Cryptography for Blockchain

Prof. James Won-Ki Hong

Distributed Processing & Network Management Lab.
Dept. of Computer Science and Engineering
POSTECH

http://dpnm.postech.ac.kr
jwkhong@postech.ac.kr

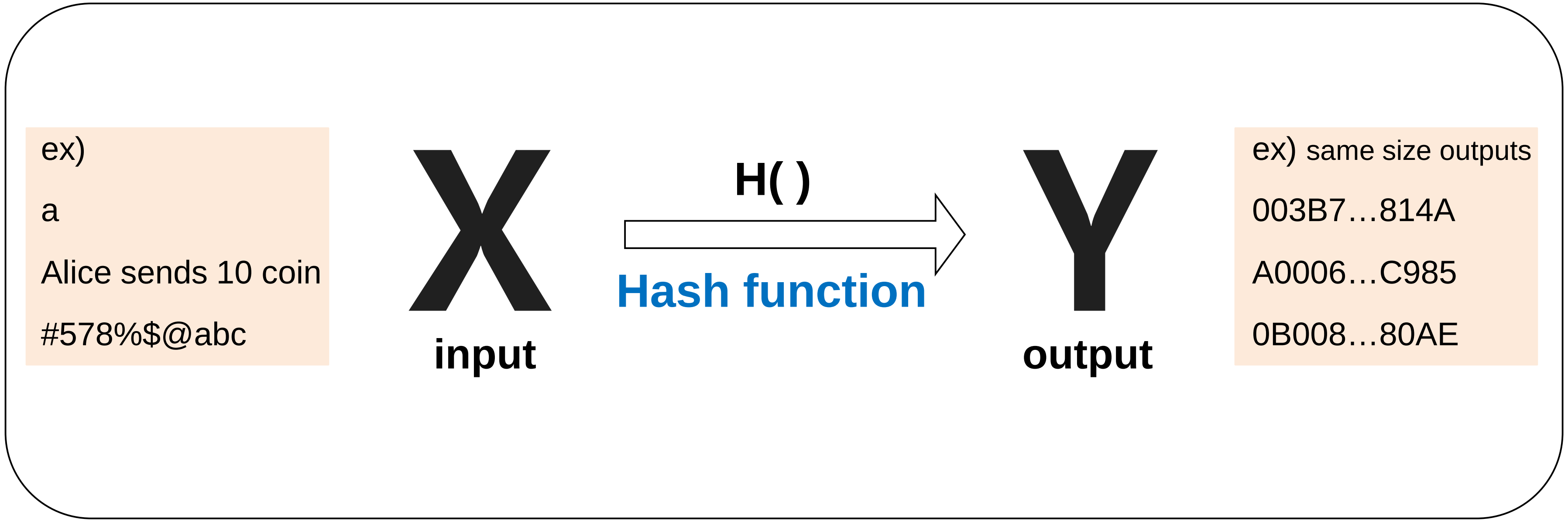
Table of Contents

Cryptographic Hash Functions (1/4)

▪ Hash function

1) Message Digest

- **Take any string as input (i.e., any string of any size)**
- **Always produce a fixed-size output**
- **One-way function**



Cryptographic Hash Functions (2/4)

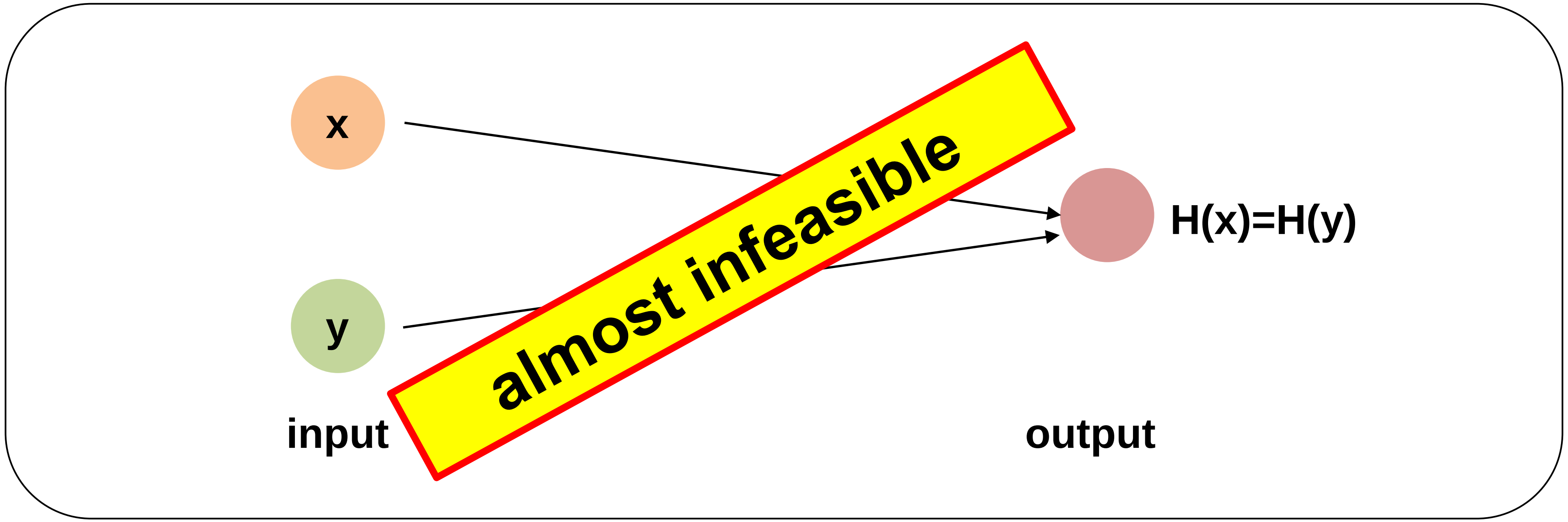
- **Hash function**

  2) Collision-free

  - **Nobody can find x and y such that x!=y and H(x)=H(y)**

    (If H(x) = H(y), it's safe to assume that x=y)

  - **When the input changes, the output also changes**



Cryptographic Hash Functions (3/4)

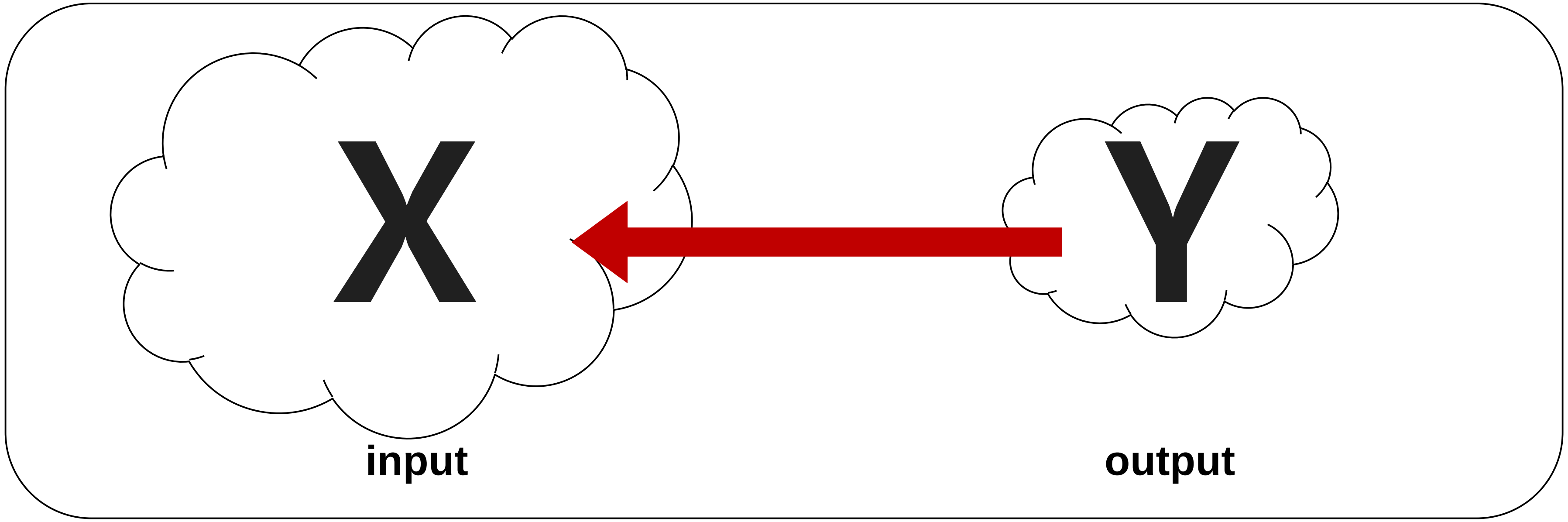
▪ Hash function

3) Hiding(Asymmetry)

- **Y=H(X) and given Y, it is infeasible to find X**
  - Ex) multiplication

mul(8*9) =72 **Easy to calculate**

Find x,y =72 → (x,y)=(1,72),(2,36),(3,24)**…** **Too many cases**



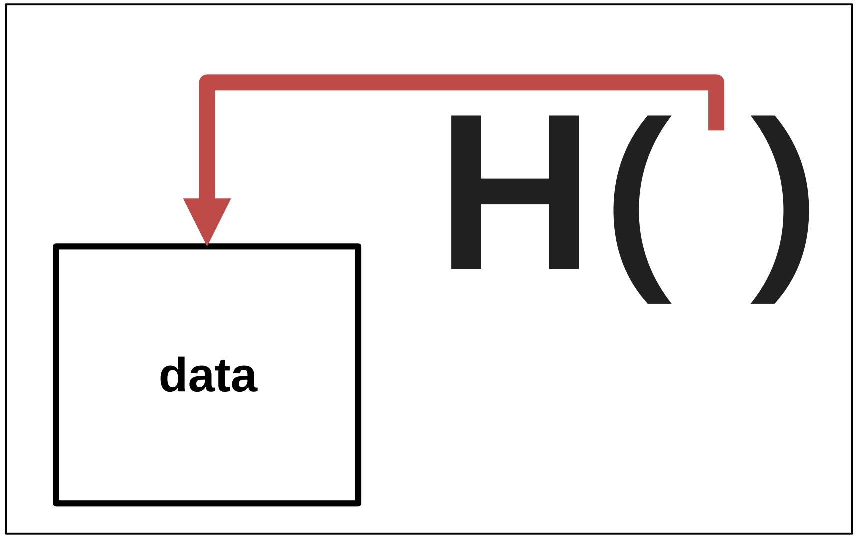
Cryptographic Hash Functions (4/4)

Kinds of Hash Functions

- **SHA (Secure Hash Algorithm)**
  - SHA-1
    - less than 2^64 bits input
    - Produces 160 bit output
  - SHA-256
    - Used in Bitcoin
    - Produces 256 bit output
- **Keccak 256**
  - Produces 256 bit output
  - Used in Ethereum
    - The first 96 bits are discarded and only the last 160 bits are used

Hash Pointers and Data Structures (1/2)

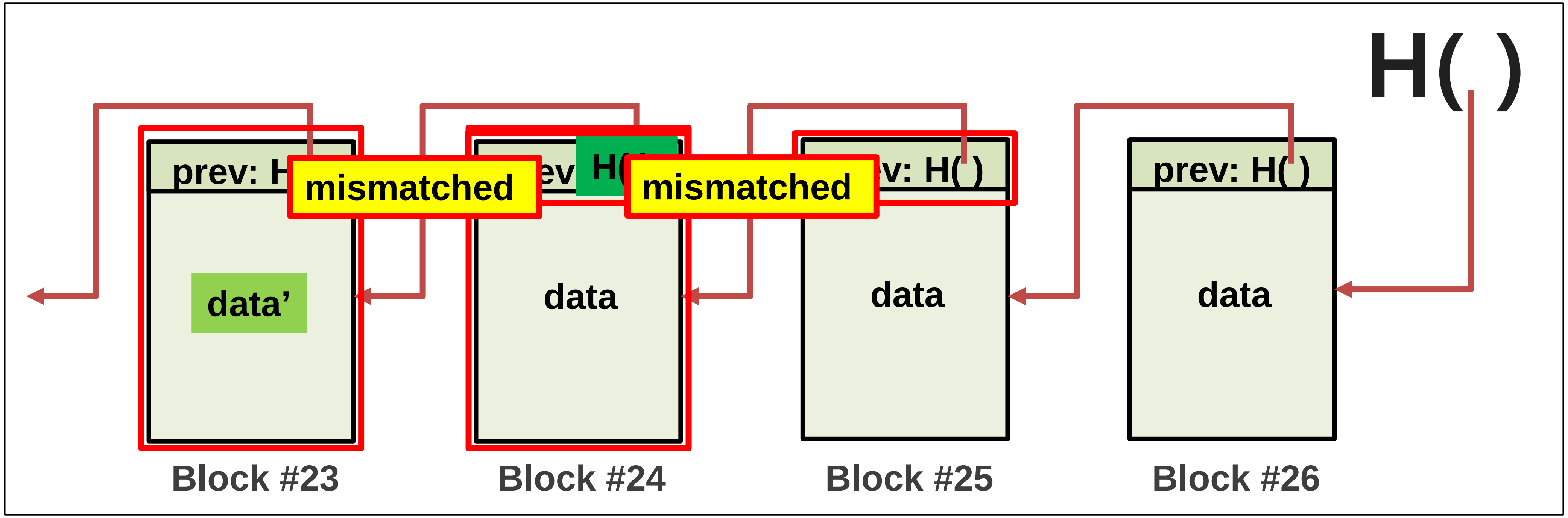
- **What is Hash Pointer?**
  - Pointer to where some information is stored



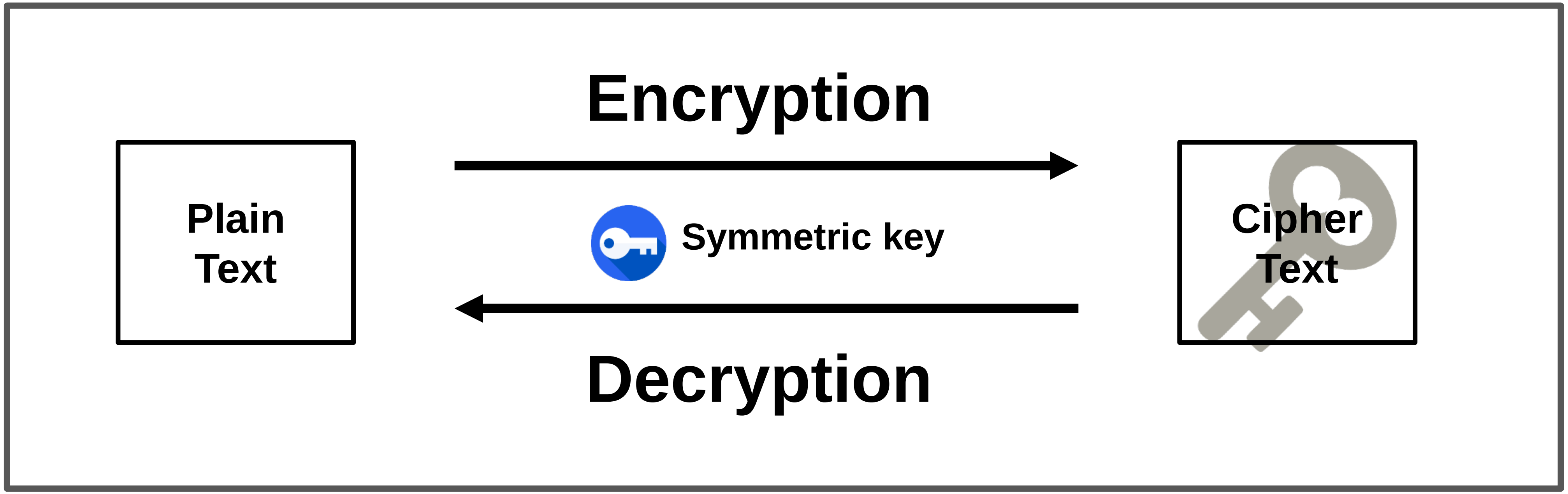
- **Why the Hash Pointer is used?**
  - For asking to get the information back
  - For verifying that the information hasn't changed

Hash Pointers and Data Structures (2/2)

- **Linked list with Hash pointers**
  - Each block has a hash pointer to the previous block in the list
  - Detecting tampering

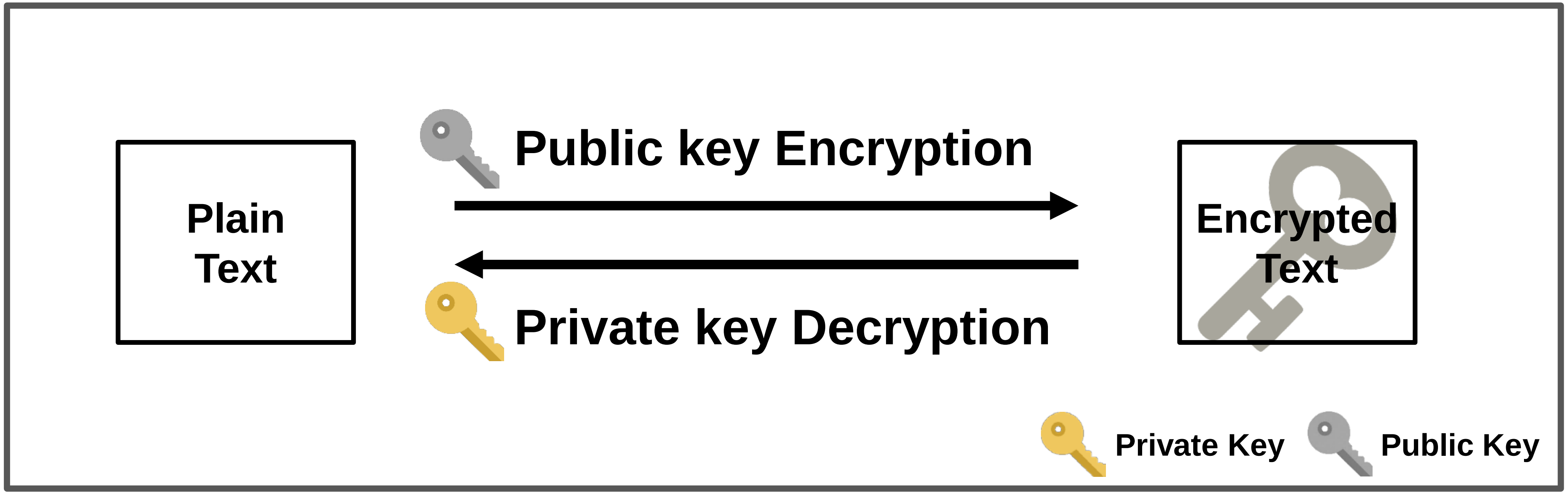


- **Symmetric Key Algorithm**
  - One symmetric key



  - Example: 3DES, AES

- **Asymmetric Key Algorithm**
  - Asymmetric Key
    - Private Key & Public Key
    - Public key Encryption



  - Example: RSA

Digital Signatures (1/5)

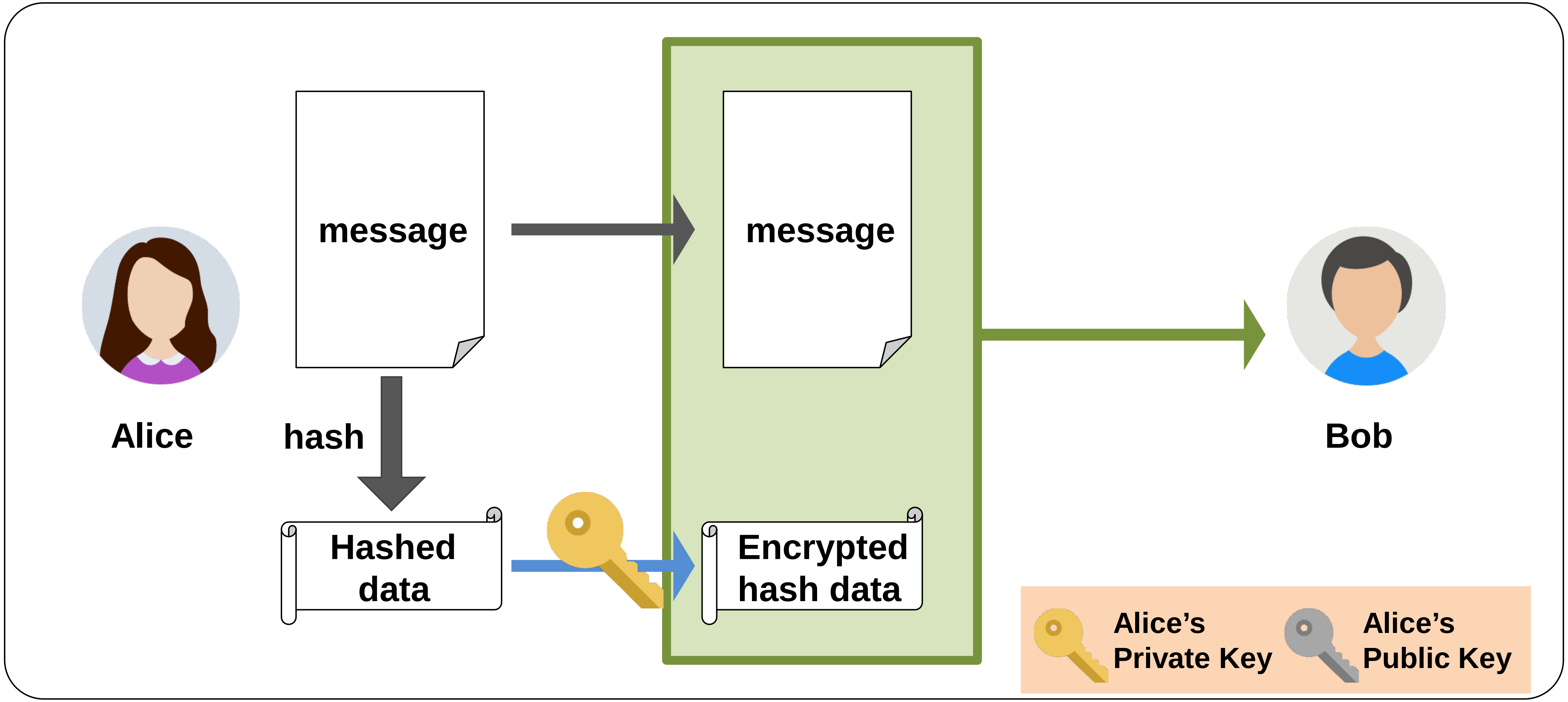
- **What is Digital Signature?**
  - Techniques for realizing functions in the computer that correspond to seal imprint or sign on the document

- **What we want from Digital Signatures**
  - No forgery
  - Authentication
  - No re-use
  - Unchangeable
  - Non-repudiation

Digital Signatures (2/5)

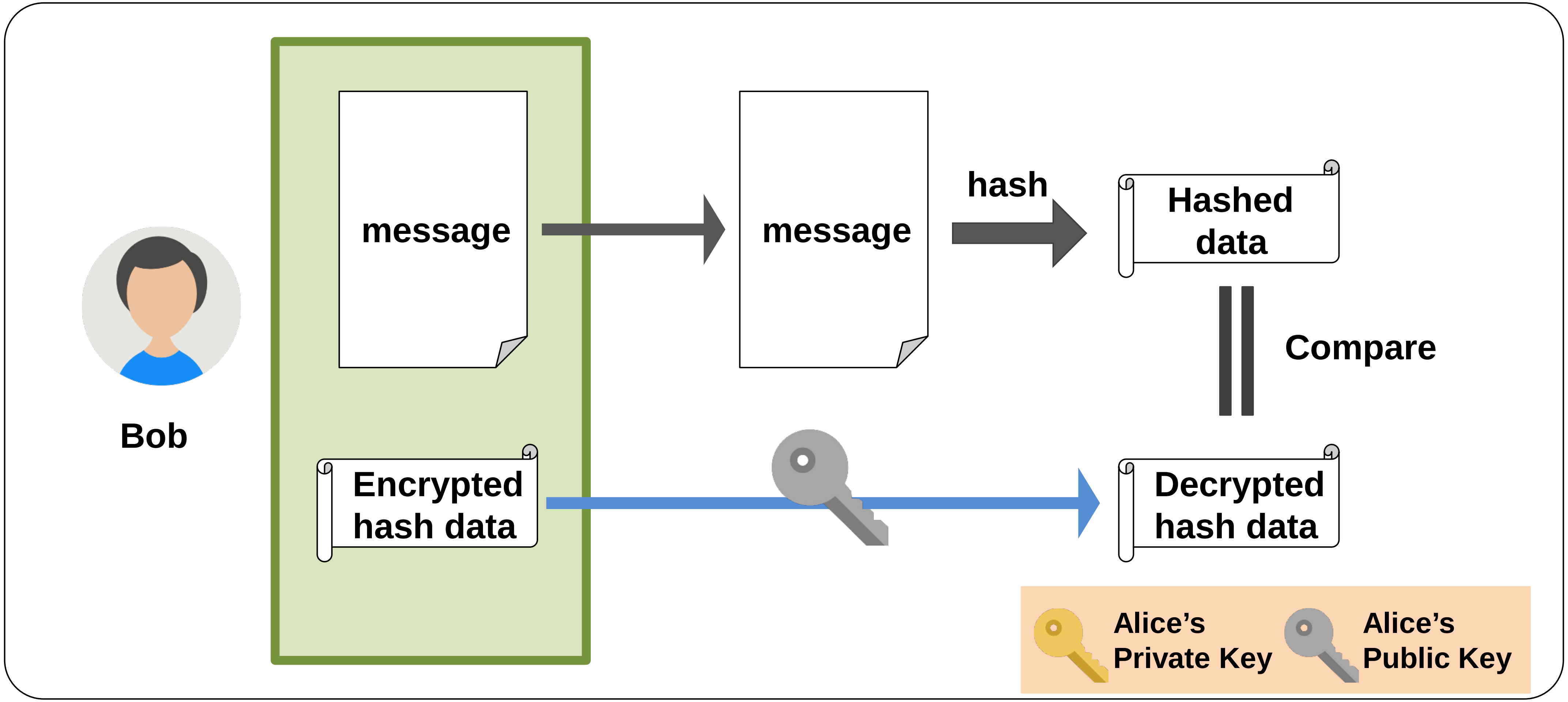
- **Basic for Digital Signatures**
  - Use a pair of Private key & Public key
  - Signing
    - Only you can sign with Private Key
  - Verification
    - Anyone can verify with Public Key
  - Hash of Original message
    - Size reduction
    - Message integrity

- **Process of Digital Signatures**
  - Signing



Digital Signatures (4/5)

- **Process of Digital Signatures**
  - Verification



Digital Signatures (5/5)

- **API for digital signatures**
  - (sk, pk) := generateKeys(keysize)
    - sk: secret signing key
    - pk: public verification key
  - sig := sign(sk, message)
  - isValid := verify(pk, message, sig)

Summary

- **Cryptographic Hash function**
  - **Produce fixed size output**
  - **Collision free**
  - **Hiding**

- **Hash pointers and data structures**

- **Basic Cryptography**

- **Digital signature**
  - **Signing**
  - **Verification**

References

- https://www.coursera.org/learn/cryptocurrency/lecture/gFEJL/cryptographic-hash-functions
- https://www.youtube.com/watch?v=lik9aaFIsl4
- https://en.wikipedia.org/wiki/Encryption
- https://en.wikipedia.org/wiki/Digital_signature
- http://www.parkjonghyuk.net/lecture/modernCrypto/lecturenote/chap09.pdf