Objectives

- Encryption techniques
- Data Encryption - Overview
- Symmetric Encryption
- Asymmetric Encryption
- Digital signature
- User Authentication

Encryption Techniques

- **Data Encryption**

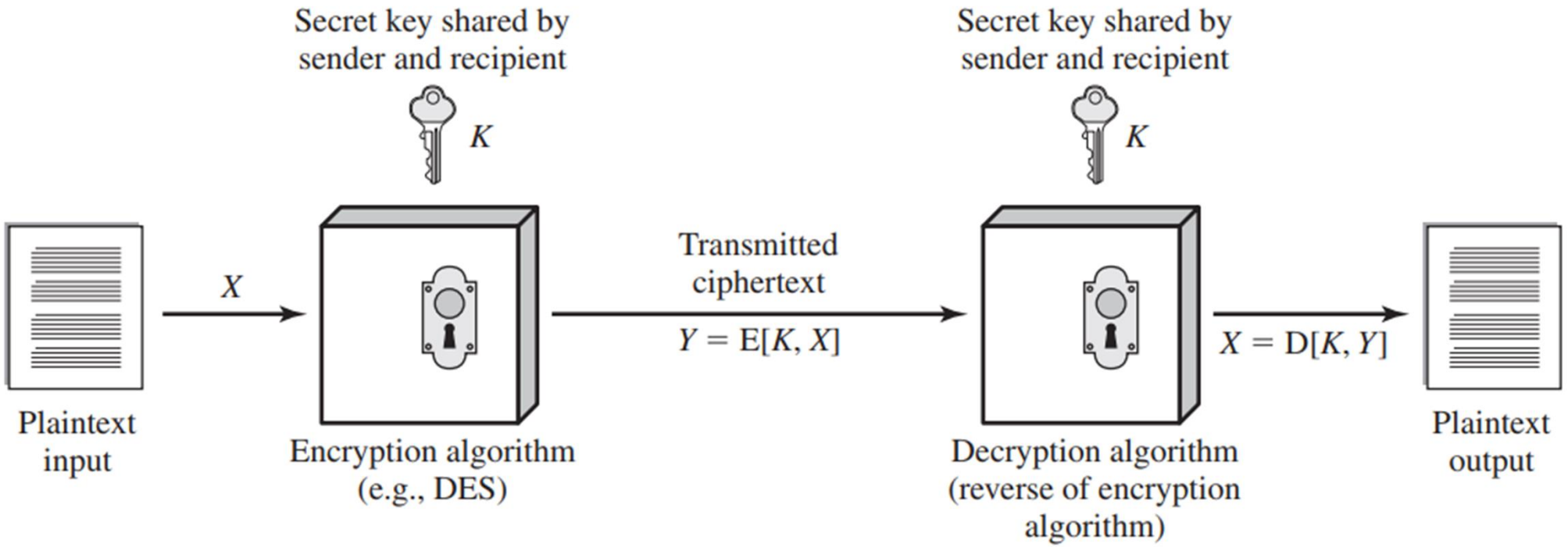
✔ Symmetric Encryption

✔ Asymmetric Encryption

Symmetric Encryption

- Also known as conventional encryption or single-key encryption.
- Used to achieve data confidentiality.

Symmetric Encryption



- **There are two requirements for secure use of symmetric encryption:**

1. We need a strong encryption algorithm.

✔ An opponent who knows the algorithm and has access to one or more ciphertexts would be unable to decipher the ciphertext or figure out the key.

2. Sender and receiver must have obtained copies of the secret key in a secure fashion and must keep the key secure.

- **Two general approaches to attacking a symmetric encryption**

1. *Cryptanalysis:* Exploit knowledge of *algorithm + pairs of plain-text and cipher-text*

2. *Brute-force Attack:* try every possible key on a piece of ciphertext.

- The most commonly used symmetric encryption algorithms are block ciphers.
- Processes the plaintext input in fixed-size blocks and produces a block of ciphertext of equal size for each plaintext block.
- ***Examples:*** Data Encryption Standard (DES), triple DES, and the Advanced Encryption Standard (AES)

	DES	Triple DES	AES
Plaintext block size (bits)	64	64	128
Ciphertext block size (bits)	64	64	128
Key size (bits)	56	112 or 168	128, 192, or 256

DES = Data Encryption Standard
AES = Advanced Encryption Standard

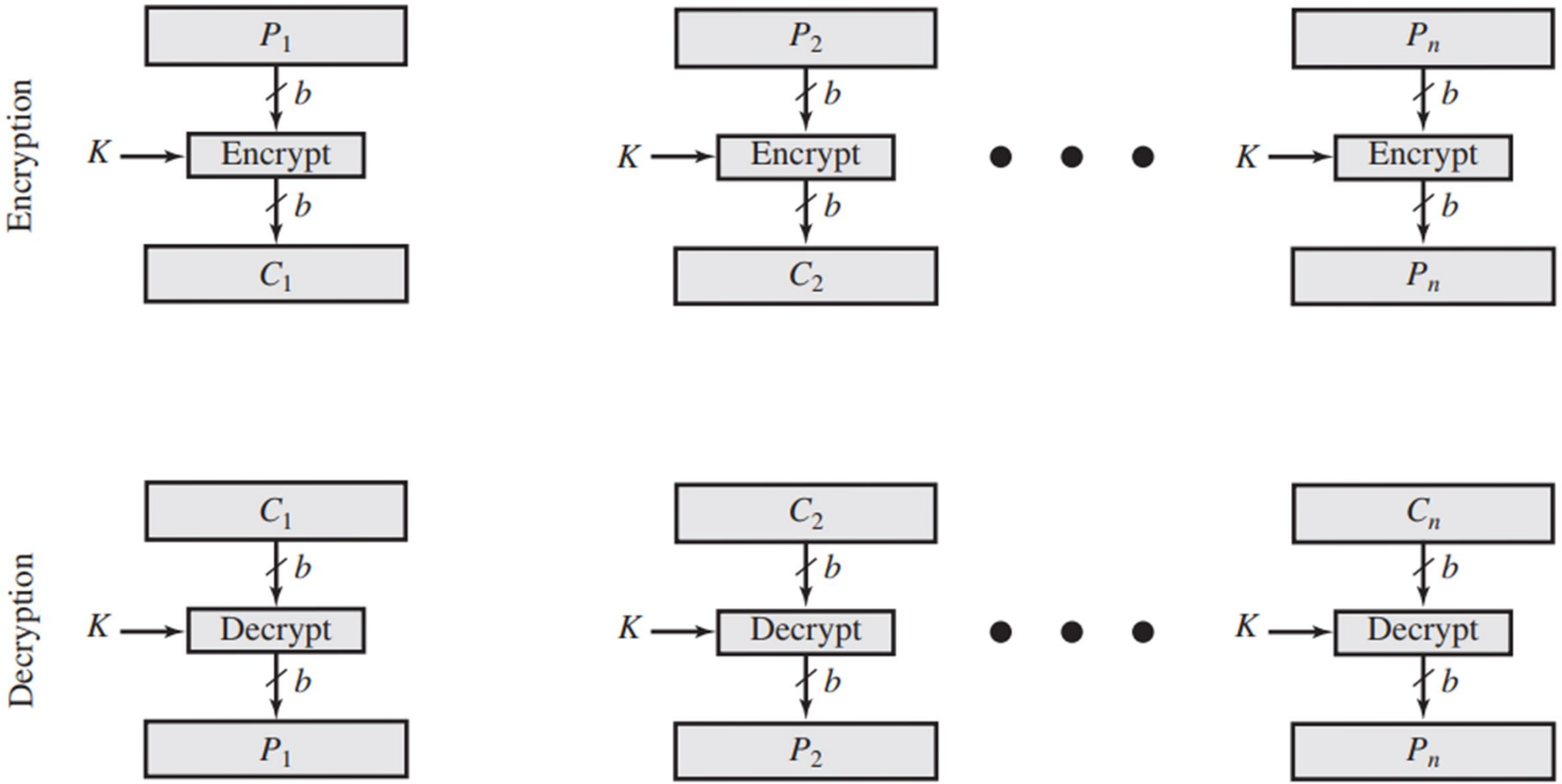
Comparison of Three Popular Symmetric Encryption Algorithms

Average Time Required for Exhaustive Key

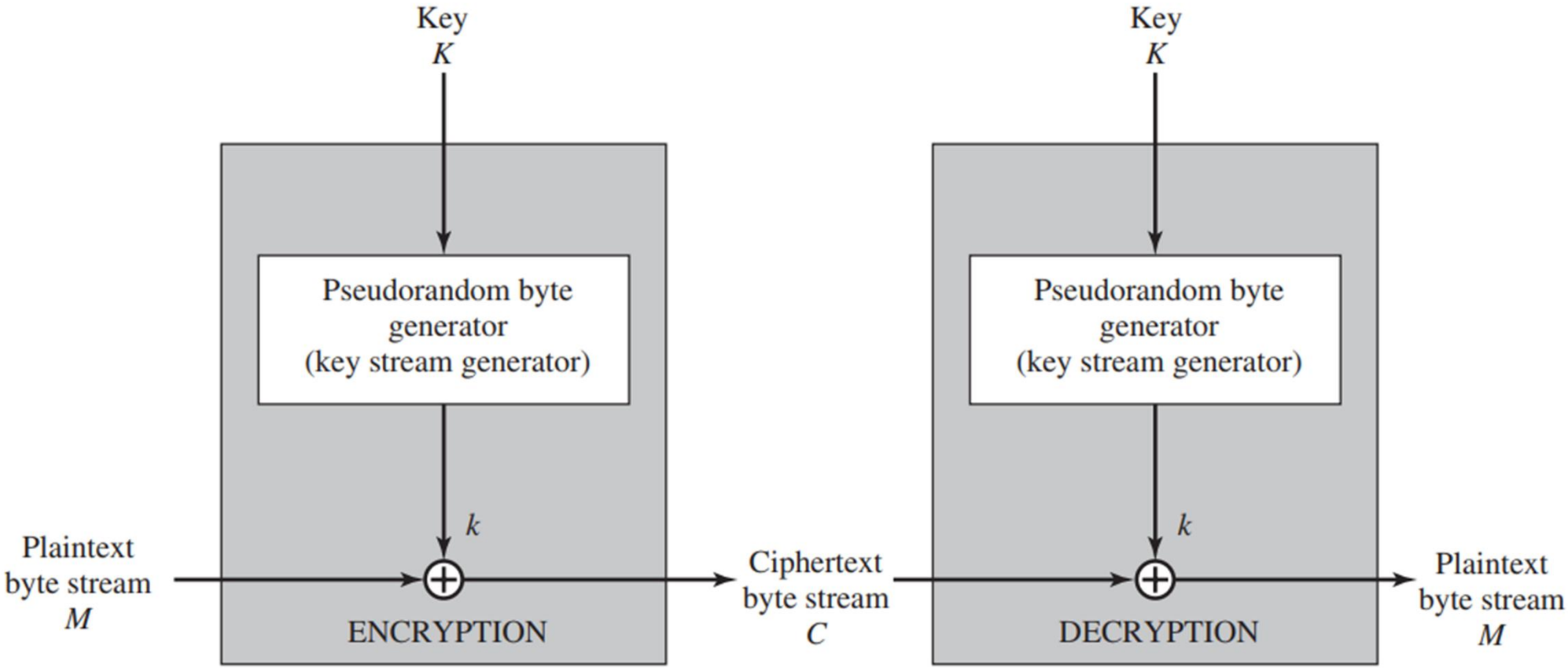
Key size (bits)	Cipher	Number of Alternative Keys	Time Required at 10^9 decryptions/μs	Time Required at 10^{13} decryptions/μs
56	DES	$2^{56} \approx 7.2 \times 10^{16}$	$2^{55}\,\mu s = 1.125$ years	1 hour
128	AES	$2^{128} \approx 3.4 \times 10^{38}$	$2^{127}\,\mu s = 5.3 \times 10^{21}$ years	5.3×10^{17} years
168	Triple DES	$2^{168} \approx 3.7 \times 10^{50}$	$2^{167}\,\mu s = 5.8 \times 10^{33}$ years	5.8×10^{29} years
192	AES	$2^{192} \approx 6.3 \times 10^{57}$	$2^{191}\,\mu s = 9.8 \times 10^{40}$ years	9.8×10^{36} years
256	AES	$2^{256} \approx 1.2 \times 10^{77}$	$2^{255}\,\mu s = 1.8 \times 10^{60}$ years	1.8×10^{56} years

- As key size increases, it reduces the vulnerability of brute-force attack.

Block Cipher Encryption (Electronic Codebook Mode)



Stream Encryption



✔ For lengthy messages, the ECB mode may not be secure. A cryptanalyst may be able to exploit regularities in the plaintext to ease the task of decryption.

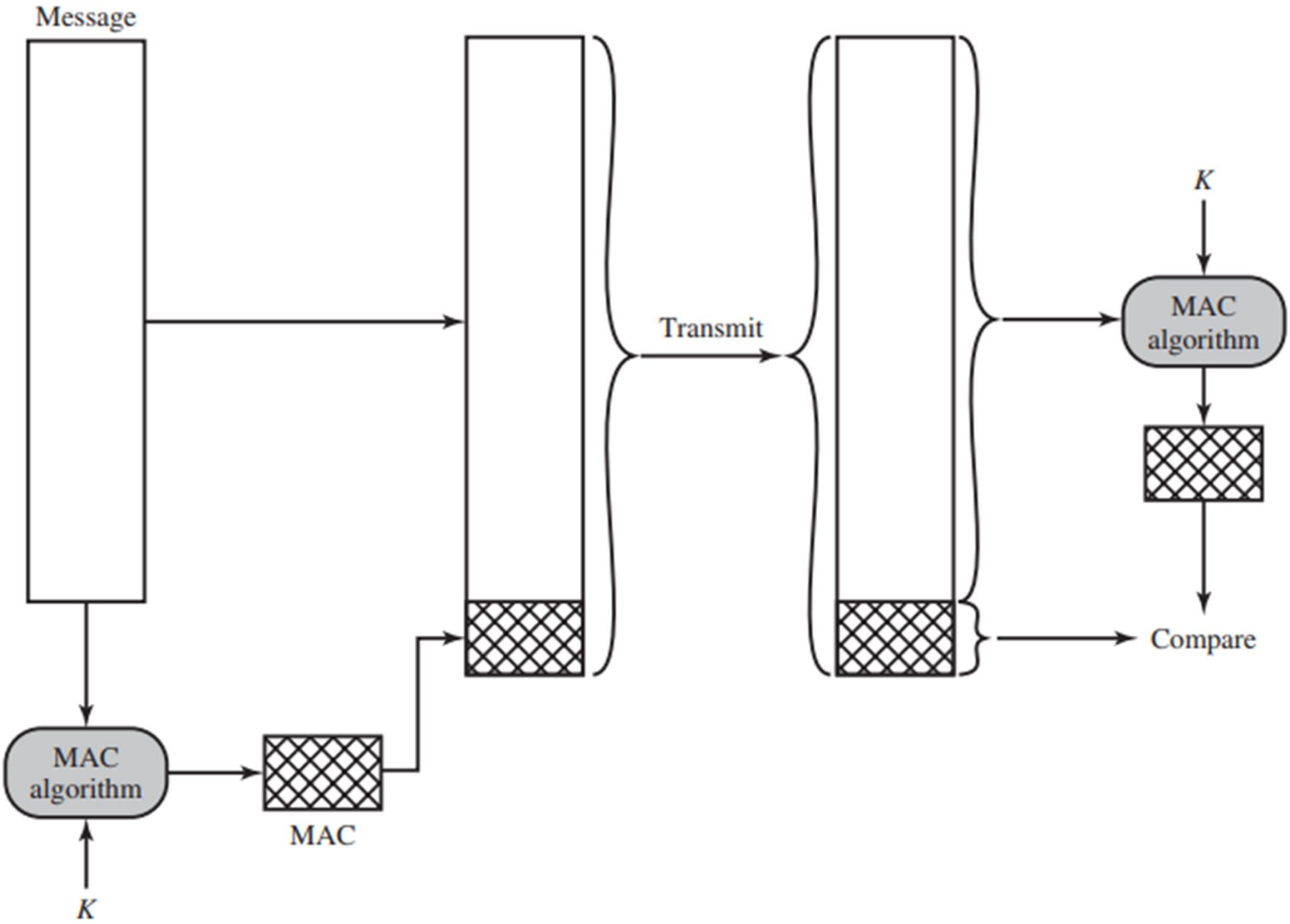
Message Authentication and Hash Function

- Encryption protects against passive attack (eavesdropping).
- A different requirement is to protect against active attack (falsification of data and transactions). Protection against such attacks is known as message or data authentication.
- Message or data authentication is a procedure that allows communicating parties to verify that received or stored messages are authentic.
- The two important aspects are to verify that the contents of the message have not been altered and that the source is authentic.

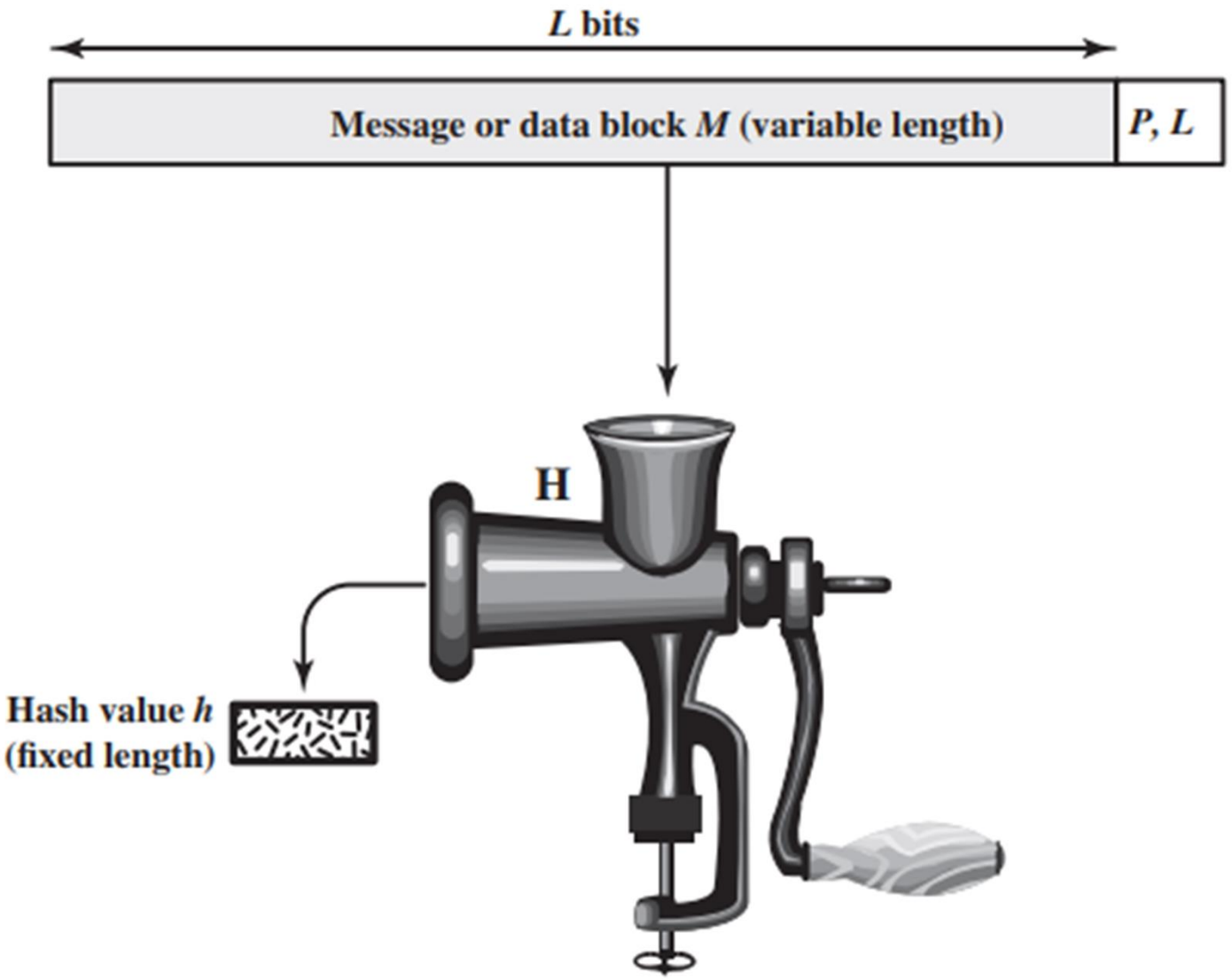
➢ **Authentication Using Symmetric Encryption**

symmetric encryption alone is not a suitable tool for data authentication.

Message Authentication without Message Encryption

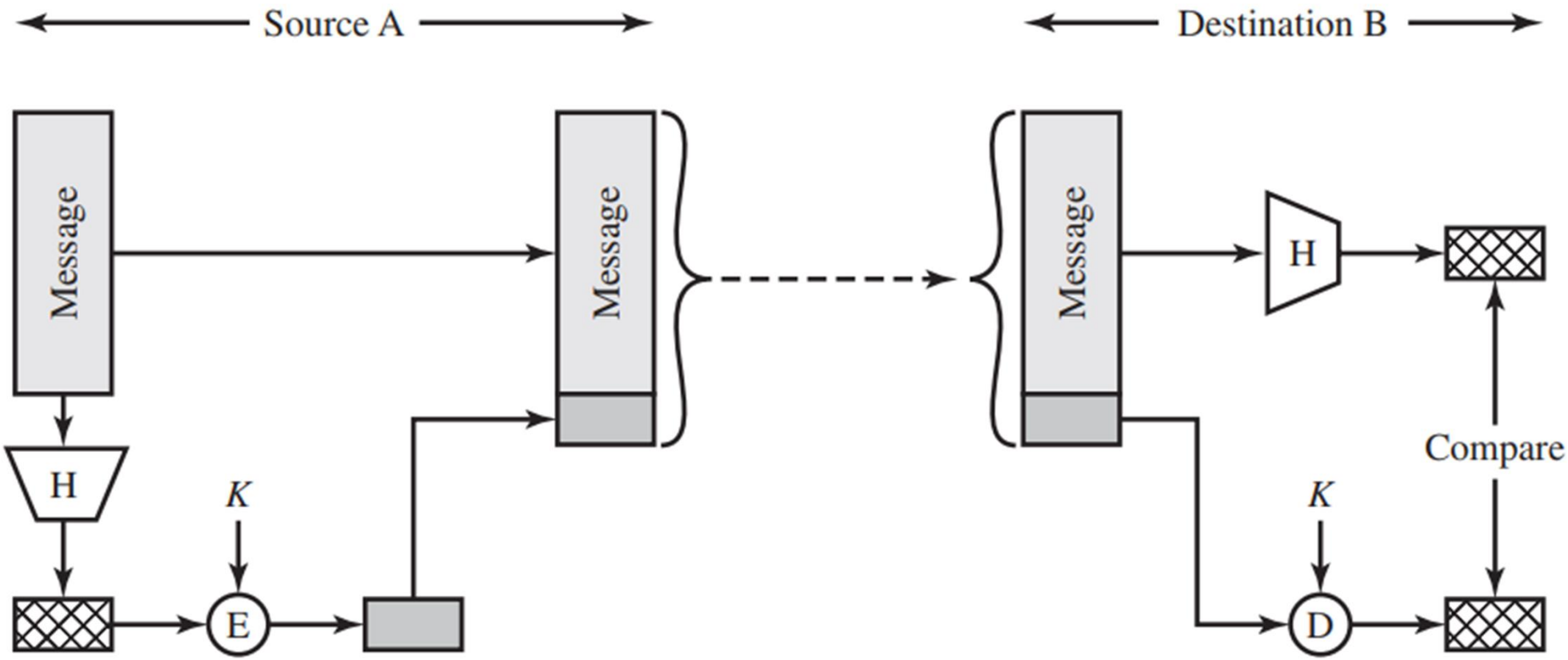


One Way Hash Function



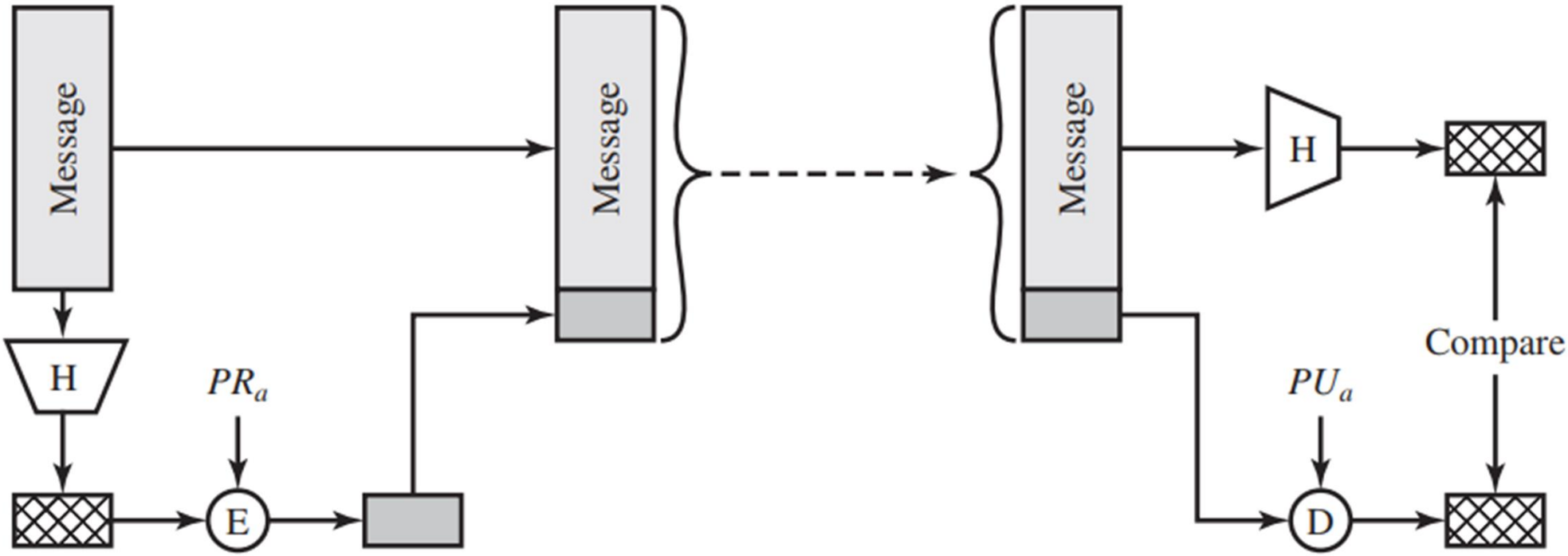
P, L = padding plus length field

Message Authentication using One-way Hash Function



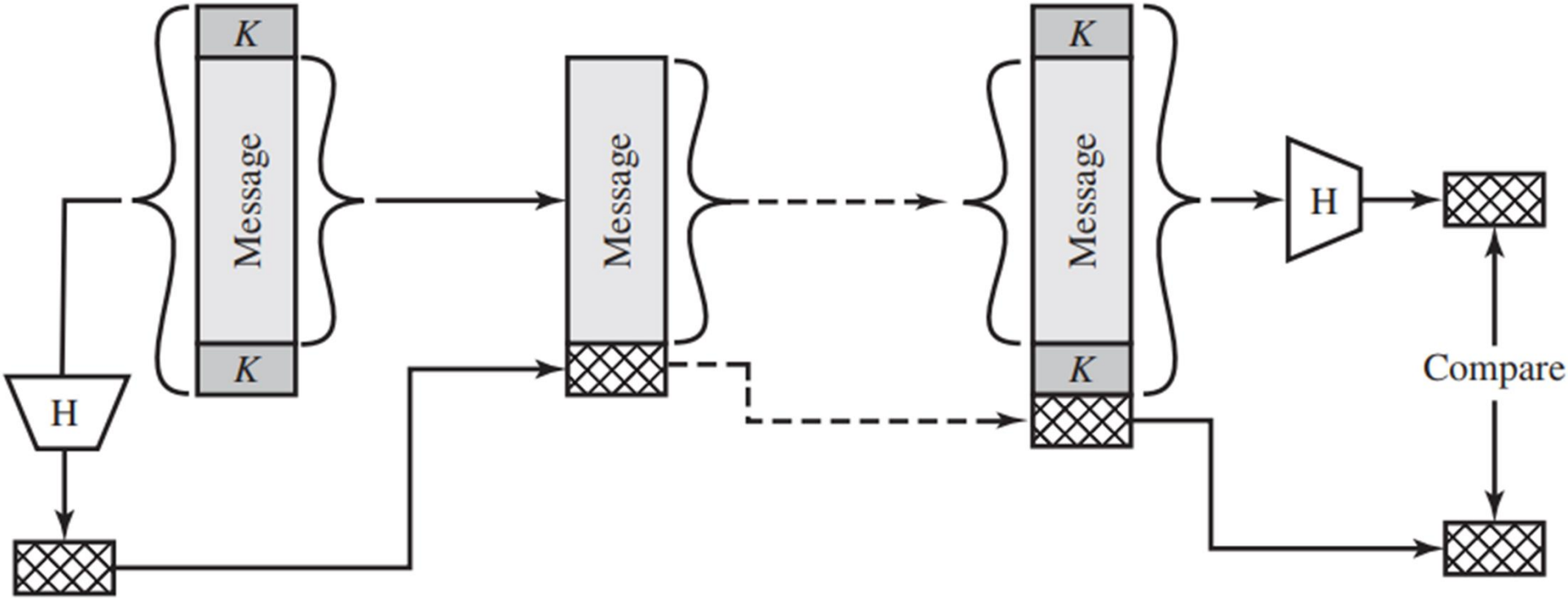
(a) Using symmetric encryption

Continued…



(b) Using public-key encryption

Continued…



(c) Using secret value

Secure Hash Function

- The one-way hash function, or secure hash function, is important not only in message authentication but in digital signatures.

Hash Function Requirements:

The purpose of a hash function is to produce a *"fingerprint"* of a file, message, or other block of data. To be useful for message authentication, a hash function *H* must have the following properties:

1. H can be applied to a block of data of any size.
2. H produces a fixed-length output.
3. H(x) is relatively easy to compute for any given x, making both hardware and software implementations practical.
4. For any given code h, it is computationally infeasible to find x such that H(x) = h. A hash function with this property is referred to as **one-way** or **preimage resistant**.

Continued…

5. For any given block x, it is computationally infeasible to find $y \neq x$ with $H(y) = H(x)$. A hash function with this property is referred to as **second preimage resistant**. This is sometimes referred to as **weak collision resistant**.

6. It is computationally infeasible to find any pair (x, y) such that $H(x) = H(y)$. A hash function with this property is referred to as **collision resistant**. This is sometimes referred to as **strong collision resistant**.

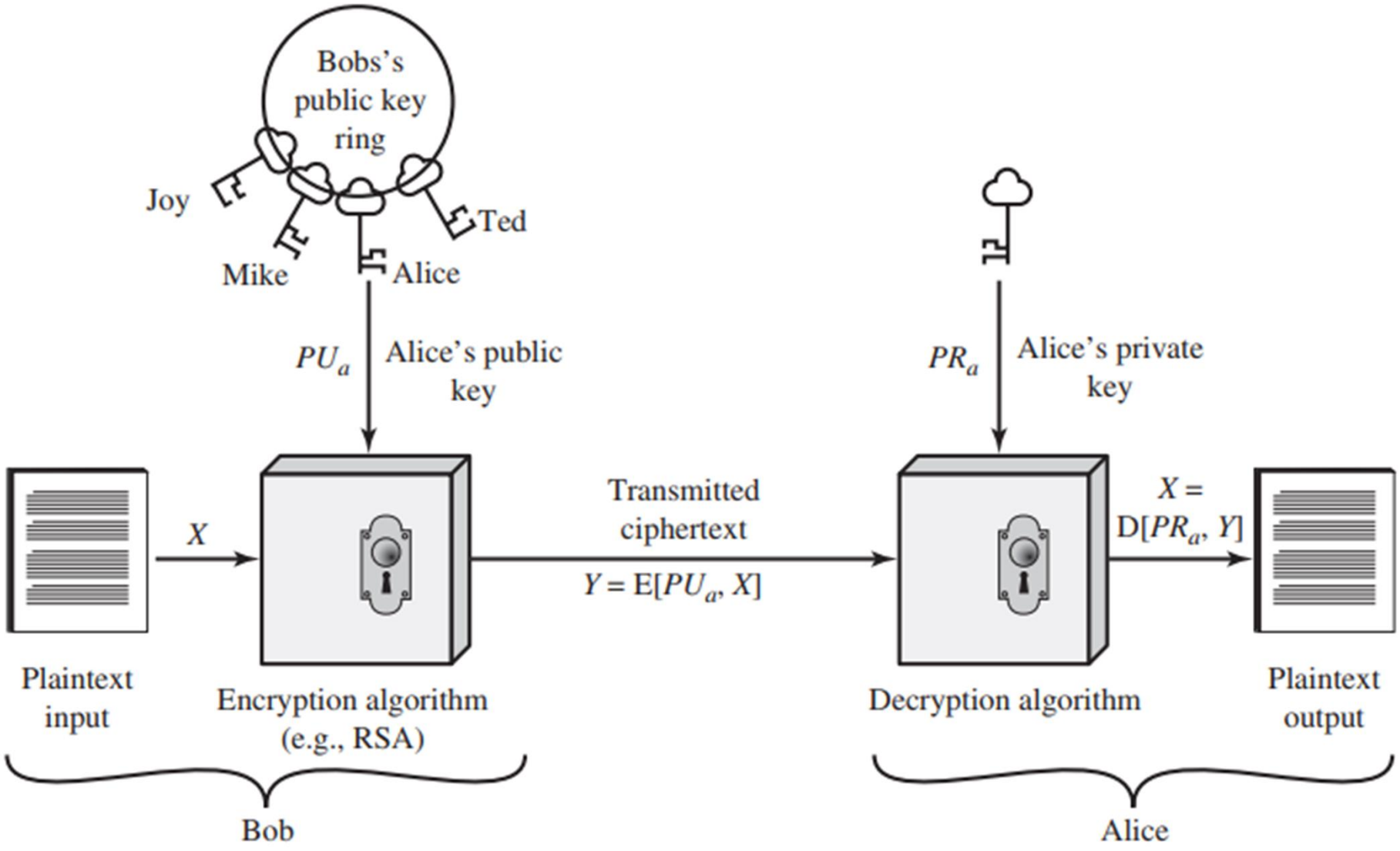
Hash Function Applications:

- Password
- Intrusion Detection

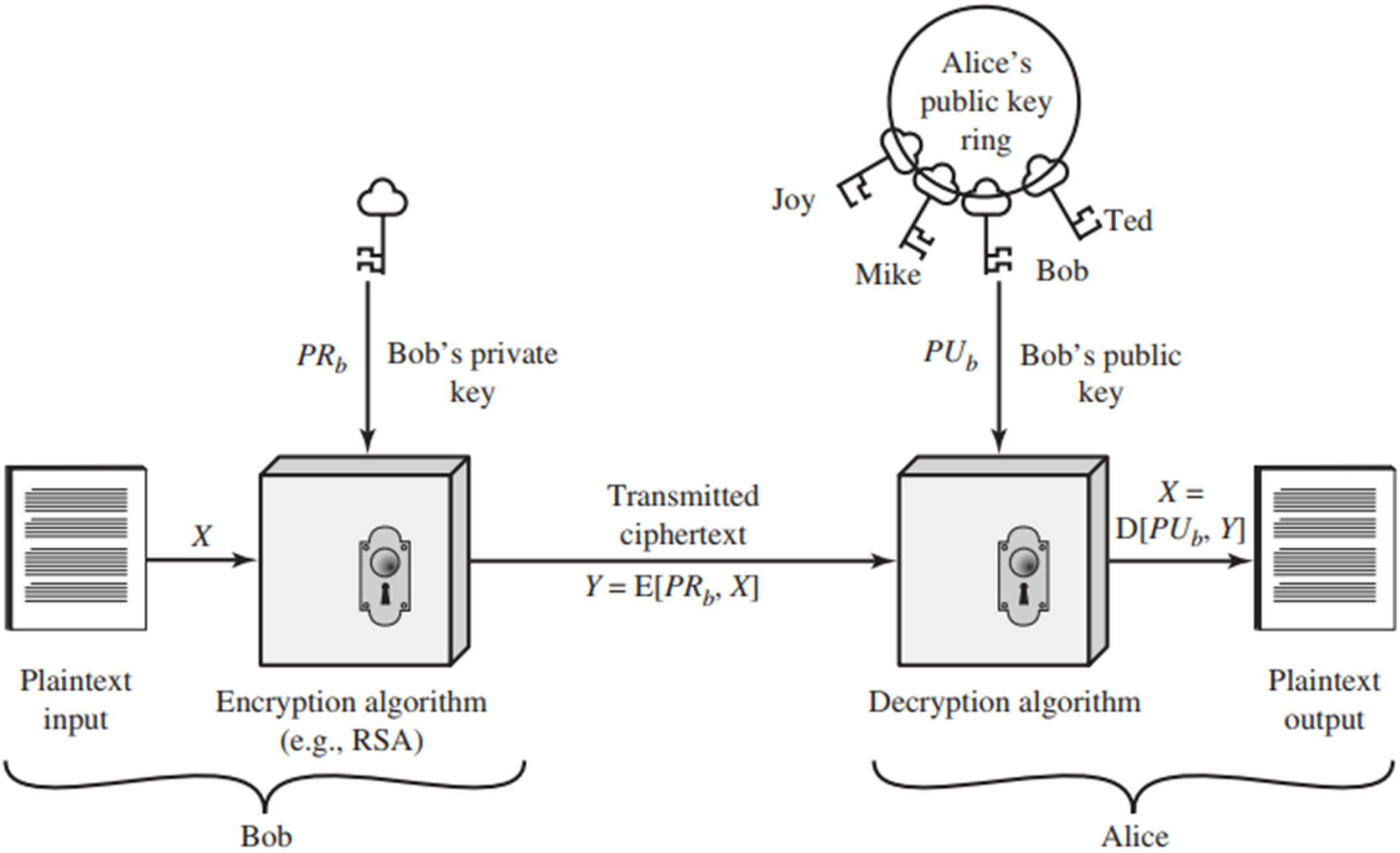
Public Key Encryption

- First publicly proposed by Diffie and Hellman in 1976.
- Public-key cryptography is asymmetric, involving the use of two separate keys, namely, public and private.
- Each user generates a pair of keys to be used for the encryption and decryption of messages.
- Each user places one of the two keys in a public register or other accessible file. This is the public key. The companion key is kept private.

Encryption with Public Key



Encryption with Private Key



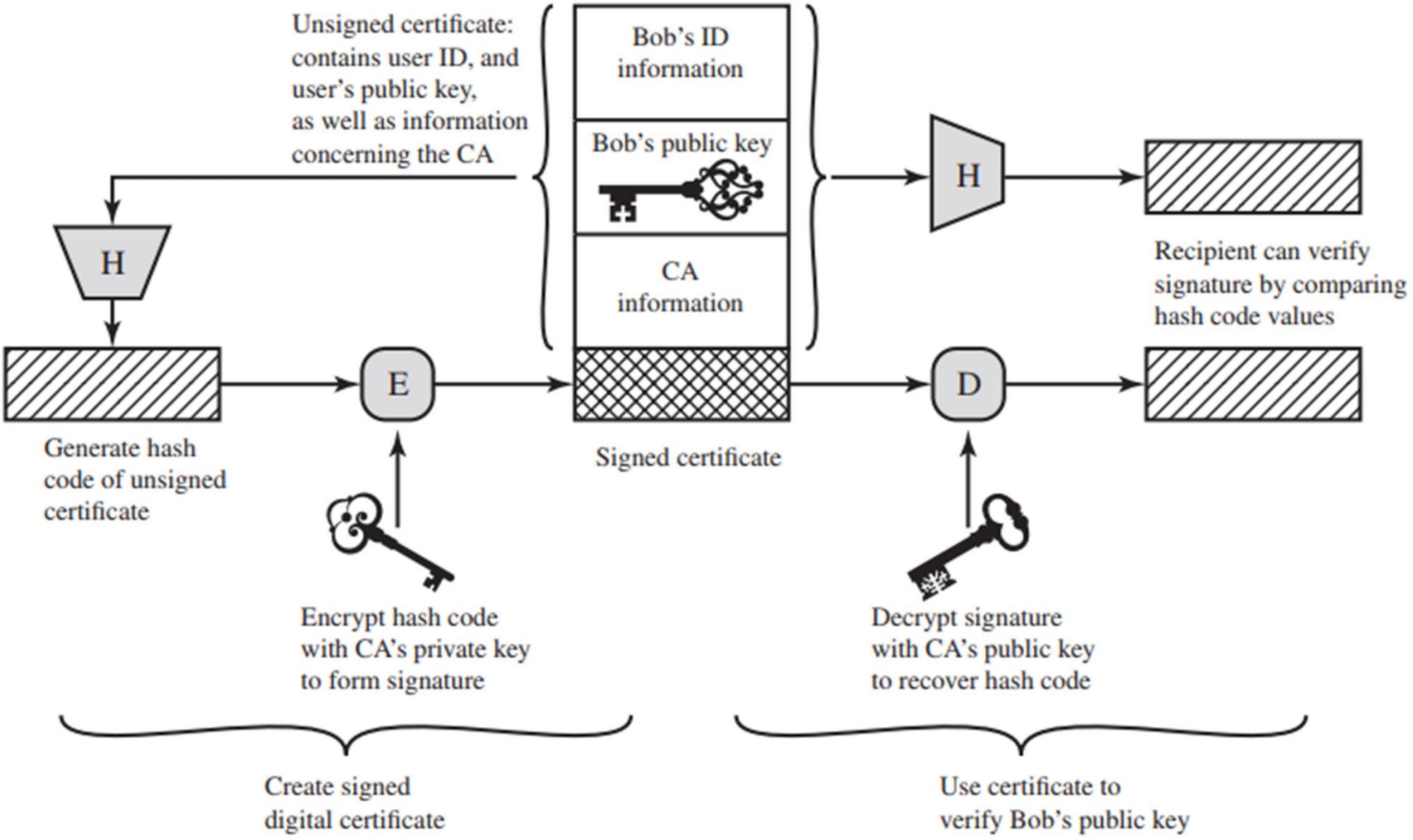
Applications of Public Key Cryptosystem

- Digital Signature
- Symmetric key distribution,
- Encryption of secret keys.

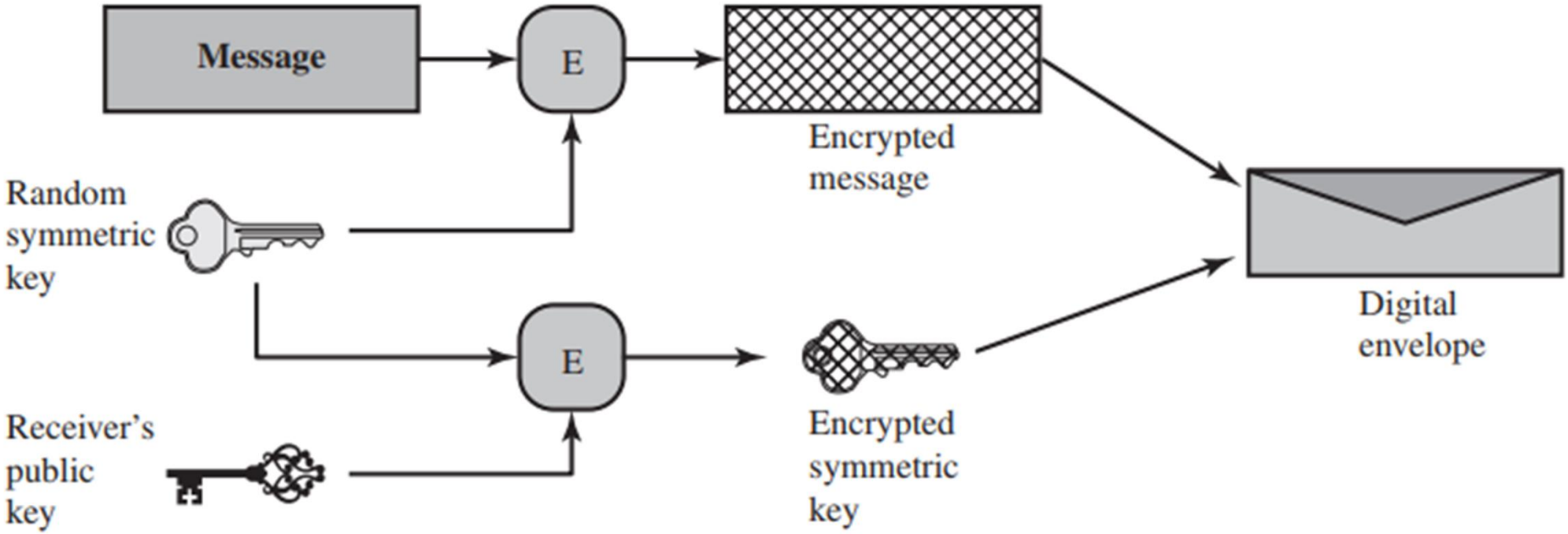
Requirements of Public Key Cryptosystem

1. It is computationally easy for a party B to generate a pair (public key PU_b, private key PR_b).
2. It is computationally easy for a sender A, knowing the public key and the message to be encrypted, M, to generate the corresponding ciphertext: $C = E(PU_b, M)$
3. It is computationally easy for the receiver B to decrypt the resulting ciphertext using the private key to recover the original message: $M = D(PR_b, C) = D[PR_b, E(PU_b, M)]$
4. It is computationally infeasible for an opponent, knowing the public key, PU_b, to determine the private key, PR_b

Digital Signature

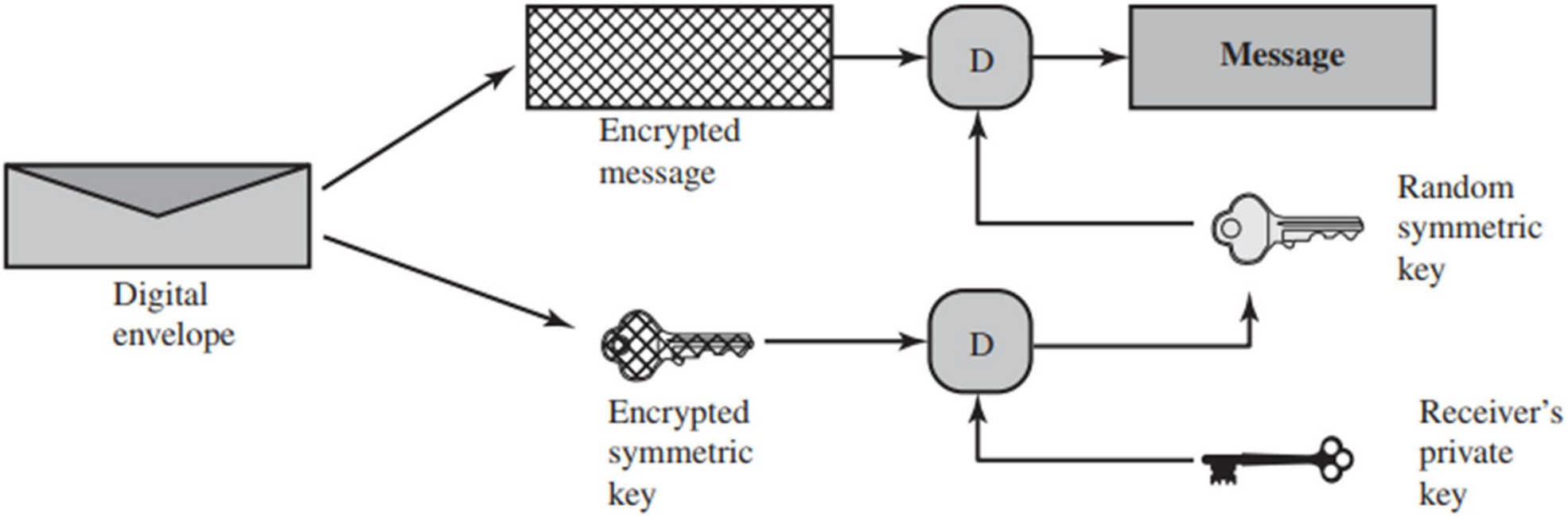


Digital Envelopes



(a) Creation of a digital envelope

Digital Envelopes Continued…



(b) Opening a digital envelope

User Authentication

- An authentication process consists of two steps:

1. **Identification step:** Presenting an identifier to the security system. (Identifiers should be assigned carefully, because authenticated identities are the basis for other security services, such as access control service.)
2. **Verification step:** Presenting or generating authentication information that corroborates the binding between the entity and the identifier.

1. **Something the individual knows:** Examples includes a password, a personal identification number (PIN), or answers to a prearranged set of questions.
2. **Something the individual possesses:** Examples include electronic keycards, smart cards, and physical keys. This type of authenticator is referred to as a *token*.
3. **Something the individual is (static biometrics):** Examples include recognition by fingerprint, retina, and face.
4. **Something the individual does (dynamic biometrics):** Examples include recognition by voice pattern, handwriting characteristics, and typing rhythm.

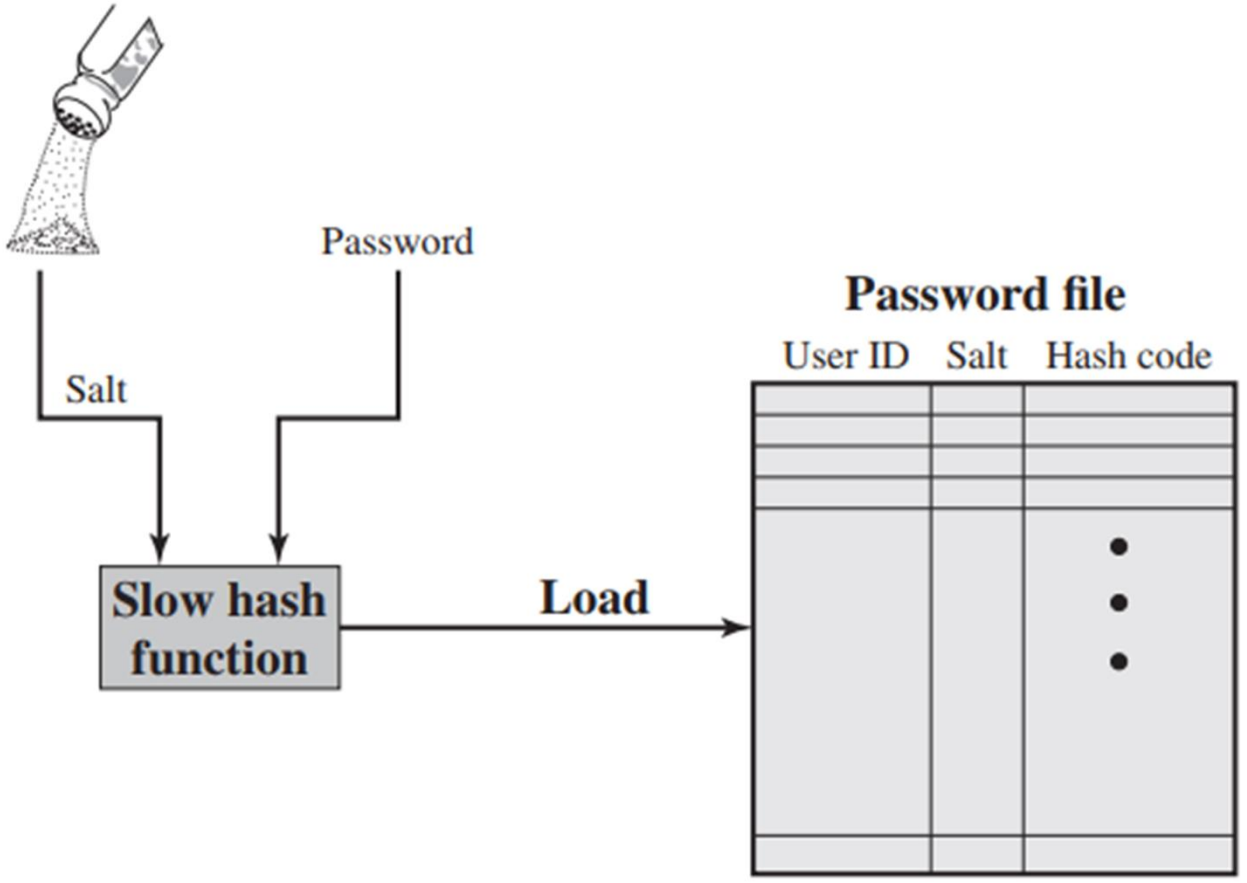
Password Based Authentication

- A widely used line of defense against intruders.

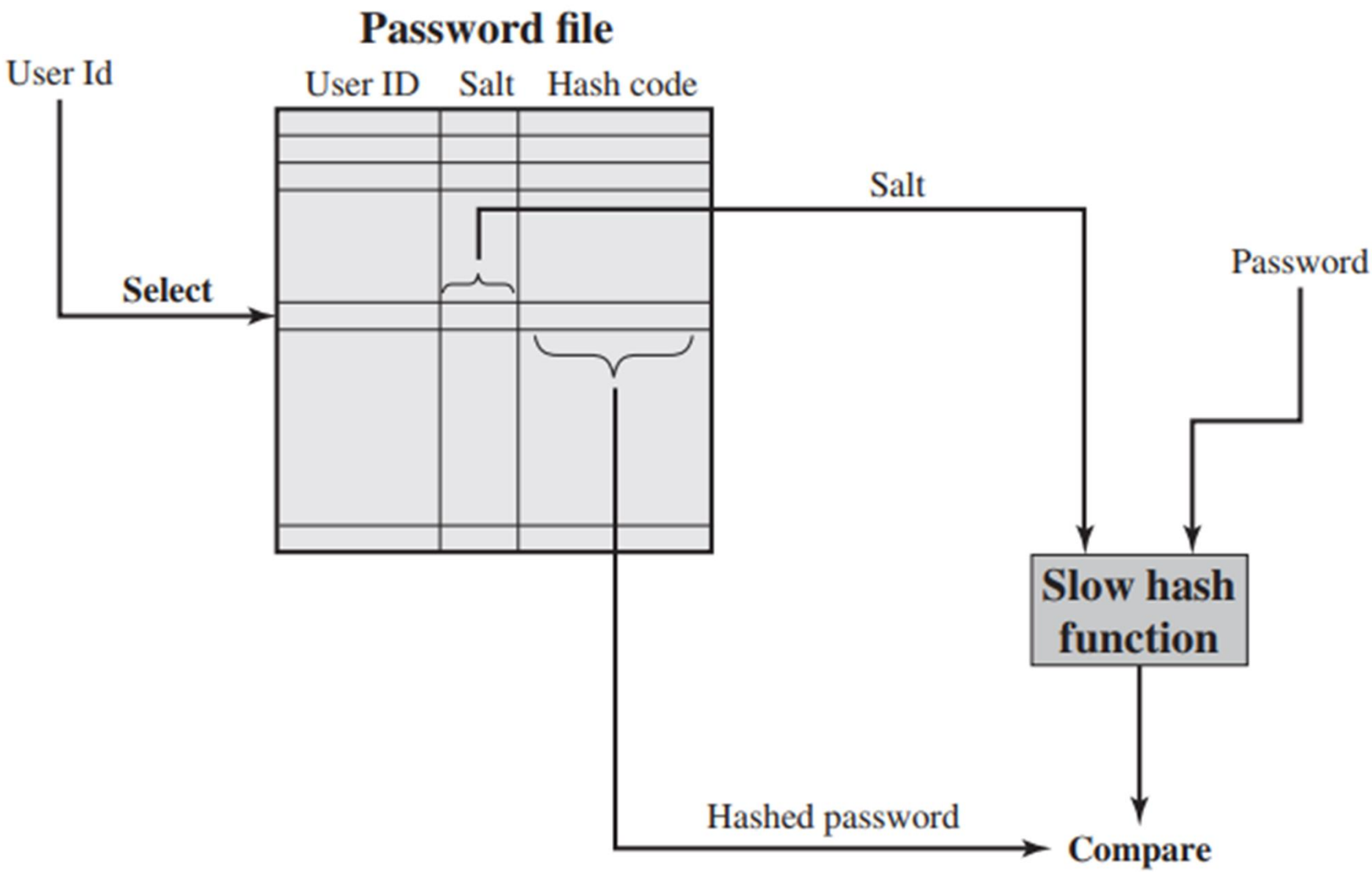
The Vulnerability of Passwords

1. Offline dictionary attack
2. Specific account attack
3. Popular password attack
4. Password guessing against single user
5. Workstation hijacking
6. Exploiting user mistakes
7. Exploiting multiple password use
8. Electronic monitoring

- A widely used password security technique is the use of hashed passwords and a salt value.
- To load a new password into the system, the user selects or is assigned a password. This password is combined with a fixed-length salt value.



(a) Loading a new password



(b) Verifying a password

The salt serves three purposes:

- It prevents duplicate passwords from being visible in the password file. Even if two users choose the same password, those passwords will be assigned different salt values. Hence, the hashed passwords of the two users will differ.
- It greatly increases the difficulty of offline dictionary attacks. For a salt of length b bits, the number of possible passwords is increased by a factor of 2^b,increasing the difficulty of guessing a password in a dictionary attack.
- It becomes nearly impossible to find out whether a person with passwords on two or more systems has used the same password on all of them.

NOTE:

In password file access control, the hashed passwords are kept in a separate file from the user IDs, referred to as a **shadow password file**.