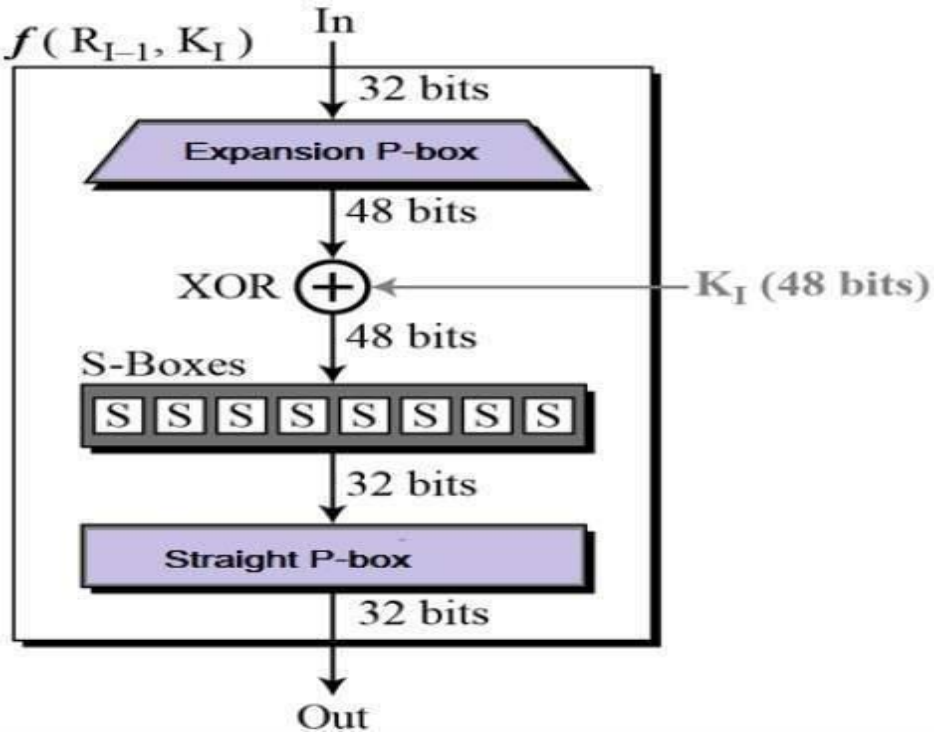
Initial and Final Permutation

The initial and final permutations are straight Permutation boxes (P-boxes) that are inverses of each other. They have no cryptography significance in DES. The initial and final permutations are shown as follows −

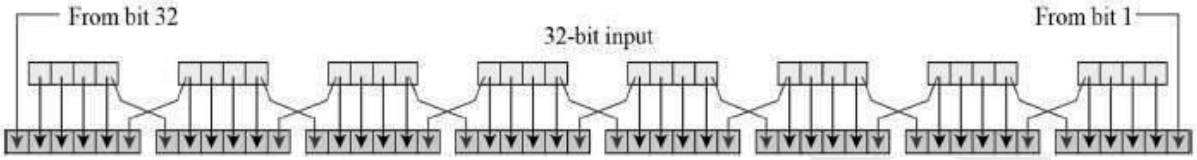


Round Function

The heart of this cipher is the DES function, f. The DES function applies a 48-bit keyto the rightmost 32 bits to produce a 32-bit output.



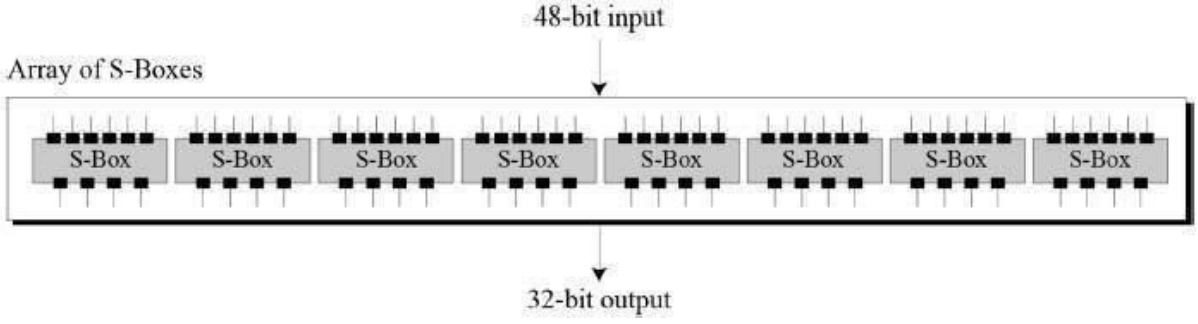
- **Expansion Permutation Box** − Since right input is 32-bit and round key is a48-bit, we first need to expand right input to 48 bits. Permutation logic is graphically depicted in the following illustration −



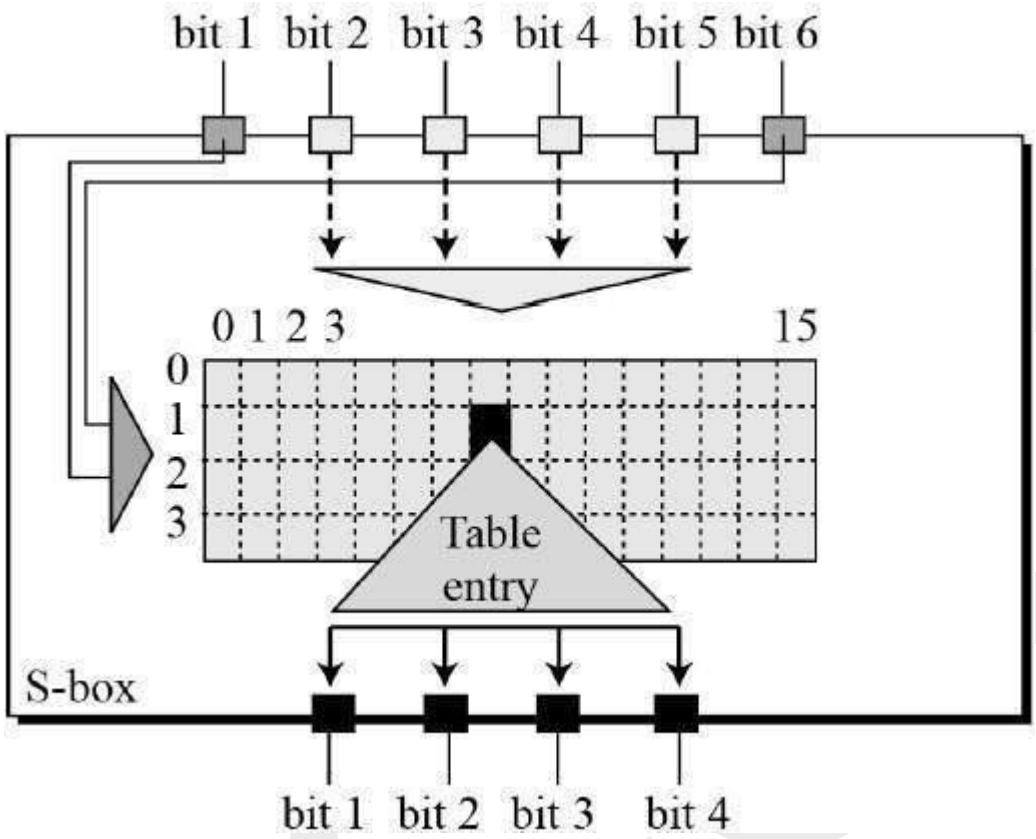
- The graphically depicted permutation logic is generally described as table in DES specification illustrated as shown −

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 −

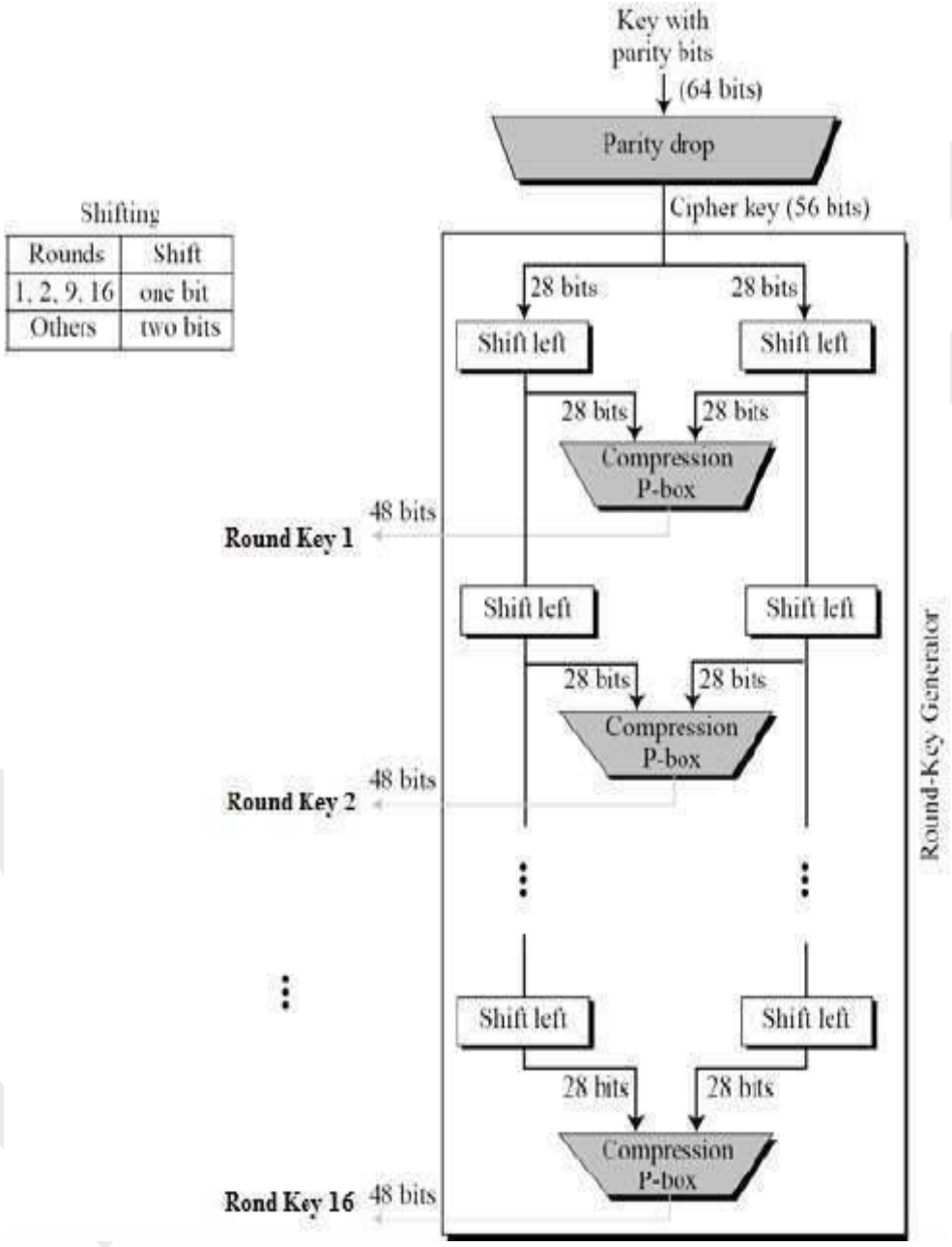


- There are a total of eight S-box tables. The output of all eight s-boxes is then combined in to 32 bit section.
- **Straight Permutation** − The 32 bit output of S-boxes is then subjected to the straight permutation with rule shown in the following illustration:

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. Theprocess of key generation is depicted in the following illustration −



Shifting

Rounds	Shift
1, 2, 9, 16	one bit
Others	two bits

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

DES Analysis

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

- **Avalanche effect** − A small change in plaintext results in the very great 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 whenkey 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.

Triple DES

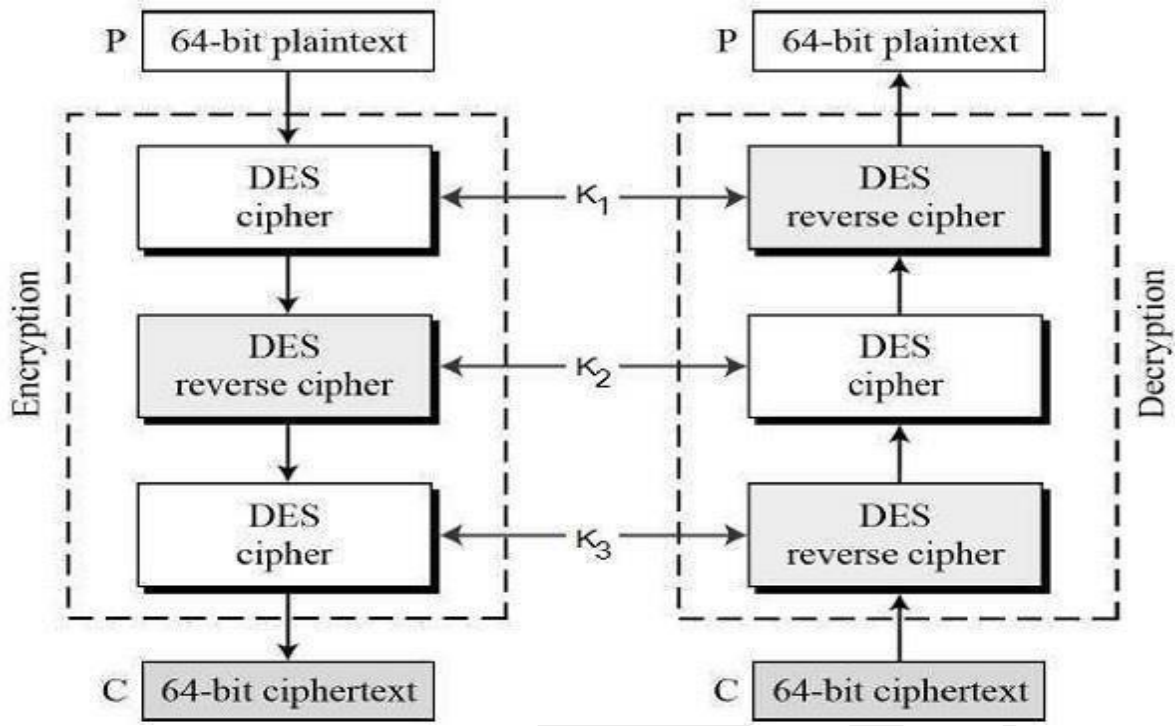
The speed of exhaustive key searches against DES after 1990 began to cause discomfort amongst users of DES. However, users did not want to replace DES as ittakes an enormous amount of time and money to change encryption algorithms thatare widely adopted and embedded in large security architectures.

The pragmatic approach was not to abandon the DES completely, but to change themanner 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 $3\times56 = 168$ 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_3isreplaced 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.

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 time 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

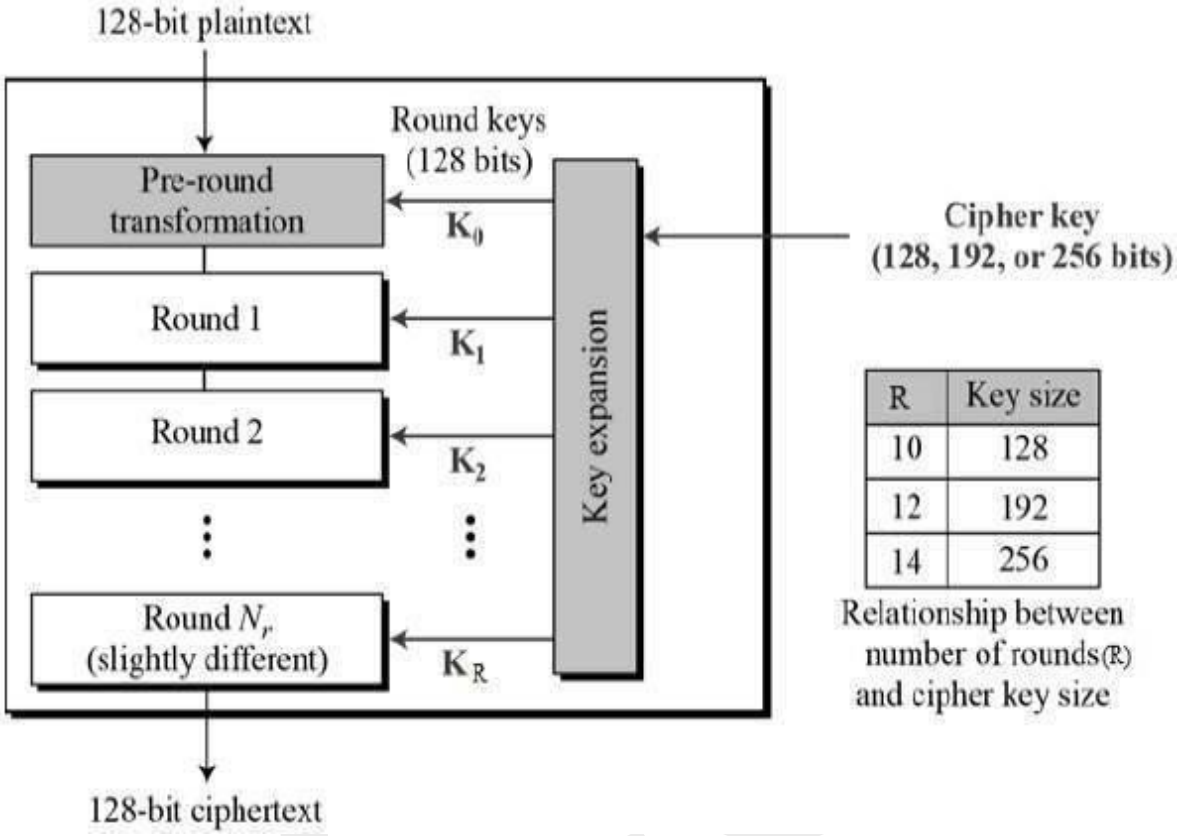
Operation of AES

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 −

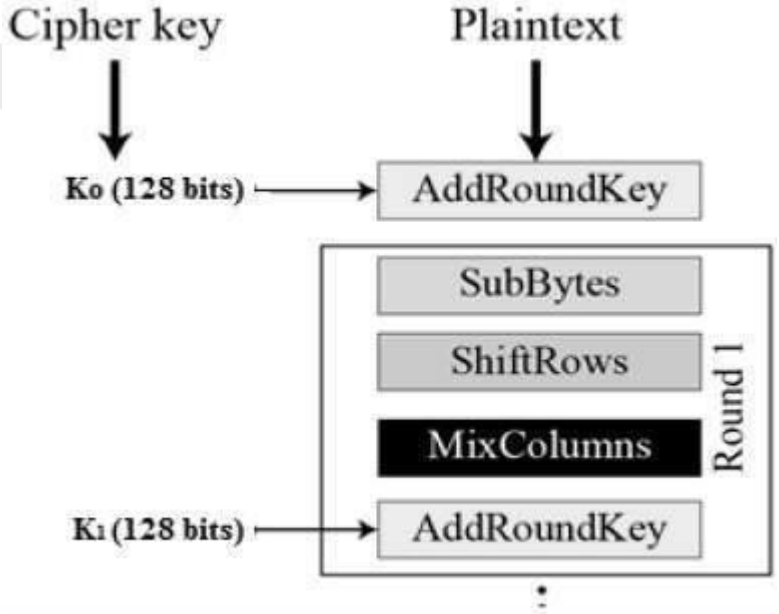
Unlike DES, the number of rounds in AES is variable and depends on the length of the key. AES uses 10 rounds for 128-bit keys, 12 rounds for 192-bit keys and 14 rounds for 256-bit keys. Each of these rounds uses a different 128-bit round key, which is calculated from the original AES key.

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 withrespect 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 needs 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 exhaustivekey searches.

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

Block Cipher 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 oneat a time.

Electronic Code Book (ECB) Mode

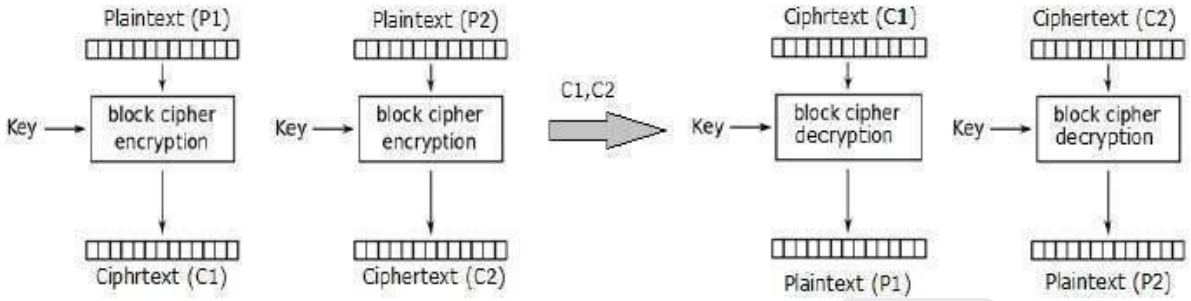
This mode is a most straightforward way of processing a series of sequentially listedmessage blocks.

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 P1, P2,…, Pm 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

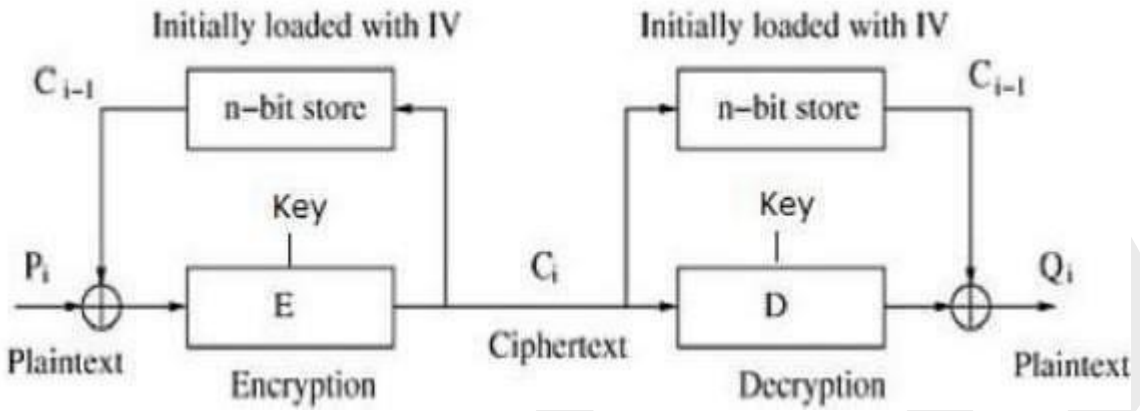
CBC mode of operation provides message dependence for generating ciphertext and makes the system non-deterministic.

Operation

The operation of CBC mode is depicted in the following illustration. The steps are asfollows –

- 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 reverseprocess, 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 propagatedto 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

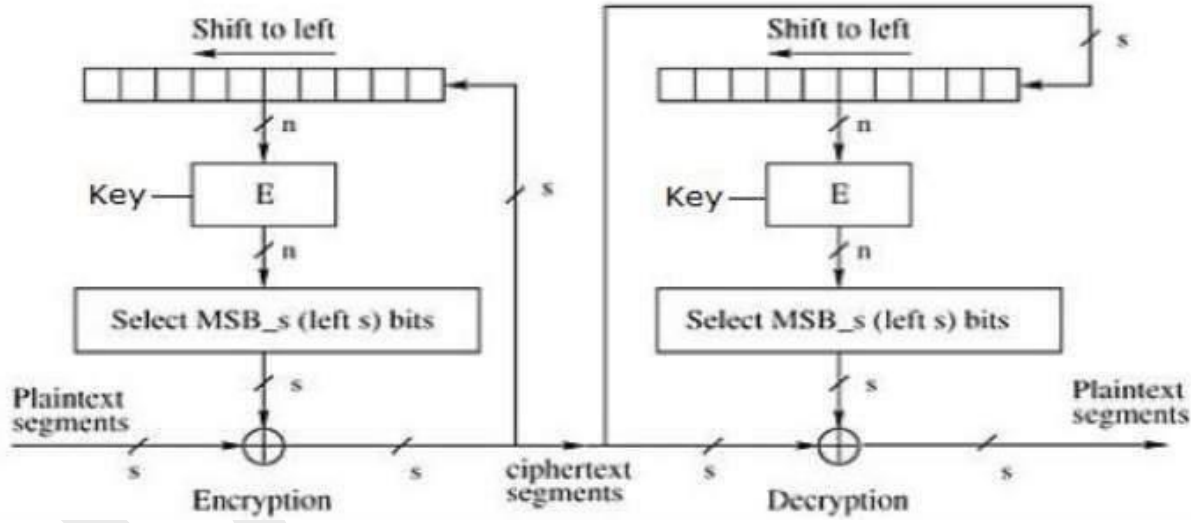
In this mode, each ciphertext block gets 'fed back' into the encryption process in order to encrypt the next plaintext block.

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. TheIV 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 alsoon the previous ciphertext block. In other words, the ciphertext block is dependentof 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 asin 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.

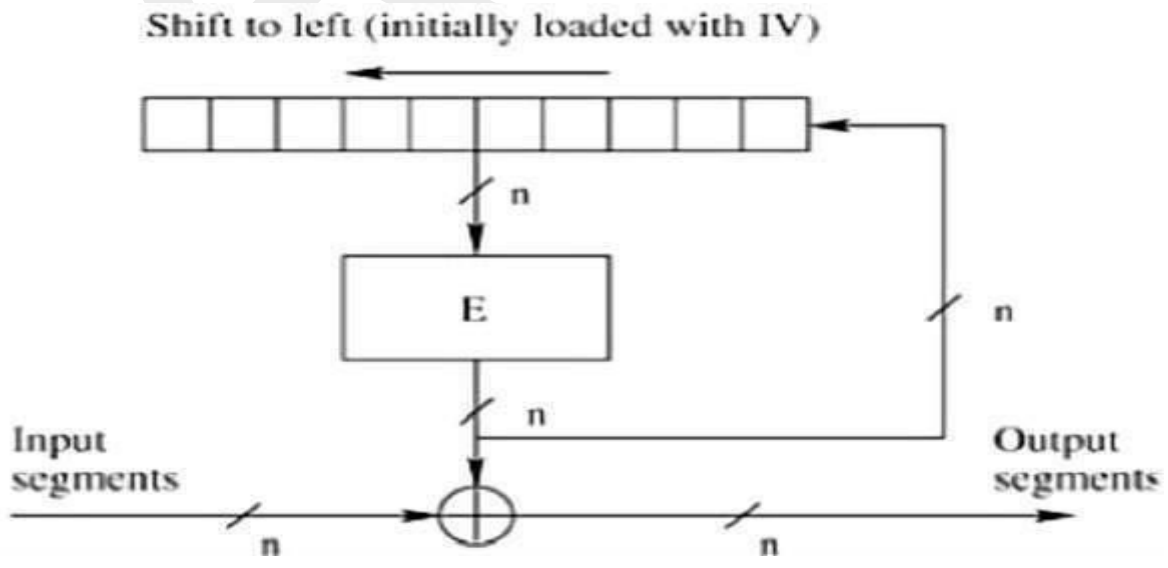
On the flip side, the error of transmission gets propagated due to changing of blocks.

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 −



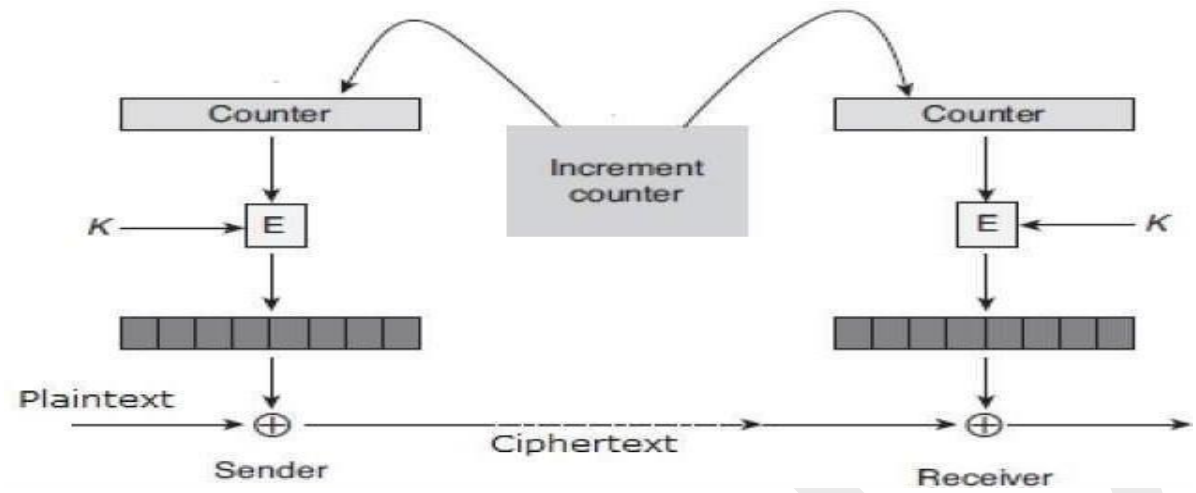
Counter (CTR) Mode

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 cipher text 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 P1 and XOR this to the contents of the bottom register. The result of this is C1. Send C1 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.

Public Key Encryption

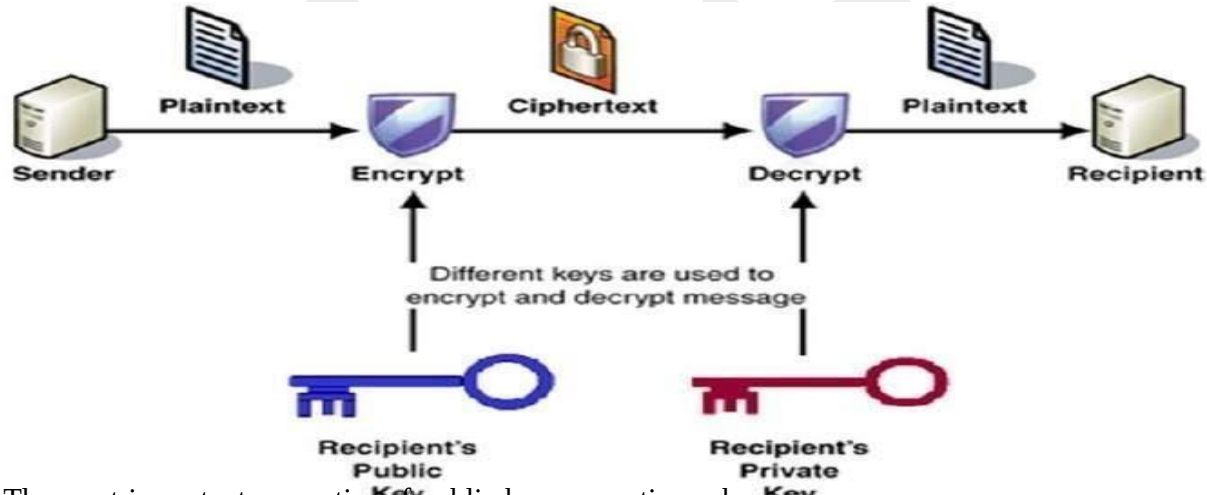
Public Key Cryptography

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 propertywhich set this scheme different than symmetric encryption scheme.
- Each receiver possesses a unique decryption key, generally referred to as hisprivate 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 schemeto 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. Theprocess followed in the generation of keys is described below −

- **Generate the RSA modulus (n)**
  - Select two large primes, p and q.
  - 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**
  - The pair of numbers (n, e) form the RSA public key and is made public.
  - Interestingly, though n is part of the public key, difficulty in factorizing alarge 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**
  - Private Key d is calculated from p, q, and e. For given n and e, there isunique number d.
  - Number d is the inverse of e modulo (p - 1)(q – 1). This means that d isthe number less than (p - 1)(q - 1) such that when multiplied by e, it is equal to 1 modulo (p - 1)(q - 1).
  - This relationship is written mathematically as follows −

```
ed = 1 mod (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 x 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 × 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 etimes 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 thereceiver 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.

$$Plaintext = C^d \bmod n$$

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

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

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 onthe 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 unlesshe 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.**
    - This number must be between 1 and p − 1, but cannot be any number.
    - 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 $g^k = a \bmod n$.

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

N	3^n	$3^n \bmod 5$
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 −

$$y = g^x \bmod p$$

- **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 6is a generator of group Z_{17}). The private key x can be any number bigger than1 and smaller than 71, so we choose x = 5. The value y is then computed as follows −

$$y = 6^5 \bmod 17 = 7$$

- Thus the private key is 62 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 −
  - Randomly generate a number k;
  - Compute two values C1 and C2, where −

$$C1 = g^k \bmod p$$

$$C2 = (P * y^k) \bmod p$$

- Send the ciphertext C, consisting of the two separate values (C1, C2), sent together.
- Referring to our ElGamal key generation example given above, the plaintextP = 13 is encrypted as follows −
    - Randomly generate a number, say k = 10
    - Compute the two values C1 and C2, where −

$C1 = 6^{10} \bmod 17$

$C2 = (13 * 7^{10}) \bmod 17 = 9$

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

ElGamal Decryption

- To decrypt the ciphertext (C1, C2) using private key x, the following two stepsare taken −
    - Compute the modular inverse of $(C1)^x$ modulo p, which is $(C1)^{-x}$, generally referred to as decryption factor.
    - Obtain the plaintext by using the following formula −

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

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

$15^{-5} \bmod 17 = 9$

- 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 mod p**. Thestrength 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 areused. 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 thediscrete 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.

RSA	ElGamal
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 keysare required.
It is widely accepted and used.	It is new and not very popular in market.

Data Integrity in Cryptography

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 tothe data.

Active Threats: In this type of threats, an attacker can manipulate the data withmalicious intent.

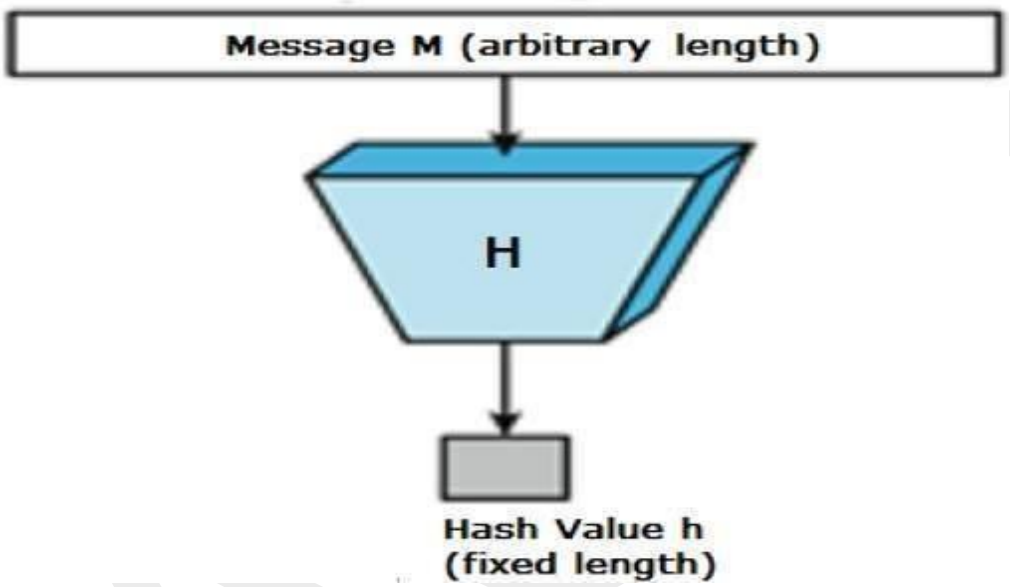
- At simplest level, if data is without digest, it can be modified without detection.The system can use techniques of appending CRC to data for detecting any active modification.
- At higher level of threat, attacker may modify data and try to derive new digest for modified data from exiting digest. 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.

Cryptography 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 intoanother 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)**
    - Hash function coverts data of arbitrary length to a fixed length. Thisprocess is often referred to as **hashing the data**.
    - In general, the hash is much smaller than the input data, hence hash functions are sometimes called **compression functions**.
    - Since a hash is a smaller representation of a larger data, it is alsoreferred to as a **digest**.

- 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**
  - Generally for any hash function h with input x, computation of h(x) is a fast operation.
  - Computationally hash functions are much faster than a symmetricencryption.

Properties of Hash Functions

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

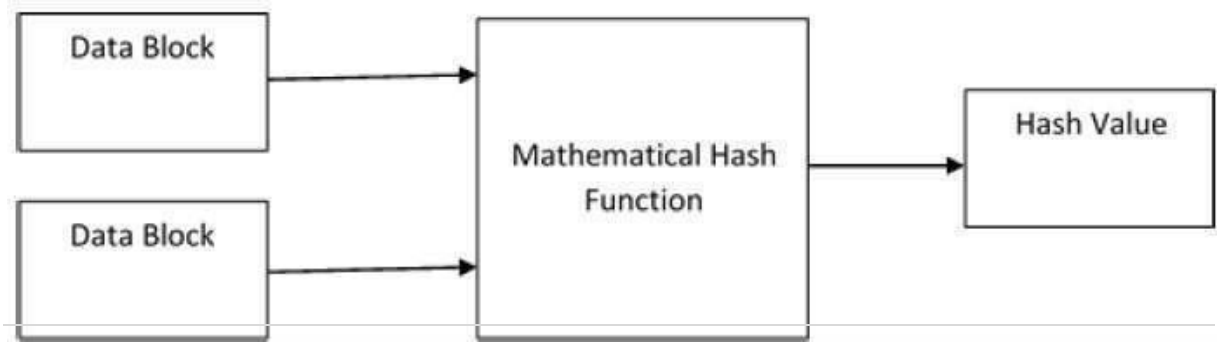
- **Pre-Image Resistance**
  - This property means that it should be computationally hard to reversea hash function.
  - In other words, if a hash function h produced a hash value z, then itshould be a difficult process to find any input value x that hashes to z.
  - This property protects against an attacker who only has a hash valueand is trying to find the input.
- **Second Pre-Image Resistance**
  - This property means given an input and its hash, it should be hard to find a different input with the same hash.
  - 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).
  - 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**
    - 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.
    - 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).
    - Since, hash function is compressing function with fixed hash length, itis impossible for a hash function not to have collisions. This propertyof collision free only confirms that these collisions should be hard to find.
    - This property makes it very difficult for an attacker to find two input values with the same hash.
    - Also, if a hash function is collision-resistant **then it is second pre- image resistant.**

Design of Hashing Algorithms

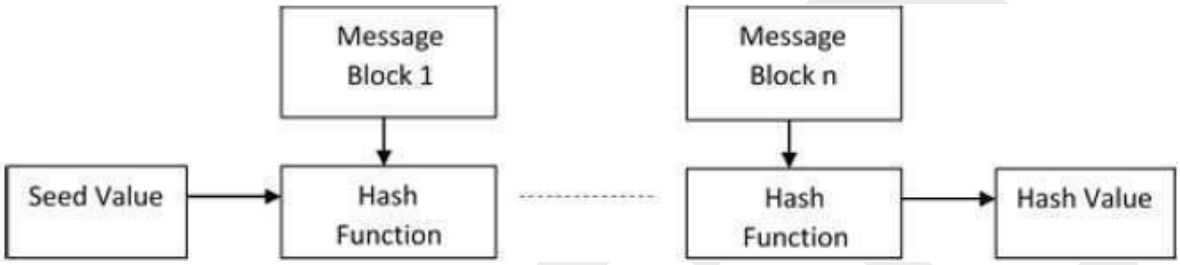
At the heart of a hashing is a mathematical function that operates on two fixed-size blocks of data to create a hash code. This hash function forms the part of the hashing algorithm.

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 thatdiffer 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

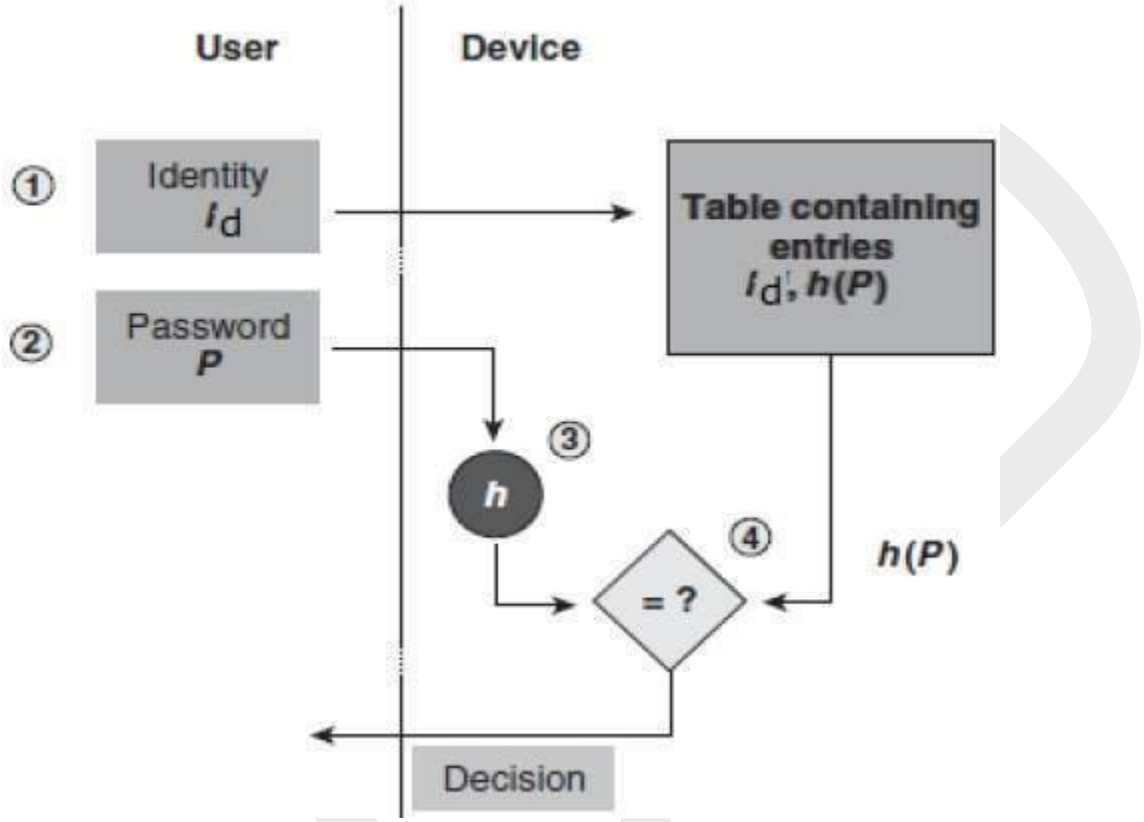
There are two direct applications of hash function based on its cryptographic properties.

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 −

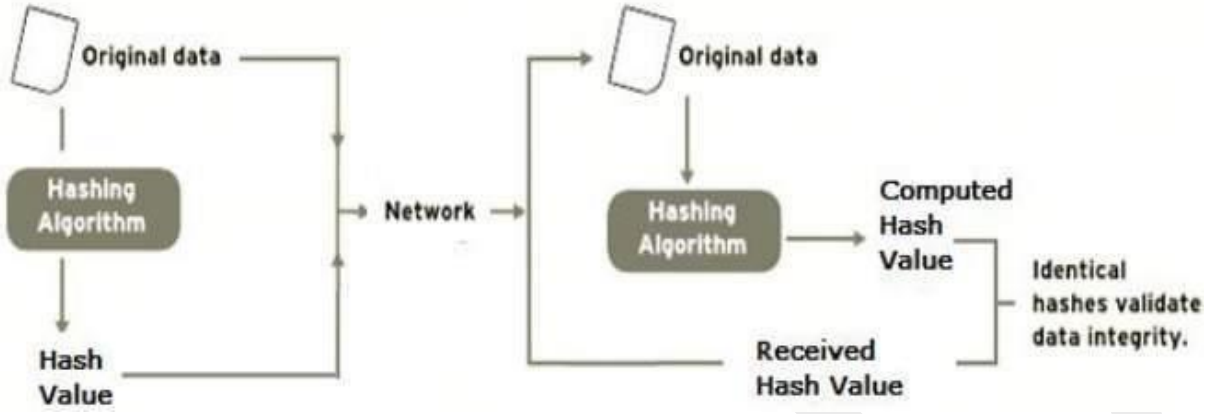


- An intruder can only see the hashes of passwords, even if he accessed the password. He can neither logon using hash nor can he derive the password from hash value since hash function possesses the property of pre-image resistance.

Data Integrity Check

Data integrity check is a most common application of the hash functions. It is usedto 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.

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.

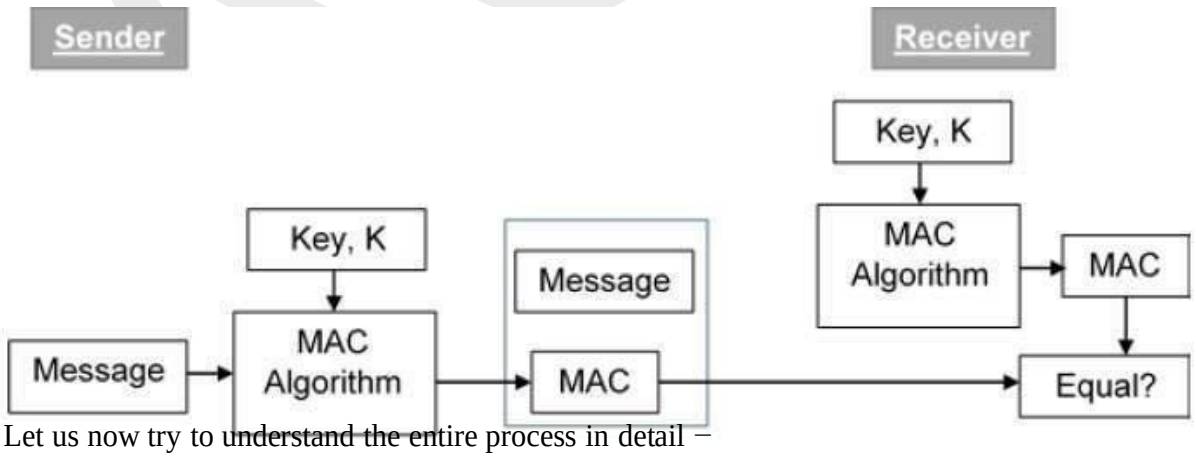
Message Authentication Code (MAC)

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 messageand 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 orit 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 ofoperation −

- **Establishment of Shared Secret.**
    - It can provide message authentication among pre-decided legitimateusers who have shared key.
    - This requires establishment of shared secret prior to use of MAC.
- **Inability to Provide Non-Repudiation**
    - Non-repudiation is the assurance that a message originator cannotdeny any previously sent messages and commitments or actions.

- MAC technique does not provide a non-repudiation service. If thesender and receiver get involved in a dispute over message origination, MACs cannot provide a proof that a message was indeed sent by the sender.
- 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 digitalsignatures discussed in following section.

Cryptography Digital signatures

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.

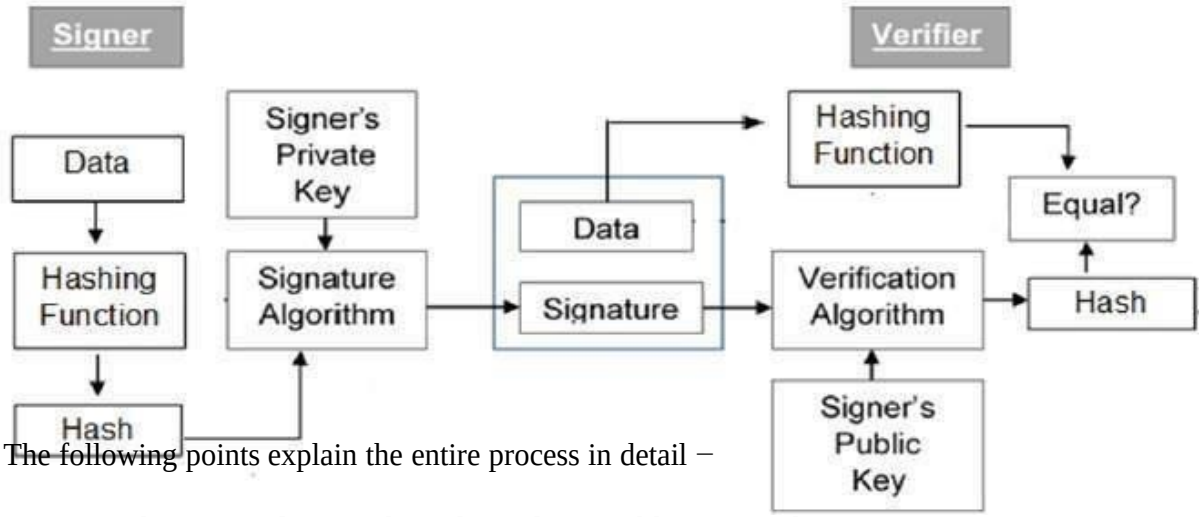
Similarly, a digital signature is a technique that binds a person/entity to the digital data. This binding can be independently verified by receiver as well as any third party.

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, usuallya 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 noone 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.

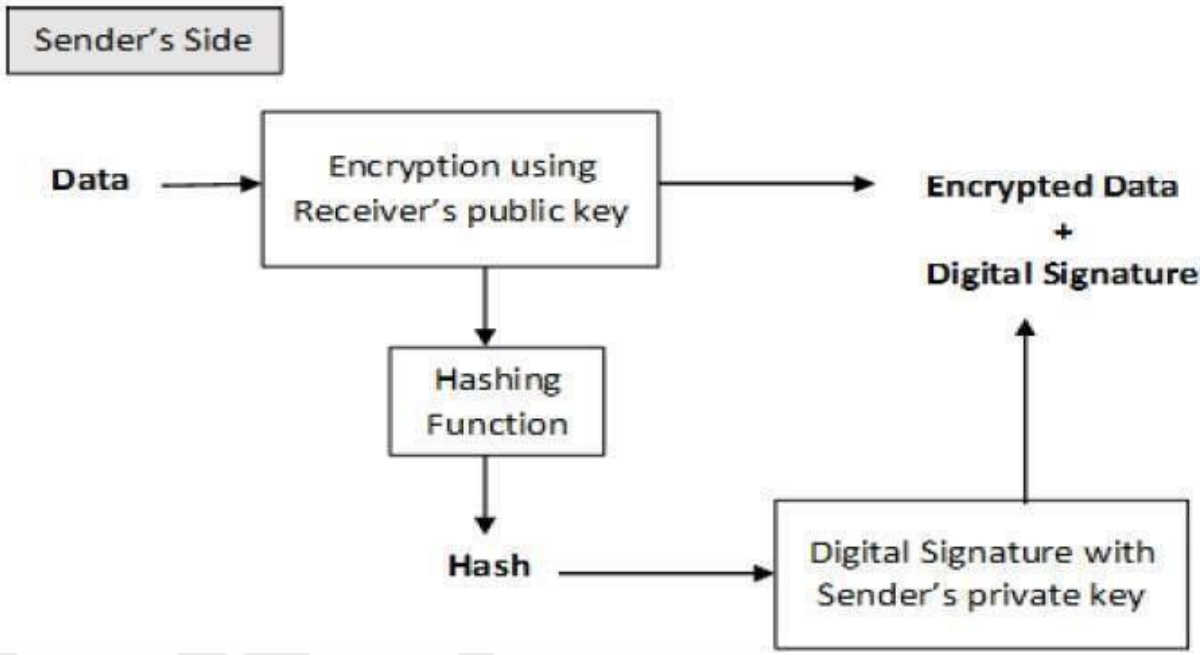
Encryption with Digital Signature

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.

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.

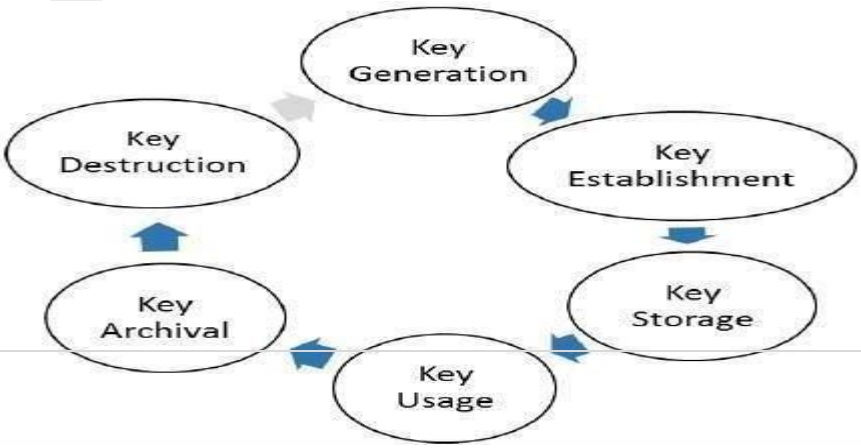
Key Management

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 managementrefers 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.
    - **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.
    - **Assurance of public keys.** In public key cryptography, the public keysare 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 theelectronic world.

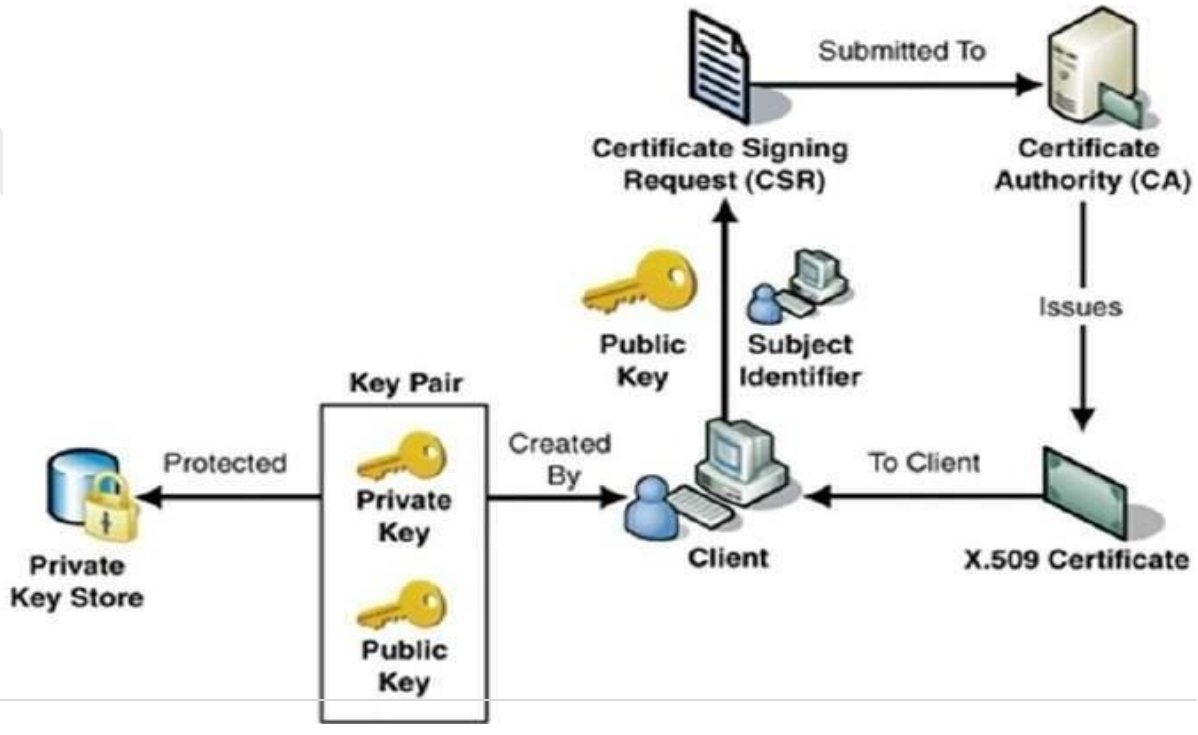
- Digital certificates are based on the ITU standard X.509 which defines a standard certificate format for public key certificates and certification validation. Hence digital certificates are sometimes also referred to as X.509 certificates.

  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 PKIequivalent 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 isto 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 dueto 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 emailaddress.
- **Class 2** − These certificates require additional personal information to be supplied.
- **Class 3** − These certificates can only be purchased after checks have beenmade about the requestor's identity.
- **Class 4** − They may be used by governments and financial organizationsneeding 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 statusat 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

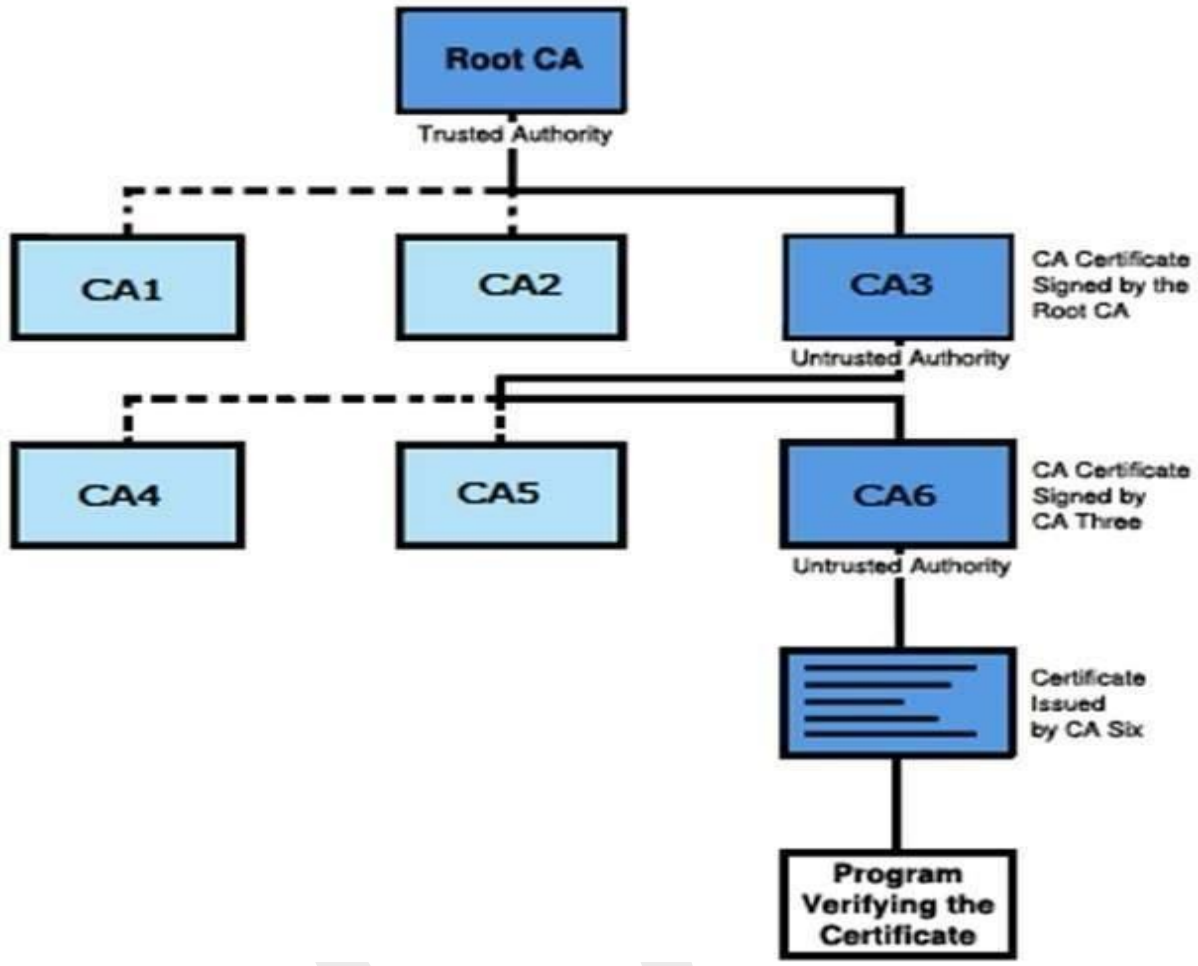
With vast networks and requirements of global communications, it is practically not feasible to have only one trusted CA from whom all users obtain their certificates. Secondly, availability of only one CA may lead to difficulties if CA is compromised.

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 nothave trust relationships with the same CA.

- The root CA is at the top of the CA hierarchy and the root CA's certificate is aself-signed certificate.
- The CAs, which are directly subordinate to the root CA (For example, CA1and CA2) have CA certificates that are signed by the root CA.
- The CAs under the subordinate CAs in the hierarchy (For example, CA5 andCA6) 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 inabove steps. This process continues till either trusted CA is found in betweenor else it continues till Root CA.

Cryptography - Benefits & 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 serviceto guard against the dispute that may arise due to denial of passing messageby 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 −
    - Addition of cryptographic techniques in the information processing leads to delay.
    - 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.

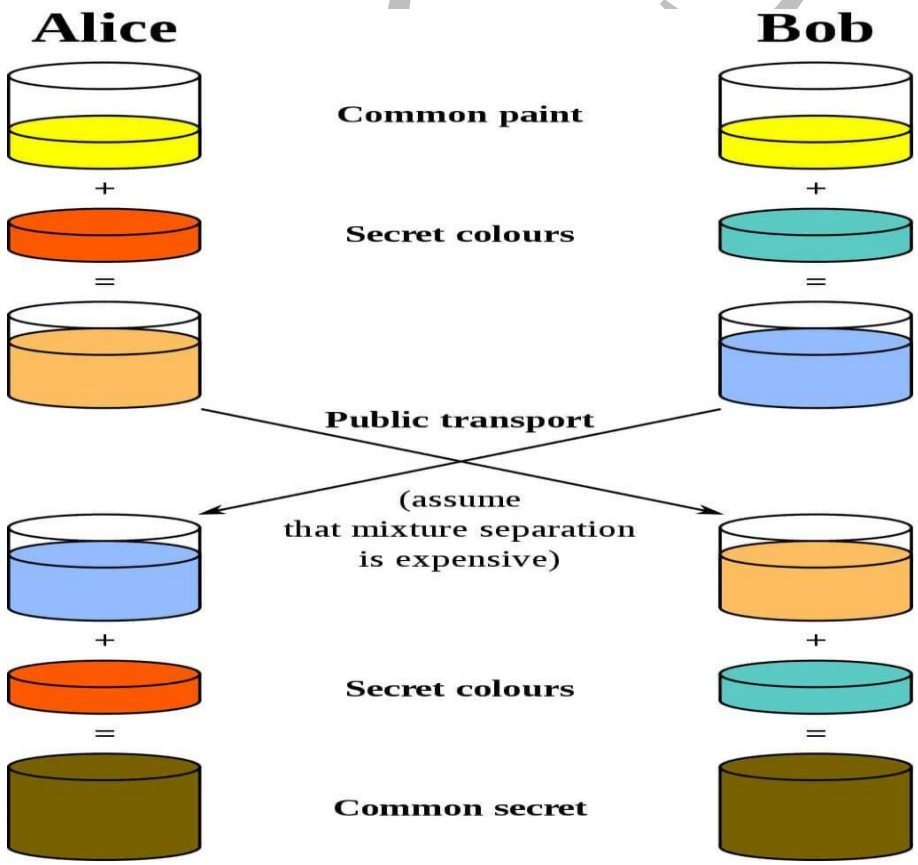
In view of these facts, modern cryptography will have to look for computationally harder problems or devise completely new techniques of archiving the goals presently served by modern cryptography.

Diffie-Hellman Key Exchange

The Diffie-Hellman key exchange is complex and it can be difficult to get your head around how it works. It uses very large numbers and a lot of math, something that many of us still dread from those long and boring high school lessons.

To make things a bit easier to understand, we will start by explaining the Diffie-Hellman key exchange with an analogy. Once you have a big-picture idea of how it works, we'll move on to a more technical description of the underlying processes.

The best analogy for the Diffie-Hellman scheme is to think of two people mixing paint. Let's use the cryptography standard, and say that their names are Alice and Bob. They both agree on a random color to start with. Let's say that they send each other a message and decide on yellow as their common color, just like in the diagram below:



They set their own color. They do not tell the other party their choice. Let's say that Alice chooses red, while Bob chooses a slightly-greenish blue.

The next step is for both Alice and Bob to mix their secret color (red for Alice, greenish-blue for Bob) with the yellow that they mutually agreed upon. According to the diagram, Alice ends up with an orangish mix, while Bob's result is a deeper blue.

Once they have finished the mixing, they send the result to the other party. Alice receives the deeper blue**,** while Bob is sent the orange-colored paint.

Once they have received the mixed result from their partner, they then add their secret color to it. Alice takes the deeper blue and adds her secret red paint**,** while Bob adds his secret greenish-blue to the orange mix he just received**.**

The result? They both come out with the same color, which in this case is a disgusting brown. It may not be the kind of color that you would want to paint your living room with, but it is a shared color nonetheless. This shared color is referred to as the common secret.

The critical part of the Diffie-Hellman key exchange is that both parties end up with the same result, without ever needing to send the entirety of the common secret across the communication channel. Choosing a common color, their own secret colors, exchanging the mix and then adding their own color once more, gives both parties a way to arrive at the same common secret without ever having to send across the whole thing.

If an attacker is listening to the exchange, all that they can access is the common yellow color that Alice and Bob start with, as well as the mixed colors that are exchanged. Since this is really done with enormous numbers instead of paint, these pieces of information aren't enough for the attack to discern either of the initial secret colors, or the common secret (technically it is possible to compute the common secret from this information, but in a secure implementation of the Diffie-Hellman

keyexchange, it would take an unfeasible amount of time and computationalresources to do so).

This structure of the Diffie-Hellman key exchange is what makes it so useful. It allows the two parties to communicate over a potentially dangerous connection and still come up with a shared secret that they can use to make encryption keys for their future communications. It doesn't matter if any attackers are listening in, because the complete shared secret is never sent over the connection.

The technical details of the Diffie-Hellman key exchange

Time for some math…

Don't worry, we'll take it slow and try to make the whole process as easy to understand as possible. It follows a similar premise as the analogy shown above, but instead of mixing and sending colors, the Diffie-Hellman scheme actually makes calculations based on exceptionally-large prime numbers, then sends them across.

To ensure security, it is recommended that the **prime (*p*) is at least 2048 bits long**, which is the binary equivalent of a decimal number of about this size:

415368757628736598425938247569843765827634879128375827365928736
842736847289385729837592834759348759384759284759287395872495 87
298739587298357928759827958375298763482736857298435793487958 27
938579287395487723975928375924785938670459867923847378267352 67
354762356873486938694567345682765949806384902487580960394790 2
794598273018743975928462095029375928704950293805892098394587 2
094860298491283750294801937109248019358103799581093750193

850791395710937597019385089103951073058710393701934701938091803984091804
9810938019850139840198350918350198309107918039581039519039518093 5
810938501984019358019384019834091809385109830918001 9

To prevent anyone's head from exploding, we'll run this explanation through with much smaller numbers. Be aware that the Diffie-Hellman key exchange would be insecure if it used numbers as small as those in our example. We are only using such small numbers to demonstrate the concept in a simpler manner.

In the most basic form of the Diffie-Hellman key exchange, Alice and Bob begin by mutually deciding upon two numbers to start with, as opposed to the single common paint in the example above. These are the modulus (*p*) and the base (*g*).

In practical use, the modulus (*p*) is a very large prime number, while the base (*g*) is relatively small to simplify calculations. The base (*g*) is derived from a cyclic group (*G*) that is normally generated well before the other steps take place.

For our example, let's say that the modulus (*p*) is **17**, while the base (*g*) is **4**.

Once they have mutually decided on these numbers, Alice settles on a secret number (**a**) for herself, while Bob chooses his own secret number (***b***). Let's say that they choose:

***a* = 3**

***b* = 6**

Alice then performs the following calculation to give her the number that she will send to Bob:

$A = g^a \bmod p$

In the above calculation, **mod** signifies a modulo operation. These are essentially calculations to figure out the remainder after dividing the left

side by the right. As an example:

15 mod 4 = 3

If you understand how modulo operations work, you can do them yourself in the following calculations, otherwise you can use an online calculator.

So let's put our numbers into the formula:

$A = 4^3 \bmod 17$

$A = 64 \bmod 17$

$A = 13$

When we do the same for Bob, we get:

$B = 4^6 \bmod 17$

$B = 4096 \bmod 17$

$B = 16$

Alice then sends her result (*A*) to Bob, while Bob sends his figure (*B*) to Alice. Alice then calculates the shared secret (*s*) using the number she received from Bob (*B*) and her secret number (*a*), using the following formula:

$s = B^a \bmod p$

$s = 16^3 \bmod 17$

$s = 4{,}096 \bmod 17$

$s = 16$

Bob then performs what is essentially the same calculation, but with the number that Alice sent him (*A*), as well as his own secret number (*b*):

$s = A^b \bmod p$

$s = 13^6 \bmod 17$

$s = 4{,}826{,}809 \bmod 17$

$s = 16$

As you can see, both parties ended up with the same result for *s*, **16**. This is the shared secret, which only Alice and Bob know. They can then use this to set up a key for symmetric encryption, allowing them to safely send information between themselves in a way that only they can access it.

*Note that although **B** and **s** are the same in the example above, this is just a coincidence based on the small numbers that were chosen for this illustration. Normally, these values would not be the same in a real implementation of the Diffie-Hellman key exchange.*

Even though much of the above data is sent across the channel in cleartext (*p*, *g*, *A* and *B*) and can be read by potential attackers, the shared secret (*s*) is never transmitted. It would not be practical for an attacker to calculate the shared secret (*s*) or either of the secret numbers(*a* and *b*) from the information that is sent in cleartext.

Of course, this assumes that the Diffie-Hellman key exchange is properly implemented and sufficiently large numbers are used. As long as these provisions are adhered to, the Diffie-Hellman key exchange is considered a safe way to establish a shared secret which can be used to secure future communications.

Establishing a shared key between multiple parties

The Diffie-Hellman key exchange can also be used to set up a shared key with a greater number of participants. It works in the same manner, except further rounds of the calculations are needed for each party to add in their secret number and end up with the same shared secret.

Just like in the two-party version of the Diffie-Hellman key exchange, some parts of the information are sent across insecure channels, but not enough for an attacker to be able to compute the shared secret.

Why is the Diffie-Hellman key exchange secure?

On a mathematical level, the Diffie-Hellman key exchange relies on one-way functions as the basis for its security. These are calculations which are simple to do one way, but much more difficult to calculate in reverse.

More specifically, it relies on the Diffie-Hellman problem, which assumes that under the right parameters, it is infeasible to calculate *gab* from the separate values of *g*, *ga* and *gb*. There is currently no publicly known way to easily find *gab* from the other values, which is why the Diffie-Hellman key exchange is considered secure, despite the fact that attackers can intercept the values *p*, *g*, *A*, and *B*.

Attacks against encryption

1. Brute force attacks

Brute-force attacks involve trying every possible character combination to find the 'key' to decrypt an encrypted message. While brute-force attacks may take a smaller amount of time for smaller key spaces, it will take an immeasurable amount of time for larger keyspaces. Hence it is impractical to try brute-force attacks modern encryption systems.

2. Cipher-only attack

In the 'cipher-only' attack, the attacker knows the ciphertext of various messages which have been encrypted using the same encryption algorithm. The attacker's challenge is to figure the 'key' which can then be used to decrypt all messages.

The 'cipher-only' attack is probably one of the easiest attacks to commit since it is easy to capture the ciphertext (by sniffing) but difficult to implement since the knowledge about the encryption process is limited.

3. Known-plaintext attack

In the 'known-plaintext' attack, the attacker knows some of the plaintext and the ciphertext. He then has to figure the 'key' by reverse

engineering and he can decipher other messages which use the same 'key' and algorithm.

The 'known-plaintext' attack was effective against simple ciphers such as the 'substitution cipher'. It was popular for breaking ciphers used during the Second World War.

4. Chosen plaintext attack

The 'chosen-plaintext' attack is similar to the 'known-plaintext' attack, but here the attacker experiments by choosing his own plaintext (say choosing a word such as 'cryptography') for a 'Vignere cipher' and with the generated ciphertext he can figure the 'key'.

Once he figures the 'key' he can learn more about the whole encryption process and understand how the 'key' is being used.
With this information, he can decrypt other messages.

5. Chosen ciphertext attack

In the 'chosen ciphertext' attack, the attacker chooses a portion of the decrypted ciphertext. He then compares the decrypted ciphertext with the plaintext and figures out the key.

This is relatively a harder type of attack and earlier versions of RSA were subject to these types of attacks.

6. Differential cryptanalysis

This was a popular type of attack against block algorithms such as DES in 1990. The primary aim of this attack, as with other attacks, is finding the 'key'.

The attacker follows several messages of plaintext into their transformed ciphertext. He observes the changes form plaintext to the ciphertext and deduces the key.

This is a type of 'chosen-plaintext' attack since the attacker chooses the plaintext to observe the transformation.

7. Linear cryptanalysis

The attacker carries out a "known-plaintext" attack against several messages which have been encrypted with the same key. This gives the attacker insight into the probability of a particular key. If more messages are attacked, there is a higher possibility of finding the particular "key".

8. Side channel attacks

Apart from just relying on mathematical ways to break into systems, attackers may use other techniques such as observing power consumption, radiation emissions and time for data processing. With this data, the attacker works in a reverse manner to figure the 'keys' to an algorithm just by observing the amount of heat released in an attack.

RSA keys in 1995 were uncovered using this type of attack.

9. Replay attacks

In a 'replay attack', the attacker captures some information (as an example, authentication information) and re-submits it back to the receiver. This dupes the receiver and they give the attacker unauthorized access.

Timestamps are one of the important countermeasures to handle 'replay attacks.'

HMAC

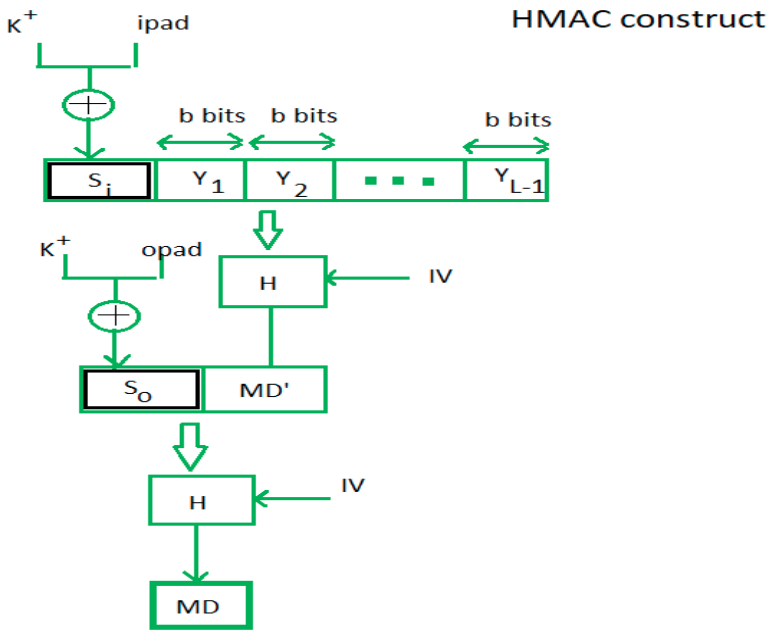
HMAC algorithm stands for Hashed or Hash-based Message Authentication Code. It is a result of work done on developing a MAC derived from cryptographic hash functions. HMAC is a great resistance towards cryptanalysis attacks as it uses the hashing concept twice. HMAC consists of twin benefits of Hashing and MAC and thus is more secure than any other authentication code. RFC 2104 has issued HMAC, and HMAC has been made compulsory to implement in IP security. The FIPS 198 NIST standard has also issued HMAC.

- As the Hash Function, HMAC is also aimed to be one way, i.e, easy to generate output from input but complex the other way round.
- It aims at being less affected by collisions than the hash functions.
- HMAC reuses the algorithms like MD5 and SHA-1 and checks to replace the embedded hash functions with more secure hash functions,in case found.
- HMAC tries to handle the Keys in a simpler manner.

HMAC algorithm –

The working of HMAC starts with taking a message M containing blocks oflength *b* bits. An input signature is padded to the left of the message and thewhole is given as input to a hash function which gives us a temporary message-digest MD'. MD' again is appended to an output signature and the whole is applied a hash function again, the result is our final message digestMD.

Here is a simple structure of HMAC:



Here, H stands for hashing function,M is
the original message
Si and So are input and output signatures respectively,
Yi is the ith block in original message M, where I ranges from [1, L)L =
the count of blocks in M
K is the secret key used for hashing IV is
an initial vector (some constant)
The generation of input signature and output signature *Si* and *So* respectively.

$S_i = K^+ \oplus ipad$ where K^+ is nothing but K padded with zeros on the left so that the result is b bits in length

$S_o = K^+ \oplus opad$ where ipad and opad are 00110110 and 01011100 respectively taken b/8 times repeatedly.

$MD' = H(S_i || M)$

$MD = H(S_o || MD')$ or $MD = H(S_o || H(S_i || M))$

To a normal hash function, HMAC adds a compression instance to the processing. This structural implementation holds efficiency for shorter MAC values.

Public Key Cryptography Standards

1. PKCS

In cryptography, PKCS stands for "Public Key Cryptography Standards". Theseare a group of public-key cryptography standards devised and published by RSASecurity LLC, starting in the early 1990s. The company published the standardsto promote the use of the cryptography techniques to which they had patents, such as the RSA algorithm, the Schnorr signature algorithm and several others.

Public Key Cryptography Standard (PKCS), which has had a significant impact on the use of public-key encryption in practice. The PKCS standard is a set of standards called PKCS 1 to 15. These standards cover RSA encryption, RSA signature, password-based encryption, encrypted message syntax, private key information syntax, selected object category and attribute type, authentication request syntax, encryption token interface, personal information exchange syntax, and encrypted token information grammar. RSA Laboratories publishes the PKCS standard. Although RSA Laboratories solicits comments and suggestions from the public on the PKCS standard, RSA Laboratories reserves the exclusive power to decide all aspects of the PKCS standard. PKCS has become the basis for many other standards, such as S/MIME.

PKCS did not become industry standards initially because RSA retained controlover them, but many of the standards were adapted by other working groups.
The standards were developed by RSA with the cooperation of industry partnerswhich included Apple, Microsoft, Lotus, Sun, DEC and MIT.

FIPS 140-2

This standard specifies the security requirements that will be satisfied by a cryptographic module utilized within a security system protecting sensitive but unclassified information (hereafter referred to as sensitive information). The standard provides four increasing, qualitative levels of security: Level 1, Level 2, Level 3, and Level 4. These levels are intended to cover the wide range of potential applications and environments in which cryptographic modules may be employed. The security requirements cover areas related to the secure design and implementation of a cryptographic module. These areas include cryptographic module specification, cryptographic module ports and interfaces; roles, services, and authentication; finite state model; physical security; operational environment; cryptographic key management; electromagnetic interference/electromagnetic compatibility (EMI/EMC); self-tests; design

assurance; and mitigation of other attacks. This standard supersedes FIPS 140- 1, Security Requirements for Cryptographic Modules, in its entirety.

The Cryptographic Module Validation Program (CMVP) validates cryptographic modules to Federal Information Processing Standard (FIPS) 140- 2 and other cryptography-based standards. The CMVP is a joint effort between NIST and the Communications Security Establishment (CSE) of the Government of Canada. Products validated as conforming to FIPS 140-2 are accepted by the Federal agencies of both countries for the protection of sensitiveinformation (United States) or Designated Information (Canada). The goal of theCMVP is to promote the use of validated cryptographic modules and provide Federal agencies with a security metric to use in procuring equipment containingvalidated cryptographic modules.

Strong Authentication

Strong Authentication is a method of user verification that is considered robust enough to withstand attacks on the system to which the users are authenticating.There are competing definitions of strong authentication, as layered systems often do not meet the required threshold for being strong. Strong authenticationis thought to be true two-factor authentication or multi-factor authentication (2FA, MFA). Such systems, in requiring two or more factors from the "something I have, something I am, and something I know), require those factorsto be a combination of different authentication factors. Technologies that enable strong authentication include information derived from the devices in use by people (e.g. mobile or laptop) as well as user biometrics, user email, one-time passwords (OTPs), and time-based one-time passwords (TOTPs). True strong authentication also takes the form of cryptographic solutions wherepublic-key infrastructure (PKI) is the underlying system. Such systems ensure that the user and the verifying system do not share sensitive information. Rather,the parties exchange non-sensitive mutually agreed upon information to verify that they are the authorized parties to a conversation. Fast Identity Online (FIDO)Alliance authentication standards are examples of such a system.

In addition, in the consumer space, Strong Customer Authentication (SCA) is defined in the recent update to the European Union's Payment Services Directive(PSD, PSD2). SCA serves as an accepted strong authentication definition.

Single-Factor Authentication (SFA)

Single-Factor Authentication (SFA) is a method of logging users into resources by having them present only one way of verifying their identity. Username and password is the dominant form of SFA.

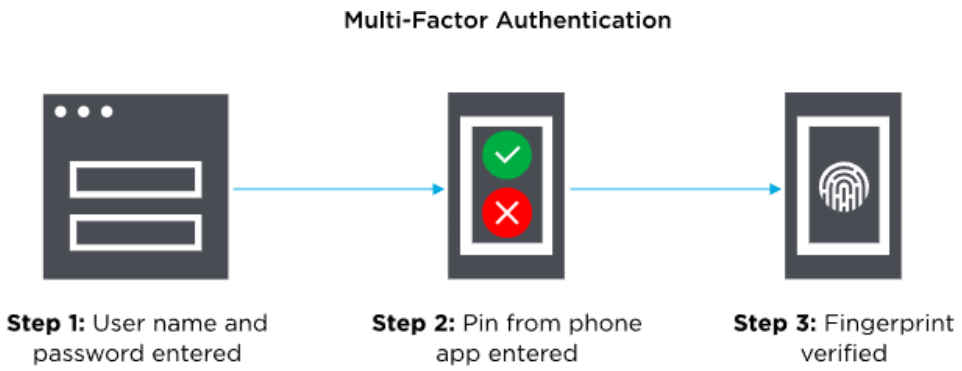
The factors of authenticating users generally take the form of either something you know (knowledge), something you have (possession), and something you are (biometric). Continuous authentication is an emerging means using one factor however even it compiles contextual information derived from knowledge and possession, in addition to device information such as geolocation.

As the number-one SFA method, password authentication relies on mutual secrets between user and the online service. Shared secrets have shown to be vulnerable to credential stuffing attacks since users recycle passwords across different services, and hackers weaponize passwords from prior breaches against unrelated enterprises. In response, enterprises implement two-factor authentication (2FA) or multi-factor authentication (MFA) to add a possession factor often using hardware or software tokens. These methods, however, add friction to the user experience (UX) and result in poor 2FA or MFA adoption, or security workarounds such as sharing tokens.

Everyday mobile devices are helping enterprises overtake the reliance on SFA and the challenges of reliable 2FA and MFA. Smartphones are de-factor MFA and have multiple authenticators, security features such hardware trust zones, and can be leveraged for public-key cryptography (PKC). This makes smartphones crucial to password less authentication based on true secrets, and fast, simple MFA.

Multi-Factor Authentication (MFA)

Multi-factor Authentication (MFA) is an authentication method that requires the user to provide two or more verification factors to gain access to a resource such as an application, online account, or a VPN. MFA is a core component of a strong identity and access management (IAM) policy. Rather than just asking for a username and password, MFA requires one or more additional verification factors, which decreases the likelihood of a successful cyber- attack.



The main benefit of MFA is it will enhance your organization's security by requiring your users to identify themselves by more than a username and password. While important, usernames and passwords are vulnerable to brute force attacks and can be stolen by third parties. Enforcing the use of an MFA factor like a thumbprint or physical hardware key means increased confidence that your organization will stay safe from cyber criminals.

MFA works by requiring additional verification information (factors). One of the most common MFA factors that users encounter is one-time passwords (OTP). OTPs are those 4–8-digit codes that you often receive via email, SMS or some sort of mobile app. With OTPs a new code is generated periodicallyor each time an authentication request is submitted. The code is generatedbased upon a seed value that is assigned to the user when they first registerand some other factor which could simply be a counter that is incremented ora time value.

Most MFA authentication methodology is based on one of three types ofadditional information:

- Things you know (knowledge), such as a password or PIN
- Things you have (possession), such as a badge or smartphone
- Things you are (inherence), such as a biometric like fingerprints or voice recognition

Difference between single-factor and multi-factor authentication systems:

Single-factor authentication	Multi-factor authentication
In single-factor authentication, the process of authentication is much simpler	In multi-factor authentication, the process of successful authentication can be complex.
There is a risk of security.	There is no risk of security.
There is a chance of information getting stolen	There is not a chance, for the information to get stolen.
There is a risk of a keylogger who can steal passwords and more.	There is no risk of keylogger activity.
The user is not in full control.	The user is in full control.
There is a chance of user password getting captured by the keylogger.	There is no risk of data getting captured through keylogger.
There is a chance of a phishing attack, where the attacker may deceive the user to enter the password or userid.	In multi-factor authentication, a phishing attack will not accomplish its purpose.

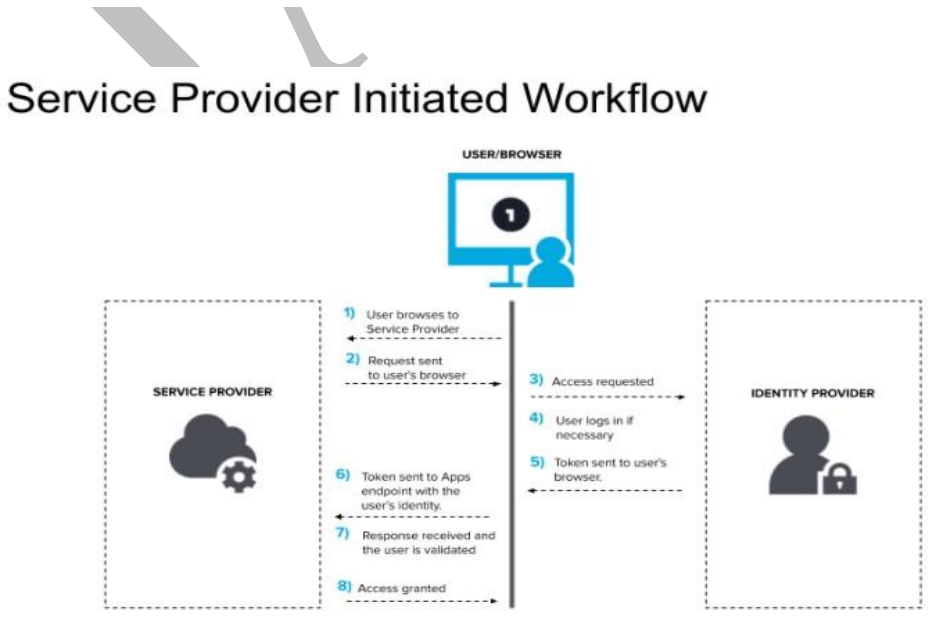
Single Sign-on Solutions

Single sign-on (SSO) is an authentication method that enables users to securely authenticate with multiple applications and websites by using just one set of credentials. SSO works based upon a trust relationship set up between an application, known as the service provider, and an identity provider, like OneLogin. This trust relationship is often based upon a certificate that is exchanged between the identity provider and the service provider. This certificate can be used to sign identity information that is being sent from the identity provider to the service provider so that the service provider knows itis coming from a trusted source. In SSO, this identity data takes the form of

tokens which contain identifying bits of information about the user like auser's email address or a username.

The login flow usually looks like this:

1. A user browses to the application or website they want access to, aka, the Service Provider.
2. The Service Provider sends a token that contains some information about the user, like their email address, to the SSO system, aka, the Identity Provider, as part of a request to authenticate the user.
3. The Identity Provider first checks to see whether the user has already been authenticated, in which case it will grant the user access to the Service Provider application and skip to step 5.
4. If the user hasn't logged in, they will be prompted to do so by providing the credentials required by the Identity Provider. This could simply be a username and password or it might include some other form of authentication like a One-Time Password (OTP).
5. Once the Identity Provider validates the credentials provided, it will send a token back to the Service Provider confirming a successful authentication.
6. This token is passed through the user's browser to the Service Provider.
7. The token that is received by the Service Provider is validated according to the trust relationship that was set up between the Service Provider and the Identity Provider during the initialconfiguration.
8. The user is granted access to the Service Provider.



When the user tries to access a different website, the new website would haveto have a similar trust relationship configured with the SSO solution and the authentication flow would follow the same steps.

An SSO token is a collection of data or information that is passed from one system to another during the SSO process. The data can simply be a user's email address and information about which system is sending the token. Tokens must be digitally signed for the token receiver to verify that the tokenis coming from a trusted source. The certificate that is used for this digital signature is exchanged during the initial configuration process.

There are many reasons why SSO can improve security. A single sign-on solution can simplify username and password management for both users and administrators. Users no longer have to keep track of different sets of credentials and can simply remember a single more complex password. SSO often enables users to just get access to their applications much faster.

SSO can also cut down on the amount of time the help desk has to spend on assisting users with lost passwords. Administrators can centrally control requirements like password complexity and multi-factor authentication (MFA). Administrators can also more quickly relinquish login privileges across the board when a user leaves the organization.
Single Sign-On does have some drawbacks. For example, you might have applications that you want to have locked down a bit more. For this reason, it would be important to choose an SSO solution that gives you the ability to,say, require an additional authentication factor before a user logs into a particular application or that prevents users from accessing certain applications unless they are connected to a secure network.

OpenID

OpenID allows you to use an existing account to sign in to multiple websites, without needing to create new passwords.

You may choose to associate information with your OpenID that can be shared with the websites you visit, such as a name or email address. With OpenID, youcontrol how much of that information is shared with the websites you visit.

With OpenID, your password is only given to your identity provider, and that provider then confirms your identity to the websites you visit. Other than your

provider, no website ever sees your password, so you don't need to worry aboutan unscrupulous or insecure website compromising your identity.

OpenID is rapidly gaining adoption on the web, with over one billion OpenID enabled user accounts and over 50,000 websites accepting OpenID for logins. Several large organizations either issue or accept OpenIDs, including Google, Facebook, Yahoo!, Microsoft, AOL, MySpace, Sears, Universal MusicGroup, France Telecom, Novell, Sun, Telecom Italia, and many more.

OAUTH

OAuth is an open-standard authorization protocol or framework that provides applications the ability for "secure designated access." For example, you can tell Facebook that it's OK for ESPN.com to access your profile or post updates to your timeline without having to give ESPN your Facebook password. This minimizes risk in a major way: In the event ESPN suffers a breach, your Facebook password remains safe.

OAuth doesn't share password data but instead uses authorization tokens to prove an identity between consumers and service providers. OAuth is an authentication protocol that allows you to approve one application interacting with another on your behalf without giving away your password.

NOTE: OpenID is used for authentication while OAuth is used for authorization.

Graphical Passwords

A graphical password or graphical user authentication is a form of authentication using images rather than letters, digits, or special characters.The type of images used and the ways in which users interact with them vary between implementations.

With increasing technical advancements, the world is becoming digital at a highpace and everything is happening online. From paying your bills to ticket bookings to paying the person sitting next to you, you prefer to pay online. Not only payments but all activities, be it, communication through e-mails and messaging apps, keeping your documents in a digital locker, etc happen online.

With everything turning online, the risk of cybercrimes and privacy breachesis also increasing. Passwords play a huge role in keeping your data safe online as well as offline platforms. Passwords are the default method of authentication

to get access to our accounts. There are various types of authenticationsavailable for users to secure their accounts.

Types of authentications

- **Token-based authentication** includes key cards, bank cards, smart cards, etc.

- **Knowledge-based authentication** includes text-based authentication and picture-based authentication.

- **Biometric authentication** includes fingerprints authentication, iris scan and facial recognition.

Considering the traditional username-password authentication, the alphanumeric passwords are either easy to guess or difficult to remember. Also,users generally keep the same passwords for all their accounts because it is difficult to remember a lot of them. Alternative authentication methods, suchas biometrics, graphical passwords are used to overcome these problems associated with the traditional username-password authentication technique.

In a graphical password authentication system, the user has to select from images, in a specific order, presented to them in a graphical user interface (GUI). According to a study, the human brain has a greater capability of remembering what they see(pictures) rather than alphanumeric characters. Therefore, graphical passwords overcome the disadvantage of alphanumeric passwords. Graphical Password Authentication has three major categories based on the activity they use for authentication of the password:

- **Recognition based Authentication:** A user is given a set of images and he has to identify the image he selected during registration. For example, *Passfaces* is a graphical password scheme based on recognizing human faces. During password creation, users are given a large set of images to select from. To log in, users have to identify the pre-selected image from the several images presented to him.
- **Recall based Authentication:** A user is asked to reproduce something that he created or selected at the registration stage. For example, in the *Passpoint* scheme, a user can click any point in an image to create the password and a tolerance around each pixel is calculated. During authentication, the user has to select the points within the tolerance in the correct sequence to login.
- **Cued Recall:** Cued Click Points (CCP) is an alternative to the PassPoints technique. In CCP, users click one point on each image rather than on five points on one image (unlike PassPoints). It offers cued- recall and instantly alerts the users if they make a mistake while enteringtheir latest click-point.

Advantages:

- It is user-friendly.
- It provides higher security than other traditional password schemes.
- Dictionary attacks are infeasible.
- CCP makes attacks based on hotspot analysis more challenging.

Disadvantages:

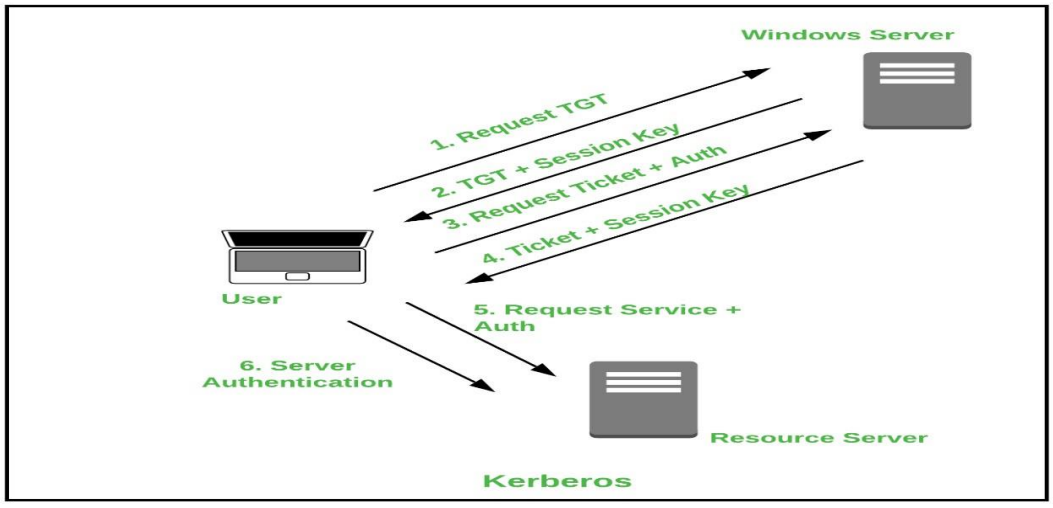
- Registration and login take too long.
- It requires more storage space because of images.
- Shoulder surfing (Watching over people's shoulders as they process information).

Authentication Protocols

User authentication is the first most priority while responding to the request made by the user to the software application. There are several mechanisms made which are required to authenticate the access while providing access tothe data. In this blog, we will explore the most common authentication protocols and will try to explore their merits and demerits.

1. Kerberos :

Kerberos is a protocol that aids in network authentication. This is used for validating clients/servers during a network employing a cryptographic key. Itis designed for executing strong authentication while reporting to applications. The overall implementation of the Kerberos protocol is openly available by MIT and is used in many mass-produced products.



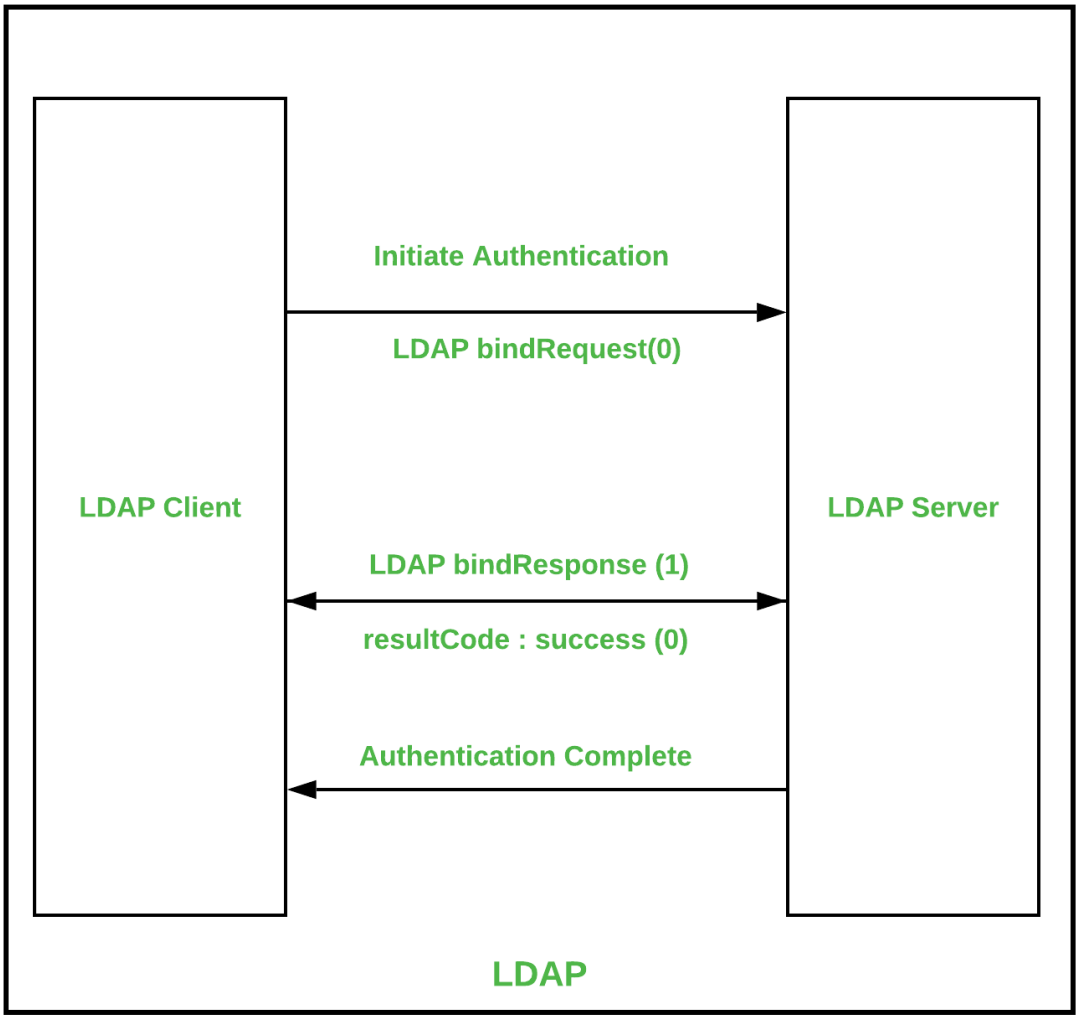
Some advantages of Kerberos:

- It supports various operating systems.
- The authentication key is shared much efficiently than public sharing.

Some disadvantages of Kerberos:

- It is used only to authenticate clients and services used by them.
- It shows vulnerability to soft or weak passwords.

LDAP refers to Lightweight Directory Access Protocol. It is a protocol that is used for determining any individuals, organizations, and other devices during a network regardless of being on public or corporate internet. It is practiced as Directories-as-a-Service and is the grounds for Microsoft building Activity Directory.



2. Lightweight Directory Access Protocol (LDAP):

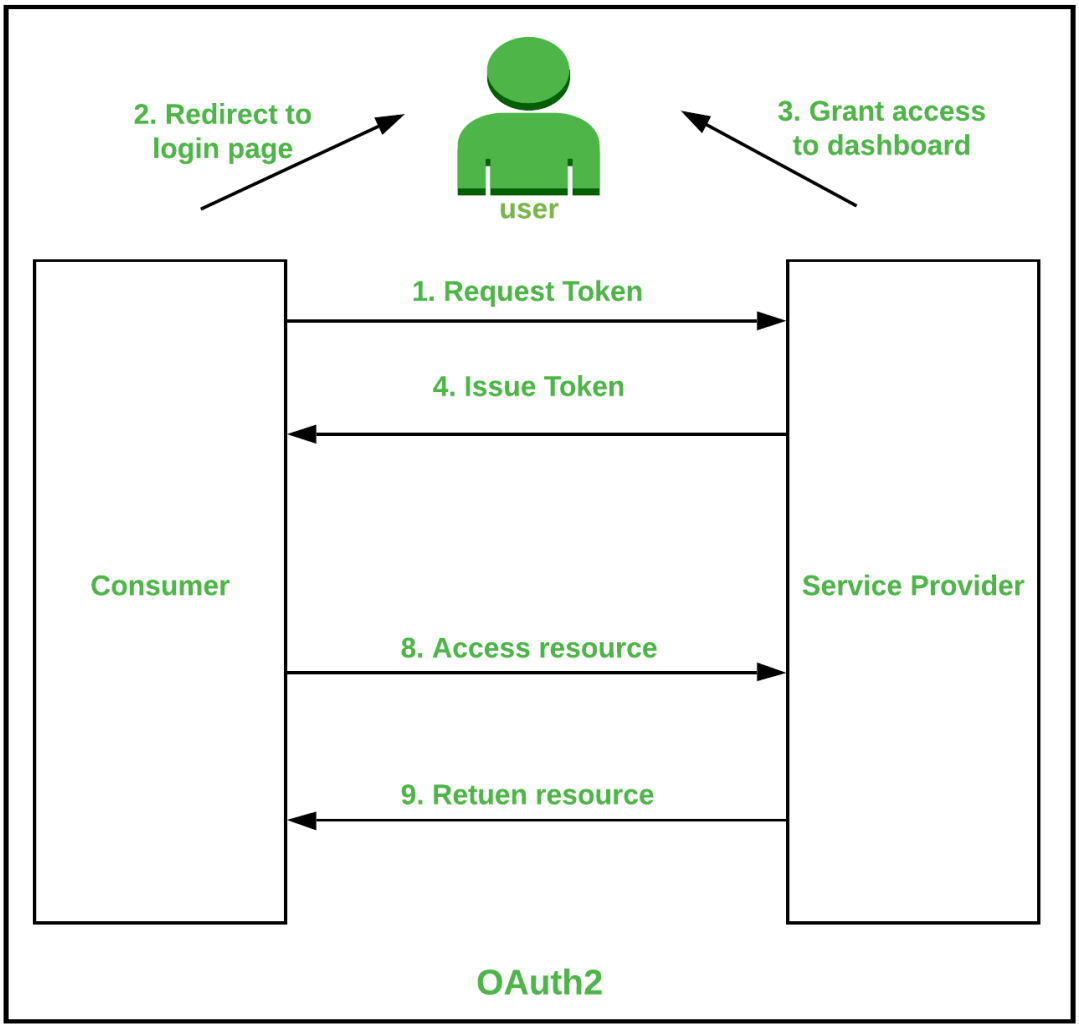
Some advantages of LDAP:

- It is an automated protocol which makes it modernizing easier.
- It supports existing technologies and allows multiple directories.

Some disadvantages of LDAP:

- It requires the experience of deployment.
- The directory servers are required to be LDAP obedient for deployment.
-

OAuth as the name suggests it is an authorization framework that promotes granting limited access to the user on its account through an HTTP service. When a user requests access to resources an API call is made and after the authentication token is passed.



3. OAuth2 :

Some advantages of OAuth2:

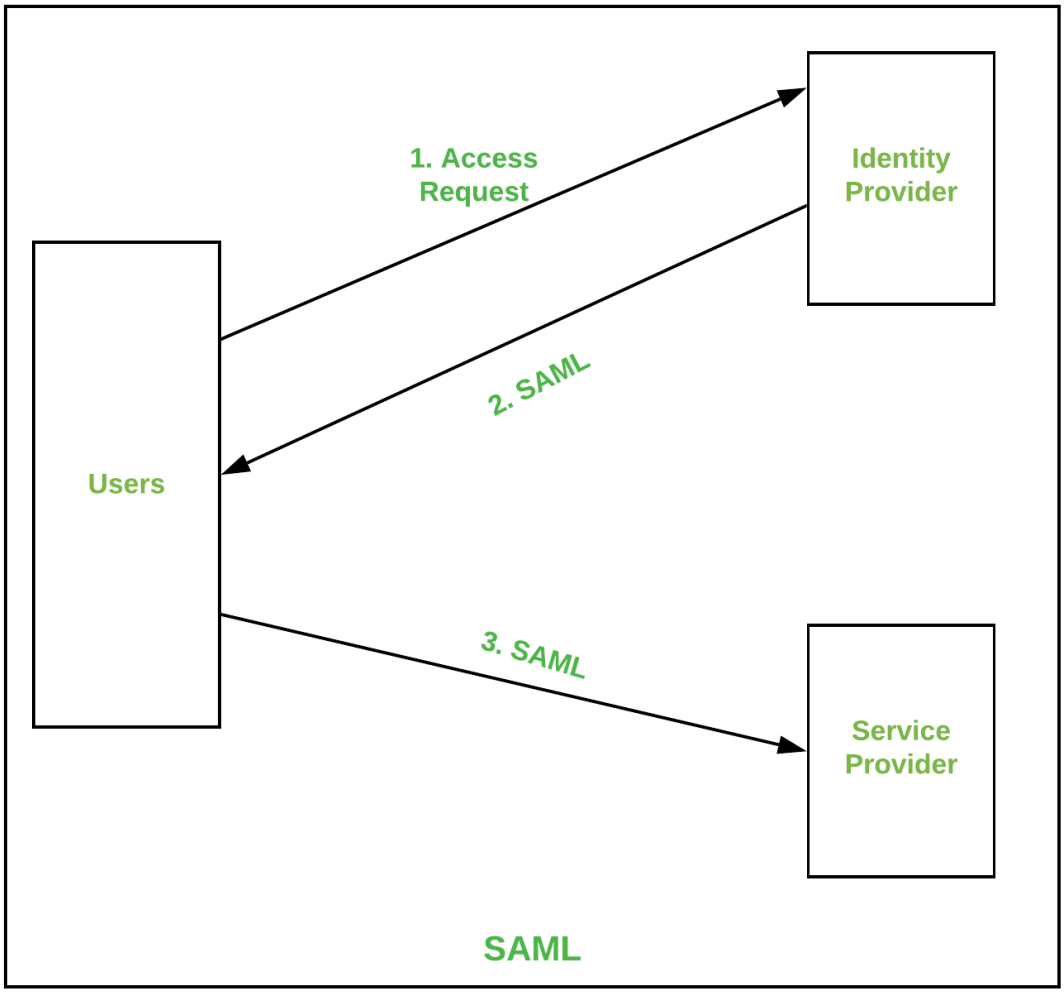
- It is a simple protocol and is easy to implement.
- It provides server-side authorization of code.

Some disadvantages of OAuth2:

- It is vulnerable to manage different sets of code.
- It shows serious effects on sites connected to another affected system.
-

4. SAML :

SAML stands for Security Assertion Markup Language which is based on XML-based authentication data format which provides the authorization between an identity provider and service provider. It serves as a product of the OASIS Security Services Technical Committee.



Some advantages of SAML:

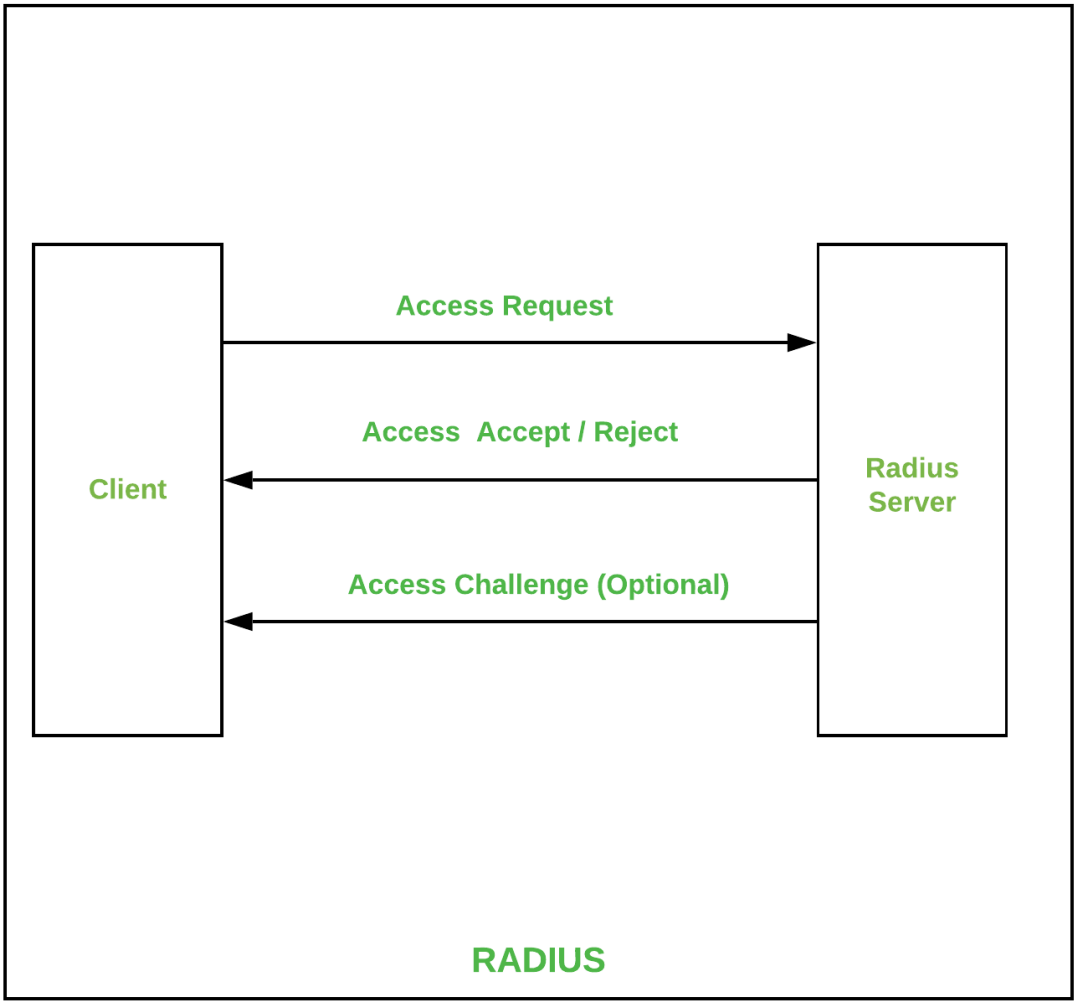
- It reduced the administrative costs for the end-users.
- It provides a single sign-in for authenticating across service providers.

Some disadvantages of SAML:

- It is dependent on the identity provider.
- All the data is managed in a single XML format.

5. RADIUS :

RADIUS stands for Remote Authentication Dial-In User Service. It is a network protocol that provides sufficient centralized Authentication, Accounting, and Authorization for the users that use and network services. The functioning of the protocol occurs when the user requests access to network resources, where the RADIUS server encrypts the credentials which are entered by the user. After this, the user credentials are mapped through the localdatabase and provide access.



Some advantages of RADIUS:

- It is a great mechanism for providing multiple access for Admins.
- It provides a unique identity to each user in a session.

Some disadvantages of RADIUS:

- Initial implementation for this mechanism is hard on hardware.
- It has a variety of models that may require a special team which is cost consuming.

FIDO

Fast Identity Online (FIDO) Authentication is a set of open technical specifications that define user authentication mechanisms that reduce the reliance on passwords.

FIDO protocols are designed from the ground up to protect user privacy. The protocols do not disclose sensitive user data that can be used by different onlineservices to collaborate and track a user across the services. Other sensitive data like biometric prints and PINs never leaves the user's device to ensure it cannotbe intercepted or compromised by an attacker.

To authenticate a user, an application – often referred to as the relying party – uses FIDO-specified client-side APIs to interact with a user's registered authenticator. For web applications, client-side APIs include WebAuthn implemented by the web browser, which in turn calls on FIDO CTAP to access the authenticator.

To authenticate a user, the relying party passes a cryptographic challenge to the registered authenticator and evaluates the response to determine the authenticityof the secrets stored on the client device and used to produce the response.

"Under the hood" FIDO utilizes asymmetric cryptography to ensure that all sensitive secrets and cryptographic key material remain on the client device at all times and are not transmitted to the authenticating service.

FIDO authentication requires an initial registration step. In cases where the userdevice supports multiple forms of authentication (i.e. fingerprint scanner, voiceprint recorder, face ID, etc.), the user is asked to choose a FIDO compliantauthenticator from the options available on the device that matches the authenticating app's acceptance policy. The user then unlocks the FIDO authenticator using whatever mechanism is built into the authenticator – e.g. byproviding a fingerprint, pressing a button on a second–factor device, or enteringPIN.

Once the authenticator is unlocked, the user's device creates a new and unique public/private cryptographic key pair that will be used for authenticating access.The public key is then sent to the online service and associated with the user's account. The private key and all other sensitive data related to the chosen authentication method – for example, biometric prints – remain on the local device and never leave it.

Authentication requires the client device to prove possession of the private key to the authenticating service by successfully responding to a cryptographic challenge. The private key can only be used after successfully authenticating using the registered authenticator, for example by swiping a finger on the fingerprint sensor, entering a PIN, speaking into a microphone, inserting a second–factor device, pressing a button, etc. The device then uses the user account identifier provided by the service to select the correct key and cryptographically sign the service's challenge. The signed challenge is sent backto the service, which verifies it with the stored public key and logs in the user.

Zero Trust Architecture

Security model that requires strict verification for every person & device tryingto access resources on a private network regardless of whether they are sitting within or outside the network perimeter.

It is also known as Zero trust security or Zero trust network.

No one trusted by default.

➢ Identity Access Management

Identity and access management (IAM) ensures that the right people and job roles in your organization (identities) can access the tools they need to do their jobs. Identity management and access systems enable your organization to manage employee apps without logging into each app as an administrator. Identity and access management systems enable your organization to manage a range of identities including people, software, and hardware like robotics and IoT devices.

Companies need IAM to provide online security and to increase employee productivity.

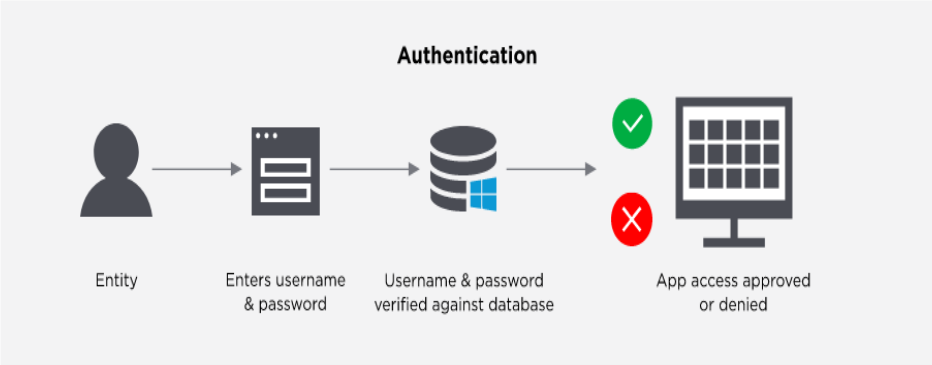
- **Security.** Traditional security often has one point of failure - the password. If a user's password is breached - or worse yet, the email address for their password recoveries - your organization becomes vulnerable to attack. IAM services

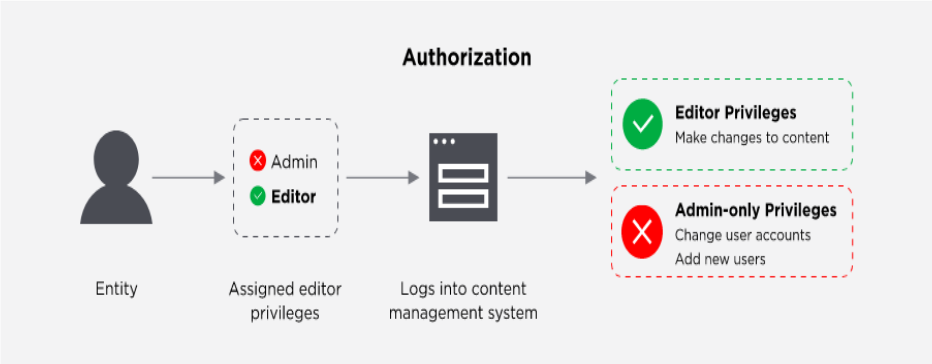
narrow the points of failure and backstops them with tools to catch mistakeswhen they're made.

- **Productivity.** Once you log on to your main IAM portal, your employee no longer has to worry about having the right password or right access level to perform their duties. Not only does every employee get access to the perfect suite of tools for their job, their access can be managed as a group or role instead of individually, reducing the workload on your IT professionals.

Identity management solutions generally perform two tasks:

1. IAM confirms that the user, software, or hardware is who they say they are by authenticating their credentials against a database. IAM cloud identity tools are more secure and flexible than traditional username and password solutions.
2. Identity access management systems grant only the appropriate level of access. Instead of a username and password allowing access to an entire software suite, IAM allows for narrow slices of access to be portioned out, i.e. *editor*, *viewer*, and *commenter* in a content management system.





Privileged Access Management

Privileged Access Management (PAM) is an information security (infosec) mechanism that safeguards identities with special access or capabilities beyond regular users. Like all other infosec solutions, PAM works through a combination of people, processes, and technology.

We treat privileged accounts with extra care because of the risk they pose tothe technology environment. For example, should the credentials of an administrator or service account fall into the wrong hands, it could lead to the compromise of the organization's systems and confidential data.

Data breaches occur when threat actors compromise privileged access accounts. As these accounts hold the keys that unlock every door in a technology environment, we need to add additional layers of protection. That extra security is a Privileged Access Management solution.

What does Privileged Access mean?

In a technology environment, privileged access refers to accounts with elevated capabilities beyond regular users. For example, in a Linux environment, the root user can add, amend, or delete users, install and uninstall software, and access restricted parts of the operating system that are off-limits to a standard user. Windows environments follow a similar security construct, but the root user in that instance is called an administrator.

Let's illustrate the concept of privileged access with a real-world bankingexample. A typical bank has customers, tellers, and managers. Each 'user' has different levels of authority when it comes to accessing the bank's cash. Customers can only access the money in their bank accounts. Tellers have more privileges than regular customers as they have access to all the cash in their respective drawers. Managers have even greater access than tellers, as they can access the money stored in the bank's vault. Technology systems also use this tiered privilege access model. Your role within the system determines what you can or cannot do.

In our banking example, the tellers and managers would be the users with privileged access. As these roles have access to more of the bank's cash than customers, the bank needs to implement additional security measures before granting tellers and manager’s access. For instance, during their job interviews, they may need to pass a criminal record check. When they start

working at the bank, their role will also determine their physical access. For example, tellers may be able to enter the secure area of the bank, but only managers will have the privileged access needed to enter the vault.

PAM vs. IAM

Privileged Access Management (PAM) is a component of a broader Identityand Access Management (IAM) solution. A PAM deals with the process and technologies needed to secure privileged accounts. On the other hand, an IAM solution offers password management, Multi-Factor Authentication, Single Sign-On (SSO), and user lifecycle management for all accounts, not just those with privileged access.

Securing Websites & Email

➢ SSL

Secure Socket Layer (SSL) is a technology to secure the communication between a client and the server. An SSL for email ensures that the email is not intercepted during the transit and nobody except the intended recipient can access it. An email SSL certificate can also authenticate the identity of the sender. The SSL certificate in your email account serves two purposes – to authenticate the sender's identity and maintain the integrity of the email. The email certificates are also known as S/MIME or email encryption certificates.

➢ TLS

Transport Layer Securities (TLS) are designed to provide security at the transport layer. TLS was derived from a security protocol called Secure Service Layer (SSL). TLS ensures that no third party may eavesdrops or tampers with any message.

There are several benefits of TLS:

- Encryption:
  TLS/SSL can help to secure transmitted data using encryption.
- Interoperability:
  TLS/SSL works with most web browsers, including Microsoft InternetExplorer and on most operating systems and web servers.

- Algorithm flexibility:
  TLS/SSL provides operations for authentication mechanism, encryption algorithms and hashing algorithm that are used during thesecure session.
- Ease of Deployment:
  Many applications TLS/SSL temporarily on a windows server 2003 operating systems.
- Ease of Use:
  Because we implement TLS/SSL beneath the application layer, mostof its operations are completely invisible to client.

Working of TLS:
The client connects to server (using TCP), the client will be something. Theclient sends number of specifications:

1. Version of SSL/TLS.
2. which cipher suites, compression method it wants to use.

The server checks what the highest SSL/TLS version is that is supported by them both, picks a cipher suite from one of the client's options (if it supports one) and optionally picks a compression method. After this the basic setup is done, the server provides its certificate. This certificate must be trusted eitherby the client itself or a party that the client trusts. Having verified the certificate and being certain this server really is who he claims to be (and nota man in the middle), a key is exchanged. This can be a public key, "PreMasterSecret" or simply nothing depending upon cipher suite.

Both the server and client can now compute the key for symmetric encryption. The handshake is finished and the two hosts can communicate securely. To close a connection by finishing. TCP connection both sides will know the connection was improperly terminated. The connection cannot be compromised by this through, merely interrupted.

➢ PGP – Authentication and Confidentiality

In 2013, when the *NSA (United States National Security Agency) scandal* was leaked to the public, people started to opt for the services which can provide

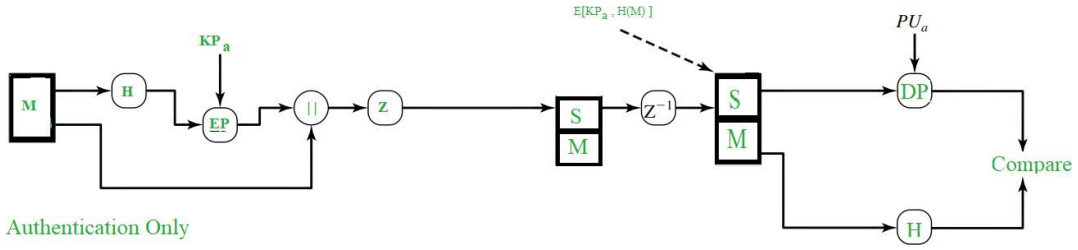
them a strong privacy for their data. Among the services people opted for, most particularly for emails, were different plug-ins and extensions for their browsers. Interestingly, among the various plug-ins and extensions that people started to use, there were two main programs that were solely responsible for the complete email security that the people needed.

As said, PGP (Pretty Good Privacy), is a popular program that is used toprovide confidentiality and authentication services for electronic mail and file storage. It was designed by Phil Zimmermann way back in 1991. He designedit in such a way, that the best cryptographic algorithms such as RSA, Diffie- Hellman key exchange, DSS are used for the public-key encryption (or) asymmetric encryption; CAST-128, 3DES, IDEA are used for symmetric encryption and SHA-1 is used for hashing purposes. PGP software is an open source one and is not dependent on either of the OS (Operating System) or the processor. The application is based on a few commands which are very easy to use.
The following are the services offered by PGP:

1. Authentication
2. Confidentiality
3. Compression
4. Email Compatibility
5. Segmentation

1.Authentication:

Authentication basically means something that is used to validate something as true or real. To login into some sites sometimes we give our account name and password, that is an authentication verification procedure.
In the email world, checking the authenticity of an email is nothing but tocheck *whether it actually came from the person it says*. In emails, authentication has to be checked as there are some people who spoof the emailsor some spams and sometimes it can cause a lot of inconvenience. The Authentication service in PGP is provided as follows:



As shown in the above figure, the Hash Function (H) calculates the Hash Valueof the message. For the hashing purpose, SHA-1 is used and it produces a 160 bit output hash value. Then, using the sender's private key (KP_a), it is encryptedand it's called as Digital Signature. The Message is then appended to thesignature. All the process happened till now, is sometimes described as *signing the message* . Then the message is compressed to reduce the transmission overhead and is sent over to the receiver.

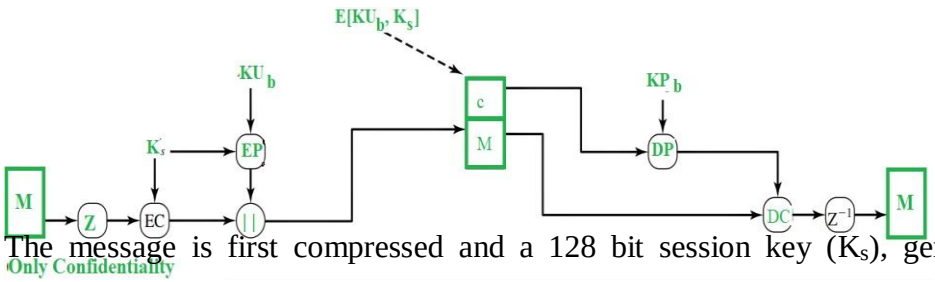
At the receiver's end, the data is decompressed and the message, signature are obtained. The signature is then decrypted using the sender's public key(PU_a) and the hash value is obtained. The message is again passed to hash function and it's hash value is calculated and obtained.

Both the values, one from signature and another from the recent output of hash function are compared and if both are same, it means that the email is actually sent from a known one and is legit, else it means that it's not a legit one.

2.Confidentiality:

Sometimes we see some packages labelled as 'Confidential', which means that those packages are not meant for all the people and only selected persons can see them. The same applies to the email confidentiality as well. Here, in the email service, only the sender and the receiver should be able to read the message, that means the contents have to be kept secret from every other person, except for those two.

PGP provides that Confidentiality service in the following manner:



The message is first compressed and a 128 bit session key (K_s), generated bythe PGP, is used to encrypt the message through symmetric encryption. Then, the session key (K_s) itself gets encrypted through public key encryption (EP) using receiver's public key(KU_b) . Both the encrypted entities are now concatenated and sent to the receiver.

As you can see, the original message was compressed and then encrypted initially and hence even if any one could get hold of the traffic, he cannot read the contents as they are not in readable form and they can only read them ifthey had the session key (K_s). Even though session key is transmitted to the receiver and hence, is in the traffic, it is in encrypted form and only the

receiver's private key (KP_b)can be used to decrypt that and thus our message would be completely safe.

At the receiver's end, the encrypted session key is decrypted using receiver's private key (KP_b) and the message is decrypted with the obtained session key. Then, the message is decompressed to obtain the original message (M).

RSA algorithm is used for the public-key encryption and for the symmetric jey encryption, CAST-128(or IDEA or 3DES) is used.

➢ Secure/Multipurpose Internet Mail Extension(S/MIME):

S/MIME is a security-enhanced version of Multipurpose Internet Mail Extension (MIME). In this, public key cryptography is used for digital sign, encrypt or decrypt the email. User acquires a public-private key pair with a trusted authority and then makes appropriate use of those keys with email applications.

Difference between PGP and S/MIME

S.NO	PGP	S/MIME
1.	It is designed for processing the plain texts	While it is designed to process email as well as many multimedia files.
2.	PGP is less costly as compared to S/MIME.	While S/MIME is comparatively expensive.
3.	PGP is good for personal as well as office use.	While it is good for industrial use.
4.	PGP is less efficient than S/MIME.	While it is more efficient than PGP.
5.	It depends on user key exchange.	Whereas it relies on a hierarchically valid certificate for key exchange.

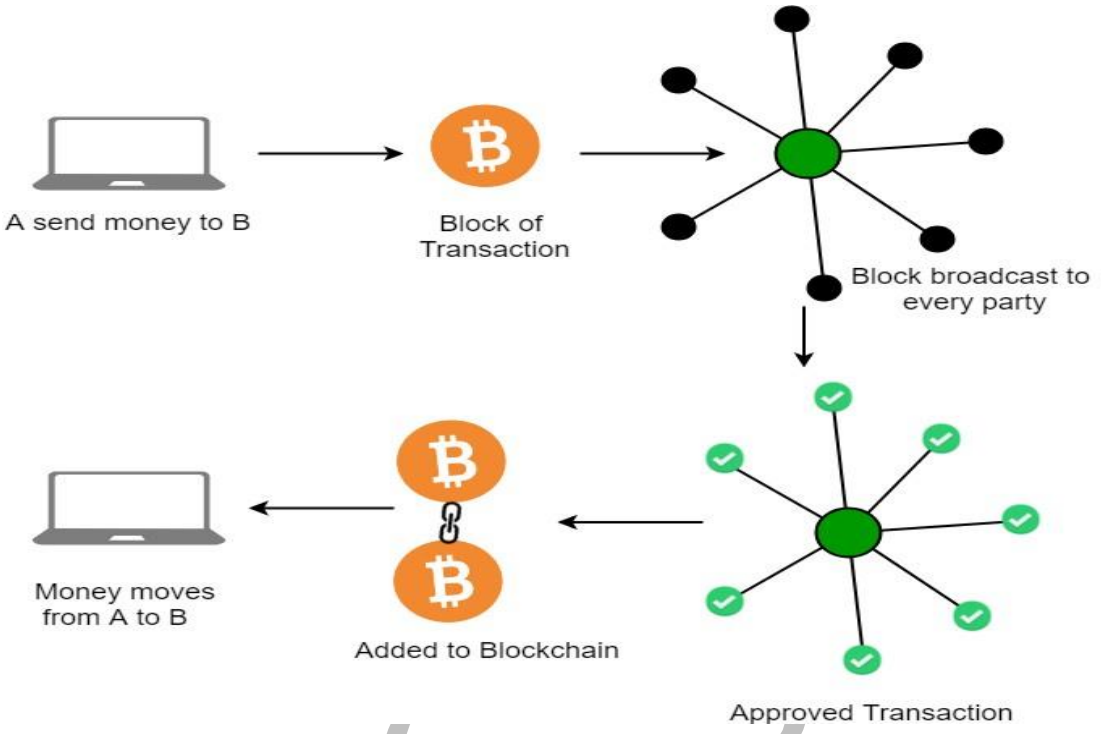
S.NO	PGP	S/MIME
6.	PGP is comparatively less convenient.	While it is more convenient than PGP due to the secure transformation of all the applications.
7.	PGP contains 4096 public keys.	While it contains only 1024 public keys.
8.	PGP is the standard for strong encryption.	While it is also the standard for strong encryption but has some drawbacks.
9.	PGP is also be used in VPNs.	While it is not used in VPNs, it is only used in email services.
10.	PGP uses Diffie hellman digital signature.	While it uses Elgamal digital signature.

Introduction to Blockchain technology

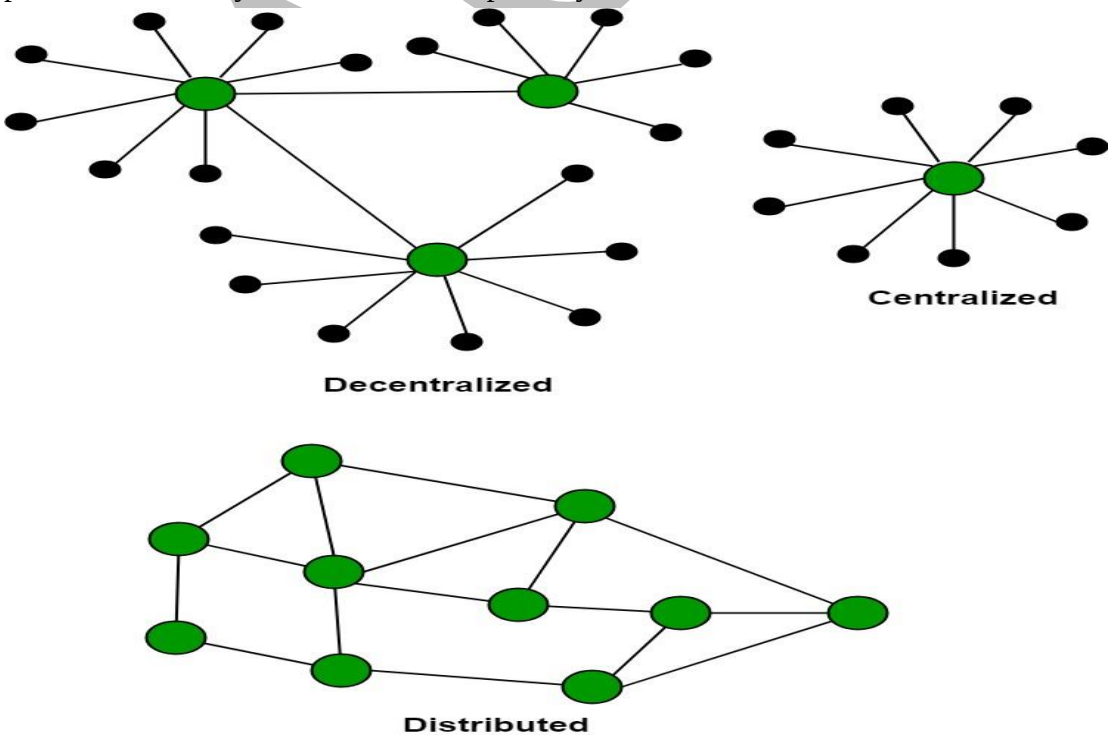
Blockchain is the backbone Technology of Digital Cryptocurrency Bitcoin. The blockchain is a distributed database of records of all transactions or digital eventthat have been executed and shared among participating parties. Each transactionverified by the majority of participants of the system. It contains every single record of each transaction. Bitcoin is the most popular cryptocurrency an example of the blockchain. Blockchain Technology first came to light when a person or Group of individuals name 'Satoshi Nakamoto' published a white paper on "*Bitcoin: A peer to peer electronic cash system*" in 2008. Blockchain Technology Records Transaction in Digital Ledger which is distributed over theNetwork thus making it incorruptible. Anything of value like Land Assets, Cars,etc. can be recorded on Blockchain as a Transaction.

How Blockchain Technology works?
One of the famous uses of Blockchain is Bitcoin. The bitcoin is a cryptocurrency

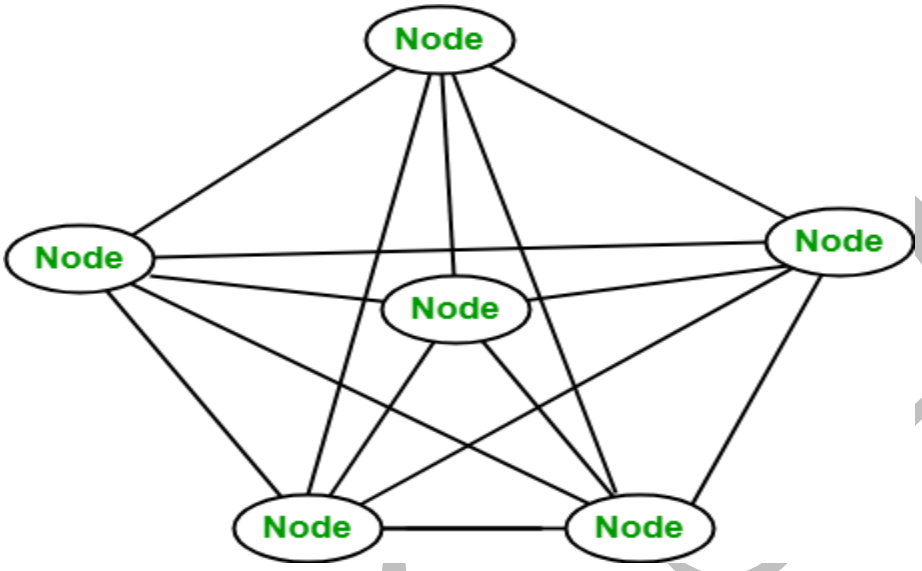
and is used to exchange digital assets online. Bitcoin uses cryptographic proof instead of third-party trust for two parties to execute transactions over the internet. Each transaction protects through digital signature.



Distributed Database: There is no Central Server or System which keeps the dataof Blockchain. The data is distributed over Millions of Computers around the world which are connected with the Blockchain. This system allows Notarizationof Data as it is present on every Node and is publicly verifiable.



A network of nodes: A node is a computer connected to the Blockchain Network.Node gets connected with Blockchain using the client. Client helps in validating and propagates transaction on to the Blockchain. When a computer connects to the Blockchain, a copy of the Blockchain data gets downloaded into the systemand the node comes in sync with the latest block of data on Blockchain. The Node connected to the Blockchain which helps in the execution of a Transactionin return for an incentive is called Miners.



Disadvantages of current transaction system:

- Cash can only be used in low amount transaction locally.
- Huge waiting time in the processing of transactions.
- Need to third party for verification and execution of Transaction make the process complex.
- If the Central Server like Banks is compromised, whole System is affected including the participants.
- Organization doing validation charge high process thus making the process expensive.

Building trust with Blockchain:
Blockchain enhances trust across a business network. It's not that you can't trustthose who you conduct business with its that you don't need to when operating on a Blockchain network.

Blockchain built trust through the following five attributes:

- Distributed: The distributed ledger is shared and updated with every incoming transaction among the nodes connected to the Blockchain. All this is done in real-time as there is no central server controlling the data.
- Secure: There is no unauthorized access to Blockchain made possible through Permissions and Cryptography.
- Transparent: Because every node or participant in Blockchain has a copyof the Blockchain data, they have access to all transaction data. They themselves can verify the identities without the need for mediators.
- Consensus-based: All relevant network participants must agree that a transaction is valid. This is achieved through the use of consensus algorithms.
- Flexible: Smart Contracts which are executed based on certain conditionscan be written into the platform. Blockchain Network can evolve in pacewith business processes.

Benefits of Blockchain Technology:

- Time-saving: No central Authority verification needed for settlements making the process faster and cheaper.
- Cost-saving: A Blockchain network reduces expenses in several ways. Noneed for third-party verification. Participants can share assets directly. Intermediaries are reduced. Transaction efforts are minimized as every participant has a copy of shared ledger.
- Tighter security: No one can temper with Blockchain Data as it shared among millions of Participants. The system is safe against cybercrimes and Fraud