Blockchain Technology

Mathematical Foundation for Blockchain

UNIT 1

Contents

Cryptography:

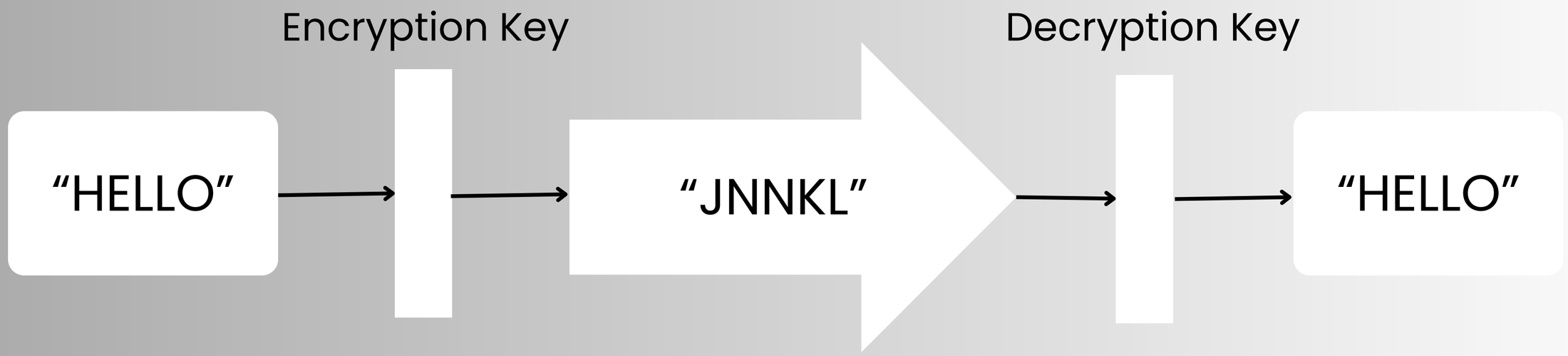
- Symmetric Key Cryptography
- Asymmetric Key Cryptography
- Elliptic Curve Cryptography (ECC)

Cryptographic Hash Functions:

- SHA256
- Digital Signature Algorithm (DSA)
- Merkel Trees.

Cryptography

- Data and information plays a significant function in todays world
- Data can be directly collected, analyzed, and delivered in digital format.
- Attackers are focusing on computer systems and open channels of communication to acquire sensitive data



“The art of protecting information by transferring it in to an unreadable format is called Cryptography”

Cryptography Basic terminology

- **Plaintext** : The original, readable message or data that needs protection.
  Example: "HELLO"

- **Ciphertext** : The scrambled or unreadable message after encryption.
  Example: "XMCKL"

- **Encryption** : The process of converting plaintext into ciphertext using an algorithm and a key.

- **Decryption** : The reverse process: converting ciphertext back into plaintext using the correct key.

- **Key** : A secret value (number/string) used in the encryption and decryption process.
  Two types: Symmetric (same key), Asymmetric (public/private key pair).

Importance of Cryptography

- **Confidentiality -** It protect data and communications against unauthorized access and disclosure.

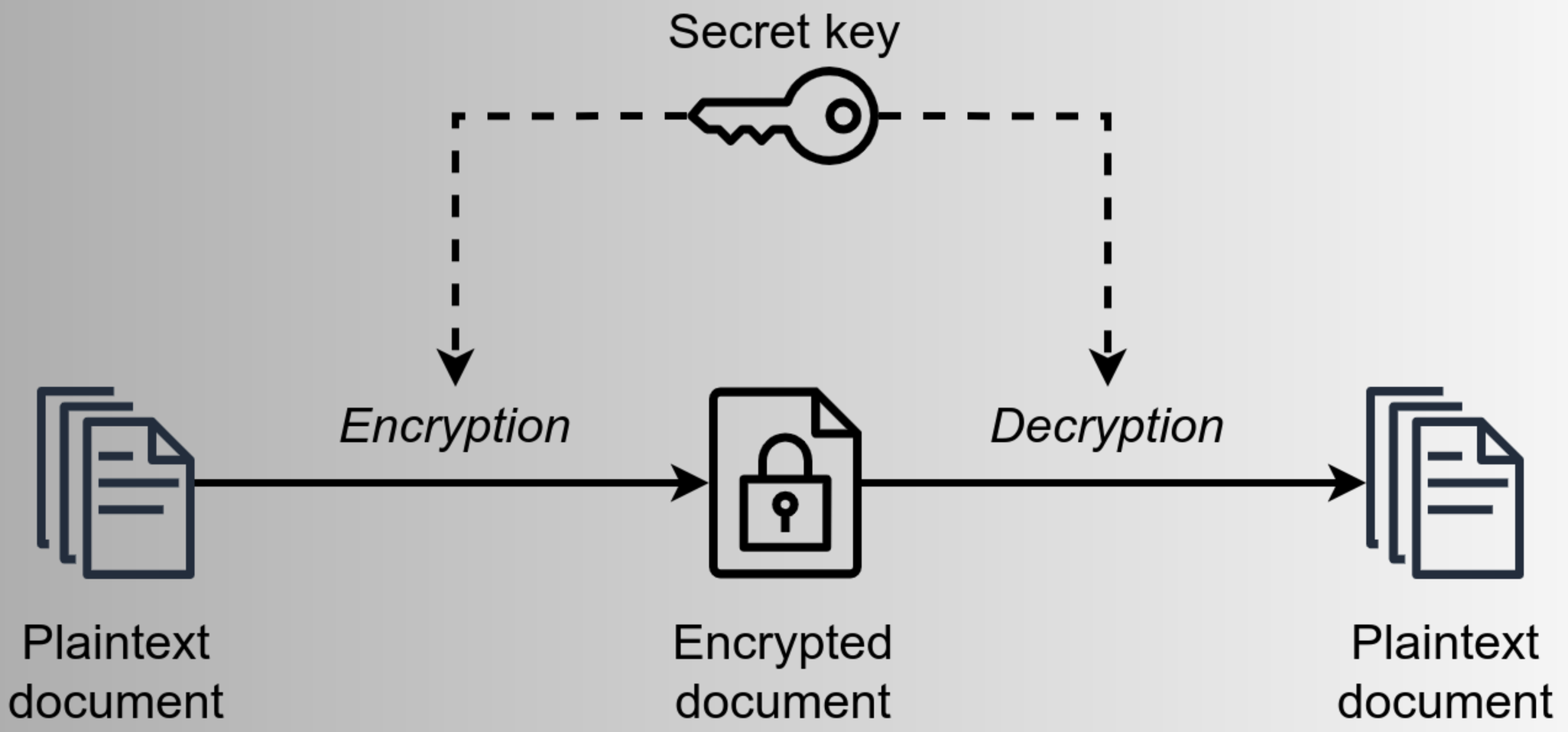
- **Authentication -** Information can be safeguarded against spoofing and forgeries using cryptographic techniques like MAC and digital signatures, which are used for authentication.

- **Data Integrity -** Cryptographic hash functions are essential in giving users confidence in the accuracy of their data.

- **Non-repudiation -** A digital signature offers the non-repudiation service to protect against disputes that can develop if the sender refuses to acknowledge receipt of the communication.

1.Symmetric Key Cryptography

- Simplest type of encryption technique
- Uses only one key for encryption and decryption
- Also called secret/private key cryptography
- Before starting communication sender and receiver shares the secret key through some external means
- Sender encrypts the message using his own copy of secret key
- Cipher text is then send to receiver over the communication channel
- Receiver decrypts the cipher text using his own copy of the key . Using decryption message is converted back to original format
- Ex. Data Encryption standard (DES),
  Advanced Encryption Standard (AES)


Secret key
Encryption
Decryption
Plaintext document
Encrypted document
Plaintext document

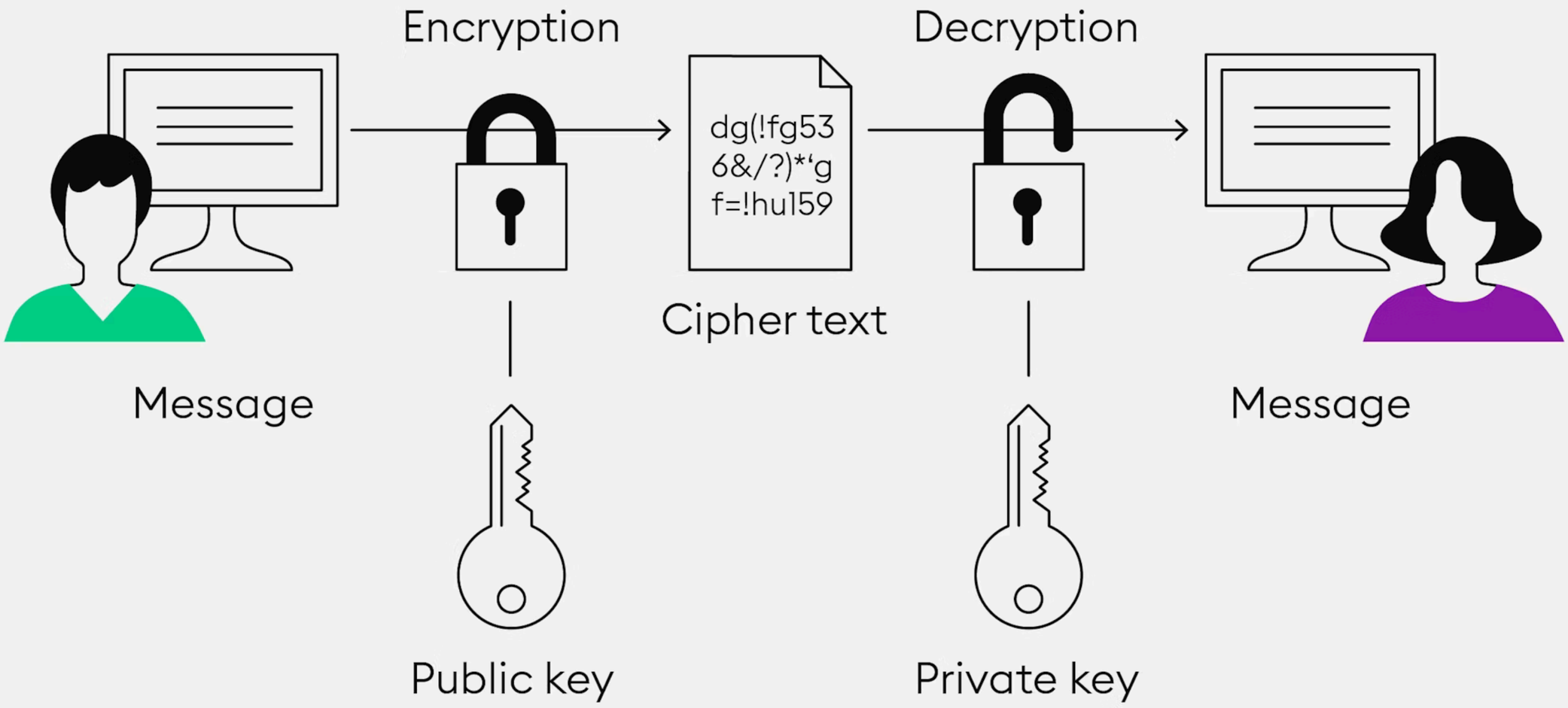
2.Asymmetric Key Cryptography

- It is also called public key cryptography
- In this technique sender and receiver use different keys to encrypt and decrypt the message
- It uses a pair of keys known as a public key and a private key
- Each public key is published and the corresponding private key is kept confidential

Working:

- Sender encrypts the message using receivers public key
- Public key of receiver is publicly available
- Encryption converts the message to ciphertext
- Cipher text is then send to receiver over the communication channel
- Ciphertext can be decrypted using receivers private key

ASYMMETRIC ENCRYPTION



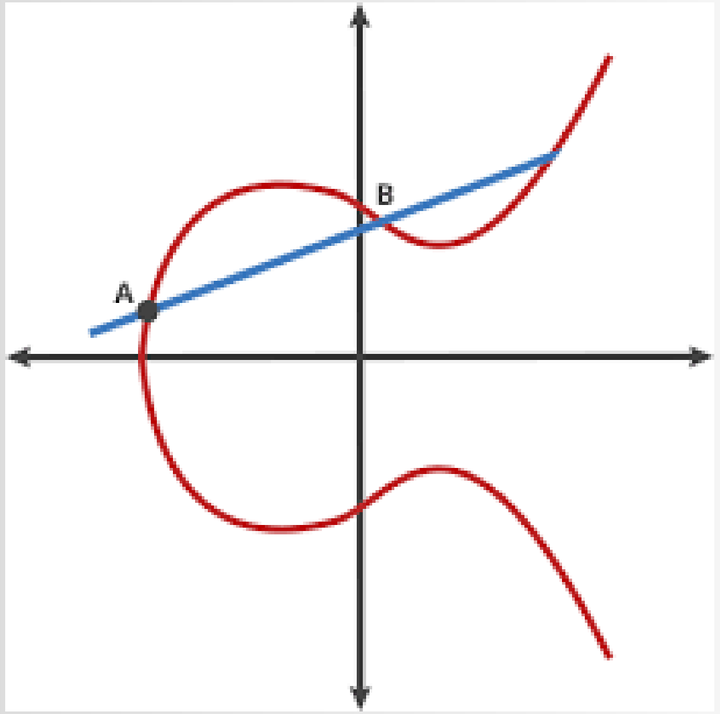
Difference

Feature	**Symmetric Cryptography**	**Asymmetric Cryptography**
Keys Used	Single key (same key for encryption and decryption)	Two keys (Public key for encryption, Private key for decryption)
Speed	Faster (less computational power needed)	Slower (requires more computation)
Security	Less secure (key distribution is difficult)	More secure (public key can be shared openly, private key remains secret)
Key Size	Smaller key sizes (e.g., 128/256 bits)	Larger key sizes (e.g., 2048/4096 bits for RSA)
Examples	AES, DES, Blowfish	RSA, ECC, DSA
Use Cases	Encrypting large amounts of data (files, databases)	Secure key exchange, digital signatures, certificates
Key Distribution Problem	Yes (needs a secure way to share the secret key)	No (public key can be shared freely)

Elliptical curve cryptography (ECC)

- **Definition**: ECC is a public key cryptography technique based on the mathematics of elliptic curves over finite fields
- **Key Idea**: Security relies on the difficulty of solving the Elliptic Curve Discrete Logarithm Problem (ECDLP).
- It is based on elliptic curve theory that can be used to create faster, smaller and more efficient cryptographic keys
- It is an alternative to RSA algorithm
- It is based on the latest mathematics and delivers a more secure foundation than the first generation public key cryptography systems for example RSA.
- It is used to perform critical security functions, including encryption, authentication, and digital signatures.

- **Equation of ECC:**
  An elliptic curve over a field is generally defined as:
  $y^2=x^3+ax+b$



- a, b are constants (define the shape of the curve).
- (x, y) are points on the curve.
- The keys are generated by making use of elliptical curve equation

ECC keys:

- **Private key:**
  ECC cryptography's private key creation is as simple as safely producing a random integer in a specific range, making it highly quick.
  Any integer in the field represents a valid ECC private key.
- **Public keys:**
  Public keys within ECC are EC points, which are pairs of integer coordinates x, and y that lie on a curve.

Difference

Feature	**ECC (Elliptic Curve Cryptography)**	**RSA (Rivest–Shamir–Adleman)**
Security Basis	Elliptic Curve Discrete Logarithm Problem (ECDLP)	Integer Factorization Problem
Key Size (for same security)	Much smaller (e.g., 256-bit ECC ≈ 3072-bit RSA)	Much larger (e.g., 3072-bit RSA ≈ 256-bit ECC)
Speed	Faster in encryption, decryption, key exchange, and digital signatures	Slower compared to ECC
Memory & Bandwidth	Requires less storage and bandwidth due to shorter keys	Needs more memory and bandwidth
Efficiency on Devices	Highly efficient, suitable for mobile & IoT devices	Less efficient on resource-constrained devices
Strength Growth	Security grows rapidly with key size	Security grows slowly with key size
Applications	Used in modern SSL/TLS, blockchain, cryptocurrencies, IoT	Still widely used in legacy systems, digital signatures, SSL/TLS

Cryptography Hash Function

Cryptographic Hash is a Hash function that takes random size input and yields a fixed-size output.

➢**properties**:

1. **Deterministic**: This means that the same message always results in the same hash.

2. **Quick**: It is quick to compute the hash value for any given message.

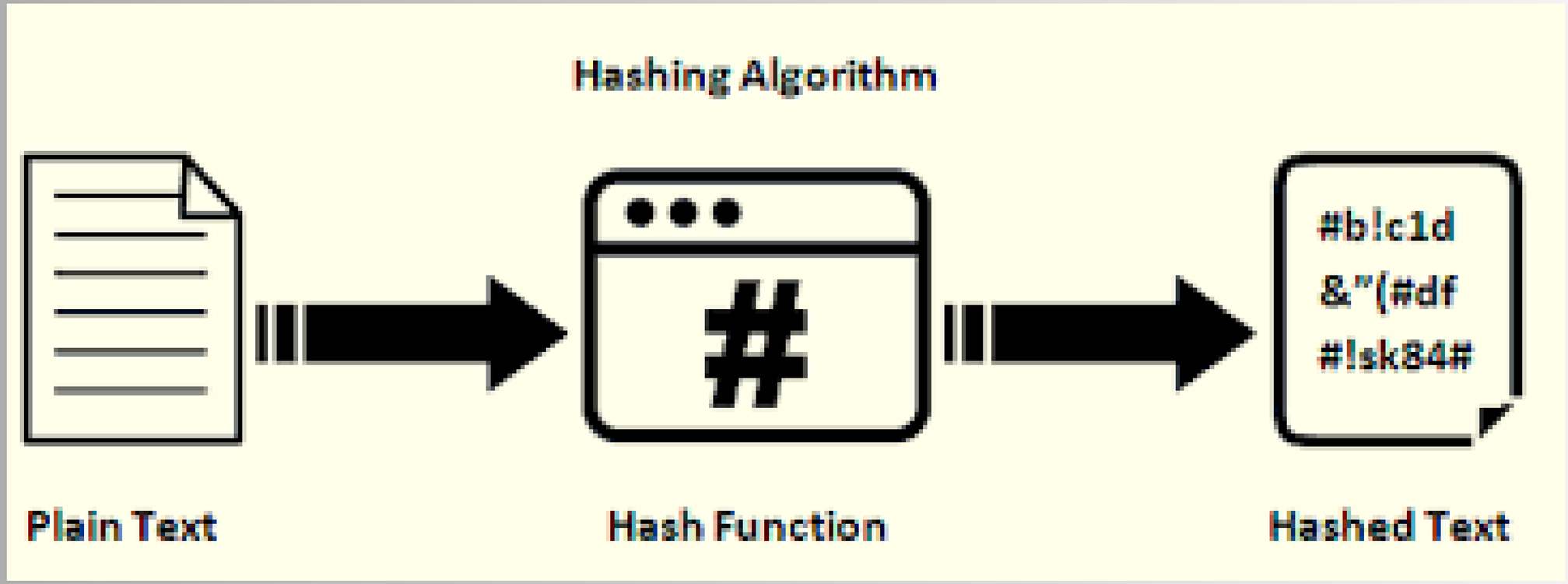
3. **Avalanche Effect**: Every minor change in the message results in a major change in the hash value.

4.**One-Way Function**: reverse operation is not possible Cryptographic Hash Function

5. **Collision Resistance**: It is infeasible to find two different messages that produce the same hash value.

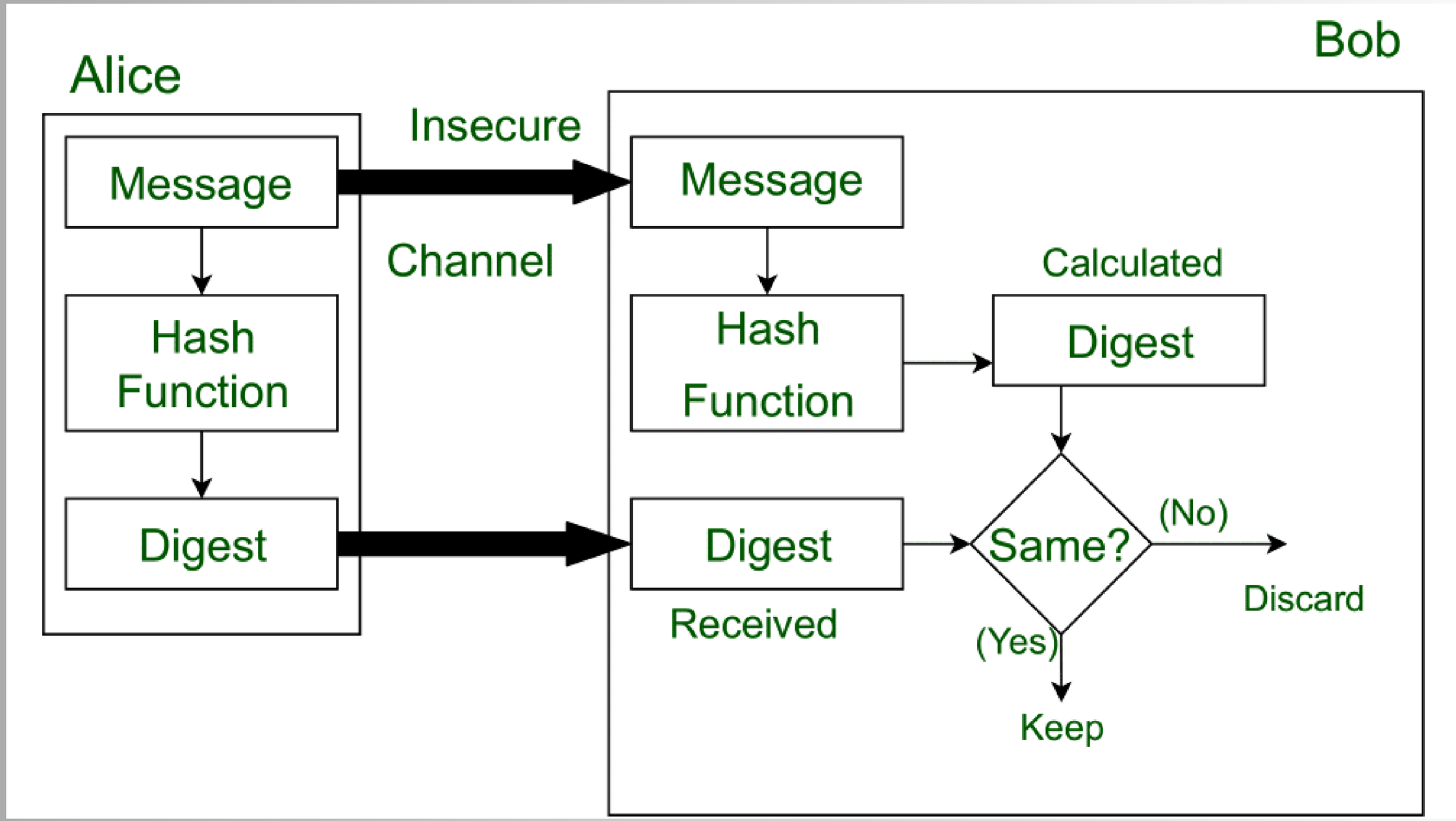
6. **Resistance**: The hash value shouldn’t be predictable from the given stringand vice versa.

7. **Second Pre-Image Resistance**: Given an input, it should be difficult to find another input that has the same hash value.


Hashing Algorithm
#
#b!c1d
&"(#df
#!sk84#
Plain Text
Hash Function
Hashed Text

Message Digest

- A Message Digest is a cryptographic hash function containing a string of digits created by a one-way hashing formula.
- The message is passed through a cryptographic hash function
- Regardless of the size of the message, the message digest produces a numeric representation of a fixed size when hashed
- This creates a compressed image of the message called Digest
- It is used to ensure the integrity of a message transmitted over an insecurechannel (where the content of the message can be changed)
- It is a one way function, that is, a function which is practically infeasible to invert.
- Digest is encrypted with sender's private key called digital signature which can be only decrypted by the receiver who has sender's public key
- The receiver can authenticate the sender and also verify the integrity of the sent message


Alice
Bob
Insecure
Channel
Message
Hash
Function
Digest
Message
Hash
Function
Calculated
Digest
Digest
Received
Same?
(No)
Discard
(Yes)
Keep

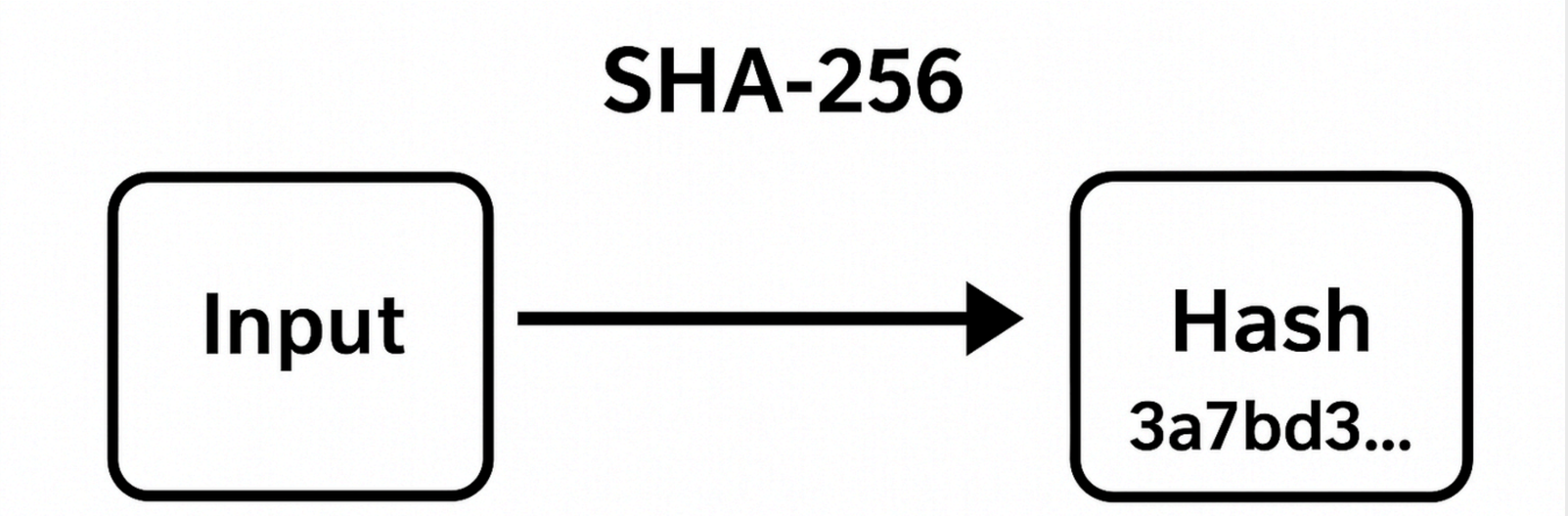
Secure Hashing Algorithm (SHA-256)

- **Definition**: SHA-256 (Secure Hash Algorithm 256-bit) is a cryptographic hash function from the SHA-2 family that produces a fixed 256-bit (32-byte) hash value.
- **Input & Output**: It can take any length of input data and always outputs a 64-character hexadecimal hash.
- **Irreversible**: It is a one-way function, meaning the original input cannot be retrieved from the hash.
- **Collision Resistance**: Two different inputs are extremely unlikely to produce the same hash, ensuring data integrity.
- **Applications**: Widely used in blockchain, digital signatures, password hashing, and verifying file/data integrity.

Working:

- **Message Preprocessing :** Input message is converted into binary. Message is padded to make its length a multiple of 512 bits.
- **Message Parsing :** The padded message is divided into 512-bit blocks. Each block is further divided into 16 words of 32 bits each.

- **Message Expansion** : The 16 words are expanded into 64 words using logical functions (shifts, XOR, etc.).
- **Initialization** : SHA-256 uses 8 fixed initial hash values (H0–H7) and 64 round constants (K0–K63).
- **Compression Function** : Each 512-bit block is processed in 64 rounds.In every round, logical operations (AND, OR, XOR, modular addition, shifts, rotations) are applied with constants and working variables (a–h).
- **Hash Value Update** : After all rounds, results are added to the current hash values.Process repeats for all blocks.
- **Final Output** : The concatenation of H0–H7 produces the 256-bit final hash value (unique fingerprint).



Digital Signature

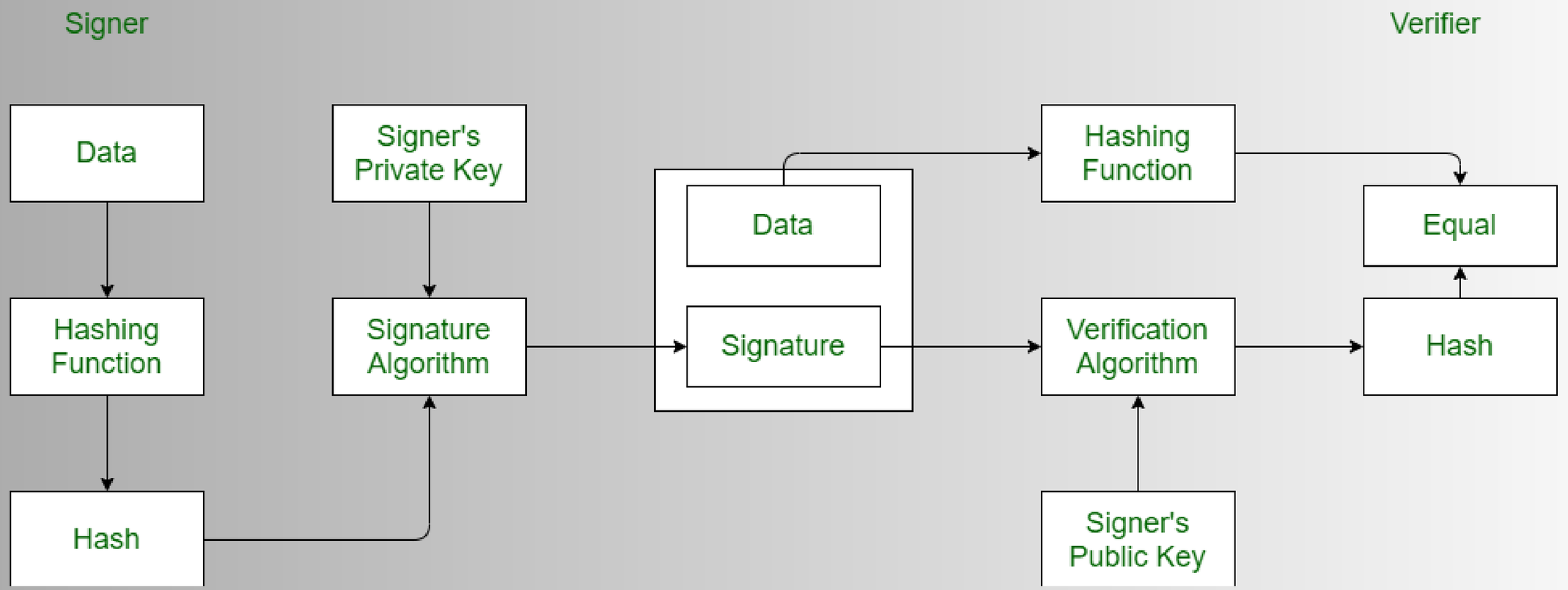
- The objective of digital signatures is to authenticate and verify documents and data.
- They work on the public key cryptography architecture.
- For digital signatures the signature is encrypted using the private key and decrypted with the public key.
- Because the keys are linked, decoding it with the public key verifies that the proper private key was used to sign the document.

Steps:

- **Step 1**: M, the original message is first passed to a hash function denoted by H# to create a digest
- **Step 2**: Next, it bundles the message together with the hash digest h and encrypts it using the sender's private key.
- **Step 3**: It sends the encrypted bundle to the receiver, who can decrypt itusing the sender's public key.

- **Step 4**: Once it decrypts the message, it is passed through the same hashfunction (H#), to generate a similar digest.

- **Step 5**: It compares the newly generated hash with the bundled hash valuereceived along with the message. If they match, it verifies data integrity.



Importance of Digital Signature

1. Ensures Authenticity
2.Offers Non-repudiation:
3. Provides Security
4.Improves Efficiency
5.Enhances Compliance

Digital Signatures Algorithm

A digital signature is a cryptographic technique used to validate the authenticity and integrity of a Message, Software, or Digital document

It ensures,

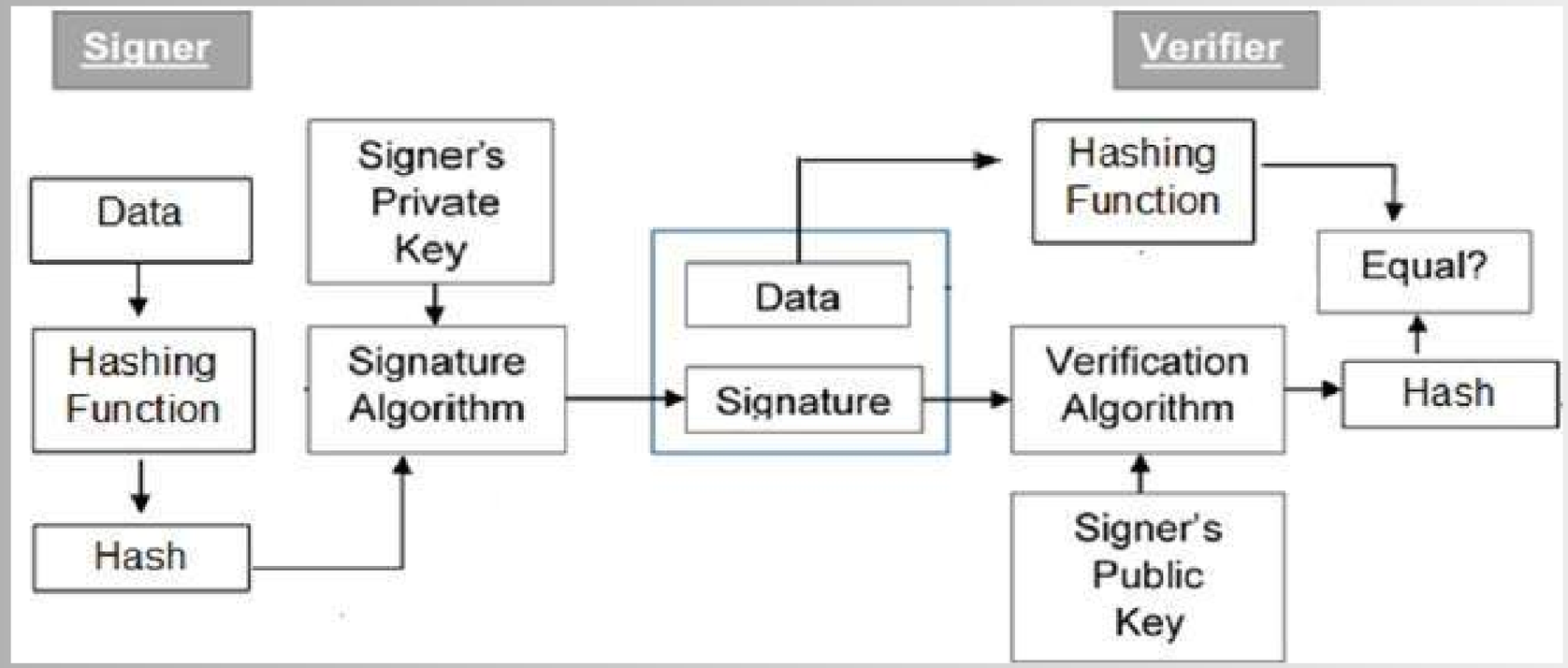
➢The message was created by a known sender (authentication).

➢The message was not altered in transit (integrity).

➢The sender cannot deny having sent the message (non-repudiation).

Applications of DSA : Digital certificates (e.g., SSL, TLS) , Government documents and ID verification Code signing and software distribution , Email security (e.g., PGP, S/MIME) , Block chain and smart contracts

- It is Public-private key pair in cryptography
- The key pair is different for every signature.
- Here, the private key is referred to as the signature key and the public key as the verification key
- Hash function generates a hash of data. Hash value and signature key are fed to the signature algorithm which produces the digital signature on a given hash of that message.
- This signature is appended to the data and then both are sent to the verifier to secure that message.



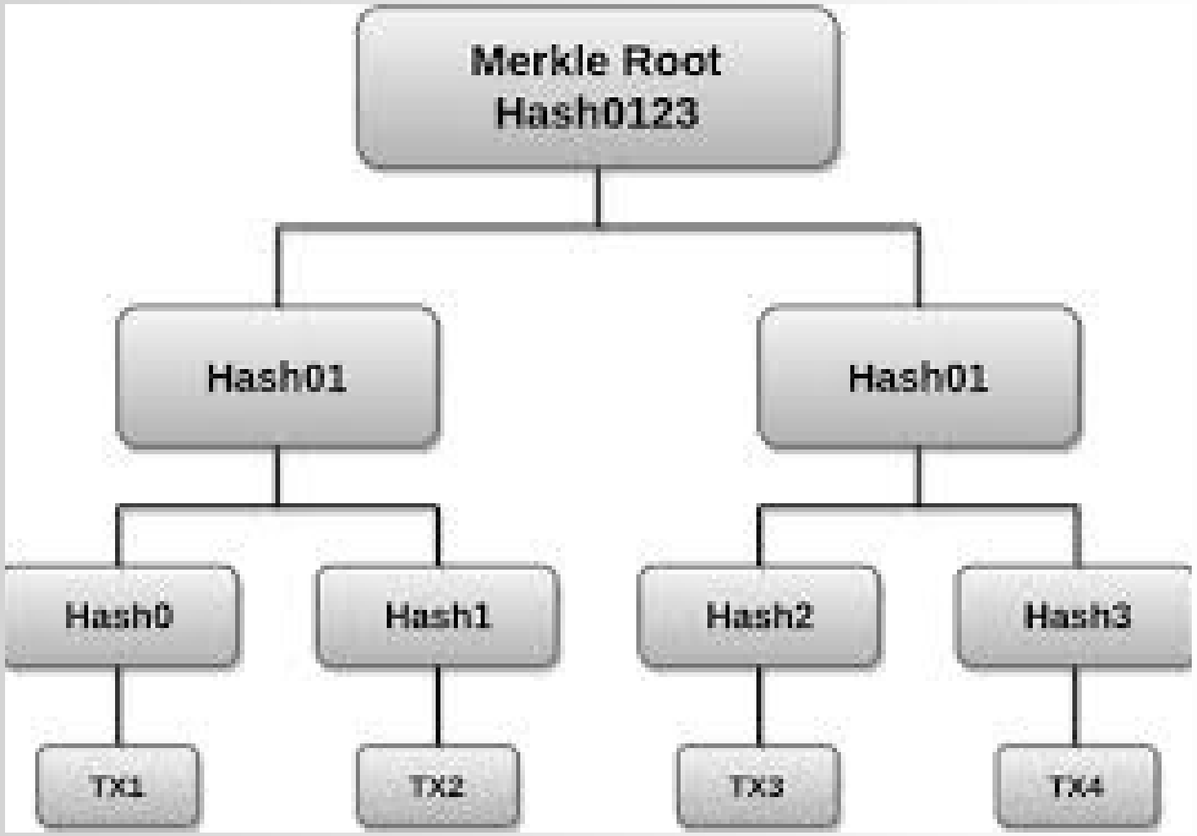
Merkel Tree

- Merkle tree also called a Hash Tree is a fundamental part of block chain technology.
- It is a binary tree structure used to encrypt block chain data more efficiently and securely
- Each leaf node contains the hash of a data block, and each non-leaf (internal) node contains the hash of its child nodes.
- Presents the summary of all the transactions in a block
- It provides a means to maintain the integrity and validity of data
- Merkle trees are created by repeatedly calculating hashing pairs of nodes until there is only one hash left.
- This hash is called the Merkle Root, or the Root Hash.
- Every leaf node is a hash of transactional data, and the non-leaf node is a hash of its previous hashes
- The Merkle Trees are constructed in a bottom-up approach.

- It is a binary tree, so it requires an even number of leaf nodes.
- In case of odd number of transactions, the last hash will be duplicated once to create an even number of leaf nodes

How do Merkle trees work?

- Four transactions in a block: TX1, TX2, TX3, and TX4
- Hashes of individual transactions 0, 1, 2, and 3
- Root Hash, or the Merkle Root is hash of entire tree
- Hash 01 = Hash(0 + 1)
- Hash 23 = Hash(2 + 3)
- Hash 0123 = Hash(01 + 23)



THANK YOU