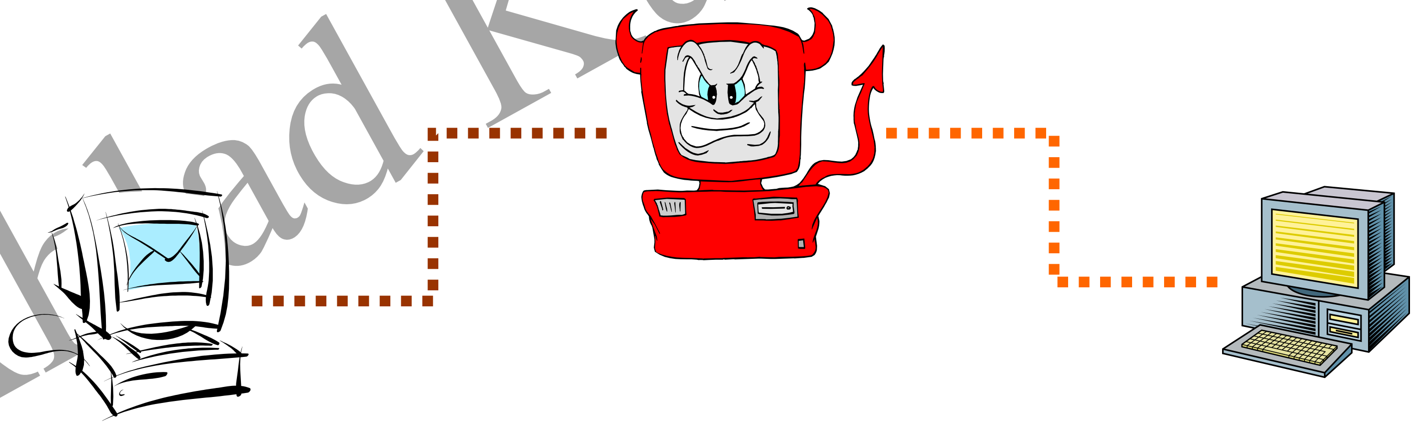
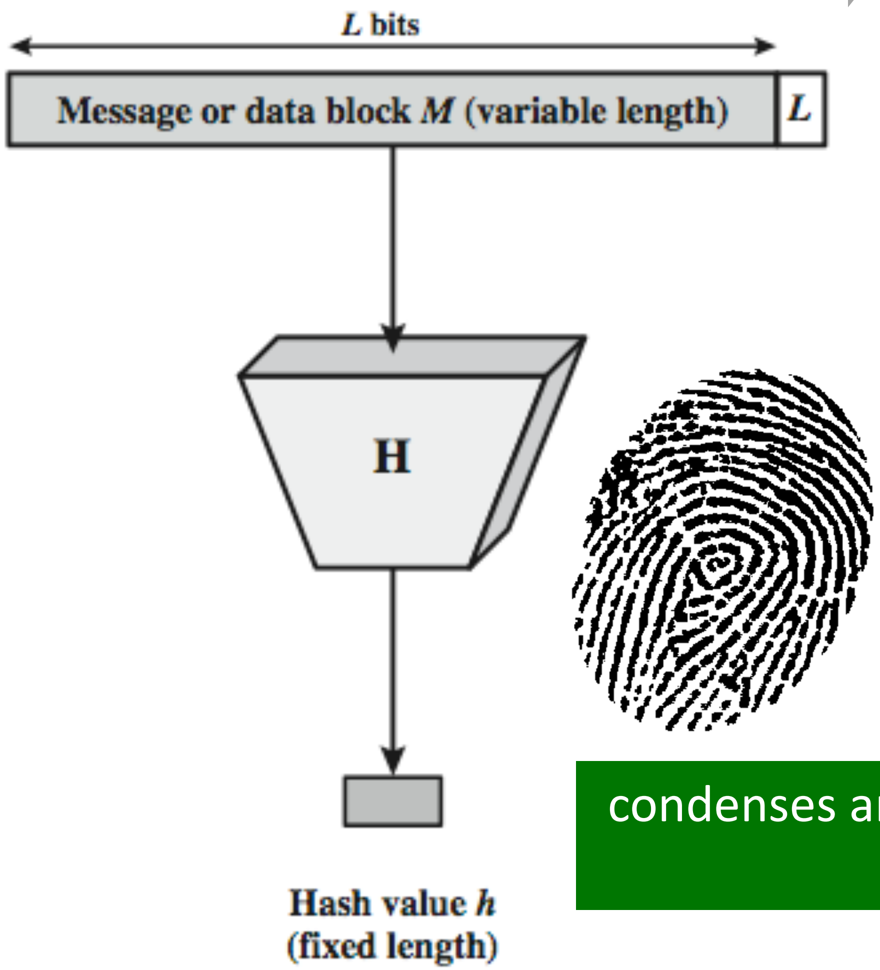
Hash Functions

Data Integrity and Source Authentication



- Encryption does not protect data from modification by another party.
- Need a way to ensure that data arrives at destination in its original form as sent by the sender and it is coming from an authenticated source.

Hash Function



- The hash value represents concisely the longer message
  - may called the *message digest*
- A message digest is as a ``digital fingerprint'' of the original document

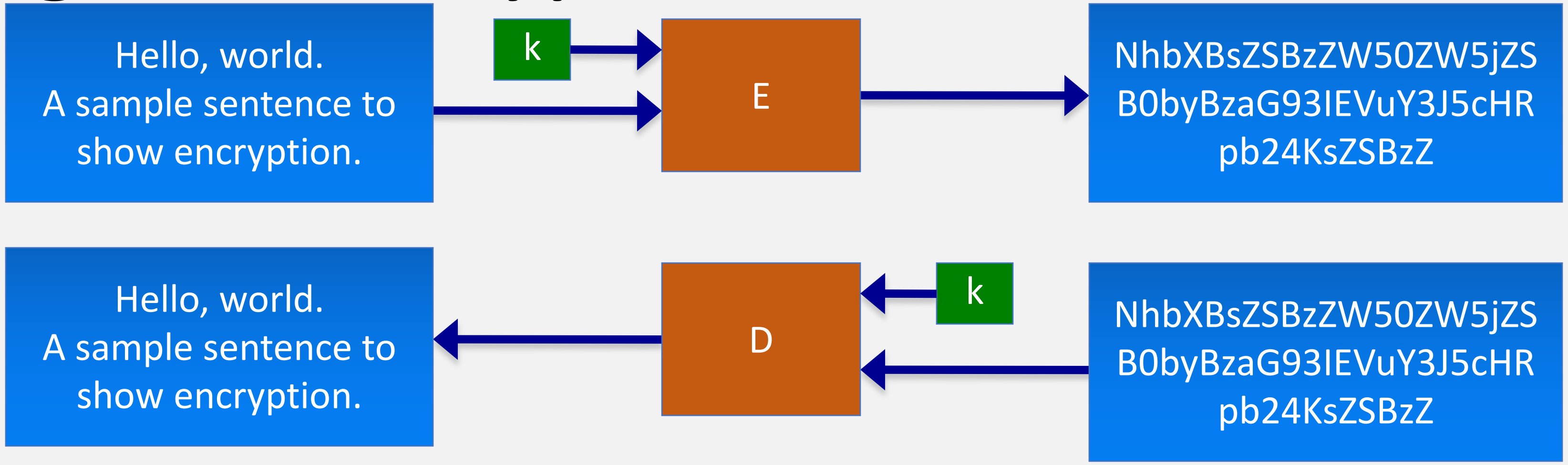
condenses arbitrary message to fixed size

$$h = H(M)$$

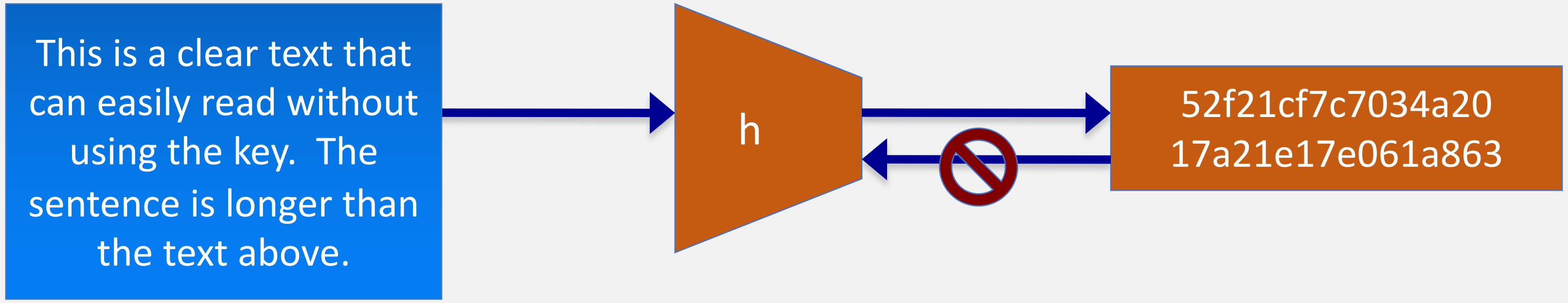
Chewing functions

- Hashing function as “chewing” or “digest” function

Hashing V.S. Encryption

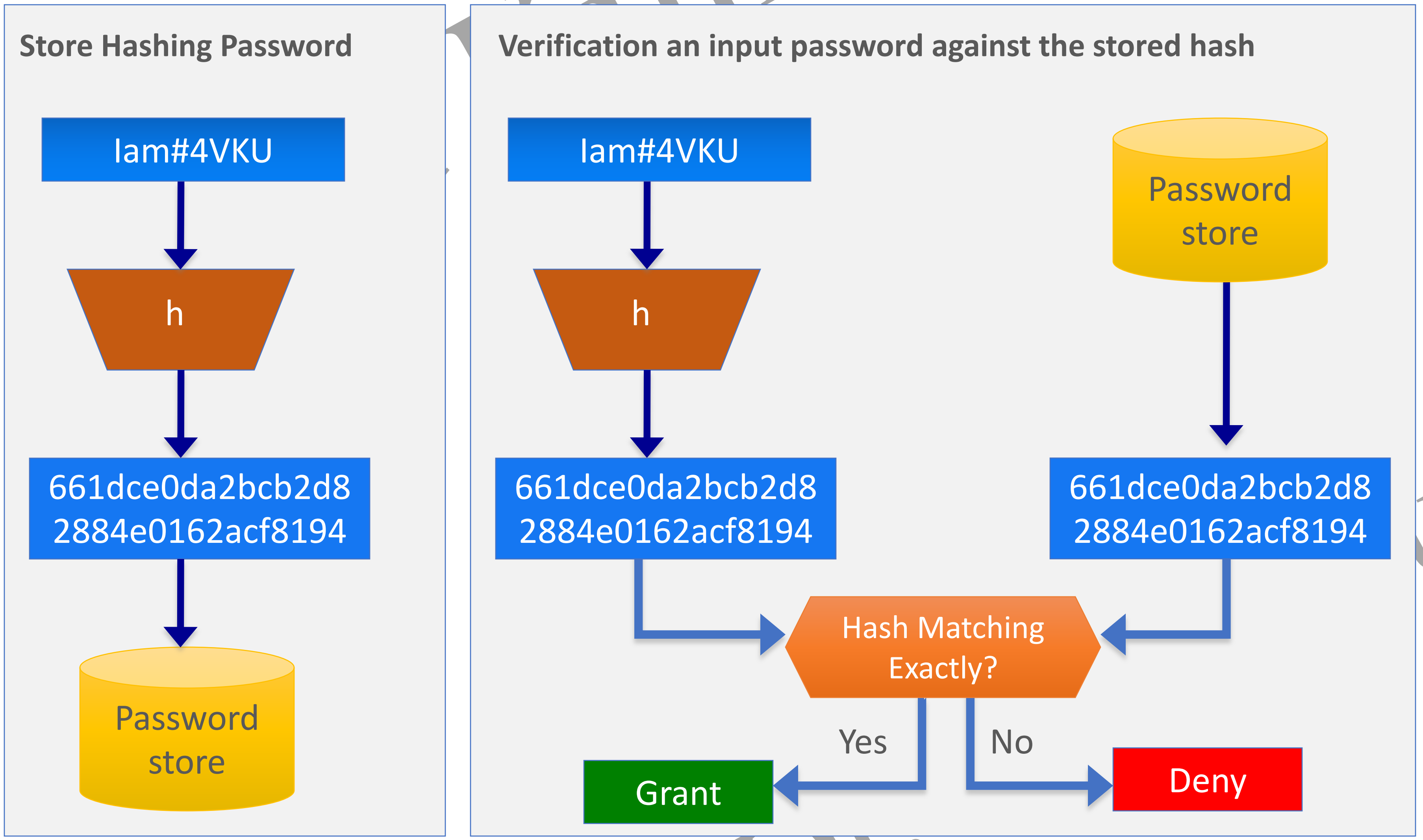


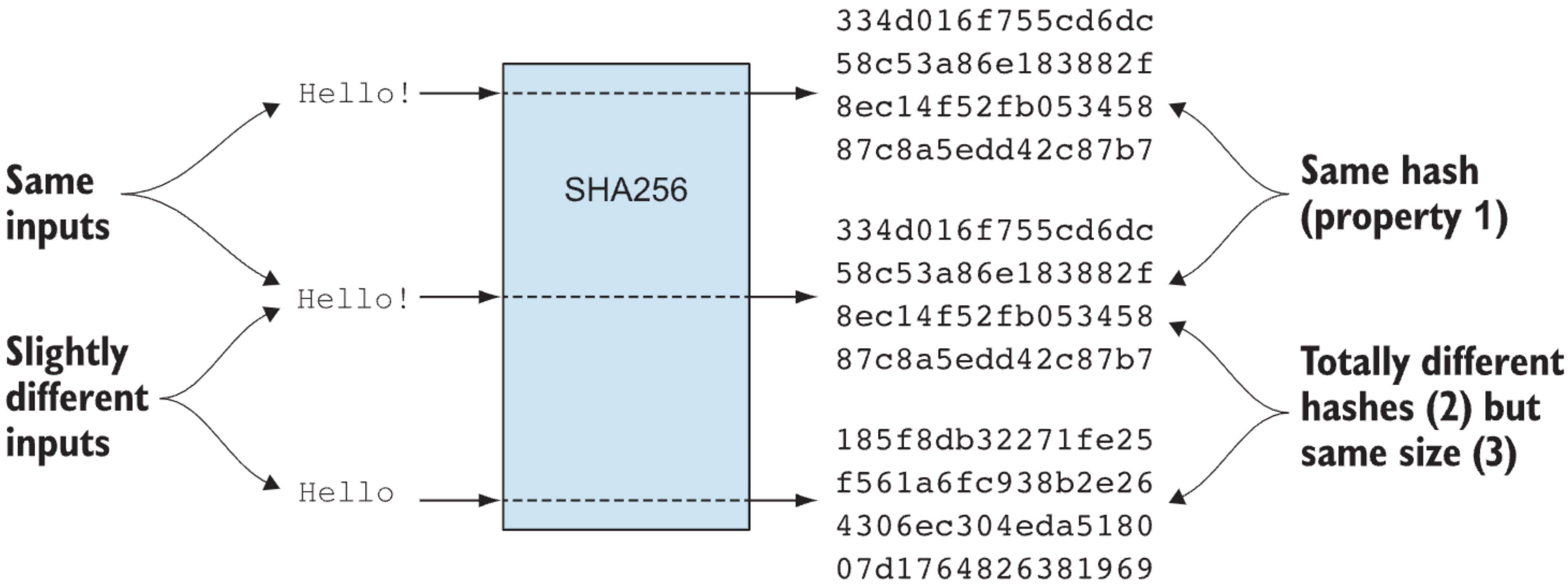
- Encryption is two way, and requires a key to encrypt/decrypt



- Hashing is one-way. There is no 'de-hashing’

Password Verification



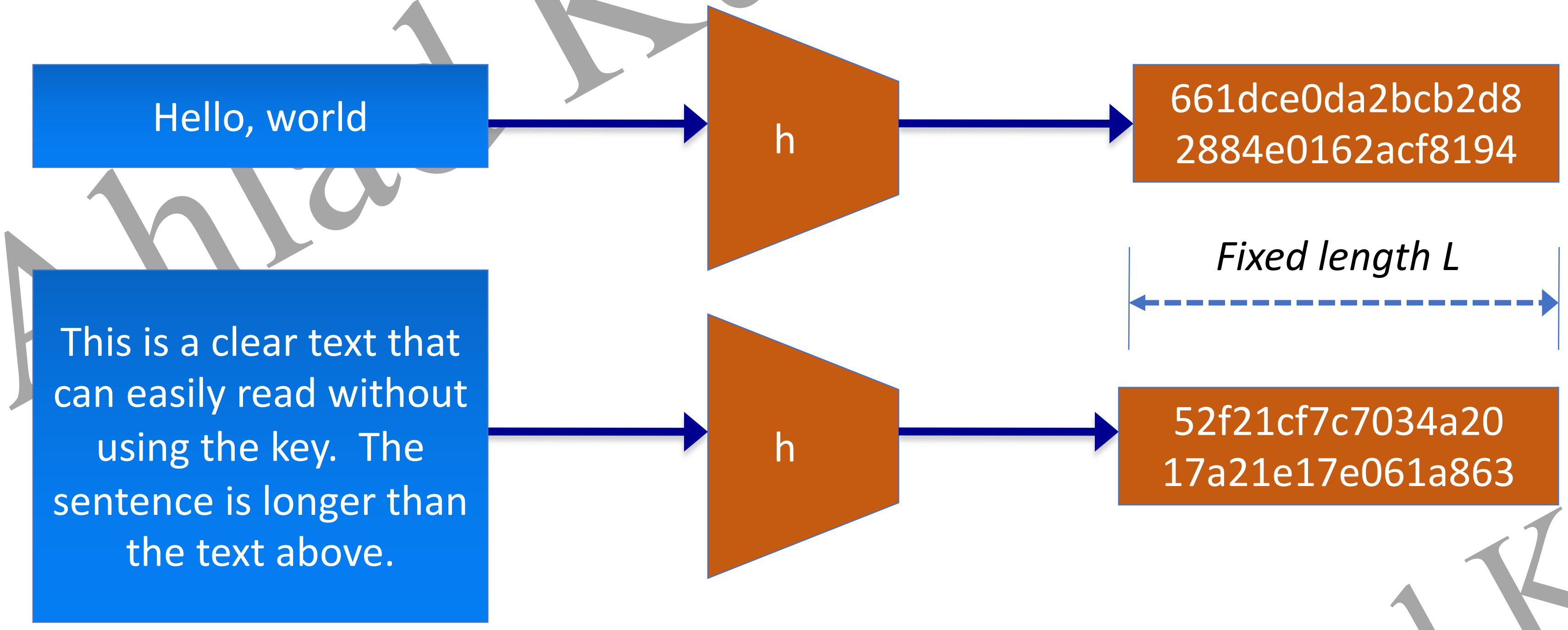


Same inputs
Slightly different inputs
Hello!
Hello!
Hello
SHA256
334d016f755cd6dc
58c53a86e183882f
8ec14f52fb053458
87c8a5edd42c87b7
334d016f755cd6dc
58c53a86e183882f
8ec14f52fb053458
87c8a5edd42c87b7
185f8db32271fe25
f561a6fc938b2e26
4306ec304eda5180
07d1764826381969
Same hash (property 1)
Totally different hashes (2) but same size (3)

Hash Function Properties

- Arbitrary-length message to fixed-length digest
- Preimage resistant (**One-way property**)
- Second preimage resistant (**Weak collision resistant**)
- Collision resistant (**Strong collision resistance**)

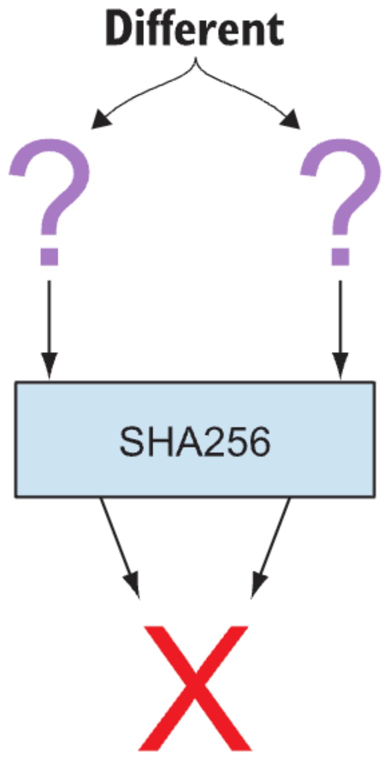
Properties : Fixed length



- Arbitrary-length message to fixed-length digest

Collision resistance

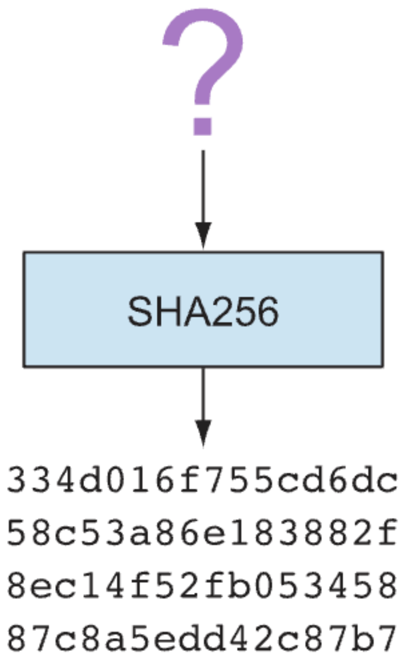
You have only the cryptographic hash function at hand. It's hard to find two *different* inputs that *result in the same hash*.



Collision resistance

Pre-image resistance

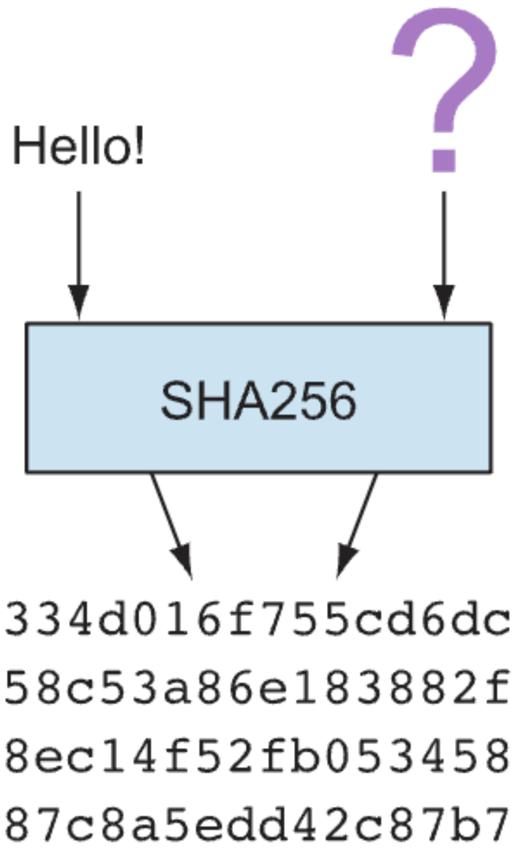
You have the hash function and a hash. It's hard to find *a pre-image of that hash*.



Pre-image resistance

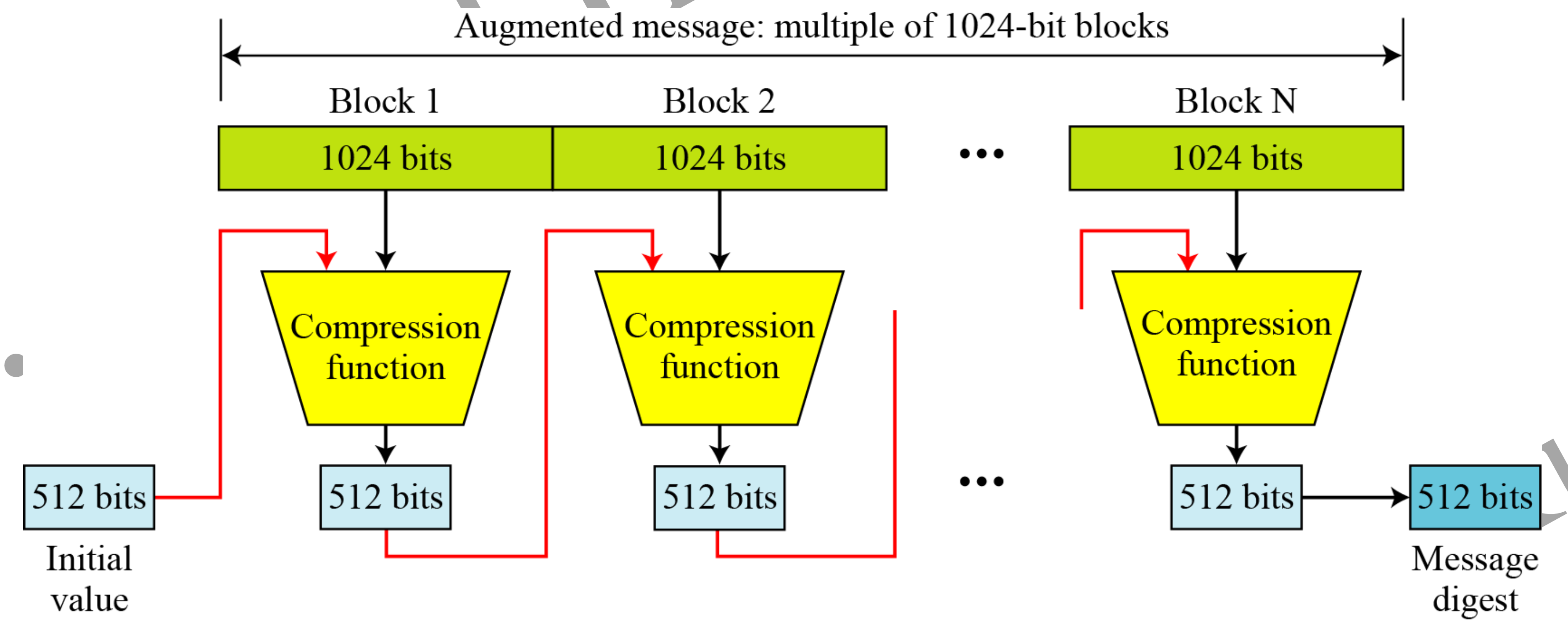
Second-pre-image resistance

You have the hash function and a pre-image (and thus the hash of that pre-image). It's hard to find *another pre-image with the same hash*.

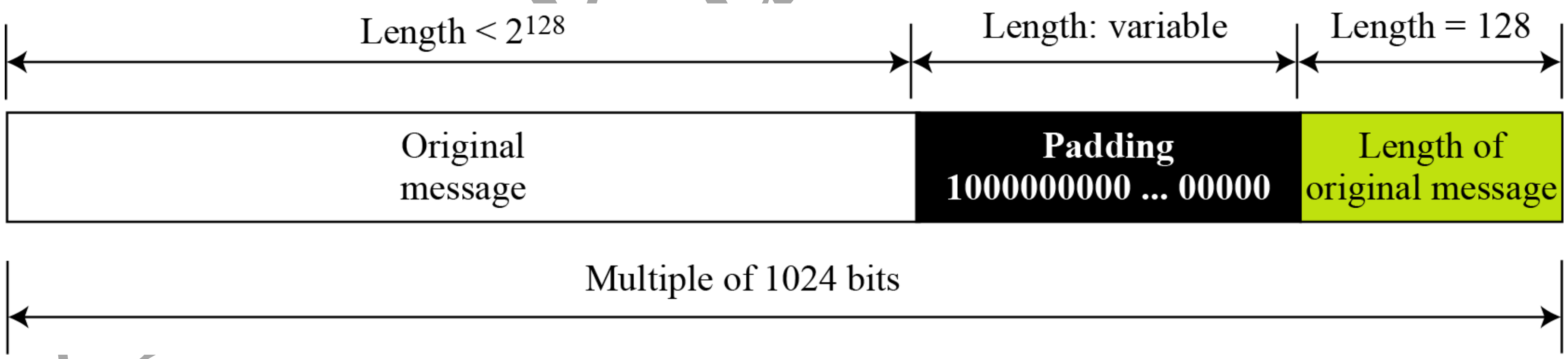


Second-pre-image resistance

SHA-512 Overview



Padding and length field in SHA-512

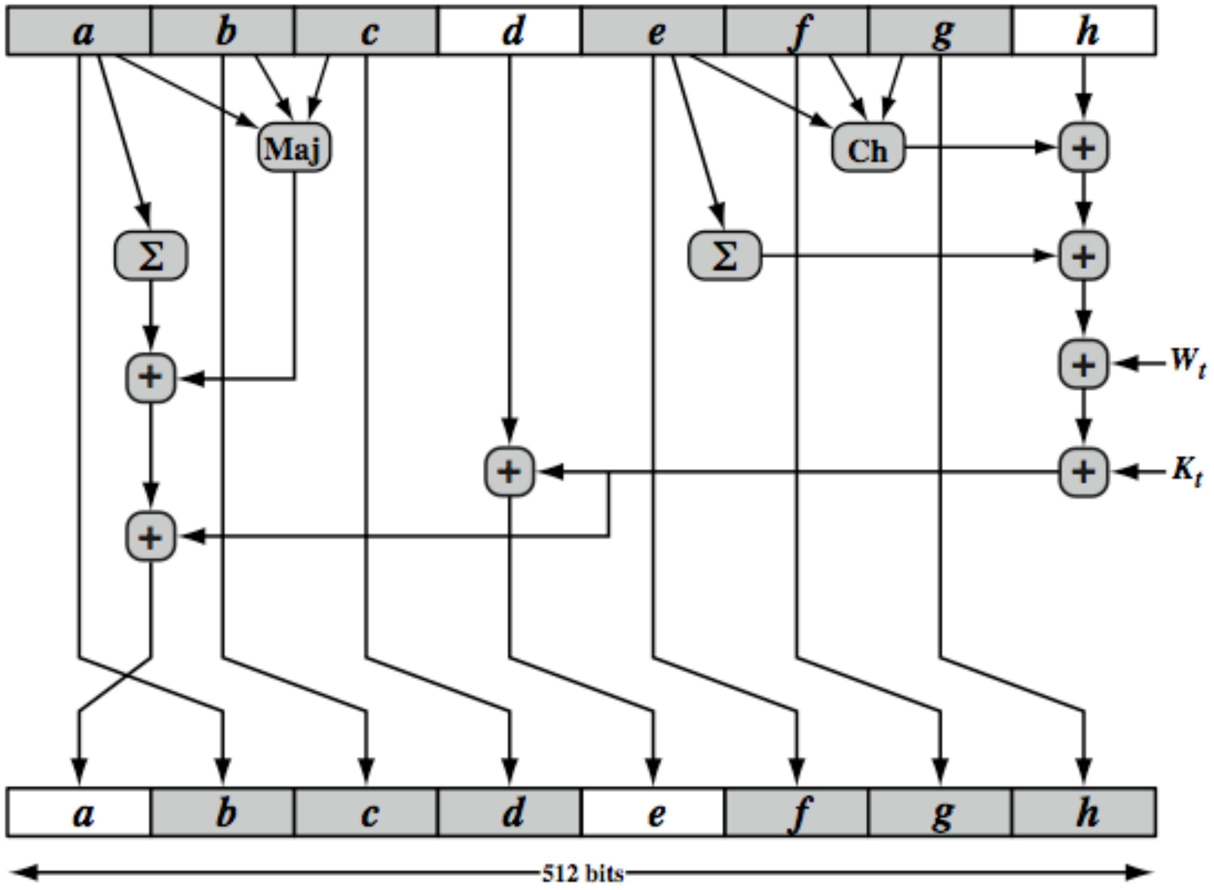


- **What is the number of padding bits if the length of the original message is 2590 bits?**
- We can calculate the number of padding bits as follows:

$$|P| = (-2590 - 128) \bmod 1024 = -2718 \bmod 1024 = 354$$

- The padding consists of one 1 followed by 353 0's.

SHA-512 Round Function



Some well-known hash functions

Name	Bits	Secure so far?	Used in Bitcoin?
SHA256	256	Yes	Yes
SHA512	512	Yes	Yes, in some wallets
RIPEMD160	160	Yes	Yes
SHA-1	160	No. A collision has been found.	No
MD5	128	No. Collisions can be trivially created. The algorithm is also vulnerable to pre-image attacks, but not trivially.	No

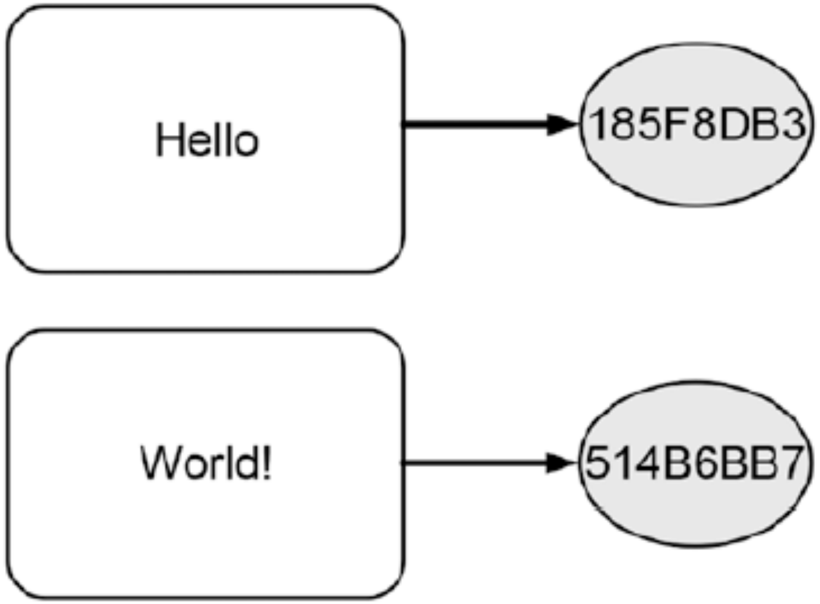
Patterns of Hashing Data

- Independent hashing
- Repeated hashing
- Combined hashing
- Sequential hashing
- Hierarchical hashing

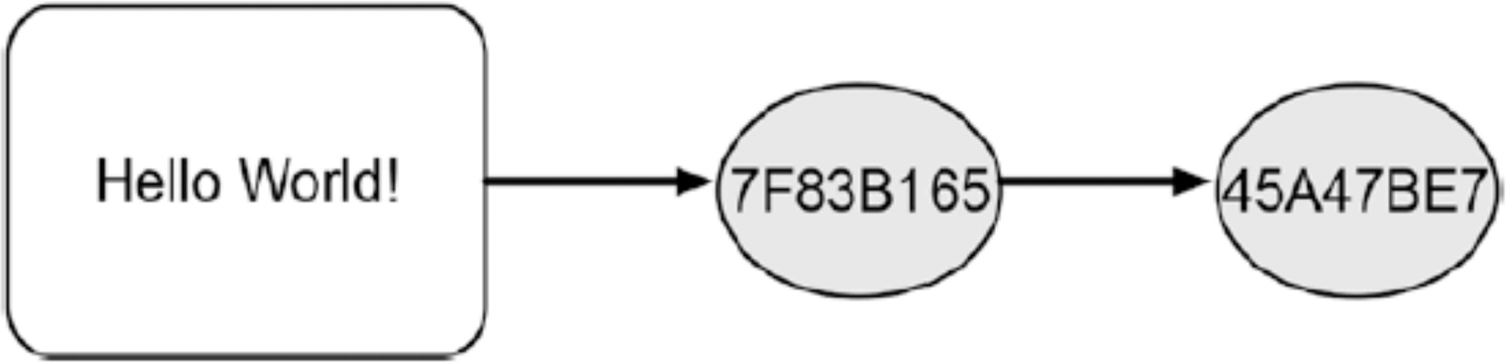
Courtesy: Blockchain Basics: A Non-Technical Introduction in 25 Steps by Daniel Drescher

Types of Hashing

- Independent hashing

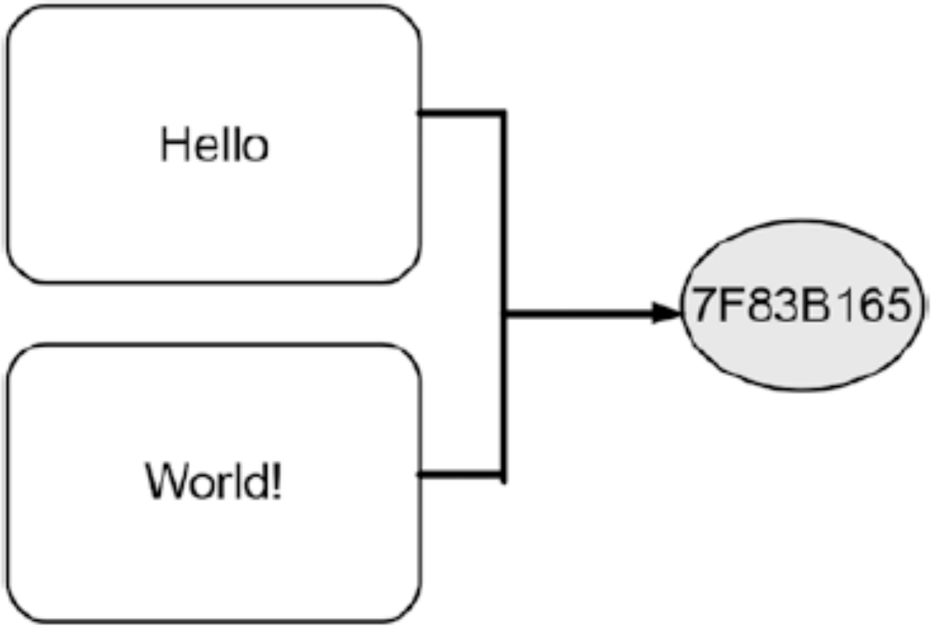


- Repeated hashing

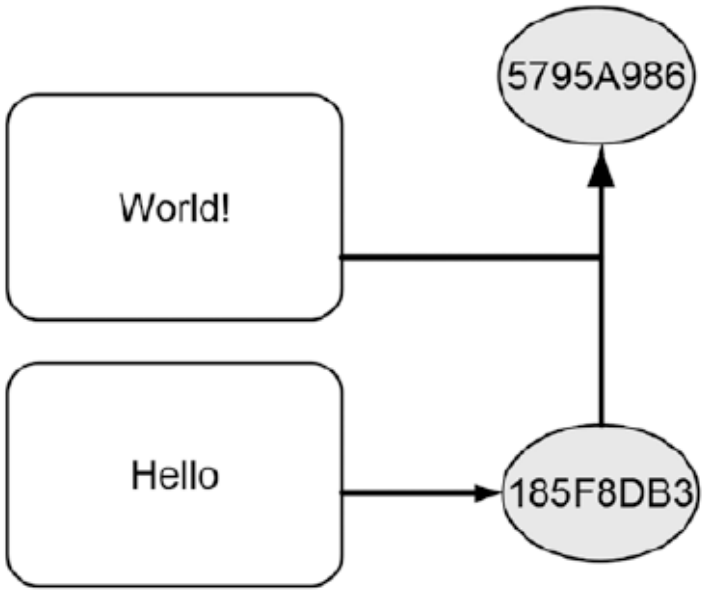


Types of Hashing

- Combined hashing

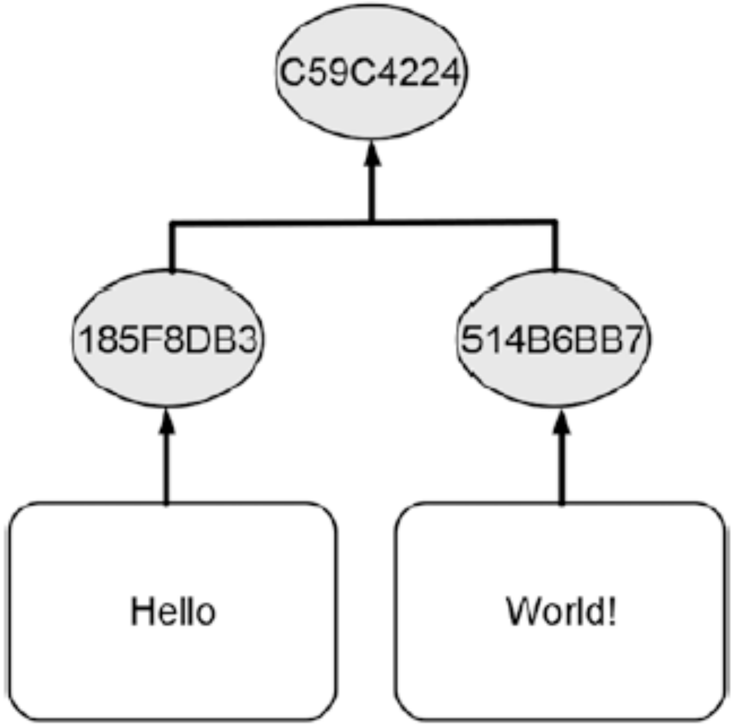


- Sequential hashing



Types of Hashing

- Hierarchical hashing

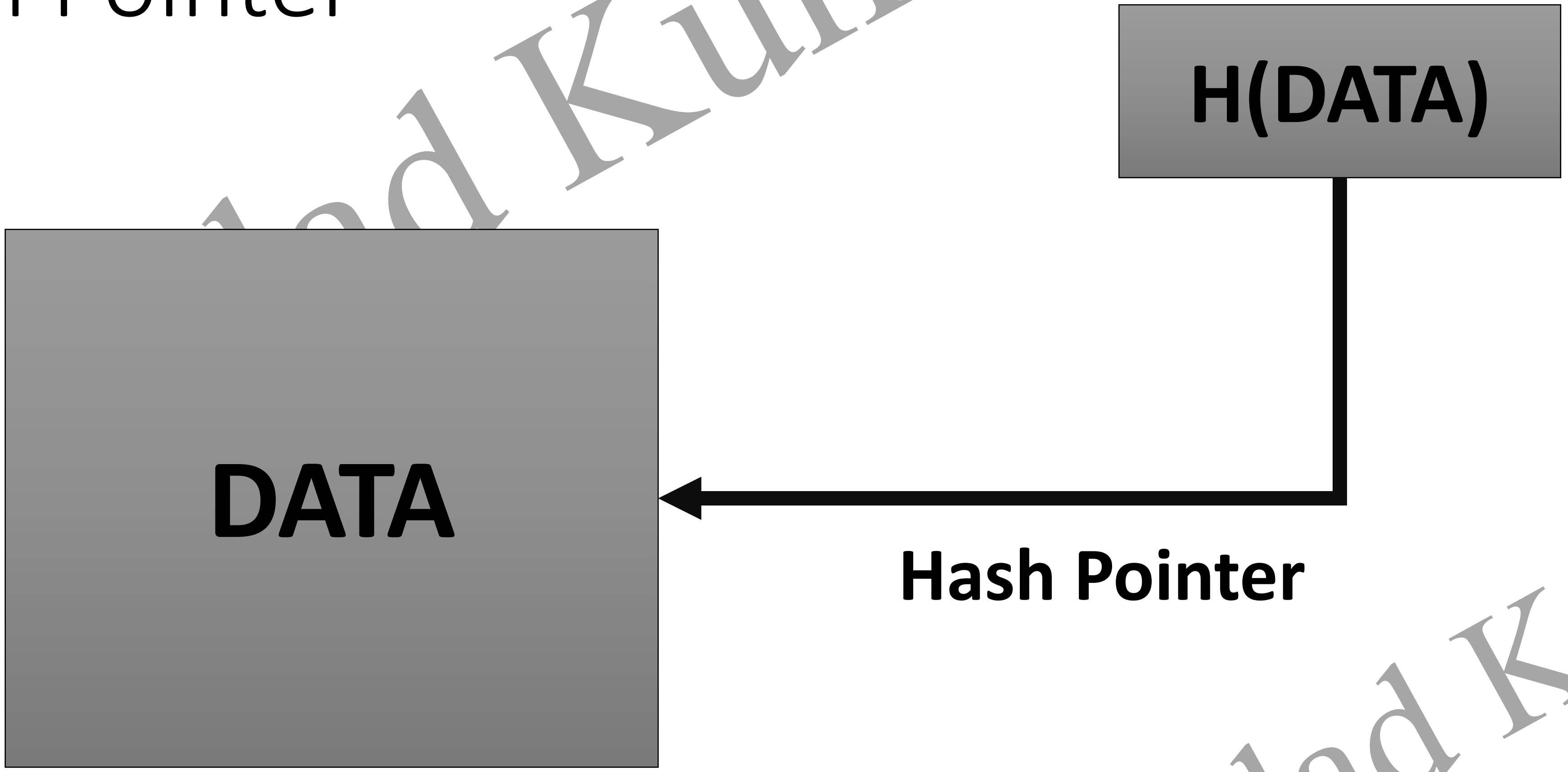


Hash Pointer

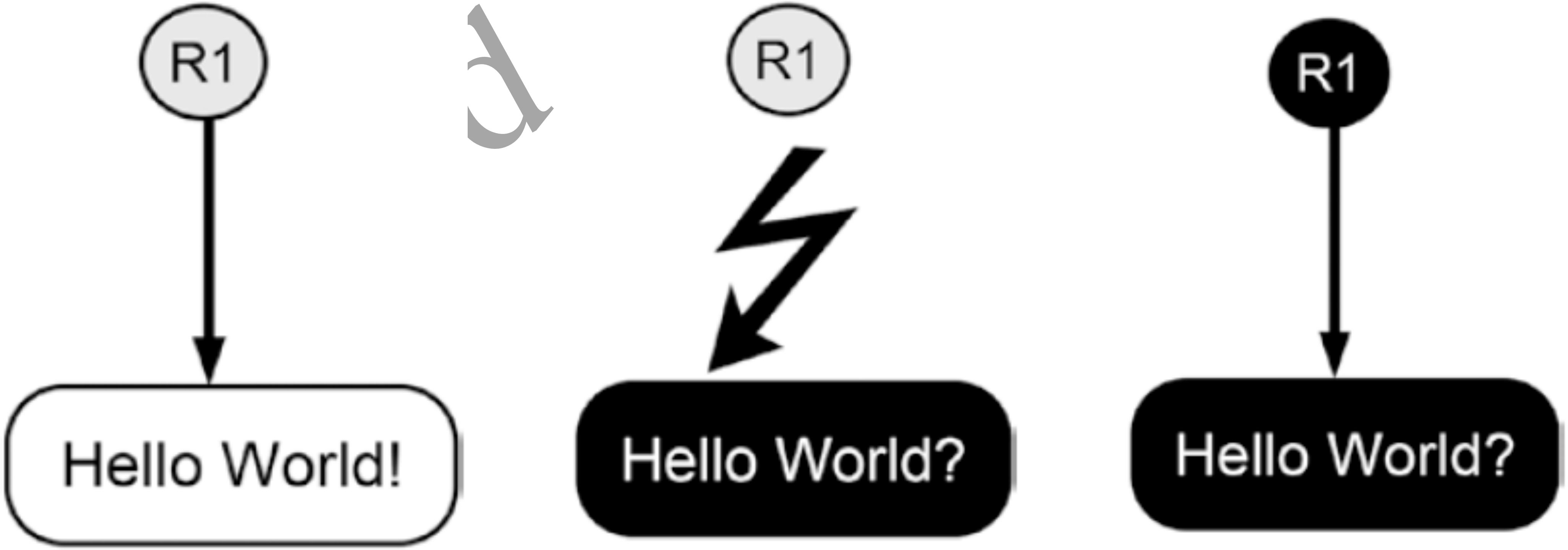
- A **Cryptographic Hash Pointer** (Often called Hash Reference) is a pointer to a location where
  - Some information is stored
  - Hash of the information is stored

- With the hash pointer, we can
  - Retrieve the information
  - Check that the information has not been modified (by computing the message digest and then matching the digest with the stored hash value)

Hash Pointer



Tamper Detection using Hash Pointer



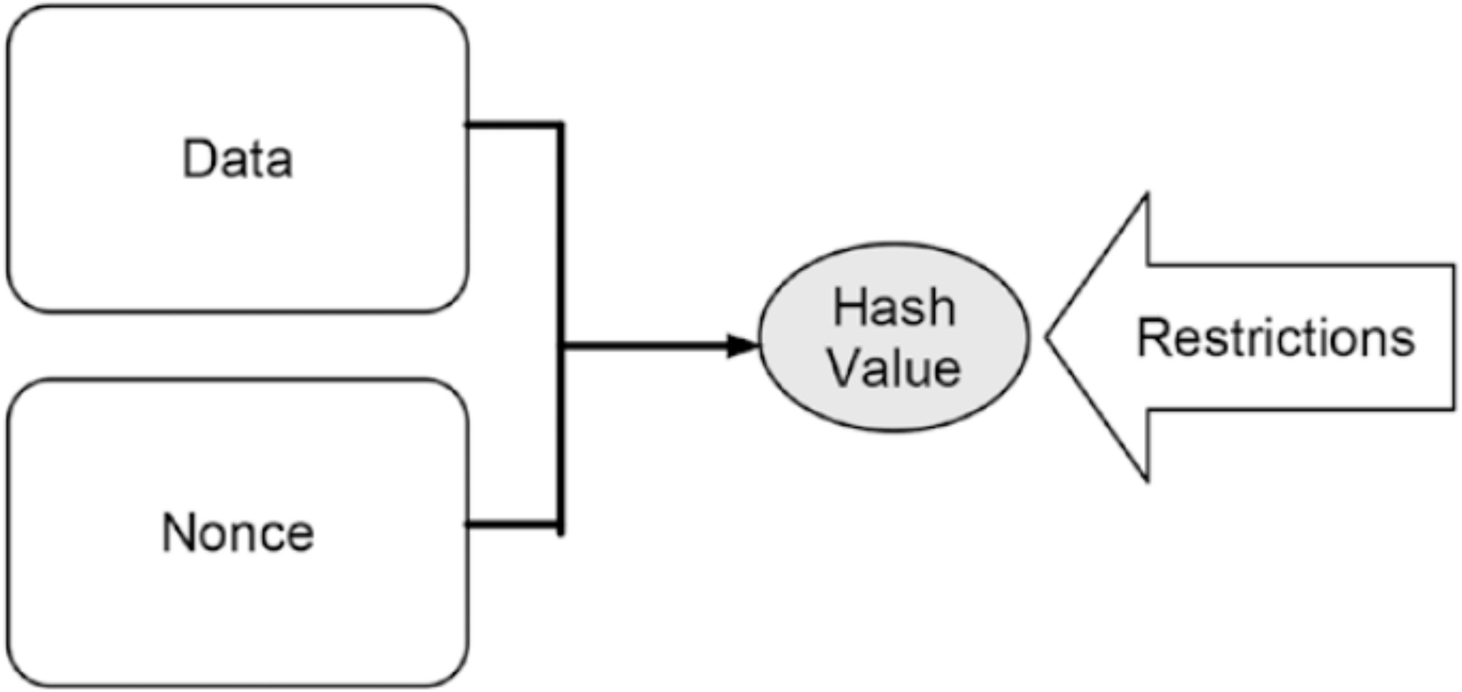
Puzzle Friendly

- Say M is chosen from a widely spread distribution; it is computationally difficult to compute k, such that $Z = H(M||k)$, where M and Z are known a priori.

- **A Search Puzzle** (Used in Bitcoin Mining)
  - M and Z are given, k is the search solution
  - Note: It might be not exactly a particular value Z, but some properties that Z satisfies, i.e., Z could be a set of possible values

- Puzzle friendly property implies that random searching is the best strategy to solve the above puzzle

Making Tampering a Hash Chain Computationally Challenging



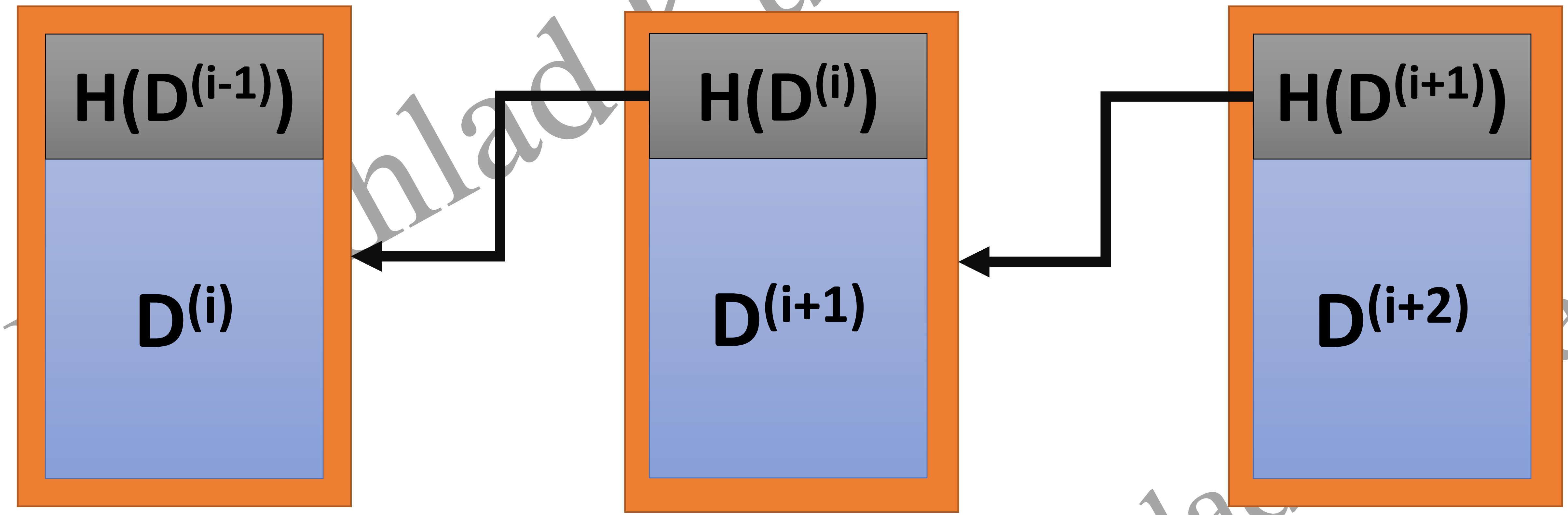
http://www.blockchain-basics.com/HashFunctions.html

Nonces for Solving a Hash Puzzle

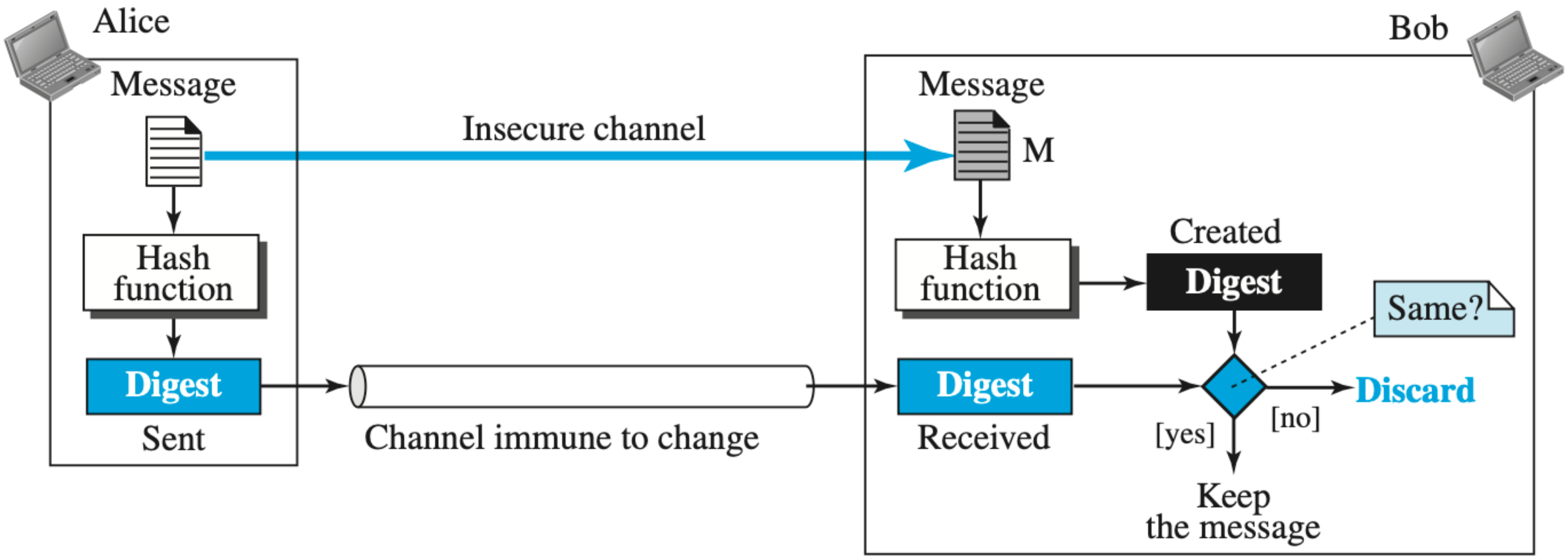
Nonce	Text to Be Hashed	Output
0	Hello World! 0	4EE4B774
1	Hello World! 1	3345B9A3
2	Hello World! 2	72040842
3	Hello World! 3	02307D5F
	...	
613	Hello World! 613	E861901E
614	Hello World! 614	**00068A3C**
615	Hello World! 615	5EB7483F

Courtesy: Blockchain Basics: A Non-Technical Introduction in 25 Steps by Daniel Drescher

Detect Tampering from Hash Pointers - Hashchain

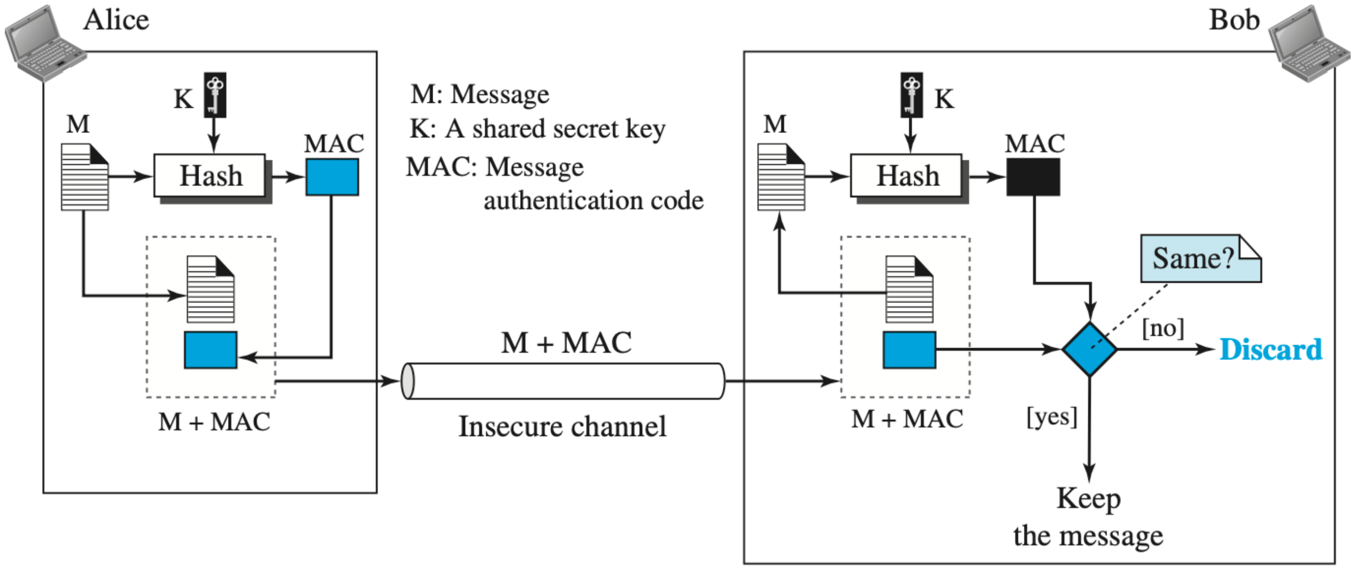


Message Integrity

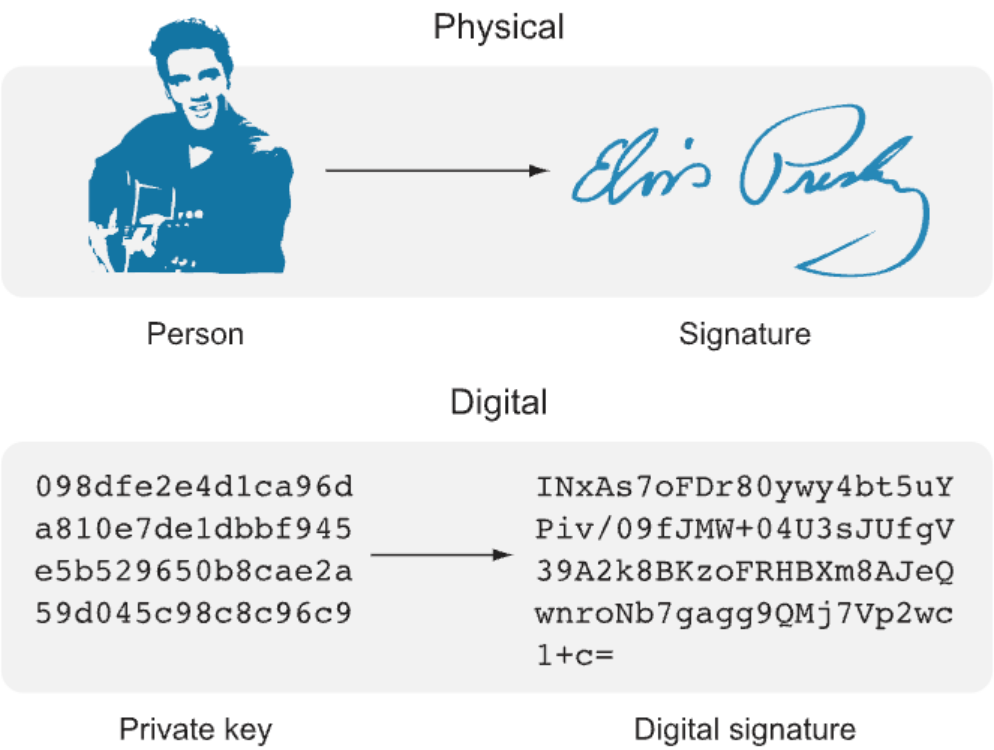


Message Authentication

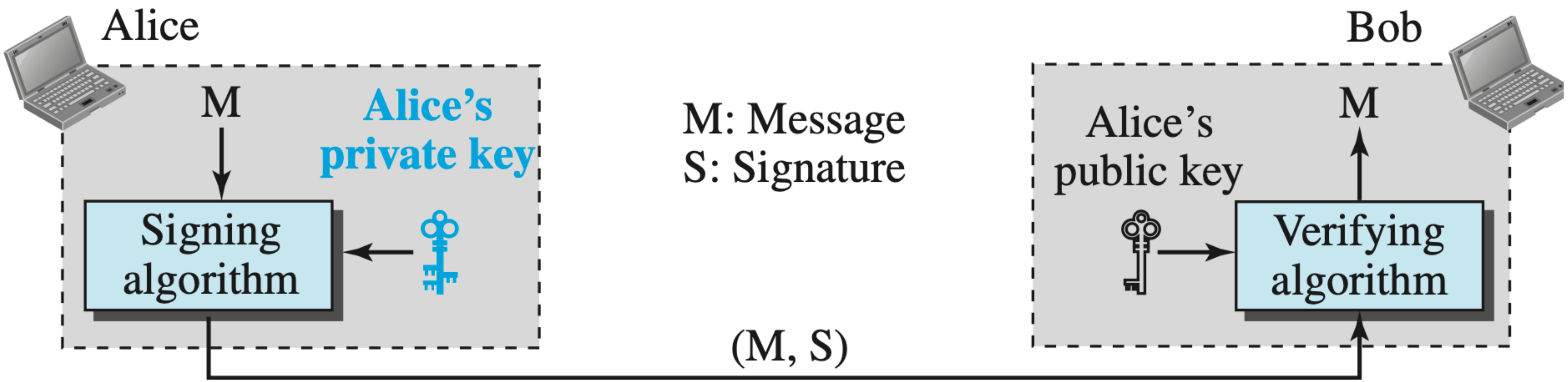
- Alice uses a hash function to create a MAC from the concatenation of the key and the message, $h(K + M)$.
- She sends the message and the MAC to Bob over the insecure channel.
- Bob separates the message from the MAC.
- He then makes a new MAC from the concatenation of the message and the secret key.
- Bob then compares the newly created MAC with the one received. If the two MACs match, the message is authentic and has not been modified by an adversary.
- Note that there is no need to use two channels in this case.
- Both the message and the MAC can be sent on the same insecure channel.
- Eve can see the message, but she cannot forge a new message to replace it because Eve does not possess the secret key between Alice and Bob. She is unable to create the same MAC that Alice did.

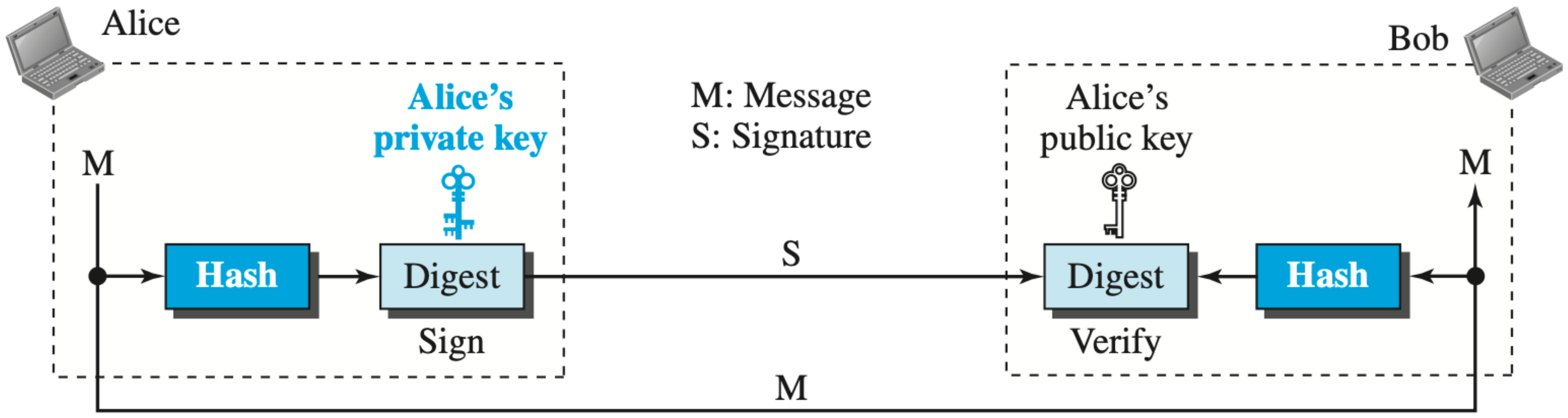


Digital Signatures

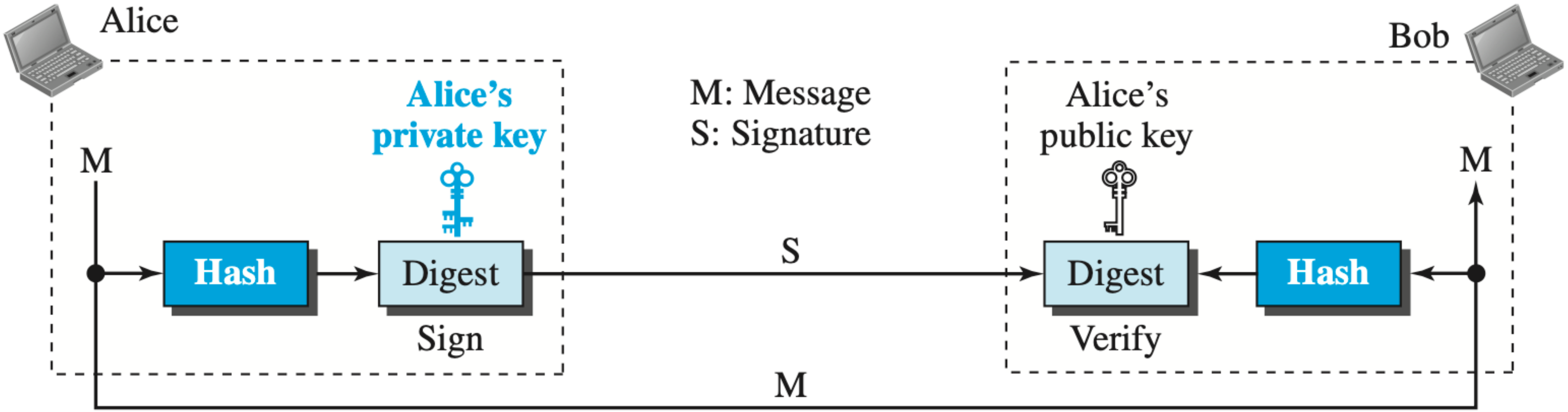


Digital Signature



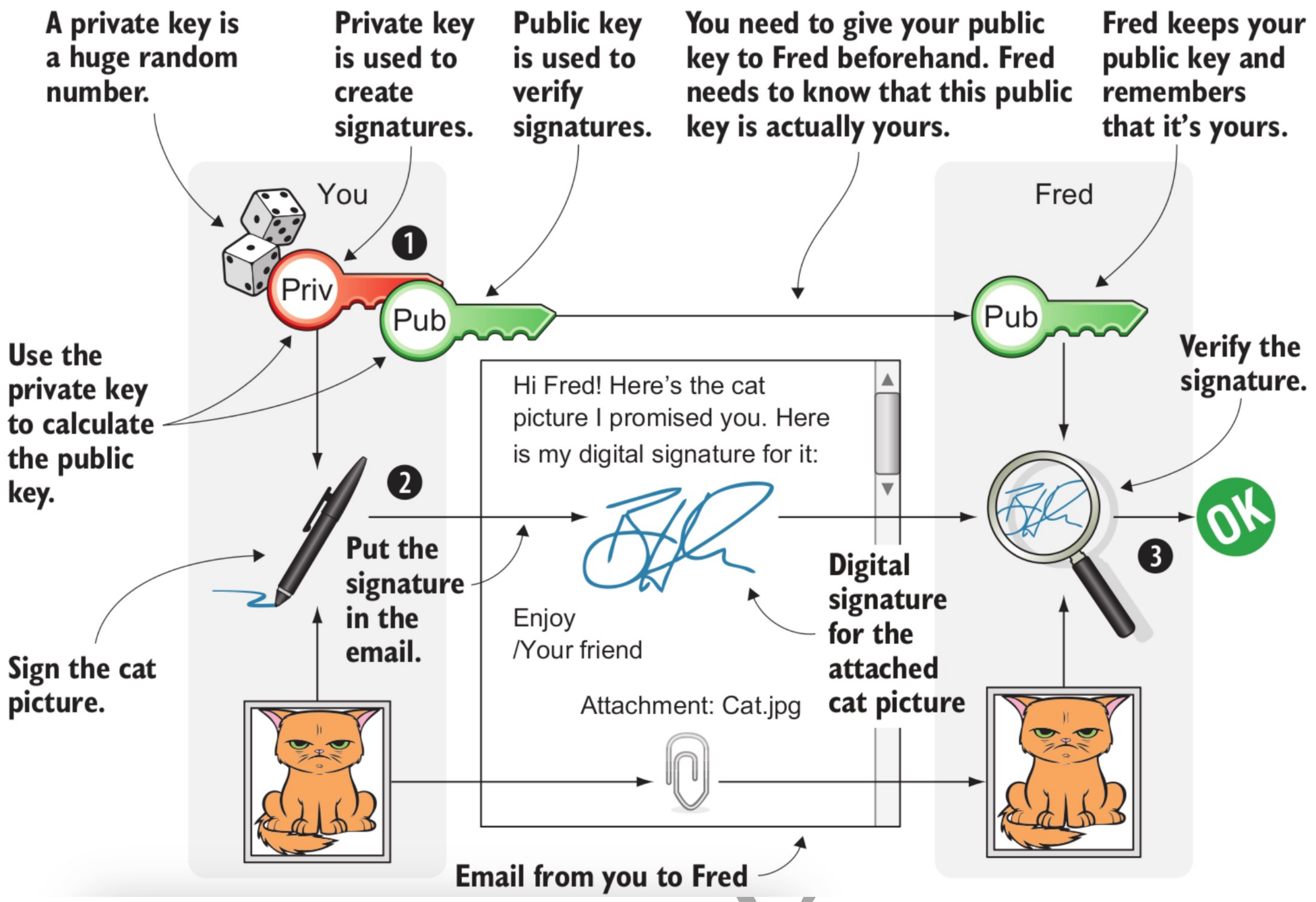


Digital Signature

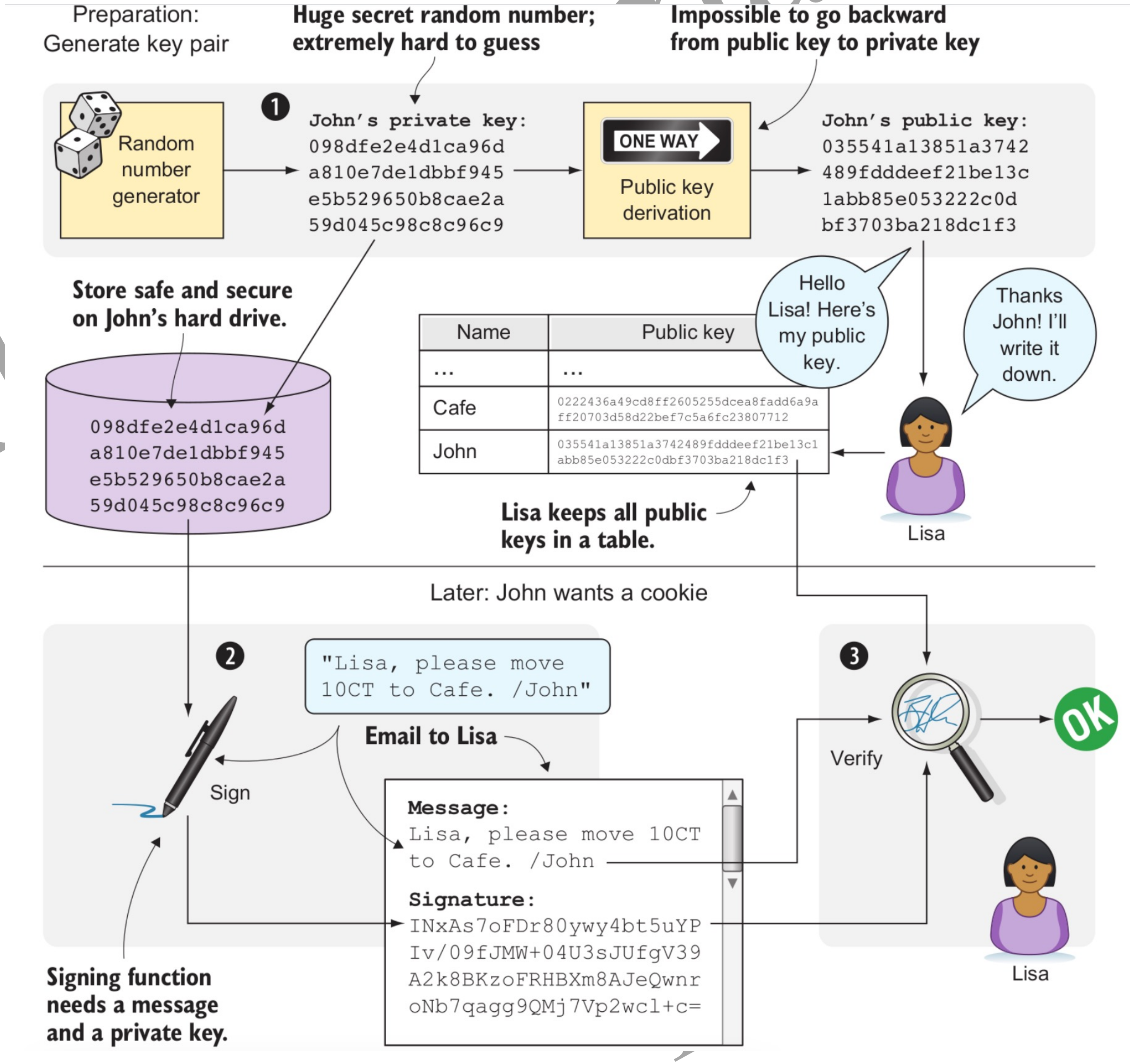


- A digital signature needs a public-key system. The signer signs with her private key; the verifier verifies with the signer's public key.
- A cryptosystem uses the public and private keys of the receiver; a digital signature uses the private and public keys of the sender.

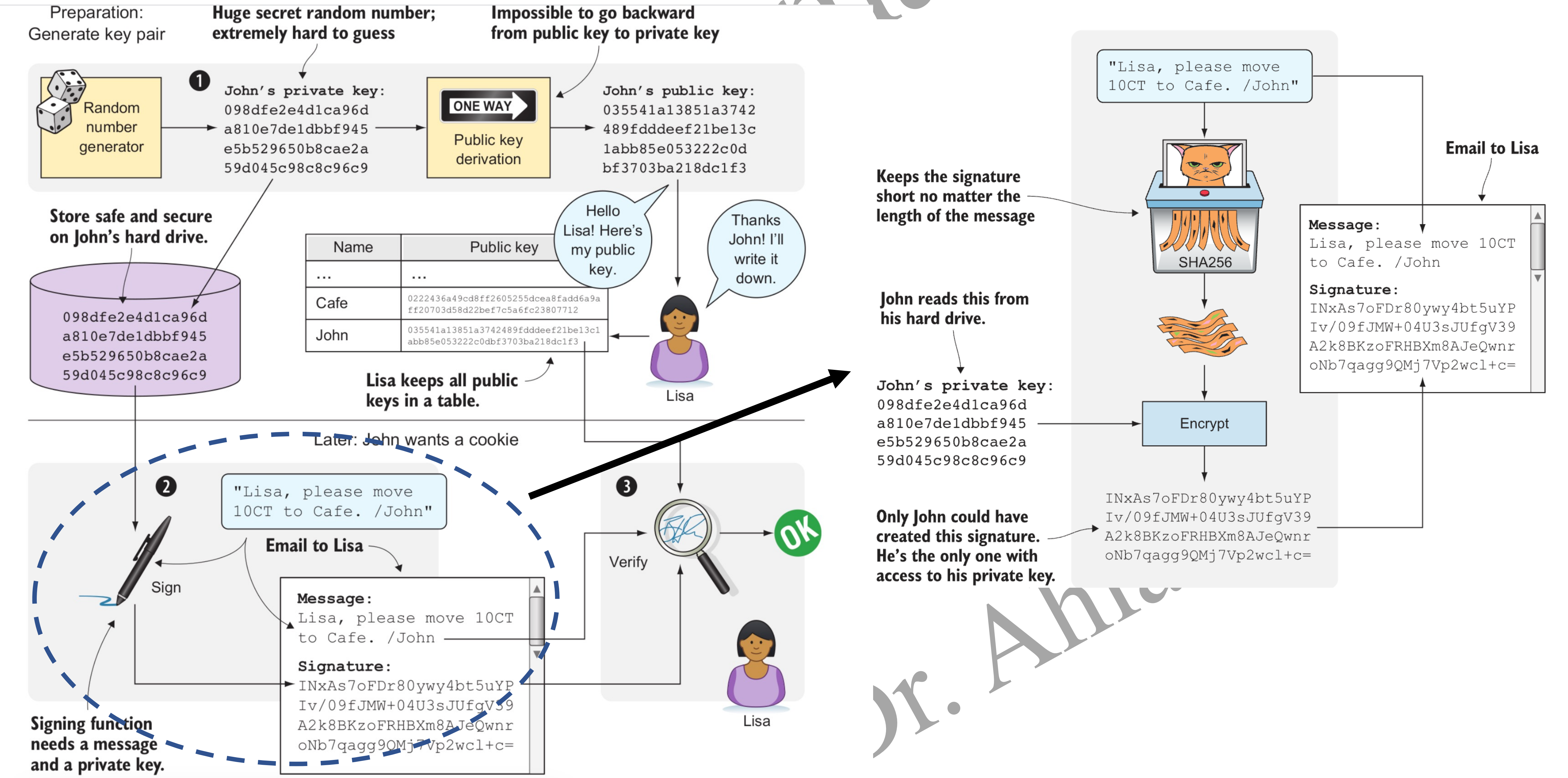
Digital Signatures



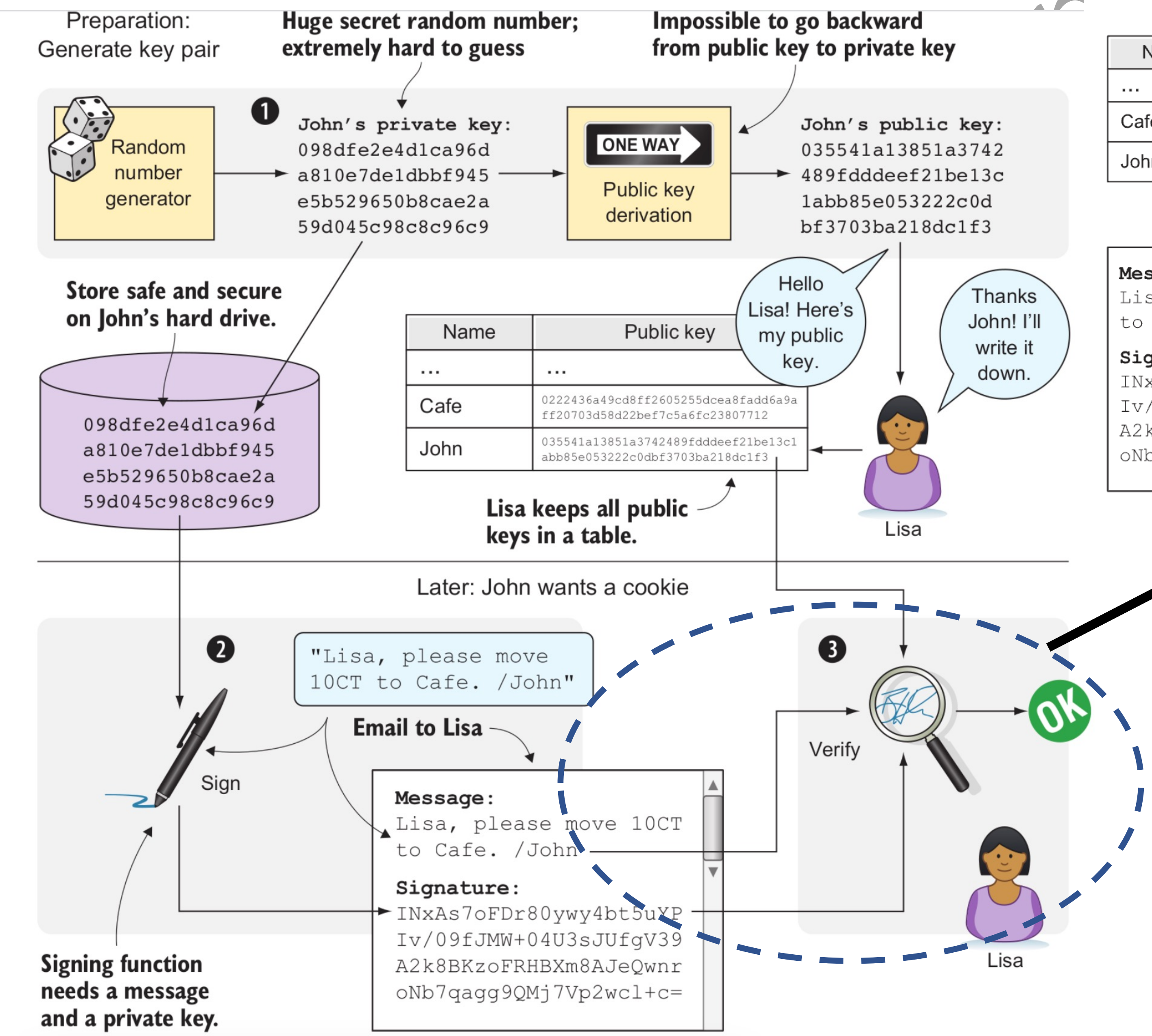
Digital Signatures

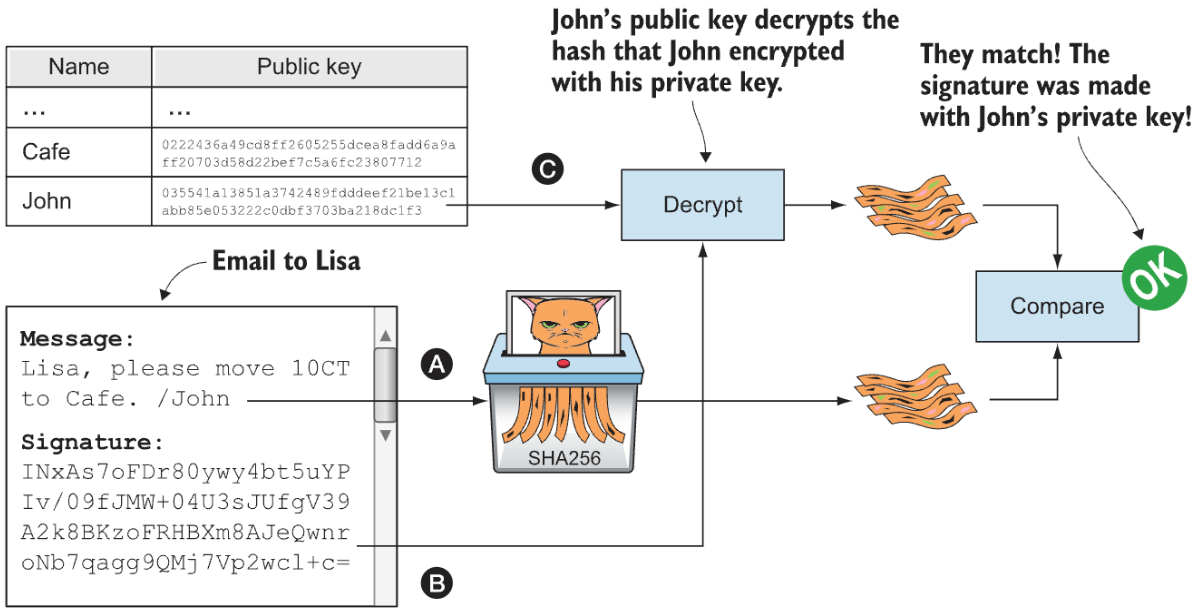


Digital Signatures

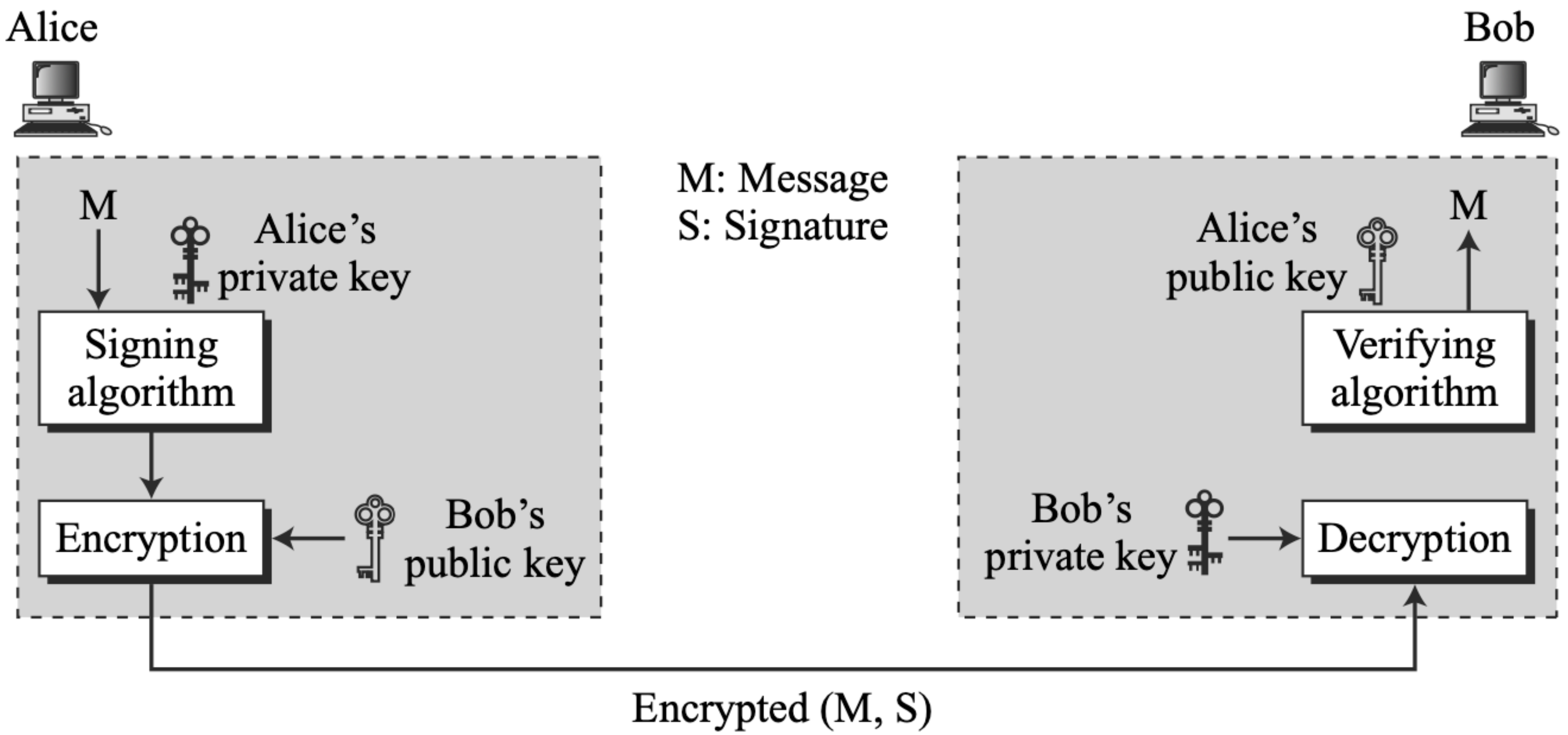


Digital Signatures





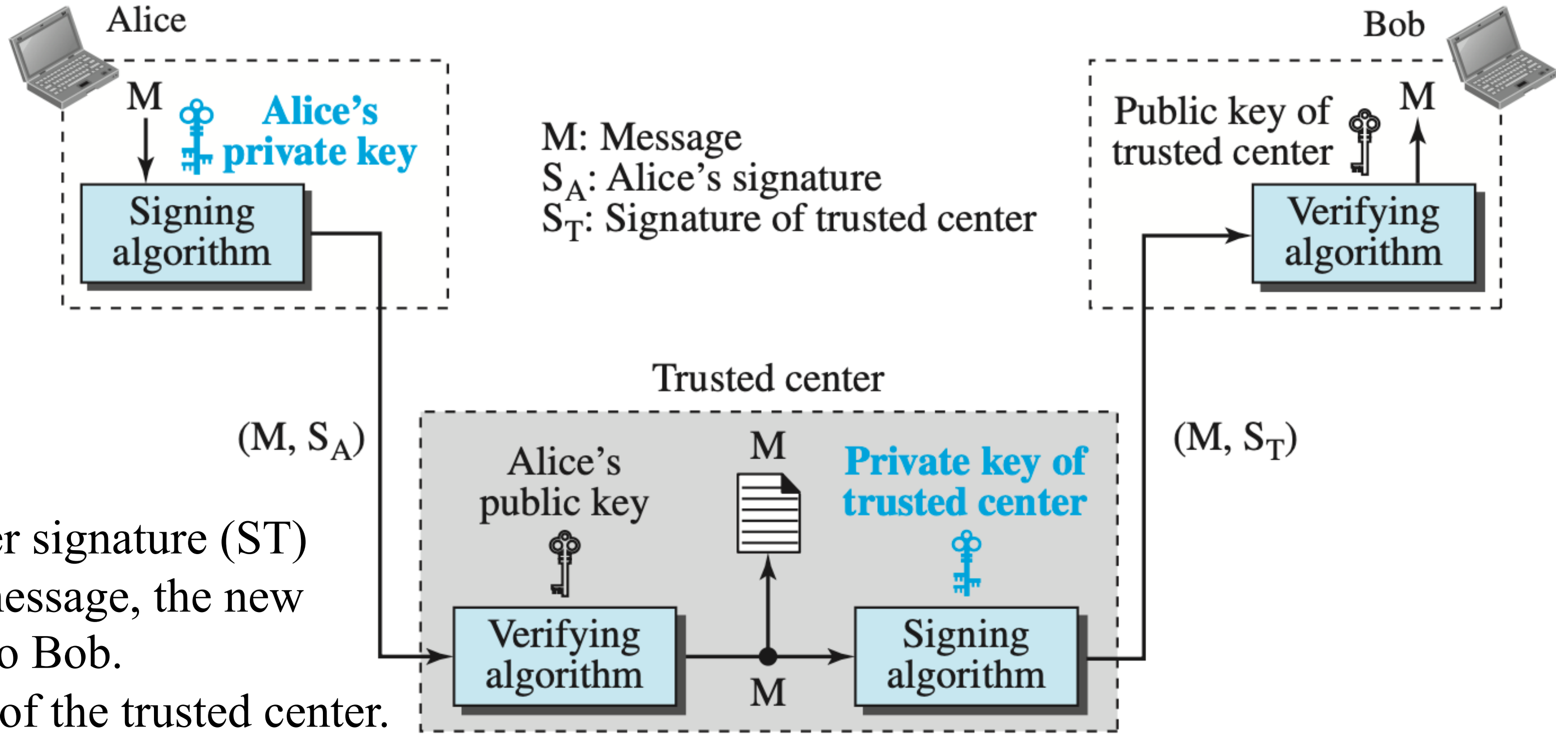
Message Confidentiality



A digital signature does not provide privacy.
If there is a need for privacy, another layer of encryption/decryption must be applied.

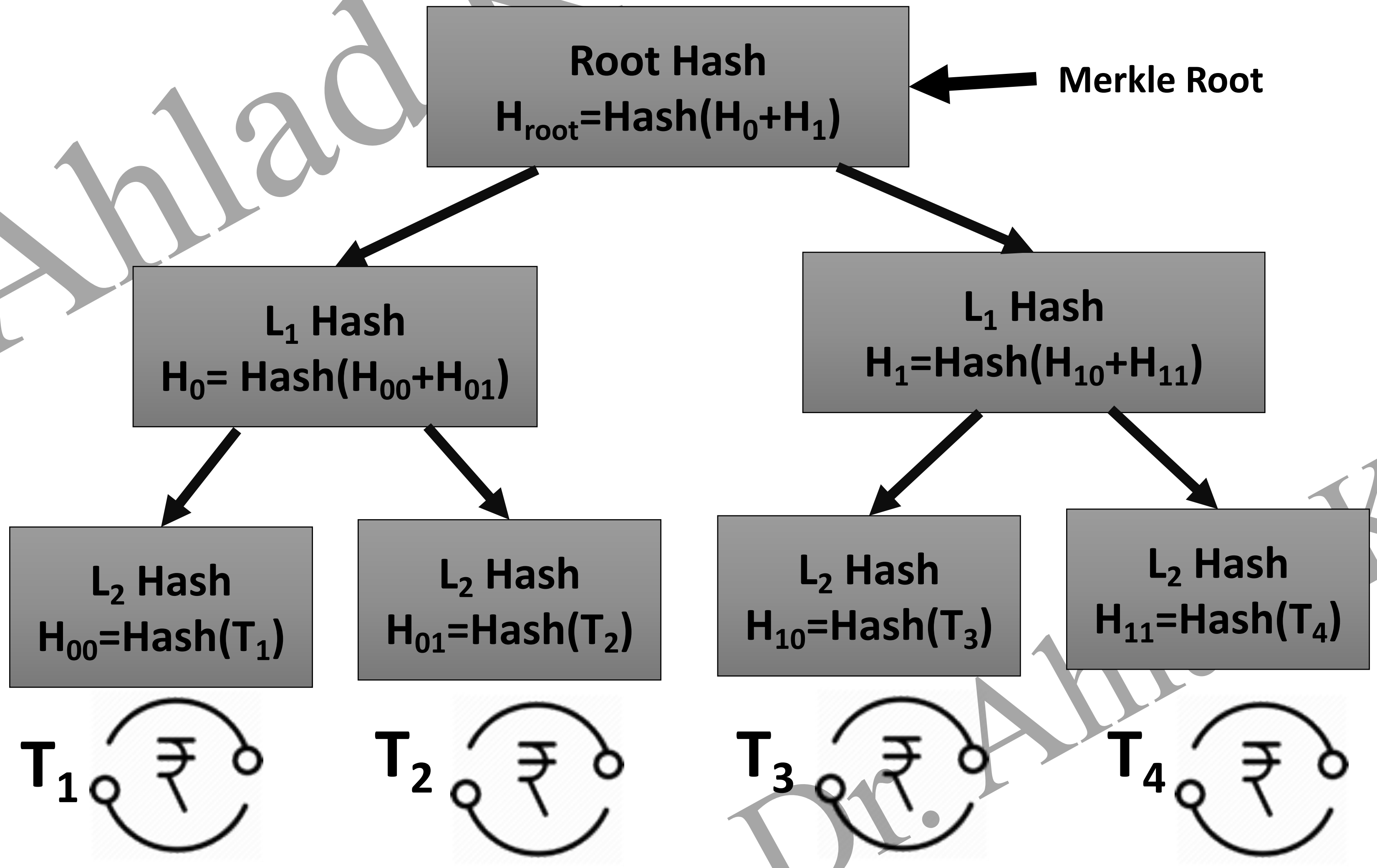
Message Non-Repudiation

- Alice creates a signature from her message (SA) and sends the message, her identity, Bob's identity, and the signature to the center.
- The center, after checking that Alice's public key is valid, verifies through Alice's public key that the message came from Alice.
- The center then saves a copy of the message with the sender's identity, recipient's identity, and a timestamp in its archive.

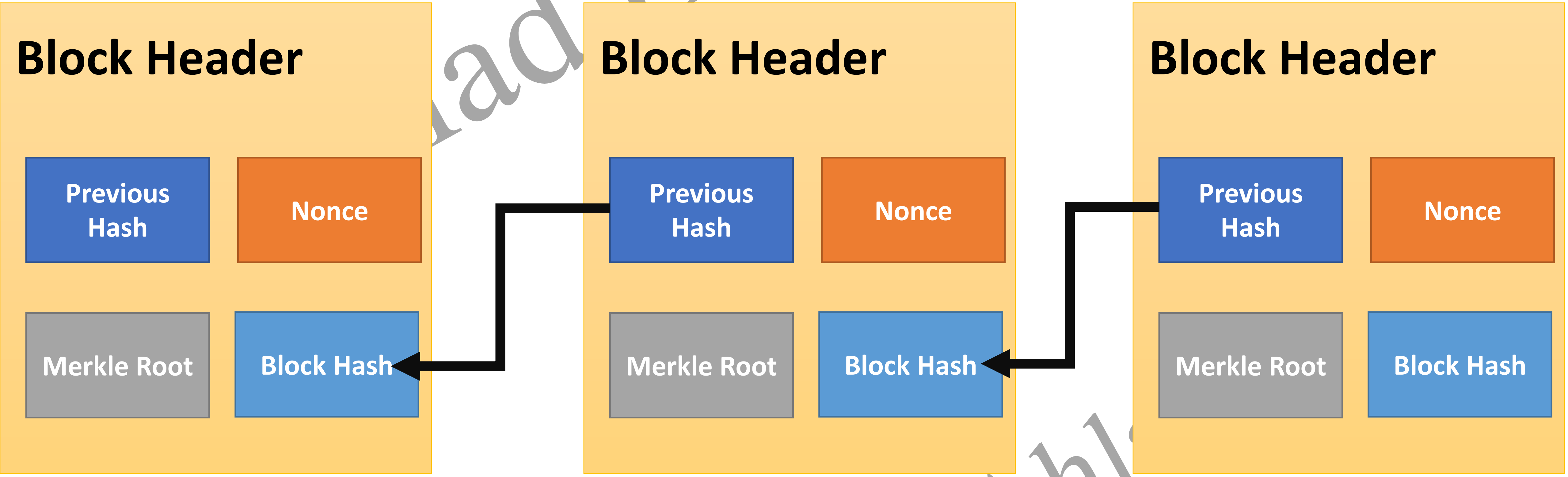


- The center uses its private key to create another signature (ST) from the message. The center then sends the message, the new signature, Alice's identity, and Bob's identity to Bob.
- Bob verifies the message using the public key of the trusted center.
- If in the future Alice denies that she sent the message, the center can show a copy of the saved message. If Bob's message is a duplicate of the message saved at the center, Alice will lose the dispute.

Merkle Tree – Organization of Hash Pointers in a Tree



Blockchain as a Hashchain



Demo

- http://www.blockchain-basics.com/HashFunctions.html

- https://andersbrownworth.com/blockchain/blockchain

- https://andersbrownworth.com/blockchain/public-private-keys/keys