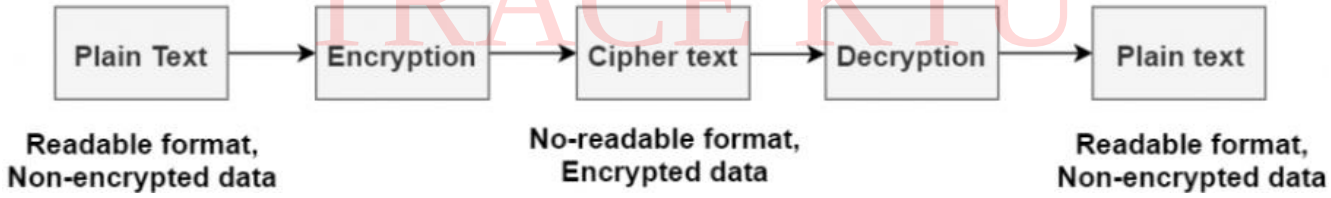
Module – 1

Fundamentals of Cryptography

Introduction to Cryptography, Symmetric cryptography – AES. Asymmetric cryptography – RSA. Elliptic curve cryptography, **Digital signatures** – RSA digital signature algorithms. **Secure Hash Algorithms** – SHA-256. **Applications of cryptographic hash functions** – Merkle trees, Distributed hash tables.

Introduction to Cryptography

- Cryptography is technique of securing information and communications through use of codes so that only those person for whom the information is intended can understand it and process it. Thus preventing unauthorized access to information.
- The prefix "crypt" means "hidden" and suffix graphy means "writing".
- In Cryptography the techniques which are used to protect information are obtained from mathematical concepts and a set of rule based calculations known as algorithms to convert messages in ways that make it hard to decode it.
- These algorithms are used for cryptographic key generation, digital signing, and verification to protect data privacy, web browsing on internet and to protect confidential transactions such as credit card and debit card transactions.



Techniques used For Cryptography:

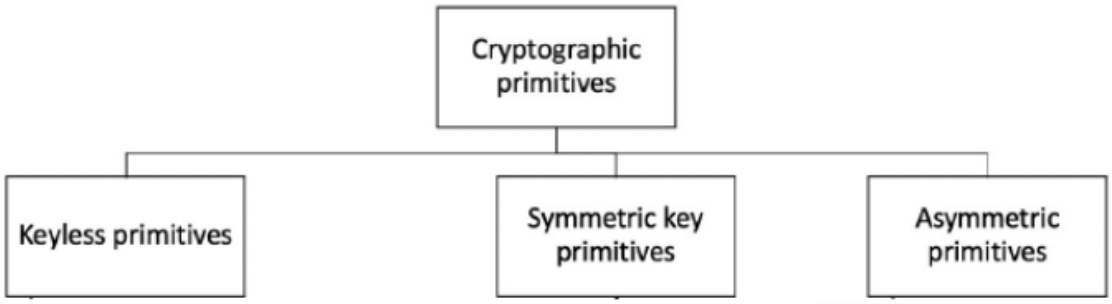
- Cryptography is often associated with the process where an ordinary plain text is converted to cipher text which is the text made such that intended receiver of the text can only decode it and hence this process is known as **encryption**.
- The process of conversion of cipher text to plain text this is known as **decryption.**

Features Of Cryptography:

1. **Confidentiality:** Information can only be accessed by the person for whom it is intended and no other person except him can access it.
2. **Integrity:** Information cannot be modified in storage or transition between sender and intended receiver without any addition to information being detected.
3. **Non-repudiation:** The creator/sender of information cannot deny his intention to send information at later stage.

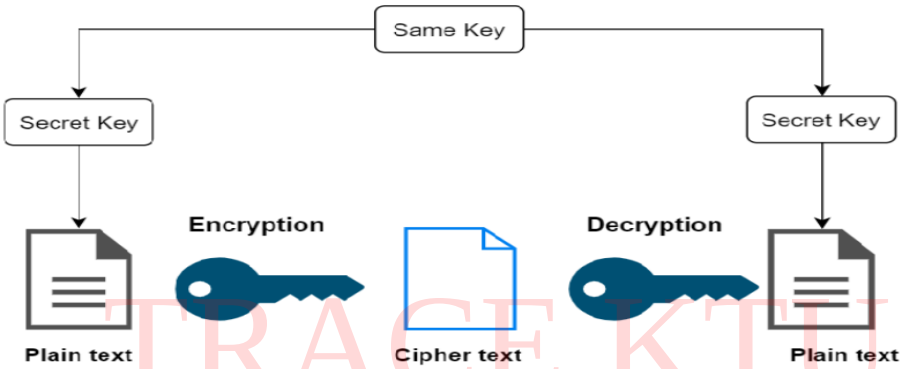
4. **Authentication:** The identities of sender and receiver are confirmed. As well as destination/origin of information is confirmed. Authentication provides assurance about the identity of an entity or the validity of a message. *There are two types of authentication mechanisms- namely, Entity authentication and data origin authentication.*
    a. **Entity authentication** is the assurance that an entity is currently involved and active in a communication session. Traditionally, users are issued a username and password that is used to gain access to the various platforms with which they are working. This practice is known as Single-factor authentication.
        - This type of authentication is not very secure for a variety of reasons, for example, password leakage; therefore, additional factors are now commonly used to provide better security. The use of additional techniques for user identification is known as multi-factor authentication (or two-factor authentication if only two methods are used).
    b. **Data origin authentication:** Also known as message authentication, data origin authentication is an assurance that the source of the information is indeed verified. Various methods, such as Message Authentication Codes (MACs) and digital signatures, are most commonly used.
    c. **Non-repudiation:** Non-repudiation is the assurance that an entity cannot deny a previous commitment or action by providing incontrovertible evidence. This is a security service that offers definitive proof that a particular activity has occurred.
    d. **Accountability:** Accountability is the assurance that actions affecting security can be traced back to the responsible party. This is usually provided by logging and audit mechanisms in systems where a detailed audit is required.

Types Of Cryptography:



1. Symmetric Key Cryptography

- ✓ This is also termed as Private or Secret key cryptography.
- ✓ Here, both the information receiver and the sender make use of a single key to encrypt and decrypt the message.
- ✓ The frequent kind of cryptography used in this method is AES (Advanced Encryption System).
- ✓ The approaches implemented through this type are completely streamlined and quicker too. Few types of Symmetric key cryptography are
    - Block cipher
    - DES (Data Encryption System)
    - RC2
    - IDEA
    - Blowfish
    - Stream cipher



- ✓ Keys can also be ephemeral (temporary) or static. Ephemeral keys are intended to be used only for a short period of time, such as in a single session between the participants, whereas static keys are intended for long-term usage.
- ✓ Another type of key is called the master key, which is used for the protection, encryption, decryption, and generation of other keys.
- ✓ Examples: Data Encryption Standard (DES), Advanced Encryption Standard (AES)

1.1. Advanced Encryption Standard (AES)

- ✓ Symmetric encryption algorithm.
- ✓ Invented by cryptographers Joan Daemen and Vincent Rijmen.
- ✓ So far, no attack has been found against AES that is more effective than the brute-force method.

How AES works?

- ✓ The encryption algorithm takes two inputs – the plaintext and key.
- ✓ AES takes the plaintext as blocks of size 128 bits (or 16 Bytes), and the key sizes of 128 bits, 192 bits or 256 bits. This 16 bytes plaintext is processed as a 4 x 4 array of bytes, known as the state.
- ✓ The state is then modified using multiple rounds.
- ✓ Full encryption requires 10 to 14 rounds, depending on the size of the key.
- ✓ The following table shows the key sizes and the required number of rounds.

Key size	Number of rounds required
128-bit	10 rounds
192-bit	12 rounds
256-bit	14 rounds

- Once the state is initiated, the following four operations are performed.
    1. **AddRoundKey:** In this step, the state array is XORed with a sub key (also known as the Round Key), which is derived from the master key.
    2. **Substitute Bytes (or Sub Bytes):** This is the substitution step where a lookup table (S-box) is used to replace all bytes of the state array.
    3. **Shift Rows:** This step is used to shift each row to the left, except for the first one, in the state array in a cyclic and incremental manner. That is, the first row of state is not altered. For the second row, a 1-byte circular left shift is performed. For the third row, a 2-byte circular left shift is performed. For the fourth row, a 3-byte circular left shift is performed.
    4. **Mix Columns:** Finally, all bytes are mixed in a linear fashion (linear transformation), column-wise
- This is one round of AES.
- In the final round (either the 10th , 12th , or 14th round, depending on the key size), stage 4 is replaced with **AddRoundKey**.

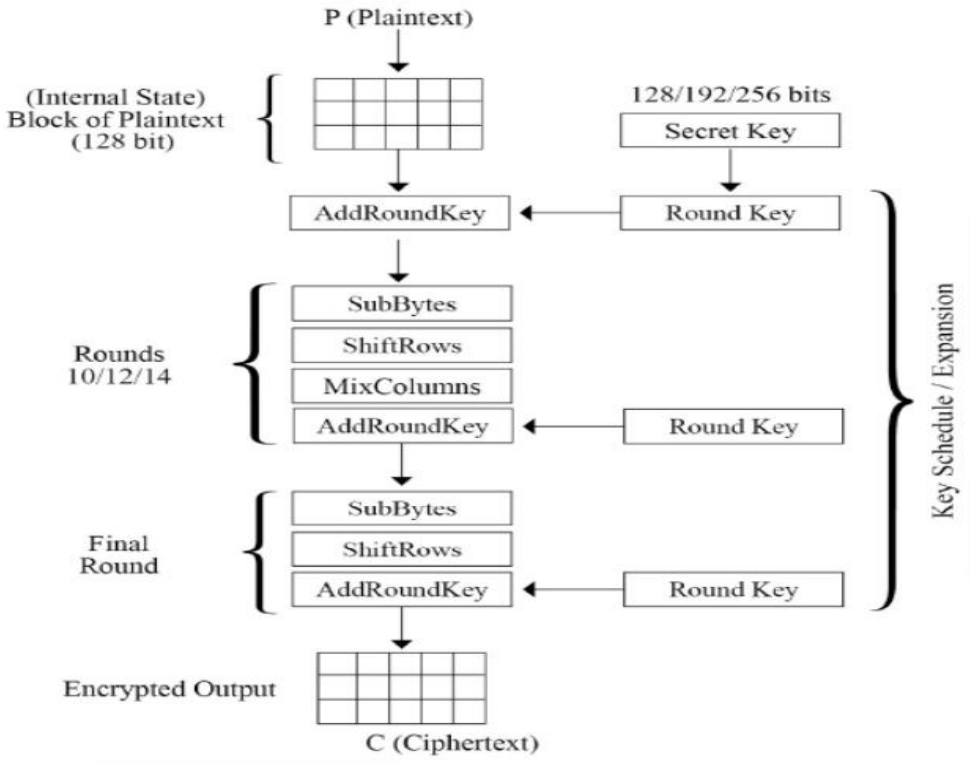


Fig: AES block diagram, showing the first round of AES encryption.

Transposition Cipher

1	2	3	4	5	6
M	E	E	T	M	E
A	F	T	E	R	P
A	R	T	Y		

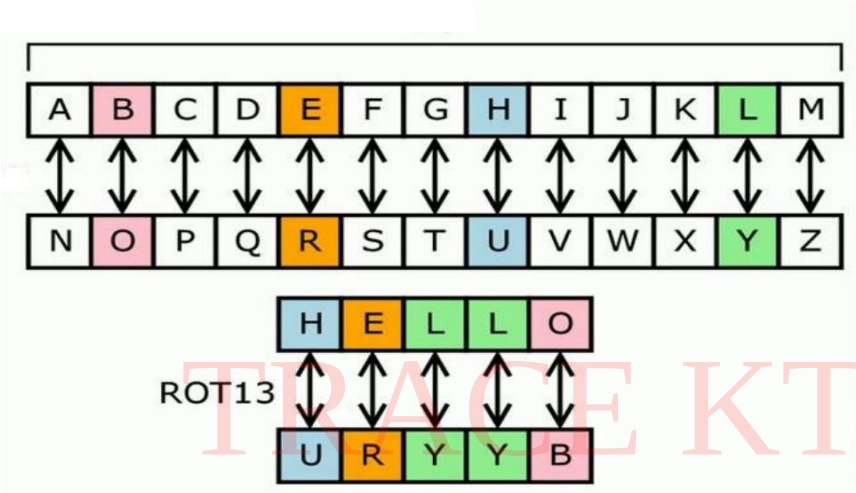
4	2	1	6	3	5
T	E	M	E	E	M
E	F	A	P	T	R
Y	R	A		T	

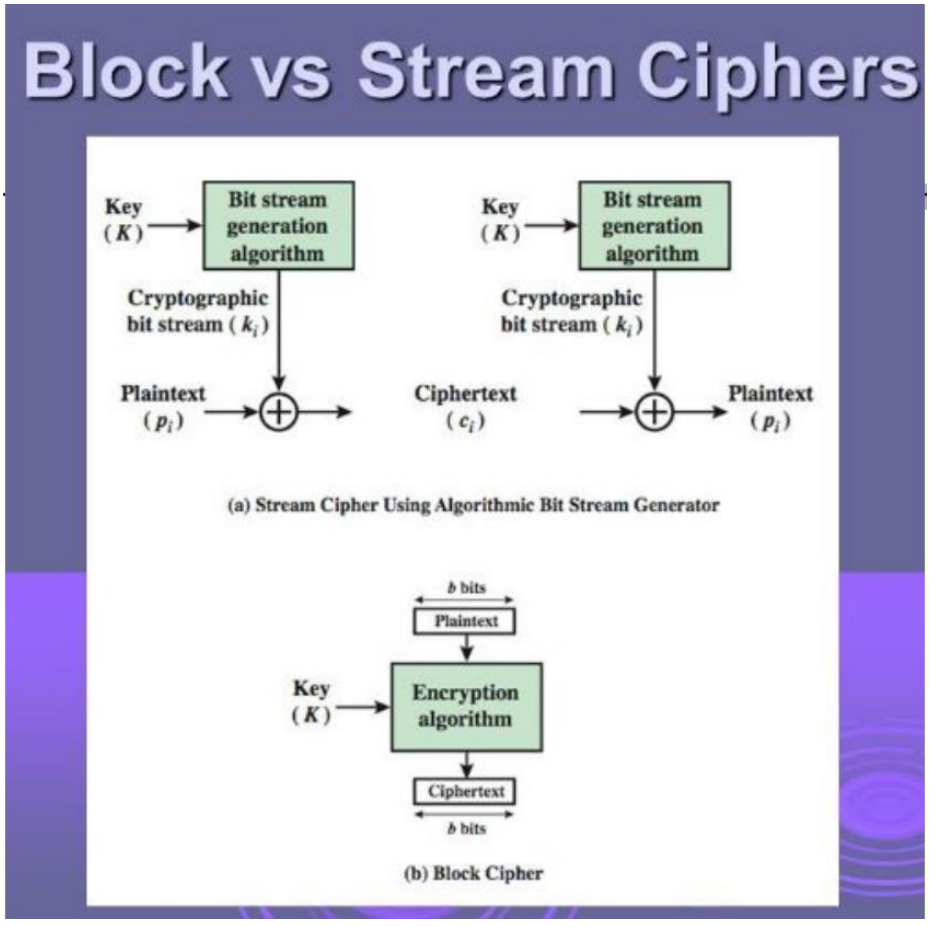
Plain Text: MEET ME AFTER PARTY

Key Used: 421635

Cipher Text: TEMEEMEFAPTRYRAT

Substitution Cipher





- also known as Rijndael algorithm
- symmetric block cipher algorithm
- block/chunk size of 128 bits
- converts these individual blocks using keys of 128, 192, and 256 bits
- Once it encrypts these blocks, it joins them together to form the cipher text
- based on a substitution-permutation network, known as SP network

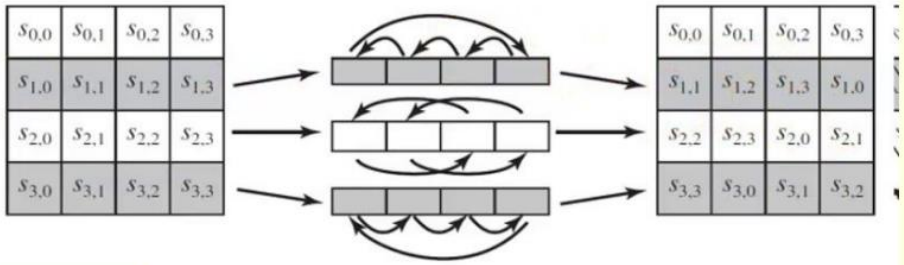
- Sub Key Generation
  - uses 128 bit Master Key
  - Key is processed in words of size 32 bit (4 words / 16 bytes)
  - Each sub key size is 32 bit / 1 word/4 bytes
  - Each round have 4 sub keys (128 bit/4 words/16 bytes)
  - For pre round calculation we use 4 sub key initially
  - Total sub key is 44

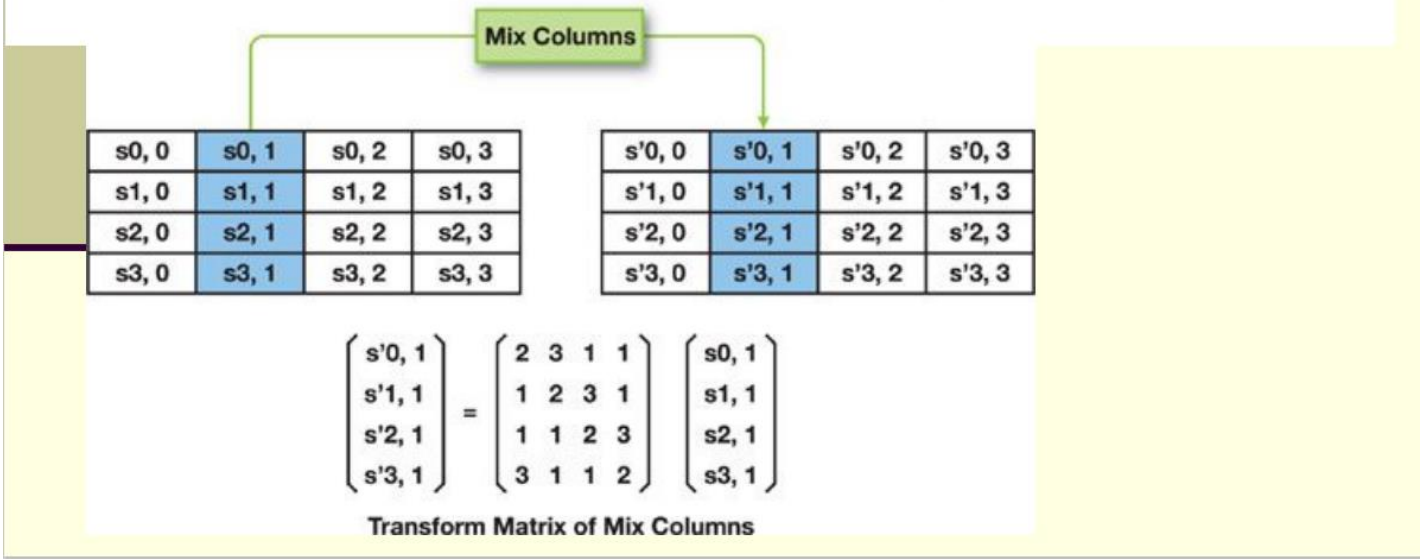
AES (Advanced Encryption Standard)

2. Shift Row transformation

- The shift row transformation is called ShiftRows.

- Rules of shifting rows,
  - Row 1 → No Shifting
  - Row 2 → 1 byte left shift
  - Row 3 → 2 byte left shift
  - Row 4 → 3 byte left shift





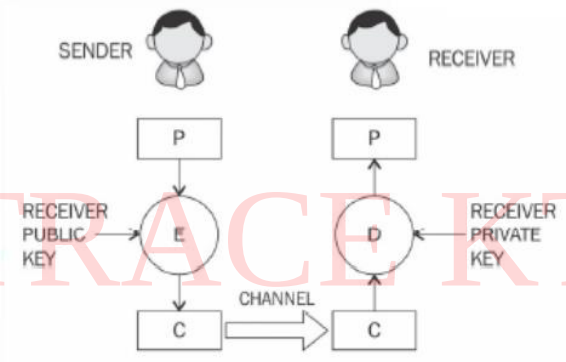
Transform Matrix of Mix Columns

Asymmetric Cryptography

- ✓ Asymmetric cryptographyrefers to a type of cryptography where the key that is used to encrypt the data is different from the key that is used to decrypt the data.
- ✓ It uses both public and private keys to encrypt and decrypt data, respectively.
- ✓ That is why asymmetric cryptography is also known as public key cryptography.
- ✓ Examples of asymmetric cryptography are RSA (named after its founders, **R**ivest, **S**hamir, and **A**delman), DSA (Digital Signature Algorithm), and **ElGamal**encryption.

Public and Private keys:

- ✓ A **private key**, as the name suggests, is a randomly generated number that is kept secret and held privately by its users.
- ✓ Private keys need to be protected and no unauthorized access should be granted to that key.
- ✓ Private keys can be of various lengths, depending on the type and class of algorithms used.
- ✓ A **public key** is freely available and published by the private key owner. An overview of public-key cryptography is shown in the following diagram:



- ✓ The sender encrypts data **P** using the recipient's public key and encryption function **E**, and produces an output encrypted data **C**, which is then transmitted over the network to the receiver.
- ✓ Once it reaches the receiver, it can be decrypted using the receiver's private using the decryption function **D**, which will output plaintext **P**.
- ✓ This way, the private key remains on the receiver's side, and there is no need to share keys in order to perform
- ✓ encryption and decryption, which is the case with symmetric encryption. The following diagram shows how the receiver uses public key cryptography to verify the integrity of the received message.

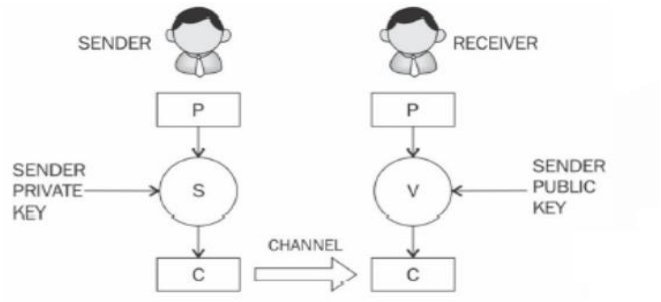


Fig: Signing and verification using public-key cryptography

In this model, the sender signs the data using their private key and transmits the message across to the receiver. Once the message is received, it is verified for integrity by the sender's public key.

- ✓ It's worth noting that there is no encryption being performed in this model.
- ✓ The preceding diagram shows that the sender digitally signs the plaintext **P** with their private key using signing function **S**, and produces data **C,** which is sent to the receiver, who verifies **C** using the sender's public key and function **V** to ensure the message has indeed come from the sender.
- ✓ Public key algorithms are slower in terms of computation than symmetric key algorithms. Therefore, they are not commonly used in the encryption of large files or the actual data that requires encryption. They are usually used to exchange keys for symmetric algorithms. Once the keys are established securely, symmetric key algorithms can be used to encrypt the data.
- ✓ Security mechanisms offered by public key cryptosystems include key establishment, digital signatures, identification, encryption, and decryption.
- ✓ Public key cryptography algorithms are based on various underlying mathematical functions. The three main categories of asymmetric algorithms are:
    - ❖ Integer factorization
    - ❖ Discrete logarithm
    - ❖ Elliptic curves
- ✓ **Integer factorization** schemes are based on the fact that large integers are very hard to factor. RSA is a prime example of this type of algorithm.
- ✓ **A discrete logarithm** scheme is based on a problem in modular arithmetic. It is easy to calculate the result of a modulo function, but it is computationally impractical to find the exponent of the generator. In other words, it is extremely difficult to find the input from the result (output).This is called a one-way function.
- ✓ **The elliptic curves algorithm** is based on the discrete logarithm problem discussed previously, but in the context of elliptic curves. An elliptic curve is an algebraic cubic curve over a field, which can be defined by an equation, as shown below. The curve is non-singular, which means that it has no cusps or self-intersections. It has two variables **a** and **b**, as well as a point of infinity:

$$y^2 = x^3 + ax + b$$

- ✓ Here, **a** and **b** are integers whose values are elements of the field on which the elliptic curve is defined.

RSA (Rivest, AdiShamir and Leonard Adelman)

RSA was invented in 1977 by Ron Rivest, AdiShamir, and Leonard Adelman, hence the name RSA. This type of public key cryptography is based on the integer factorization problem, where the multiplication of two large prime numbers is easy, but it is difficult to factor the product (the result of the multiplication) back to the two original numbers. The crux of the work involved with the RSA algorithm happens during the key

Generation process.

An RSA key pair is generated by performing the following steps:

1. Modulus generation:
   - Select **p** and **q**, which are very large prime numbers
   - Multiply **p** and **q**, **n=p.q** to generate modulus **n**
2. Generate the co-prime:
   - Assume a number called **e**.
   - **e** should satisfy a certain condition; that is, it should be greater than 1 and less than **(p-1) (q-1).** In other words, **e** must be a number such that no number other than 1 can be divided into **e** and (p-1) (q-1). This is called a **co-prime**, that is, **e** is the co-prime of **(p-1)(q-1).**
3. Generate the public key:
   - The modulus generated in step 1 and co-prime generated in **step2** as a pair is the public key, i.e., **(n, e)**. This part is the public part that can be shared with anyone; however, **p** and **q** need to be kept secret.
4. Generate the private key:
   - The private key,d is calculated from **p, q**, and **e**. The private key is basically the inverse of **e** modulo **(p-1)(q-1).**

   As an equation, it is as follows:

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

Now, let's see how encryption and decryption operations are performed using RSA.

- ✓ **Encryption in RSA** is provided using the following equation:

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

- ✓ **Decryption in RSA** is provided using the following equation:

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

- ✓ This means that the receiver who has a public key pair (n, e) can decipher the data using their private key d.

Symmetric vs. Asymmetric Cryptograph

Symmetric Cryptography	Asymmetric Cryptography
It only requires a single key for both encryption and decryption.	It requires two keys, a public key and a private key, one to encrypt and the other one to decrypt.
The size of cipher text is the same or smaller than the original plain text.	The size of cipher text is the same or larger than the original plain text.
It is used when a large amount of data is required to transfer.	It is used to transfer small amounts of data.
It only provides confidentiality	It provides confidentiality, authenticity, and non-repudiation.
The encryption process is very fast.	The encryption process is slow.
The Mathematical Representation is as follows **P = D (K, E(P))** where K –> encryption and decryption key P –> plain text D –> Decryption E(P) –> Encryption of plain text	The Mathematical Representation is as Follows **P = D(Kd, E (Ke,P))** where Ke –> encryption key Kd –> decryption key D –> Decryption E(Ke, P) –> Encryption of plain text using encryption key Ke . P –> plain text
Examples - AES, DES	Examples - RSA, DSA

Elliptic curve cryptography

Asymmetric Public key cryptosystem

- Provides security with smaller key size
- alternative to the RSA algorithm
- used for digital signatures in cryptocurrencies, such as Bitcoin and Ethereum, as well as one-way encryption of emails, data and software
- fast key generation, fast key agreement and fast signature

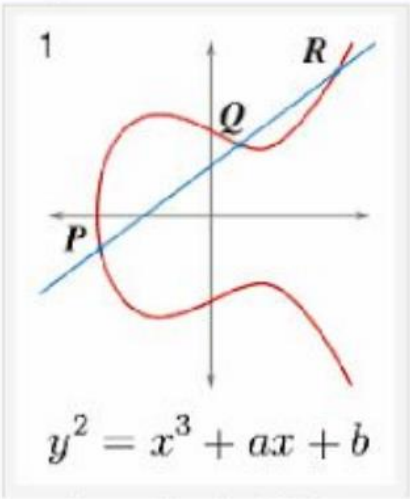


Fig. 2 shows simple elliptic curve.

2 families of Elliptic curves

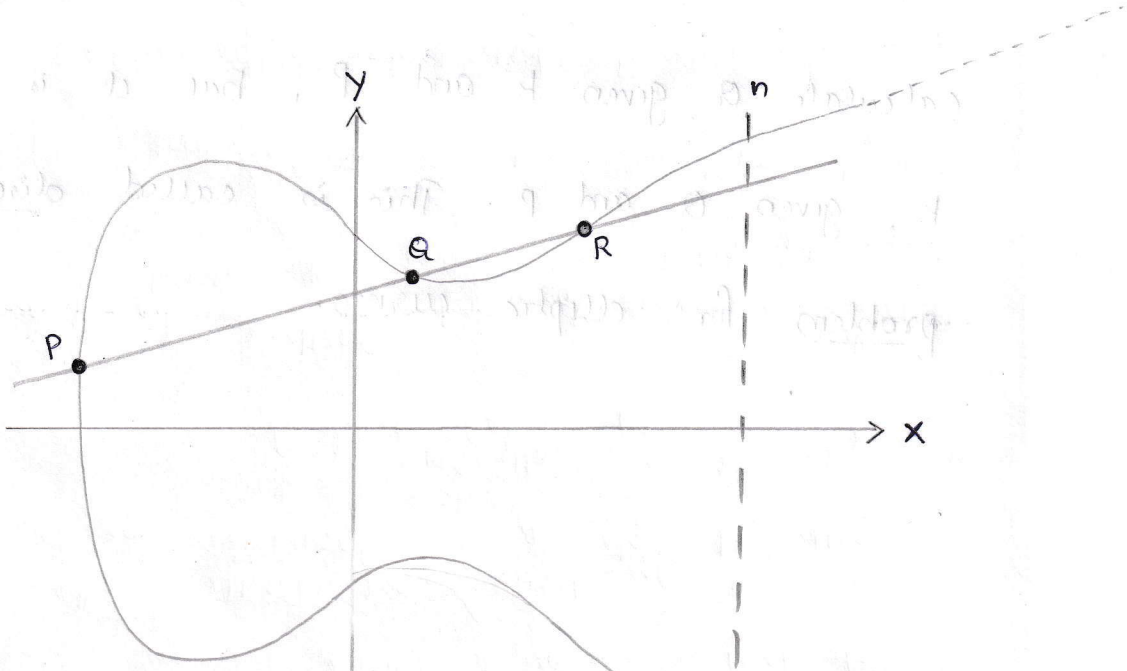
- Prime curves over Z_p
  - uses cubic equation in which variables and coefficients from 0 through p-1
  - Best for software applications
- Binary curves over GF(2^m)
  - Variables and coefficients in GF (2^m)
  - best for hardware applications

Elliptic - Curve Cryptography :- (ECC)

advantage over RSA:

- The principle attraction of ECC, compared to RSA, is that it appears to offer equal security for a far smaller key size, thereby reducing processing overhead.
- ECC is an assymetric key encryption scheme.
- It makes use of elliptic curves.
- An elliptic curve is defined by an equation in two variables. (Note that elliptic curves are not ellipses).
- Elliptic curves are defined by some mathematical function (by an equation of degree 3. ie cubic fn)

$$y^2 = x^3 + ax + b$$



→ curve is symmetric to x-axis

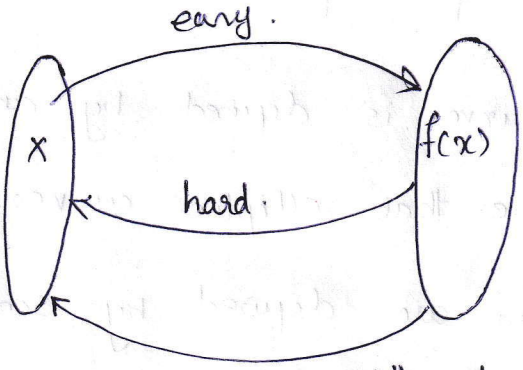
- If we draw a line, it will touch a maximum of 3 points.

We are limiting the value till n.

•) what is a trap-door function ?

- A trap-door function is a fn that is easy to compute in one direction, yet difficult to compute in the opposite direction (ie difficult to find the inverse) without some special information.

ie,



easy if given "t" → trapdoor value.

- Consider the equation $Q = kP$, where Q and P are points on the curve and $k < n$. It is relatively easy to calculate Q given k and P, but it is hard to determine k, given Q and P. This is called discrete logarithm problem for elliptic curves.

•) ECC Key Exchange :

① Global public elements : $[E_q(a,b) \quad \& \quad G]$

- first pick a large integer q, which is either a prime no. or an integer of the form 2^m.
- pick the elliptic curve parameters 'a' and 'b'
- Thus our elliptic curve will be,

$$E_q(a,b)$$

- Next pick a base point $G = (x_1, y_1)$ in $E_q(a,b)$, whose order is a very large value 'n'.
- The order 'n' of a point G on an elliptic curve is the smallest positive integer n such that $nG = 0$,

② User A key generation :

- Select private key(n_A). ($n_A < n$)
- calculate the public key (P_A)

$$\boxed{P_A = n_A \times G}$$

③ User B key generation :

- Select private key n_B , $n_B < n$
- calculate public key P_B, $\boxed{P_B = n_B \times G}$

④ Calculation of secret key by user A.

$$K = n_A \times P_B$$

⑤ calculation of secret key by user B.

$$K = n_B \times P_A$$

Thus the secret key, K is shared among both the users A and B.

•) ECC encryption / decryption :-

Encryption:

- let our plain text message be 'm'.
- encode the msg 'm' as an x-y point P_m on the elliptic curve.
- Now the point P_m is encrypted into a cipher text point C_m.
- To encrypt the message point P_m, user A chooses a random positive integer k.
- Thus the cipher point will be,

$$C_m = (kG \ , \ P_m + k \cdot P_B)$$

- This point, C_m, will be sent to the user B (receiver)

For decryption:

- for decryption, B multiplies the first point in the pair by B's secret key and subtracts the result from the second point.

ie our $C_m = \{ k.G_1 \bullet, P_m + k.P_B \}$

$k.G_1$ ⇓ first point

$P_m + k.P_B$ ⇓ second point.

- ie, first multiply the first point in C_m with the receiver's secret key (or private key).

ie $\boxed{kG_1 \times n_B}$

- then subtract this value from the 2nd point in C_m

ie $\boxed{P_m + k.P_B - (kG_1 \times n_B)}$

- But we know that,

$n_B \times G_1 = P_B$, ie the public key of the receiver.

$\therefore P_m + k.P_B - kG_1 \times n_B$

$= P_m + k.P_B - k.P_B$

$= P_m + 0 = P_m$ ⇒ plain text point. Thus the decryption is successfully done.

Digital Signatures

Digital signatures provide a means of associating a message with an entity from which the message has originated. Digital signatures are used to provide data origin authentication and non-repudiation.Digital signatures are used in blockchains, where transactions are digitally signed by senders using their private key, before the sender broadcasts the transaction to the network. This digital signing proves that the sender is the rightful owner of the asset; for example, bitcoins. These transactions are verified again by other nodes on the network to ensure that the funds indeed belong to the node (user) who claims to be the owner.

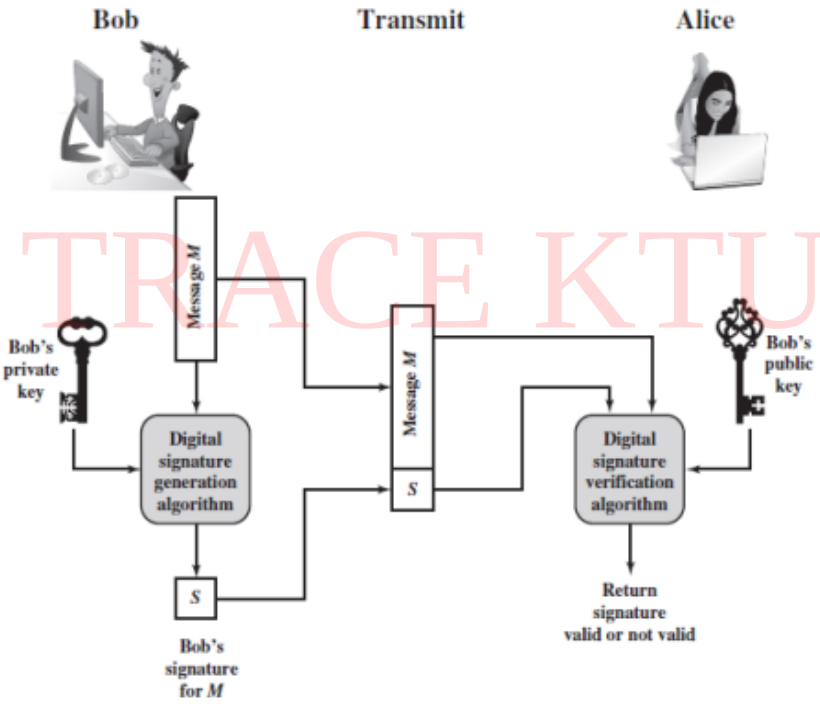


Fig: A generic model of digital signature

Digital Signature must have the following **properties**:

- **Authenticity** means that the digital signatures are verifiable by a receiving party.
- The **unforgeability** property ensures that only the sender of the message can use the signing functionality using the private key. Digital signatures must provide

protection against forgery.In other words, unforgeability means that no one else can produce the signed message produced by a legitimate sender. This is also called the property of **non-repudiation**.

- **Non-reusability** means that the digital signature cannot be separated from a message and used again for another message. In other words, the digital signature is firmly bound to the corresponding message and cannot be simply cut from its original message and attached to another.

Various schemes, such as RSA-, DSA-, and ECDSA-based digital signature schemes, are used in practice. RSA is the most commonly used; however, with the traction of ECC, ECDSA-based schemes are also becoming quite popular. This is beneficial in blockchains because ECC provides the same level of security that RSA does, but it uses less space. Also, the generation of keys is much faster in ECC compared to RSA, so it helps with the overall performance of the system.

RSA Digital Signature Algorithm

RSA-based digital signature algorithms are calculated using the two steps listed here. Fundamentally, the idea is to first compute the hash of the data and then sign it with the private key:

1. Calculate the hash value of the data packet. This will provide the data integrity guarantee, as the hash can be computed at the receiver's end again and matched with the original hash to check whether the data has been modified in transit. Technically, message signing can work without hashing the data first, but that is not considered secure.
2. Sign the hash value with the signer's private key. As only the signer has the private key, the authenticity of the signature and the signed data is ensured.

The operation of a generic digital signature function using RSA is shown in the following diagram:

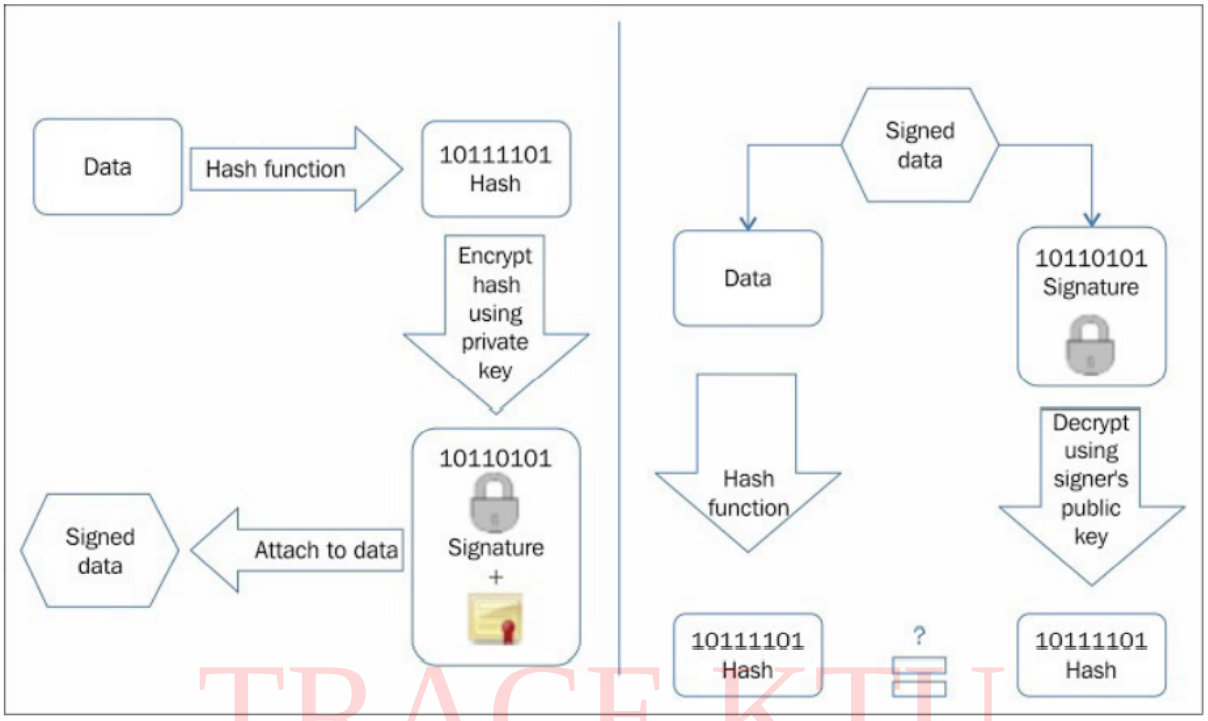


Fig: Digital signing (left) and verification process (right) (example of RSA digital signatures)

If a sender wants to send an authenticated message to a receiver, there are two methods that can be used: sign then encrypt and encrypt then sign. These two approaches of using digital signatures with encryption are as follows.

- **Sign then encrypt:**With this approach, the sender digitally signs the data using the private key, appends the signature to the data, and then encrypts the data and the digital signature using the receiver's public key. This is considered a more secure scheme compared to the 'encrypt then sign' scheme.

- **Encrypt then sign:**With this method, the sender encrypts the data using the receiver's public key and then digitally signs the encrypted data.

Elliptic Curve Digital Signature Algorithm (ECDSA)

In order to sign and verify using the ECDSA scheme, the first key pair needs to be generated:

1. First, define an elliptic curve E with the following:
    - Modulus q
    - Coefficients a and b
    - Generator point G that forms a cyclic group of prime order n
2. An integer n_A is chosen randomly so that $0 < n_A < n$.
3. Calculate public key P_A so that $P_A = n_A G$.
    - The public key is a six tuple in the form shown here:

      $$K_{pb} = (q,a,b,n,G, P_A)$$

    - The private key is a randomly chosen integer n_A in step 2:

      $$K_{pr} = n_A$$

      Now, the signature can be generated using the private and public key.
4. An ephemeral key K_e is chosen, where $0 < K_e < n$. It should be ensured that K_e is truly random and that no two signatures have the same key; otherwise, the private key can be calculated.
5. Another value R is calculated using $R = K_e G$; that is, by multiplying G (the generator point) and the random ephemeral key.
6. Initialize a variable r with the x coordinate value of point R so that r = xR.
7. The signature can be calculated as follows:

   $$S = (h(m) + n_A r) K_e\text{-}1 \bmod n$$

   Here, m is the message for which the signature is being computed, and h(m) is the hash of the message m.
8. Signature verification is carried out by following this process:
    - Auxiliary value w is calculated as, $w = s\text{-}1 \bmod n$
    - Auxiliary value $u1 = w.\ h(m) \bmod n$

- Auxiliary value $u2 = w.\ r\ mod\ n$
- Calculate point P, $P = u1G + u2P_A$

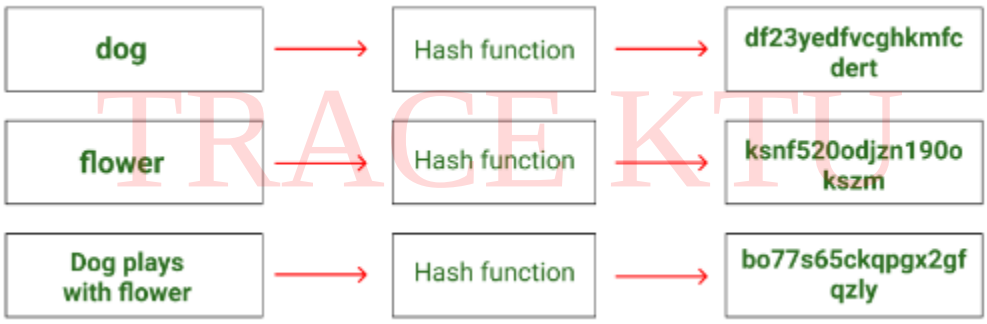
9. Verification is carried out as follows:
   - $r,\ s$ is accepted as a valid signature if the x-coordinate of point P has the same value as r mod n ; that is:

$$Xp = r\ mod\ n\ means\ valid\ signature$$

$$Xp\ != r\ mod\ n\ means\ invalid\ signature$$

Hash Functions

Hash functions are used to create fixed-length digests of arbitrarily long input strings. Hash functions are keyless cryptographic primitives, and they provide a data integrity service. Various families of hash functions are available, such as MD, SHA-1, SHA-2, SHA-3 and Whirlpool.



Hash functions are typically used to provide data integrity services. These can be used both as one-way functions and to construct other cryptographic primitives, such as MACs and digital signatures.

Different properties of hash functions are given below. There are two practical and three security properties for hash functions.

The **practical properties** are:

- **Compression of arbitrary messages into fixed-length digests**

  A hash function must be able to take an input text of any length and output a fixed-length compressed message. Hash functions produce a compressed output in various bit sizes, usually between 128-bits and 512-bits.

- **Easy to compute**

  Hash functions are efficient and fast one-way functions. It is required that hash functions should be very quick to compute regardless of the message size.

The **security properties** of hash functions are:

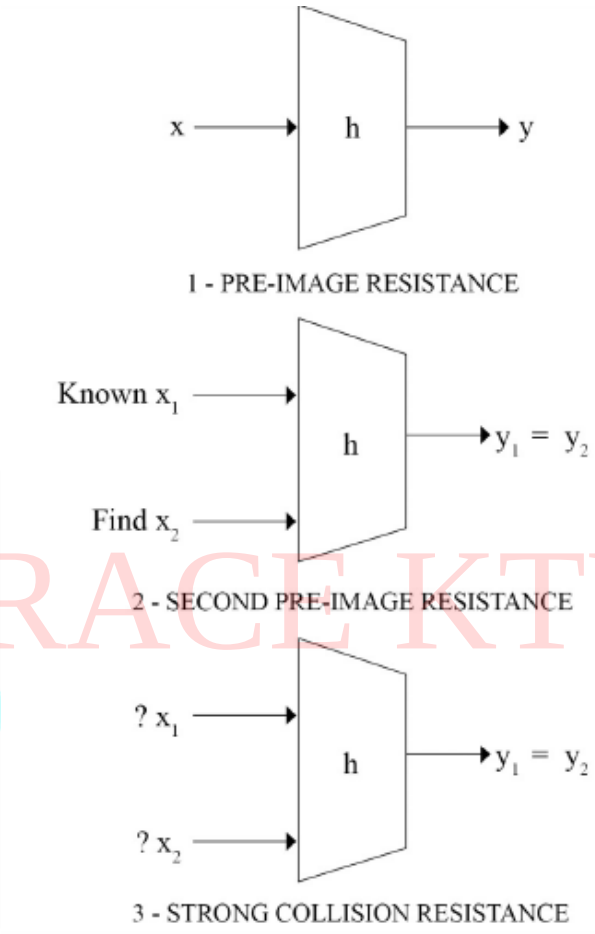


Fig: The three security properties of hash functions

- **Pre-image Resistance (One-way Property)**

  This property can be explained by using the simple equation:

  $$h(x) = y$$

  Here, h is the hash function, x is the input, and y is the hash. This property requires that y cannot be reverse-computed to x. x is considered a pre-image of y, hence the name pre-image resistance. This is also called a one-way property.

- **Second Pre-image Resistance (Weak Collision Resistance)**

  The second pre-image resistance property requires that given x and h(x), it is almost impossible to find any other message m, where m! = x and hash of m = hash of x or h(m) = h(x). This property is also known as weak collision resistance.

- **Collision Resistance (Strong Collision Resistance)**

  The collision resistance property requires that two different input messages should not hash to the same output. In other words, h(x) != h(z). This property is also known as strong collision resistance.

Avalanche Effect:

A concept known as the avalanche effect is desirable in all cryptographic hash functions. The avalanche effect specifies that a small change, even a single character change in the input text, will result in an entirely different hash output.

Secure Hash Algorithms (SHA - 256)

SHA-256 is included in the SHA-2 category. SHA-256 is used in Bitcoins.

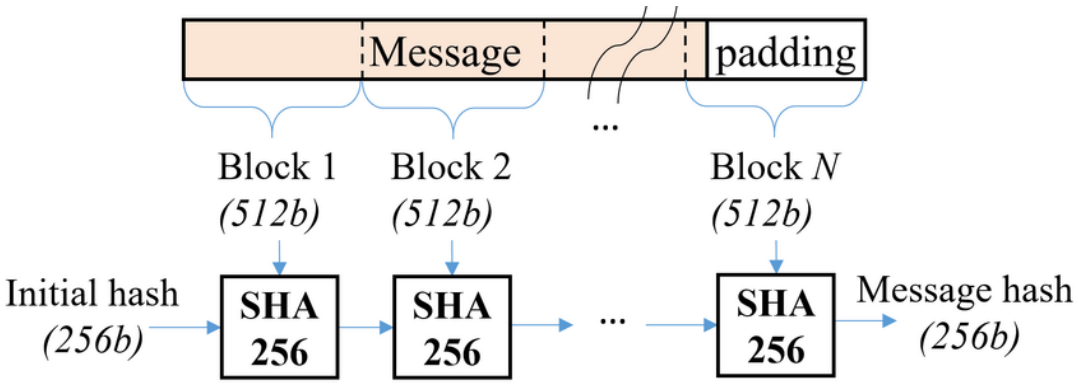
Design of SHA-256

SHA-256 has an input message size limit of 2^{64} - 1 bits. The block size is 512 bits, and it has a word size of 32 bits. The output is a 256-bit digest. The algorithm works as follows, in nine steps:

Pre-processing:

1. **Padding** of the message is used to adjust the length of a block to a multiple of 512 bits if it is smaller than the required block size.

2. **Parsing** the message into message blocks, which ensures that the message and its padding is divided into equal blocks of 512 bits



3. **Initializing the buffers.** Setting up the initial hash value, which consists of the eight 32-bit words obtained by taking the first 32 bits of the fractional parts of the square roots of the first eight prime numbers. These initial values are fixed and chosen to initialize the process.

H_0 = 6a09e667
H_1 = bb67ae85
H_2 = 3c6ef372
H_3 = a54ff53a
H_4 = 510e527f
H_5 = 9b05688c
H_6 = 1f83d9ab
H_7 = 5be0cd19

Hash Computing:

4. Each message block is then processed in a sequence, and it requires 64 rounds to compute the full hash output. Each round uses slightly different constants to ensure that no two rounds are the same.
5. The message schedule is prepared.
6. Eight working variables are initialized.
7. The compression function runs 64 times.
8. The intermediate hash value is calculated.

9. Finally, after repeating steps 5 through 8 until all blocks in the input message are processed, the output hash is produced by concatenating intermediate hash values.

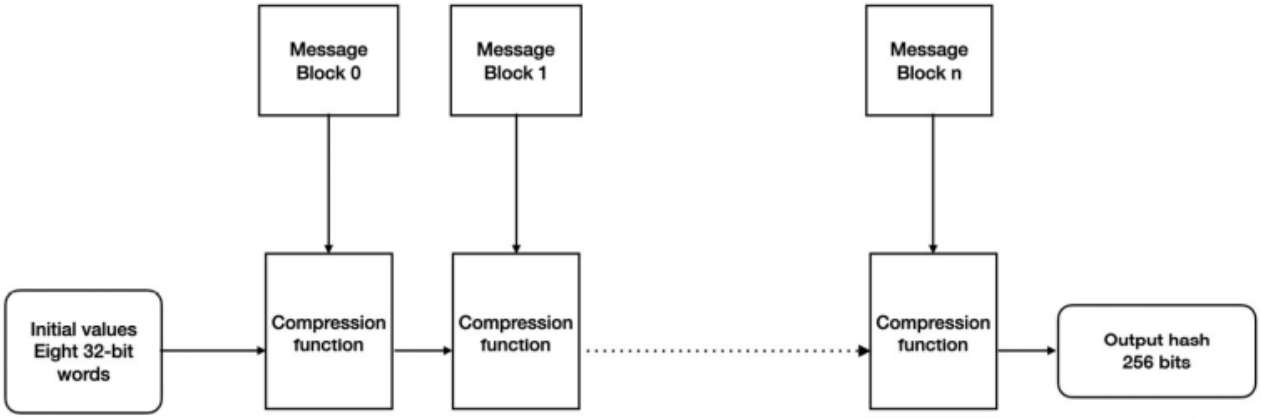


Fig: SHA-256 high level overview

As shown in the above diagram, SHA-256 takes the input message and divides it into equal blocks (chunks of data) of 512 bits. Initial values (or initial hash values) or the initialization vector are composed of eight 32 bit words (256 bits) that are fed into the compression function with the first message. Subsequent blocks are fed into the compression function until all blocks are processed and finally, the output hash is produced.

The compression function of SHA-256 is shown in the following diagram:

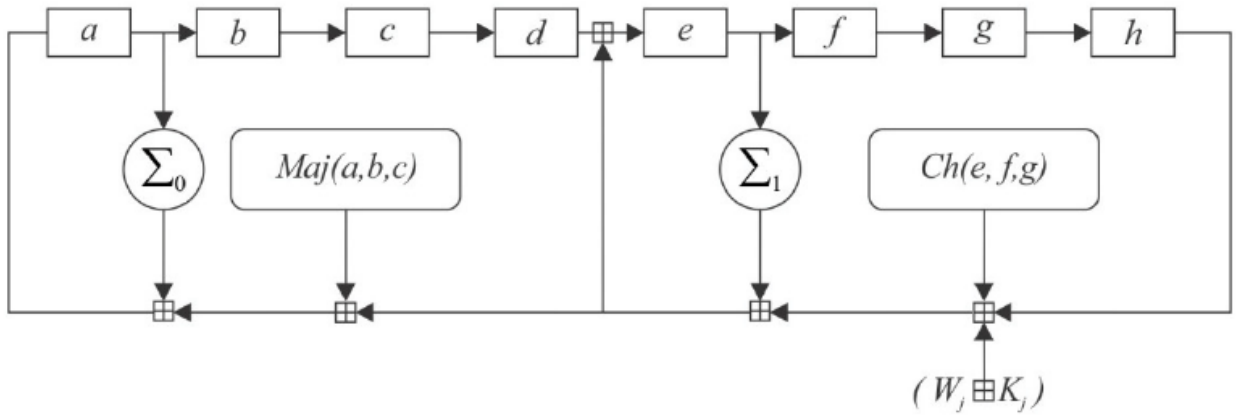


Fig: One round of an SHA-256 compression function

In the preceding diagram, a, b, c, d, e, f, g, and h are the registers for 8 working variables. Maj (majority) and Ch (choose) functions are applied bitwise.

$$Ch(x,y,z) = (x \wedge y) \oplus (x \wedge z)$$
$$Maj(x,y,z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$$

Σ_0 and Σ_1 perform bitwise rotation. The round constants are Wj and Kj, which are added in the main loop (compressor function) of the hash function.

Applications of cryptographic hash functions

- Hash functions are used to build Merkle trees, which are used to efficiently and securely verify large amounts of data in distributed systems.
- Hash functions are used in cryptographic puzzles such as the **Proof of Work (PoW)** mechanism in Bitcoin. Bitcoin's PoW makes use of the SHA-256 cryptographic hash function.
- The generation of addresses in blockchains. For example, in Ethereum, blockchain accounts are represented as addresses.
- Message digests (or hash values) in digital signatures.

Merkle Trees

- The concept of Merkle trees was introduced by Ralph Merkle.
- Merkle trees enable the secure and efficient verification of large datasets.
- A Merkle tree is a binary tree in which the inputs are first placed at the leaves (nodes with no children).
- Then the values of pairs of child nodes are hashed together to produce a value for the parent node (internal node), until a single hash value known as a **Merkle root** is achieved.

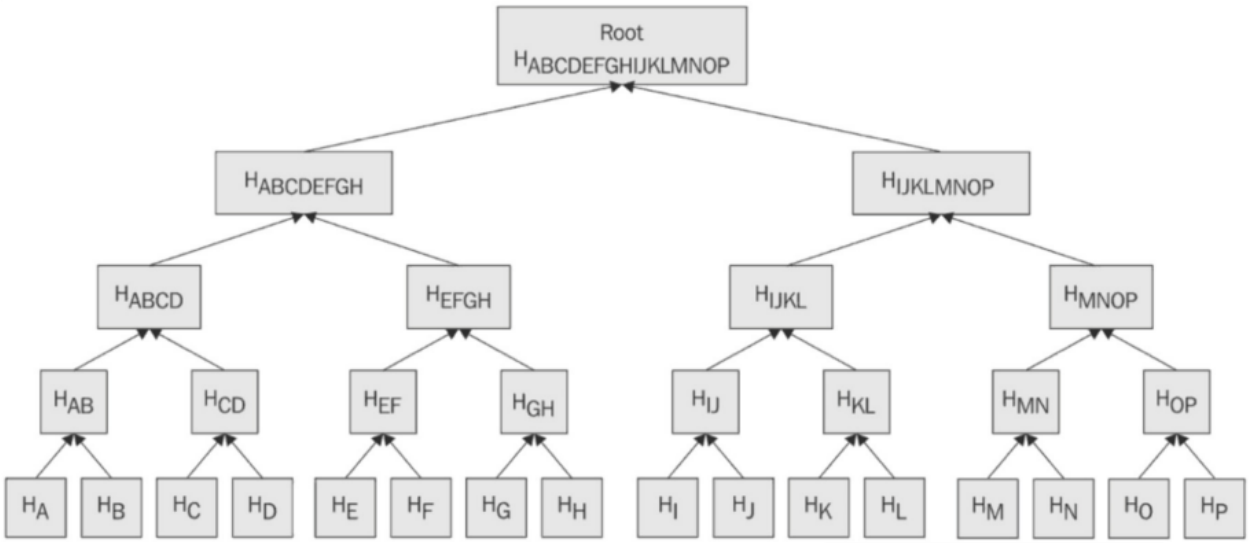


Fig: A Merkle Tree

- This structure helps to quickly verify the integrity of the entire tree (entire dataset), just by verifying the Merkle root on top of the Merkle tree, because if any change occurs in any of the hashes in the tree, the Merkle root will also change.
- Another advantage of Merkle trees is that there is no requirement of storing large amounts of data, only the hashes of the data, which are fixed-length digests of the large dataset need to be stored. Due to this property, the storage and management of Merkle trees is easy and efficient.
- Also, due to the fact that the tree is storage efficient, it is also bandwidth efficient over the network.

Distributed hash tables

A distributed hash table (DHT) is a type of distributed system that provides a lookup service similar to a hash table. In a hash table, data is stored and retrieved using keys, and the keys are used to determine the location of the data in the table. A distributed hash table is similar, but the data is distributed across multiple nodes in a network rather than being stored in a single table.

In a DHT, each node is responsible for storing and managing a portion of the data. When a client wants to retrieve or store data, it sends a request to the network. The request is then forwarded to the appropriate node based on the key of the data being requested. The node then responds to the request and either retrieves or stores the data. DHTs are used in a variety of applications, including peer-to-peer (P2P) networks, distributed databases, and distributed file systems.

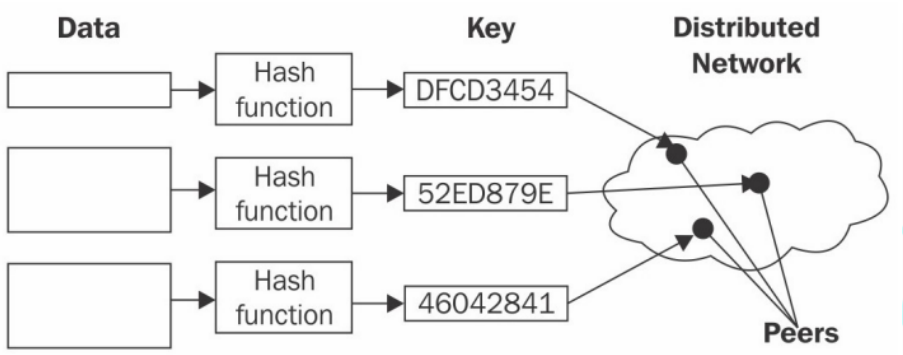


Fig: Distributed Hash Table

1. **What is cryptography? What is its role in Blockchain?**

   Blockchain uses cryptography to secure users' identities and ensure transactions are done safely with a hash function.

   Cryptography uses public and private keys in order to encrypt and decrypt data. In the Blockchain network, a public key can be shared with all the Bitcoin users but a private key (just like a password) is kept secret with the users.

   Blockchain uses SHA - 256 which is secure and provides a unique hash output for every input. The basic feature of this algorithm is whatever input you pass, it will give you a standard alphanumeric output of 64 characters. It is a one-way function from which you can derive an encrypted value from the input, but not vice-versa.

2. **What is a Genesis Block?**

- The genesis block is the first block in the Blockchain which is also known as block 0
- In Blockchain, it is the only block that doesn't refer to its previous block.
- It defines the parameters of the Blockchain such as,
  - level of difficulty,
  - consensus mechanism etc. to mine blocks

3. **How is the hash (Block signature) generated?**

The process of generating a block signature involves:

- Passing transaction details through a one-way hash function i.e., SHA-256.
- Running the output value through a signature algorithm (like ECDSA) with the user's private key.
- Following these steps, the encrypted hash, along with other information (such as the hashing algorithm), is called the digital signature.

4. **List down some of the extensively used cryptographic algorithms.**

Here are a few popular algorithms:

- SHA - 256
- RSA (Rivest-Shamir-Adleman)
- Triple DES
- Ethash
- **Blowfish**

5. **What is a smart contract and list some of its applications?**

- Smart contracts are self-executing contracts which contain the terms and conditions of an agreement between the peers
- Some of the applications are:
- Transportations: Shipment of goods can be easily tracked using smart contracts
- Protecting copyrighted content: Smart contracts can protect ownership rights such as music or books
- Insurance: Smart contracts can identify false claims and prevent forgeries
- Employment contract: Smart contracts can be helpful to facilitate wage payments

6. **What is a Dapp and how is it different from a normal application?**

Dapp:

- A Dapp is a decentralized application which is deployed using smart contract
- A Dapp has its back-end code (smart contract) which runs on a decentralized peer-to-peer network
- Process:
  - Front-end
  - Smart contract (backend code)
  - Blockchain (P2P contract)

Normal application:

- Normal application has a back-end code which runs on a centralized server
- It's a computer software application that is hosted on a central server
- Process:
  - Front-end
  - API
  - Database (runs on the server)

7. **What is the nonce and how is it used in mining?**

In Blockchain, mining is a process to validate transactions by solving a difficult mathematical puzzle called proof of work. Now, proof of work is the process to determine a number (nonce) along with a cryptographic hash algorithm to produce a hash value lower than a predefined target. The nonce is a random value that is used to vary the value of hash so that the final hash value meets the hash conditions.

8. **Differentiate between Proof of Work vs Proof of Stake.**

Proof of Work (PoW):

In Blockchain, PoW is the process of solving a complex mathematical puzzle called mining. Here, the probability of mining a block is based upon the amount of computational work done by a miner. Miners **spend a lot of computing** power (with hardware) for solving the cryptographic puzzle.

Proof of Stake (PoS):

PoS is an alternative to PoW in which the Blockchain aims to achieve distributed consensus. The probability of validating a block relies upon the number of tokens you own. The more tokens you have, the more chances you get to validate a block. It was created as a solution to minimize the use of expensive resources spent in mining.

9. **What is a 51% attack?**

In Blockchain, a 51% attack refers to a vulnerability where an individual or group of people controls the majority of the mining power (hash rate). This allows attackers to prevent new transactions from being confirmed. Further, they can double-spend the coins. In a 51% attack, smaller cryptocurrencies are being attacked.

10. **What is Merkel Tree?**

Merkel Tree is a data structure that is used for verifying a block. It is in the form of a binary tree containing cryptographic hashes of each block. A Merkle tree is structured similarly to a binary tree where each leaf node is a hash of a block of transactional data and each non-leaf node is a hash of its leaf node. The Merkel root or hash root is the final hash root of all the transaction hashes. It encompasses all the transactions that are underlying all the non-leaf nodes.

11. **What do you mean by blocks in Blockchain technology?**

Blockchain is a distributed database of immutable records called blocks, which are secured using cryptography. Refer to the video to see the various attributes of a block.

There are a previous hash, transaction details, nonce, and target hash value. A block is like a record of the transaction. Each time a block is verified, it gets recorded in chronological order in the main Blockchain. Once the data is recorded, it cannot be modified.

12. **How is Blockchain distributed ledger different from a traditional ledger?**

- A Blockchain distributed ledger is highly transparent as compared to a traditional ledger.
- Blockchain distributed ledgers are irreversible. Information registered on a distributed ledger cannot be modified whereas on a traditional ledger it is reversible.
- A distributed ledger is more secure. It uses cryptography and every transaction is hashed and recorded whereas in traditional ledger security can be compromised.

- In a distributed ledger, there is no central authority. It is a distributed system and the participants hold the authority to maintain the sanity of the network and are responsible for validating the transactions. Traditional ledgers are based on the concept of centralized control, which controls all transactions.
- In a distributed ledger, identities are unknown and hidden whereas in traditional ledger identities of all participants have to be known before the transactions happen.
- In a distributed ledger, there is no single point of failure as the data is distributed and information is shared across multiple nodes. If one node fails, the other nodes carry the same copy of the information. In comparison, traditional ledgers have a single point of failure. If a single system crashes, the entire network comes to a standstill.
- In a distributed ledger, data modification or change cannot be done but for a traditional ledger, it is possible.
- In a distributed ledger, validation is done by the participants in the network while in a traditional ledger, validation is done by a centralized authority.
- The copy of the ledger is shared amongst participants in a distributed ledger while in a traditional ledger, a single copy is maintained in a centralized location. It is not shared amongst the participants.

13. How can you identify a block?

Every block consists of four fields -

- The hash value of the previous block (thereby getting linked in a blockchain)
- It contains details of several transaction data
- It has a value called the nonce. The nonce is a random value which is used to vary the value of the hash in order to generate hash value less than the target
- Hash of the block itself. It is the digital signature of the block and an alphanumeric value used to identify a block

The hash address is the unique identification of the block. It is a hex value of 64 characters that have both letters and digits. It is obtained by using the SHA - 256 algorithms.

Refer to the video to see how a block is structured. The hash of the previous block, transaction data, and the nonce consolidate the header of the block. They are together passed through a hashing function and then the hash value is generated.

14. **What are the different types of Blockchain?**

There are three different types of Blockchain - Public, Private, and Consortium Blockchain.

Public Blockchain ledgers are visible to all the users on the internet and any user can verify and add a block of transactions to the Blockchain. Examples, Bitcoin, and Ethereum.

Private Blockchain ledgers are visible to users on the internet but only specific users in the organization can verify and add transactions. It’s a permissioned blockchain, although the information is available publicly, the controllers of the information are within the organization and are predetermined. Example, Blockstack.

In Consortium Blockchain, the consensus process is controlled by only specific nodes. However, ledgers are visible to all participants in the consortium Blockchain. Example, Ripple

15. Where is a blockchain stored?

The blockchain can be either stored as a flat file or as a database.

16. What are the types of records that are present in the blockchain database?

There are two types of records in a blockchain database.

1. Transactional Records
2. Block Records

Both the records can easily be accessed and can integrate with each other without following any complex algorithm.

17. Can you modify the data in a block?

No, it's not possible to modify the data in a block. In case any modification is required, you would have to erase the information from all other associated blocks too.

18. Illustrate architectural view of a generic blockchain

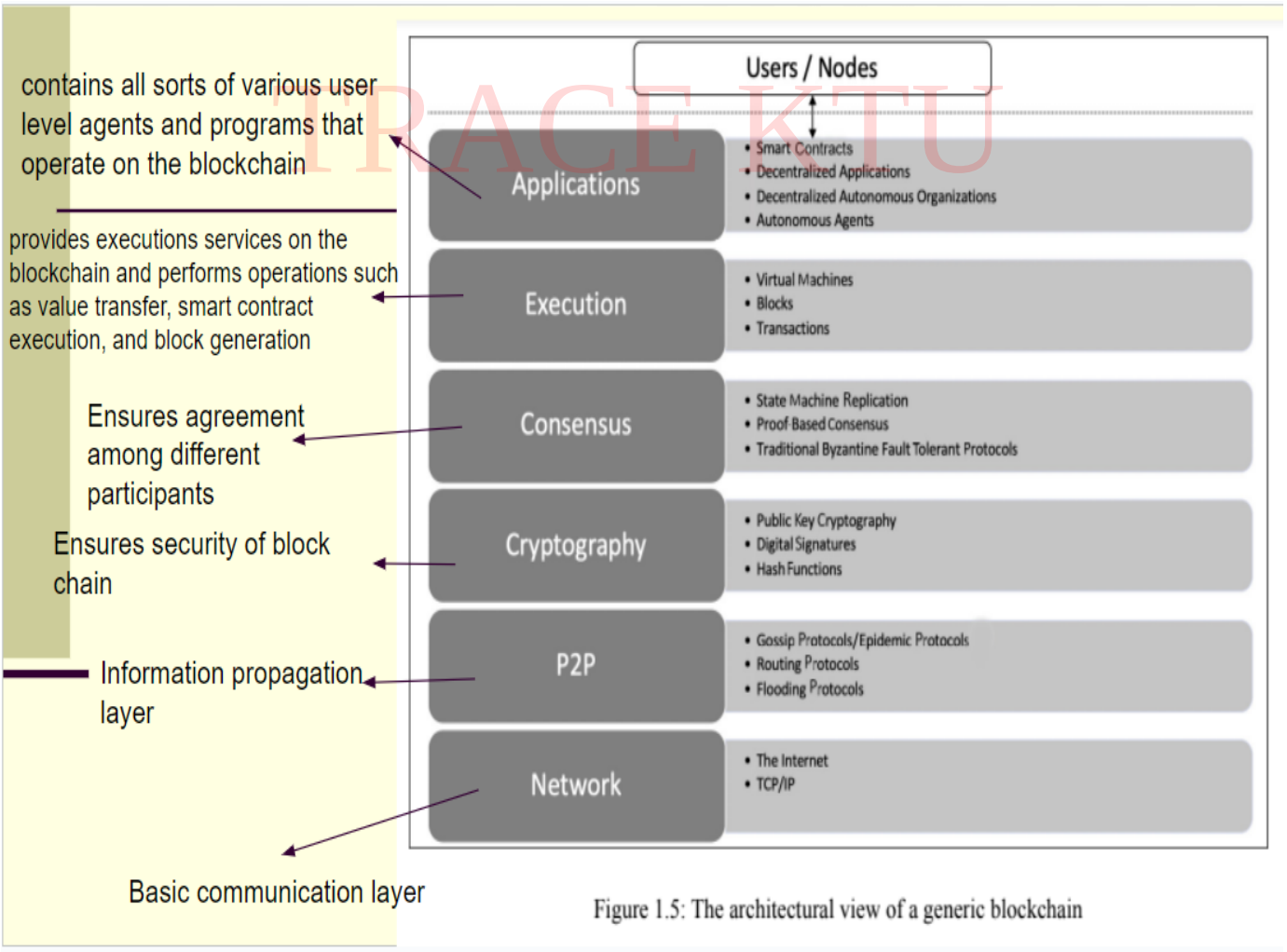


Figure 1.5: The architectural view of a generic blockchain

19. Draw generic structure of a Blockchain

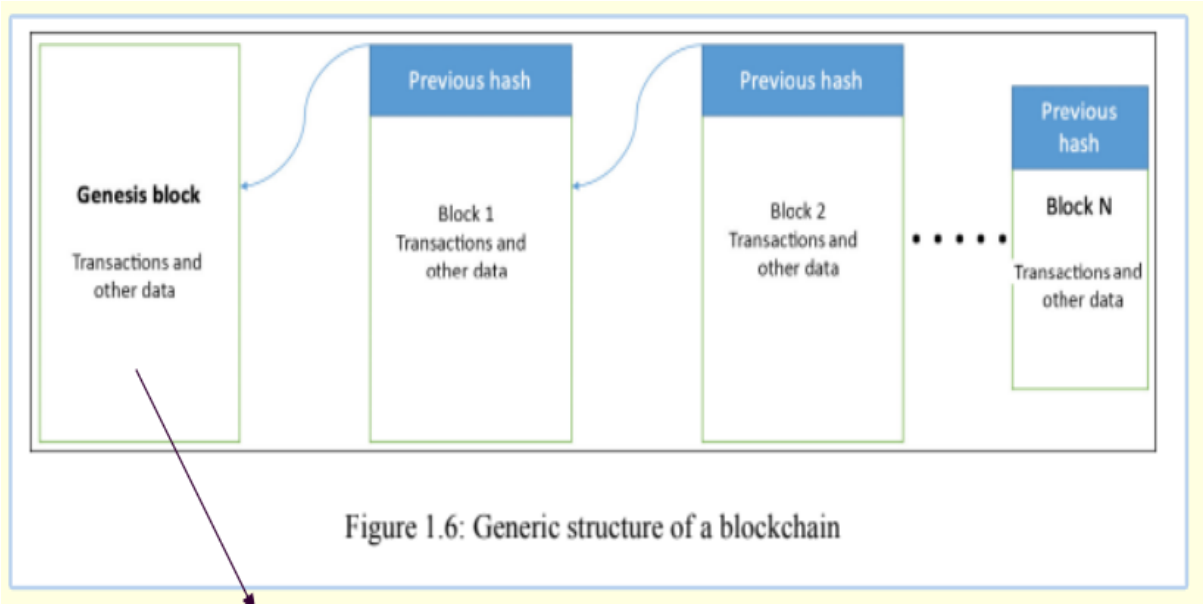


Figure 1.6: Generic structure of a blockchain

20.Explain generic structure of a block

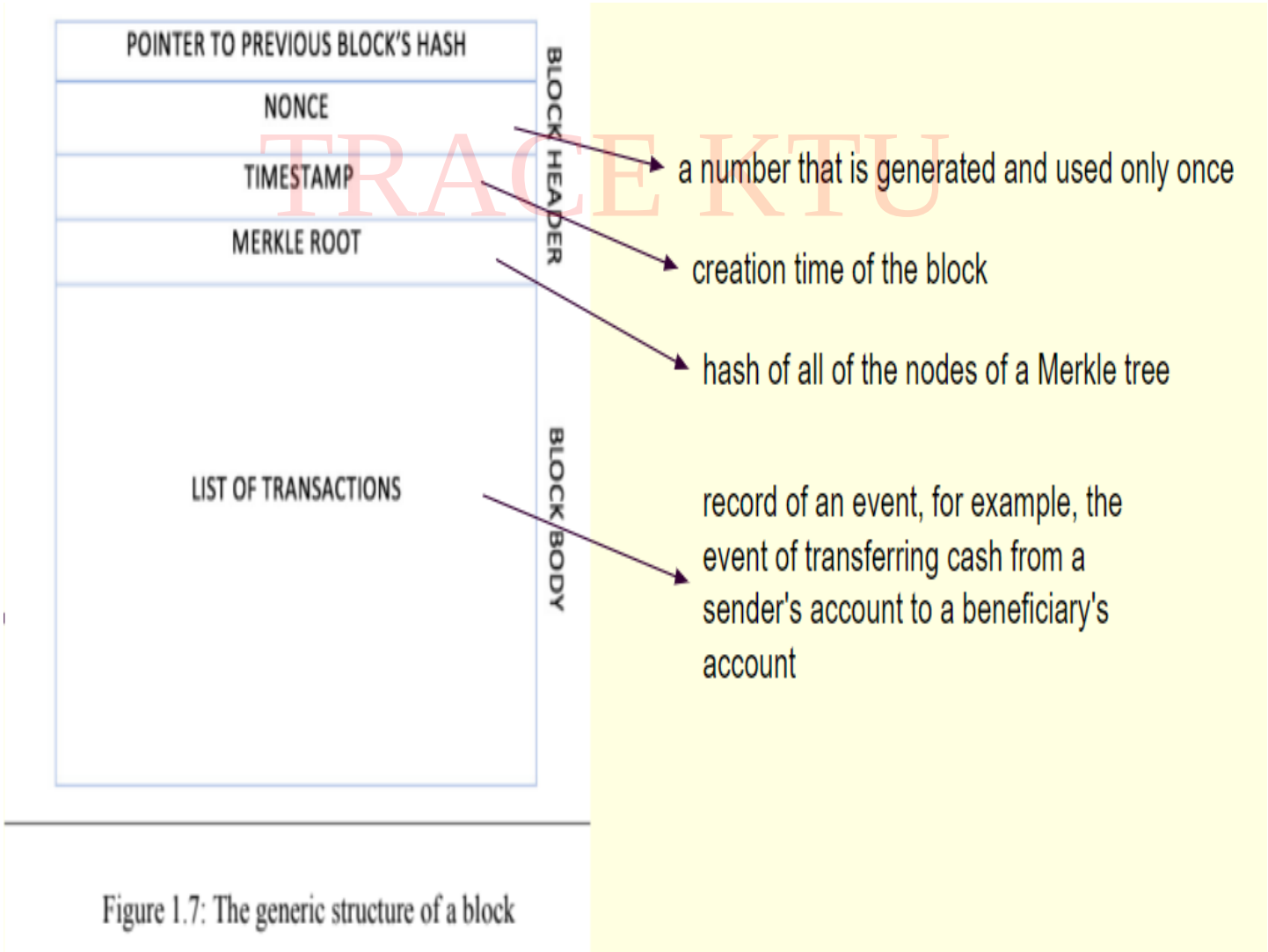


Figure 1.7: The generic structure of a block

21. illustrate Generic structure of a Blockchain Network

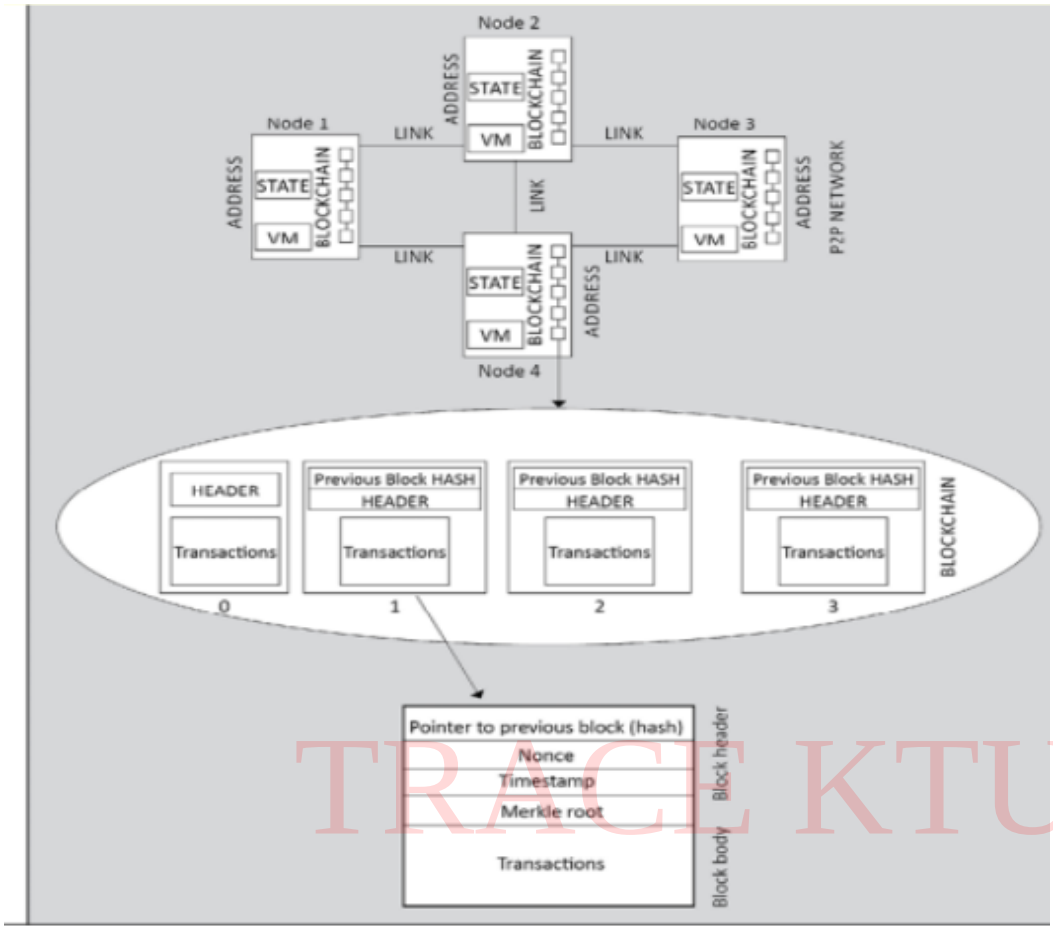


Figure 1.8: Generic structure of a blockchain network

22. How a block is generated in blockchain

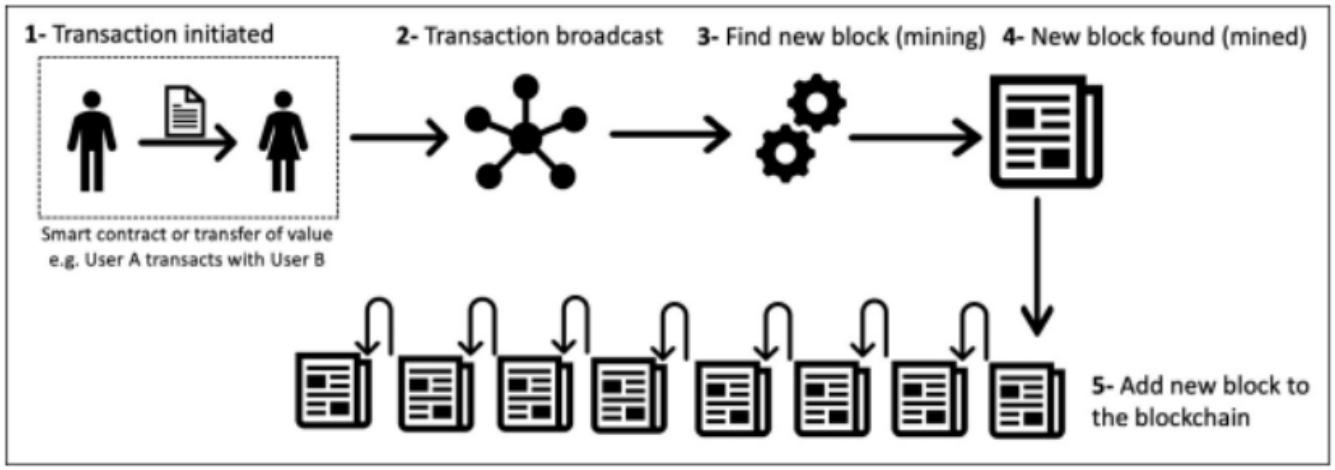


Figure 1.9: How a block is generated