# Introduction to Computer Security

## Chapter 2: Cryptographic Tools

Chi-Yu Li (2019 Spring)
Computer Science Department
National Chiao Tung University

# Outline

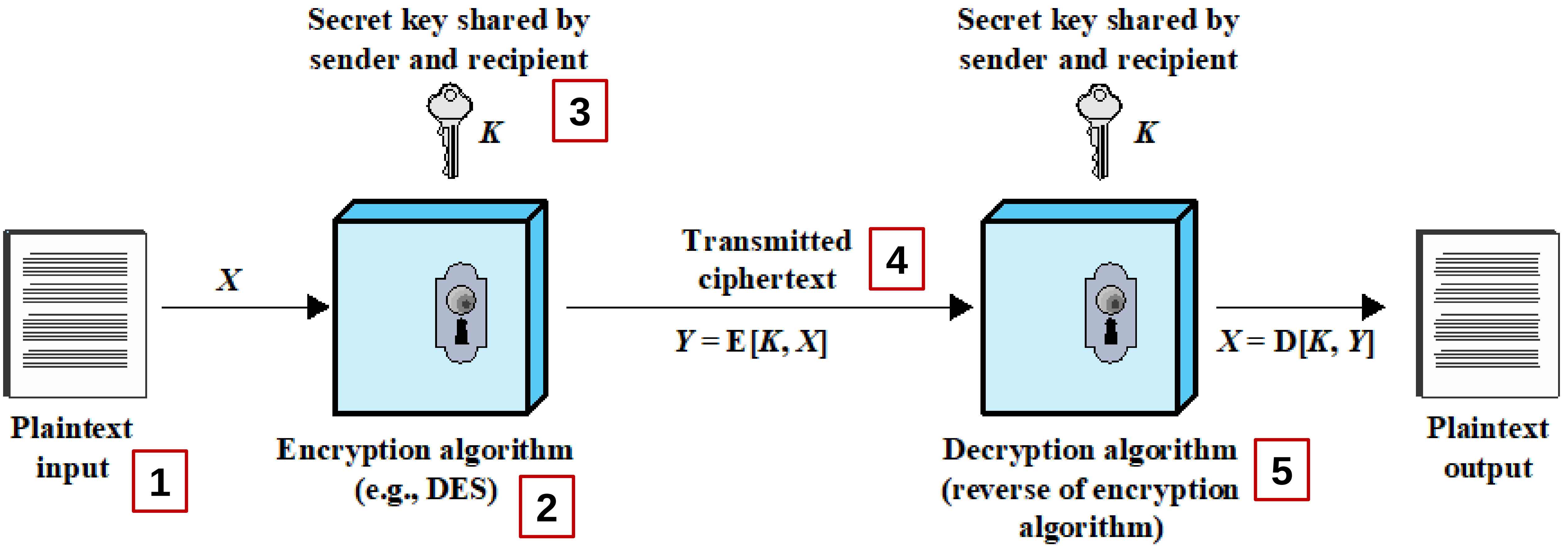
- Confidentiality with Symmetric Encryption
- Message Authentication and Hash Function
- Public-Key Encryption
- Digital Signatures and Key Management
- Random and Pseudorandom Numbers

# Confidentiality with Symmetric Encryption

- Symmetric encryption
  - Universal technique for providing confidentiality for transmitted or stored data
  - Also referred to as conventional encryption or single-key encryption
- Two requirements for its secure use
  - Need a strong encryption algorithm
    - The opponent should be unable to decrypt ciphertext or discover the key even if he or she is in possession of pairs of ciphertexts and plaintexts, as well as the algorithm
  - Sender and receive must have obtained copies of the secret key in a secure fashion and must keep the key secret

# Simplified Model of Symmetric Encryption

- There are five ingredients



# Attacking Symmetric Encryption

## Cryptanalytic Attacks

- Rely on
  - Nature of the algorithm
  - General characteristics of the plaintext
  - Sample plaintext-ciphertext pairs
- Exploits the characteristics of the algorithm to deduce a specific plaintext or the key

## Brute-Force Attacks

- Try all possible keys on some ciphertexts until an intelligible translation into plaintext is obtained
  - On average half of all possible keys must be tried to achieve success
- Knowledge about the expected plaintext

# Symmetric Block Encryption Algorithms

- Block ciphers: most commonly used symmetric encryption
  - Fixed-size blocks of plaintext → blocks of ciphertext of equal size

| | 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

# DES Encryption Standard (DES)

- Adopted in 1977 by the NIST (FIPS PUB 46)
  - Most widely used encryption scheme is based on it: Data Encryption Algorithm (DEA)
  - 64-bit plaintext blocks and a 56-bit key ➔ 64-bit ciphertext blocks

- Strength concerns
  - Algorithm: characteristics may be exploited
    - Most-studied encryption algo: numerous attempts to find weakness, but no fatal one yet
  - Key length: too short
    - 56 bits ➔ $2^{56} = 7.2 \times 10^{16}$ possible keys (Inadequate for today's processor speed)

**<u>FIPS (Federal Information Processing Standard)</u>**: describes document processing, encryption algorithms and other technology standards for use within non-military government agencies and by government contractors and vendors who work with the agencies.

# DES Encryption Standard (DES): Brute-Force Attacks

- On average, half the key space has to be searched
  - One DES encryption per micro second ➔ more than 1000 years (3.6 x $10^{16}$ keys)
- In July 1998, EFF announced it had broken a DES encryption
  - DES cracker: less than $250,000, less than three days
    - http://cs-exhibitions.uni-klu.ac.at/index.php?id=263

DES cracker: key discovery in 56 hours

- Encryption speeds advance
  - Seagate Technology [SEAG08]
    - Multicore computers (2008): $10^9$ per second
  - EE Times [AROR12]
    - Contemporary supercomputer (2012): $10^{13}$ per second ➔ break DES within 1 hour

**<u>EFF (Electronic Frontier Foundation):</u>** the leading nonprofit organization defending civil liberties in the digital world

# Triple DES (3DES)

- Repeats basic DES algorithm 3 times using either 2 or 3 unique keys
  - a key size of 112 or 168 bits
- Standardization
  - First standardized for use in financial apps in ANSI standard X9.17 in 1985
  - Part of the DES in 1999: FIPS PUB 46-3
- Two attractions
  - 168-bit key length: overcomes brute-force attack of DES
  - Underlying encryption algorithm is the same as in DES
- Two drawbacks
  - Sluggish algorithm/software: not efficient software code and three times as DES
  - Uses a 64-bit block size: not efficient and not secure

# Advanced Encryption Standard (AES)

- AES: now widely available in commercial products
  - A replacement for 3DES
  - 3DES was not reasonable for long term use
- NIST called for proposals for a new AES in 1997
  - With a security strength equal to or better than 3DES
  - Significantly improved efficiency
  - Symmetric block cipher
  - 128-bit data and 128/192/256 bit keys
- Selected Rijndael in Nov. 2001: published as FIPS 197

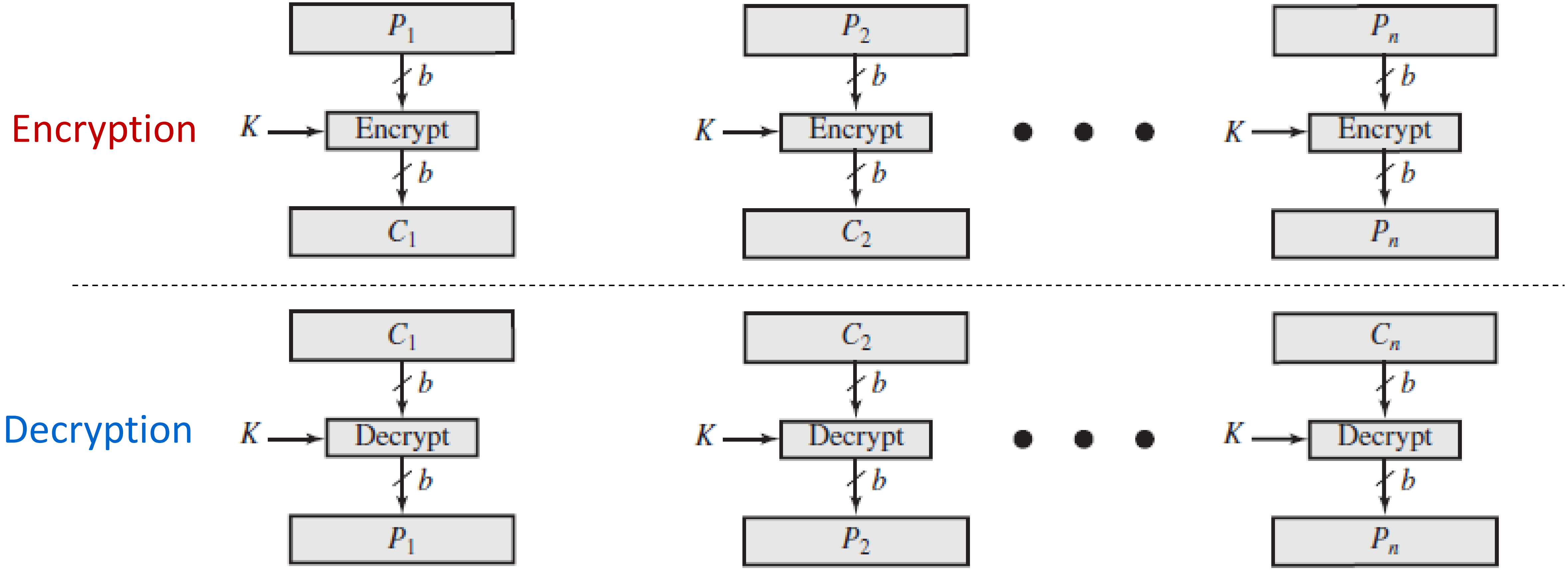
# Average Time Required for Exhaustive Key Search

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

# Practical Security Issues

- Symmetric encryption: applied to a unit of data larger than a single 64-bit or 128-bit block
  - e.g., E-mail messages, network packets, database records, etc.
- Simplest approach: electronic codebook (ECB)
  - Multiple-block encryption
  - Each block of plaintext is encrypted using the same key
- Issue: Cryptanalysts may exploit regularities in the plaintext

# Illustration of the ECB Mode

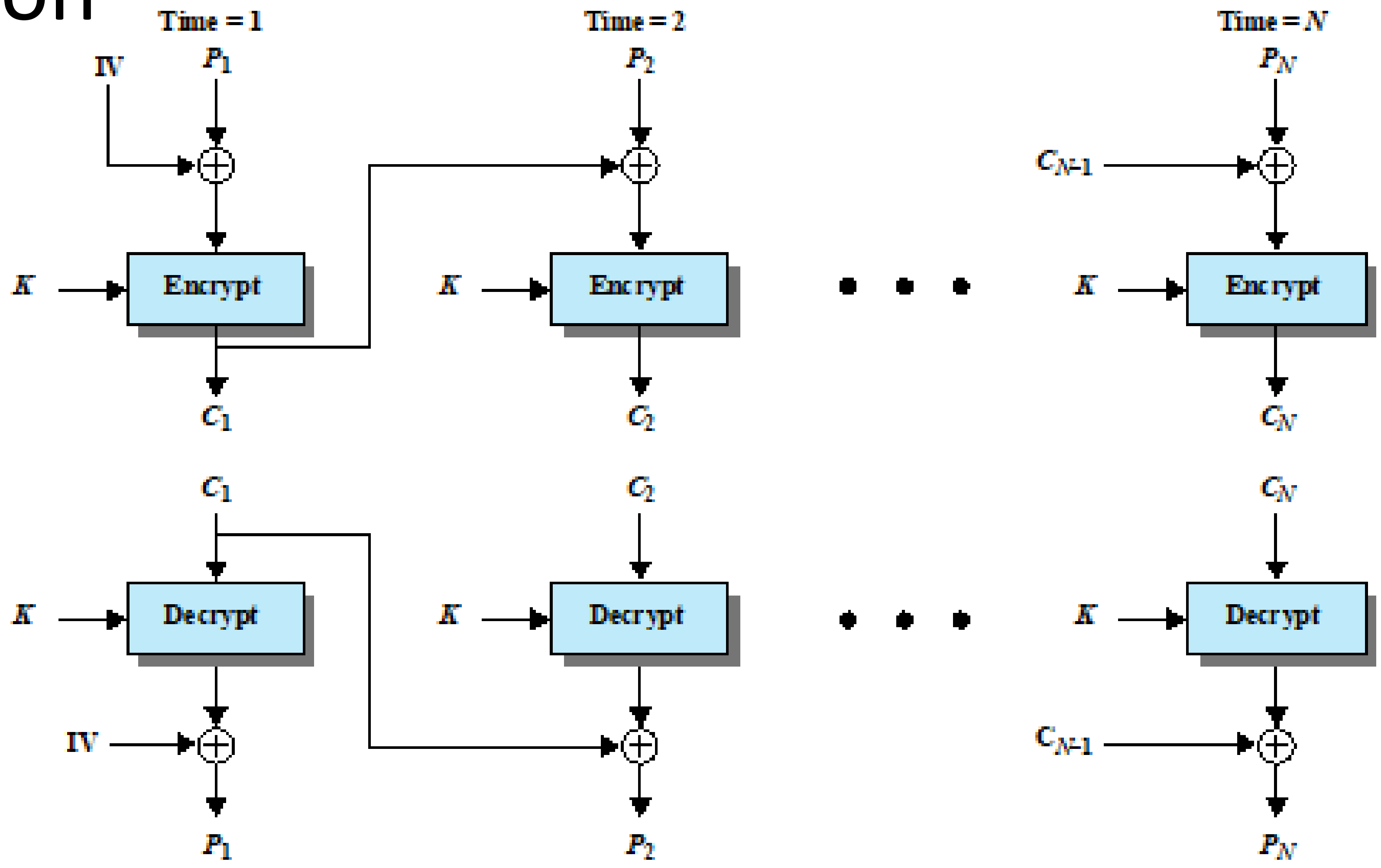



How to address the ECB issue (regularities)?

# Modes of Operation

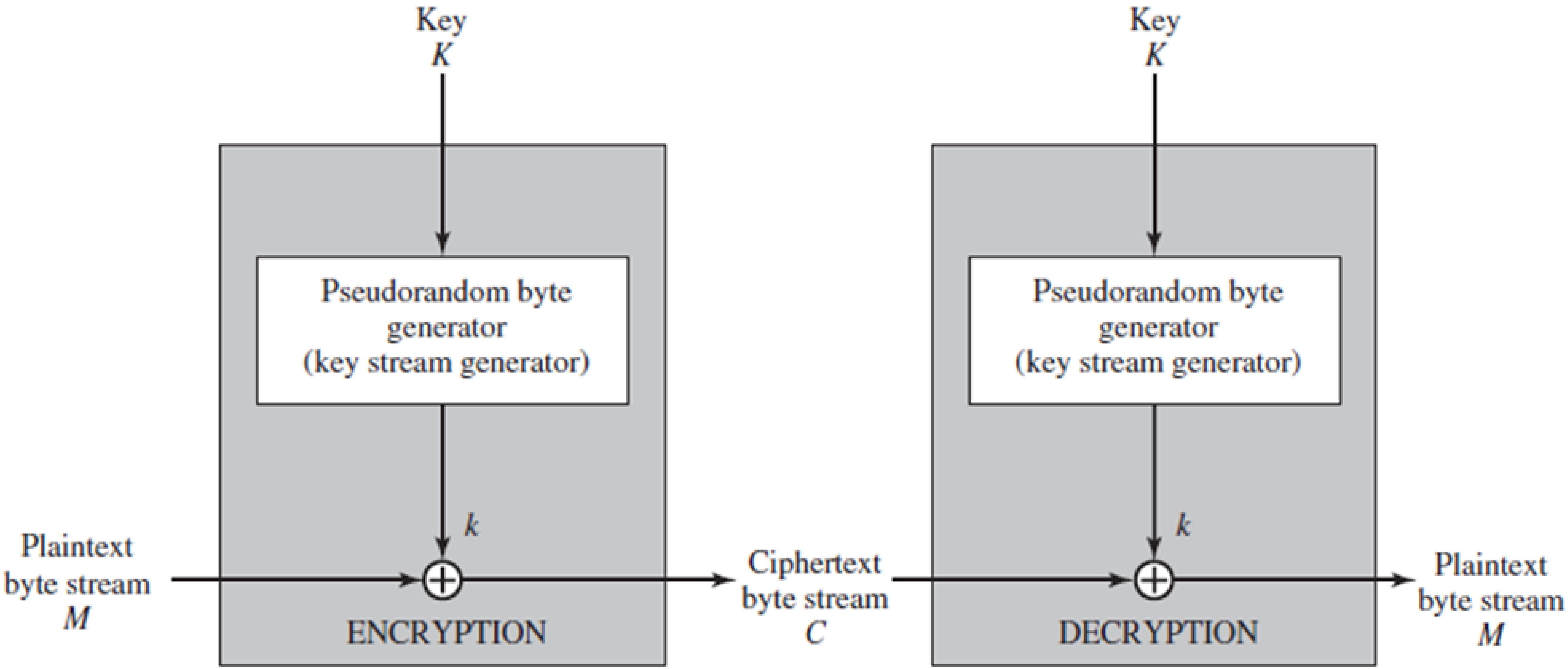
- Five modes of operation defined by NIST
  - ECB, CBC, etc.
- CBC can overcome the weakness of ECB

## Cipher Block Chaining (CBC) Mode




See Figure 20.6

# Stream Cipher



# Block & Stream Ciphers

## Block Cipher

- Processes the input one block of elements at a time
- Produces an output block for each input block
- Primary advantage: Can reuse keys
- More common
- Apps: file transfer, e-mail, and database

## Stream Cipher

- Processes the input elements continuously
- Produces one element at a time
  - Typically: one byte; one bit or larger units are also allowed
- Primary advantage: almost always faster (XOR) and use far less code
- Apps: data over a communication channel or a browser/Web link

# Message Authentication and Hash Functions

- Message (data) authentication: a procedure that allows communicating parties to verify that received/stored messages are authentic
  - Against falsification of data and transactions
- Two major aspects to verify
  - Message contents: not altered
  - Message source: authentic
- Another aspect
  - Message timeliness/sequence: not artificially delayed or replayed

# Can We Use Symmetric Encryption?

- Authentic source
  - Only the sender and the receiver share the key
- No altered contents
  - An error-detection code
- Proper message timeliness
  - A sequence number or a timestamp

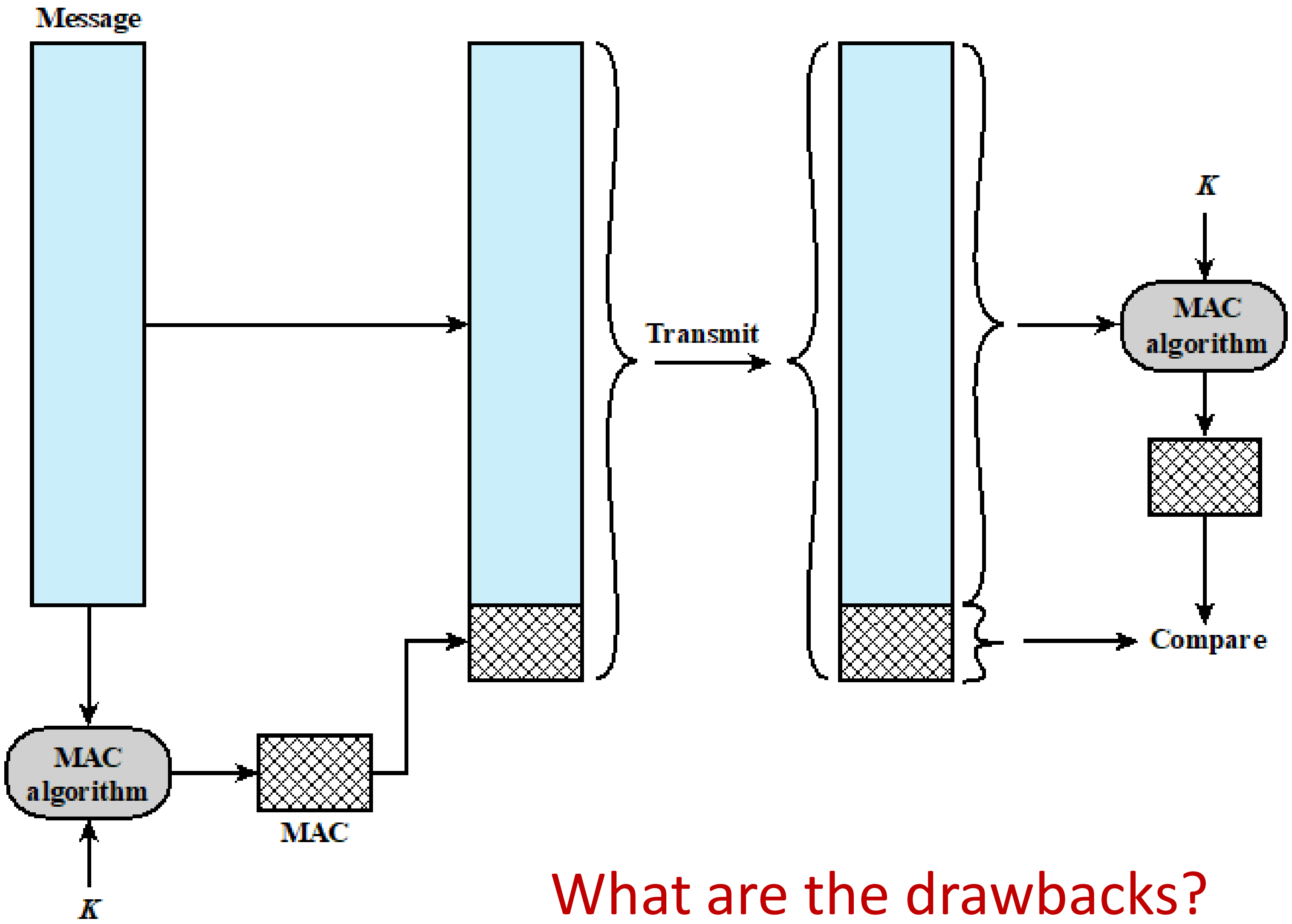
It seems proper, but is not a suitable tool for data authentication. Why?

# Message Authentication w/o Encryption

- Message authentication: a separate function from message encryption
  - e.g., a broadcast message to many destinations, not able to decrypt all incoming messages, wasteful of processor resources
- How? an auth. tag is generated and appended to each message
  - Message Authentication Code (MAC)
  - One-Way Hash Function

# Message Authentication Code (MAC)

- Use of a secret key to generate a small block of data
- Assumption: two communicating parties share the secret key




What are the drawbacks?

- MAC algorithms: NIST recommends DES
  - MAC: last 16- or 32-bit code of the encrypted message

# Message Authentication Code (Cont.)

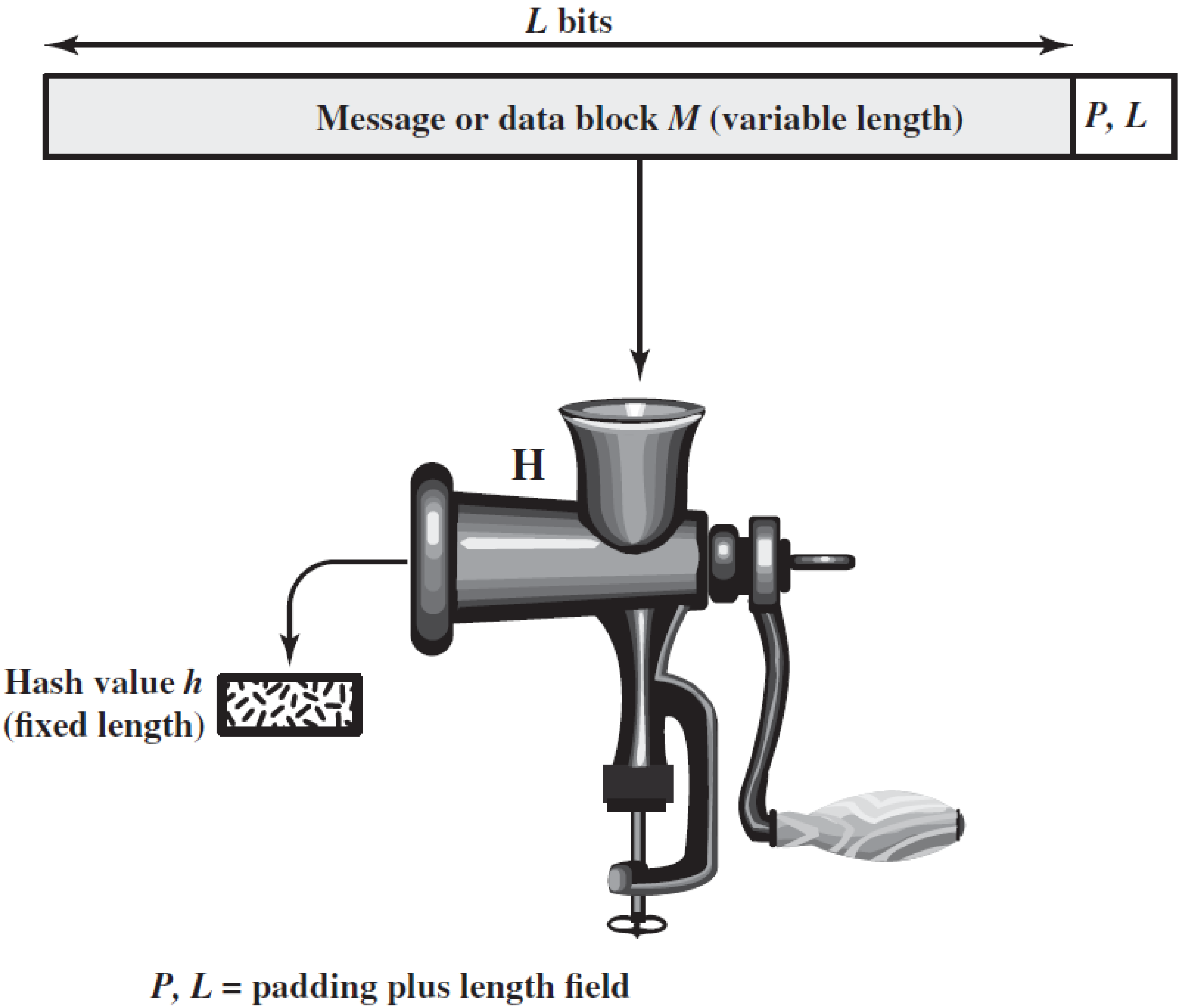
- Drawbacks
  - Encryption software is quite low
  - Encryption hardware costs are nonnegligible
  - Encryption hardware is optimized toward large data sizes

Does message authentication really need encryption of the message?

- No. Why?
  - Authentication need not be reversible
  - Only need a way to generate a tag which can verify messages

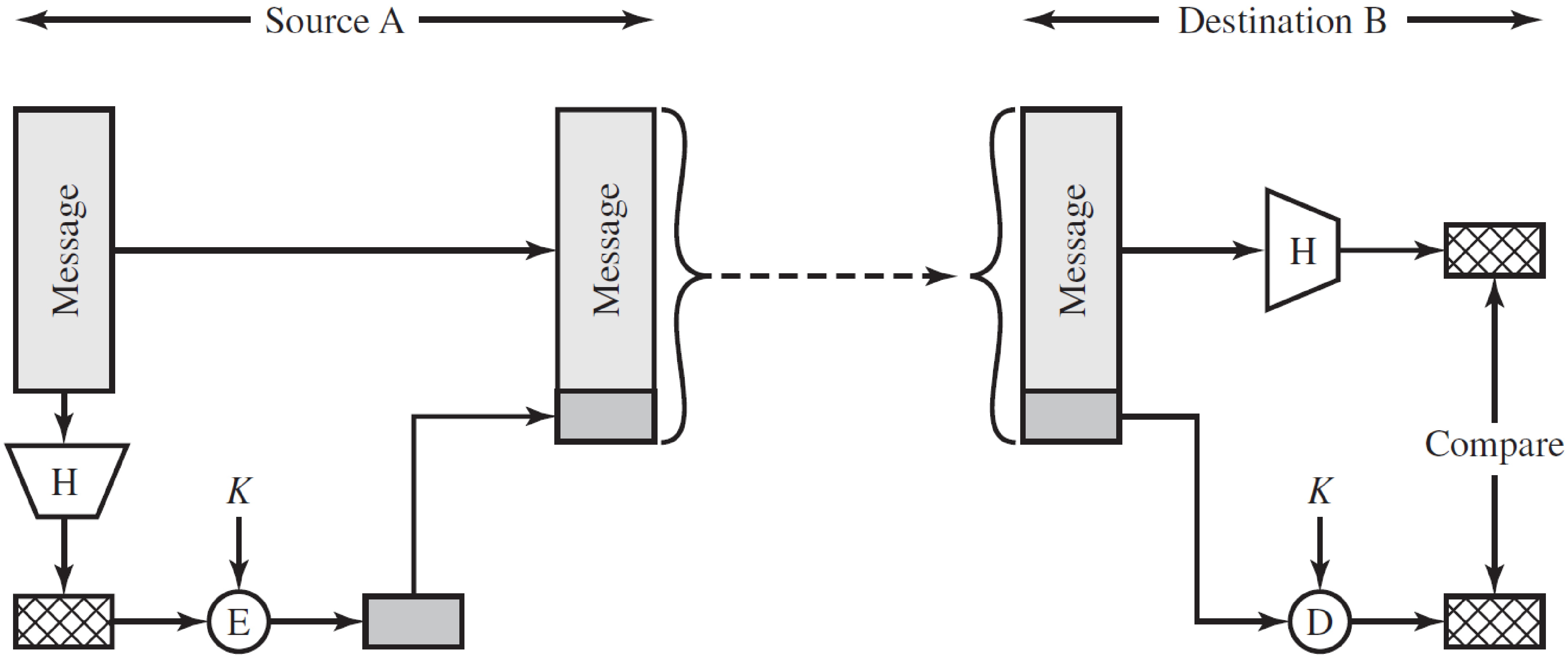
# One-way Hash Function

- A variable-size message M ➔ Tag: a fixed-size message digest H(M)
  - Length field: a security measure to increase the difficulty to produce an alternative message with the same hash value
- Unlike MAC
  - does not take a secret key as input

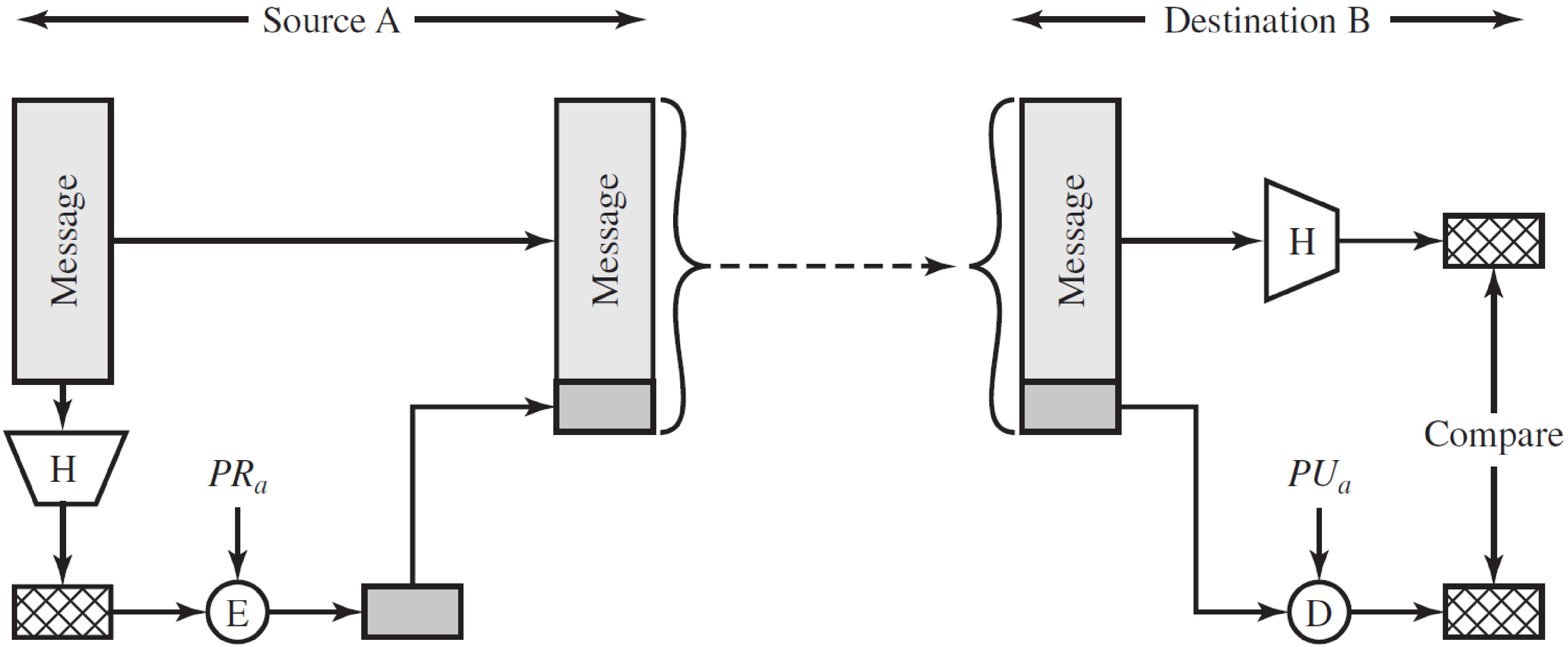



The hash value ensures only unaltered contents.
How about authentic source?

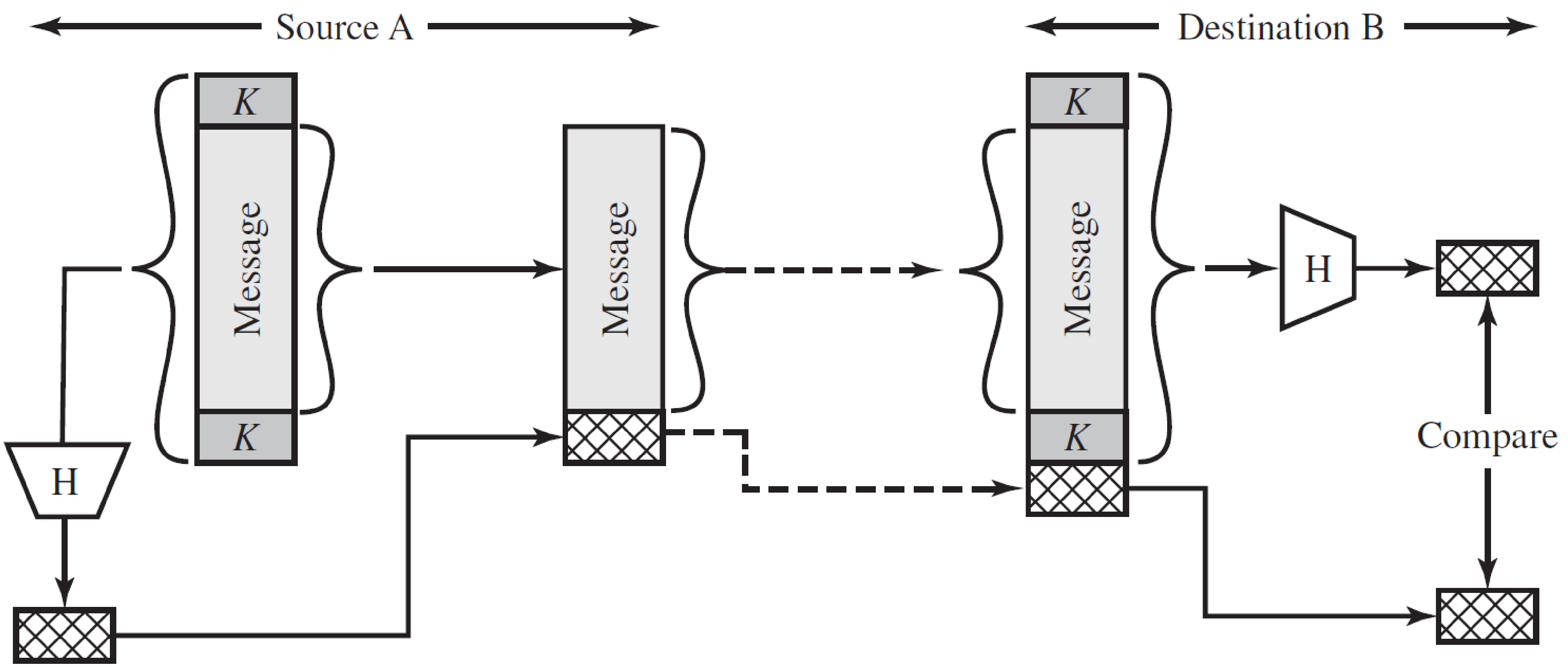
# Hash Function w/ Symmetric Encryption



# Hash Function w/ Public-key Encryption



# Hash Function w/o Encryption: Keyed Hash MAC



# Secure Hash Functions

- A hash function $H$ must have the following properties
  - $H$ can be applied to a block of data of any size
  - $H$ produces a fixed-length output
  - $H(x)$ is relatively easy to compute for any given $x$
    - Making both hardware and software implementations practical
  - One-way (pre-image resistant)
    - For any given code $h$, it is computationally infeasible to find $x$ such that $H(x) = h$
  - Second pre-image (weak collision) resistant
    - For any given block $x$, it is computationally infeasible to find $y \neq x$ with $H(y) = H(x)$
  - Collision (strong collision) resistant
    - It is computationally infeasible to find any pair $(x, y)$ such that $H(x) = H(y)$

# Security of Hash Functions

- Two attack approaches
  - Cryptanalysis: exploits logical weaknesses in algorithms → algorithm dependent
  - Brute-force → depends solely on the length of the hash code

- Security strength against brute-force attacks
  - Preimage resistant: $2^n$
  - Second preimage resistant: $2^n$
  - Collision resistant: $2^{n/2}$
    - Based on that a birthday attack on a message digest of size n produces a collision
    - 128-bit MD5 [VANO94]: 24 days
    - 160-bit SHA: > 4000 years

# Security of Hash Functions (Cont.)

- Most widely used hash algorithm: Secure Hash Algorithm (SHA)
  - Developed by NIST, and published in FIPS 180, 1993
  - SHA-1 160-bit (1995)
  - SHA-256, SHA-384, SHA-512 (2002)

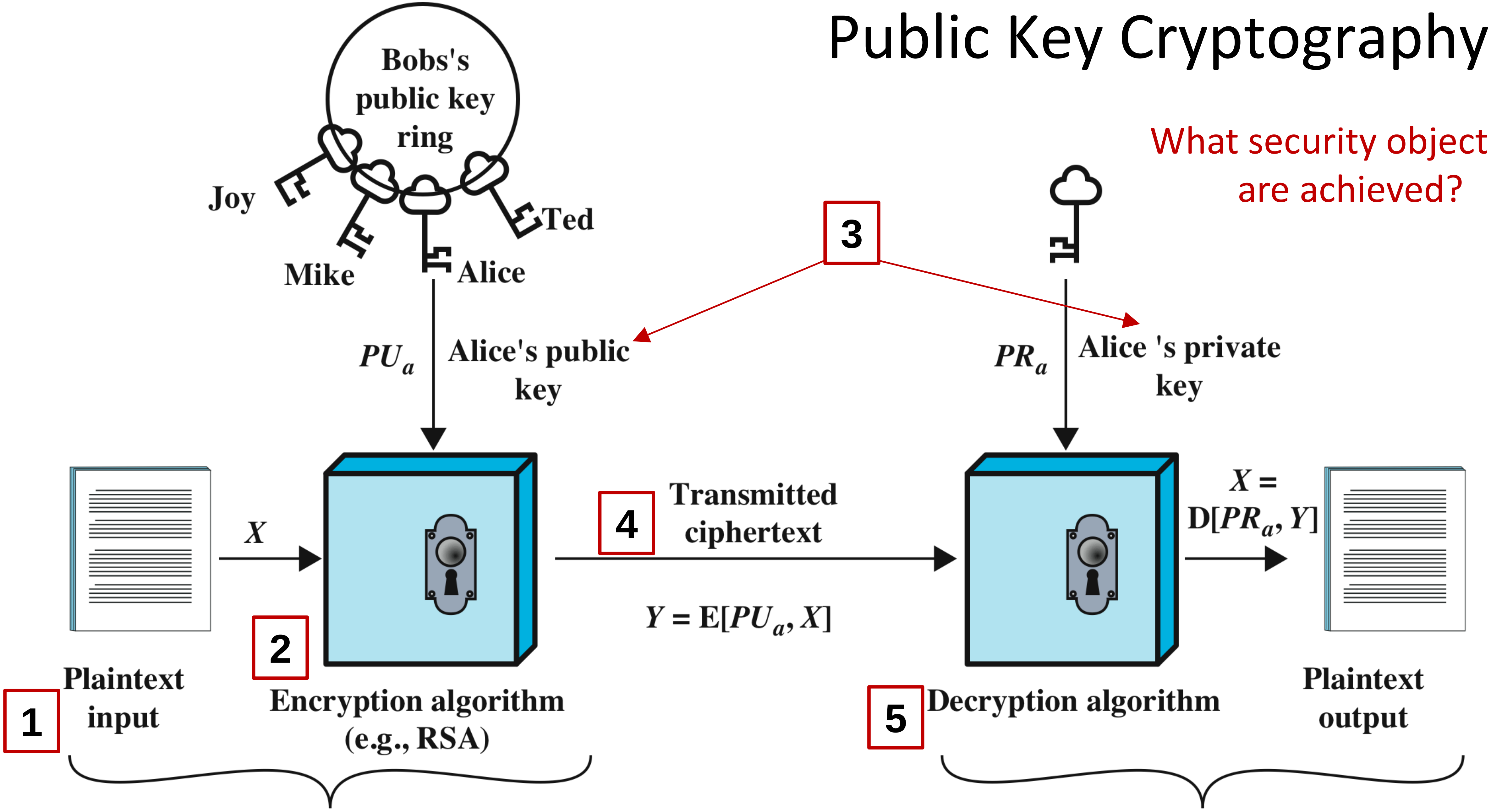
- Applications of hash functions
  - Message authentication
  - Digital signatures (later in this Chapter)
  - Passwords (Chapter 3)
  - Intrusion detection (Chapter 8)
    - Store the hash value of a file $H(F)$ for detecting file falsification

# Public-Key Encryption

- Public-key cryptography
  - Proposed by Diffie and Hellman in 1976
  - Asymmetric algorithm
    - Two separate keys: public and private keys
  - Based on mathematical functions
    - Symmetric algorithms: simple operations on bit patterns
- Three common *misconceptions* for public-key encryption
  - More secure than symmetric ones
  - A general-purpose technique that has made symmetric ones obsolete
  - Key distribution is trivial

# Public Key Cryptography

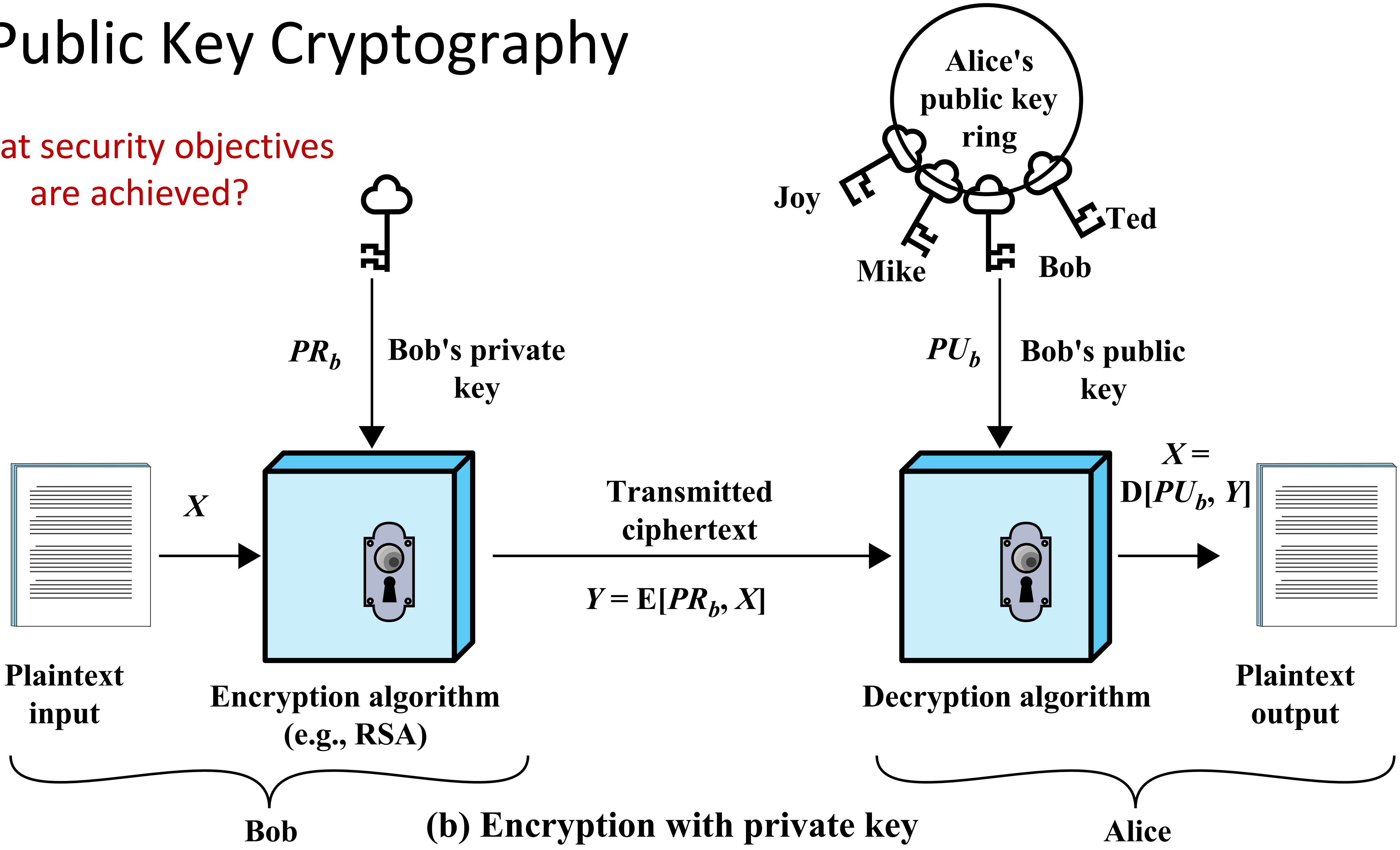
What security objectives are achieved?




(a) Encryption with public key

# Public Key Cryptography

What security objectives are achieved?



(b) Encryption with private key

# Requirements for Public-Key Cryptosystems

- Computationally easy
  - to create key pairs
  - for sender knowing public key to encrypt messages
  - for receiver knowning private key to decrypt ciphertext

- Computationally infeasible for opponent knowing public key
  - to determine private key
  - to recover original message, which is encrypted by public key

- Either of private and public keys can be used for encryption

# Encryption Algorithms and Applications

| Algorithm | Digital Signature | Symmetric Key Distribution | Encryption of Secret Keys |
|---|---|---|---|
| RSA | Yes | Yes | Yes |
| Diffie-Hellman | No | Yes | No |
| DSS | Yes | No | No |
| Elliptic Curve | Yes | Yes | Yes |

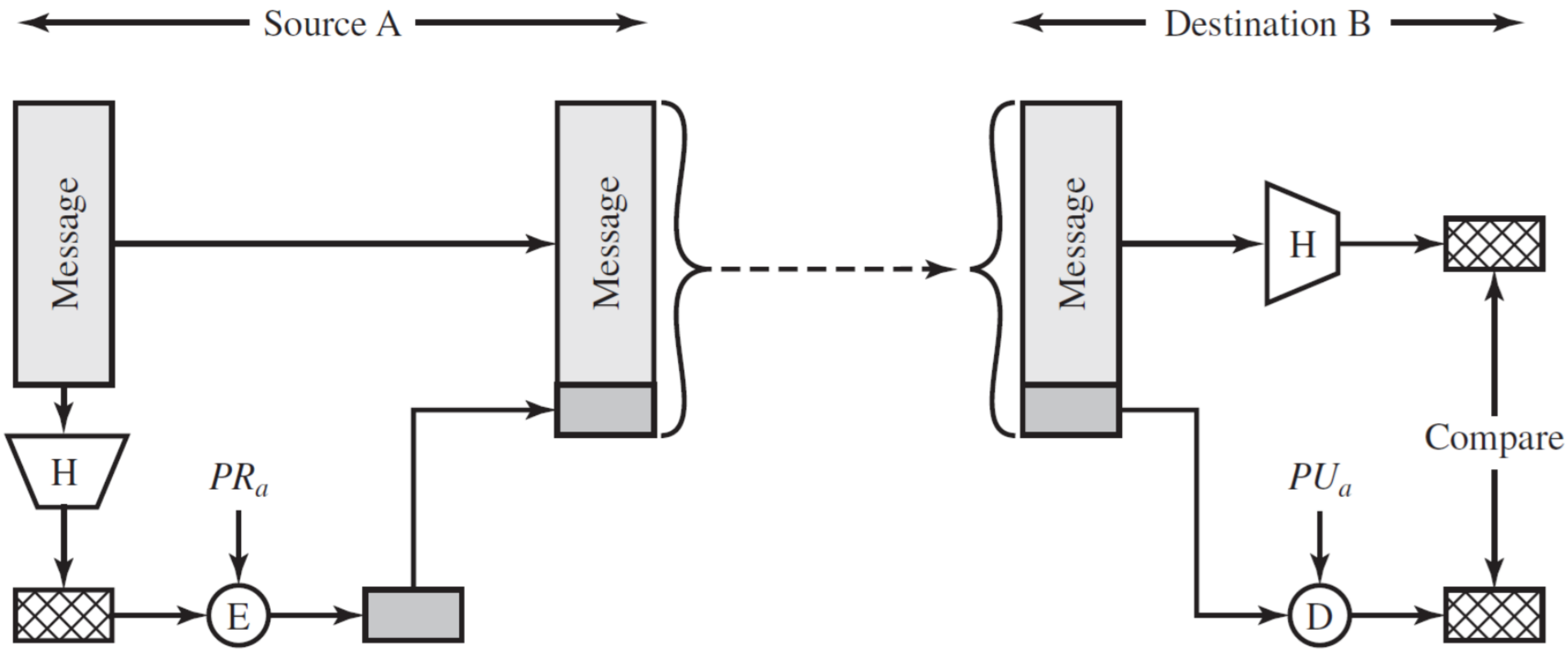
- **RSA (Rivest, Shamir, Adleman)**: most widely accepted and implemented
- **Diffie-Hellman**: first published; limited to the exchange of secret keys
- **DSS (Digital Signature Standard)**: use SHA-1; only the digital signature function
- **Elliptic Curve**: offer equal security for a far smaller bit size

# Digital Signatures and Key Management

- Digital signatures
- Public-key certificates: secure distribution of public keys
- Symmetric key exchange using public-key encryption: temporary key creation for message encrytion
- Digital envelopes: distribution of secret keys

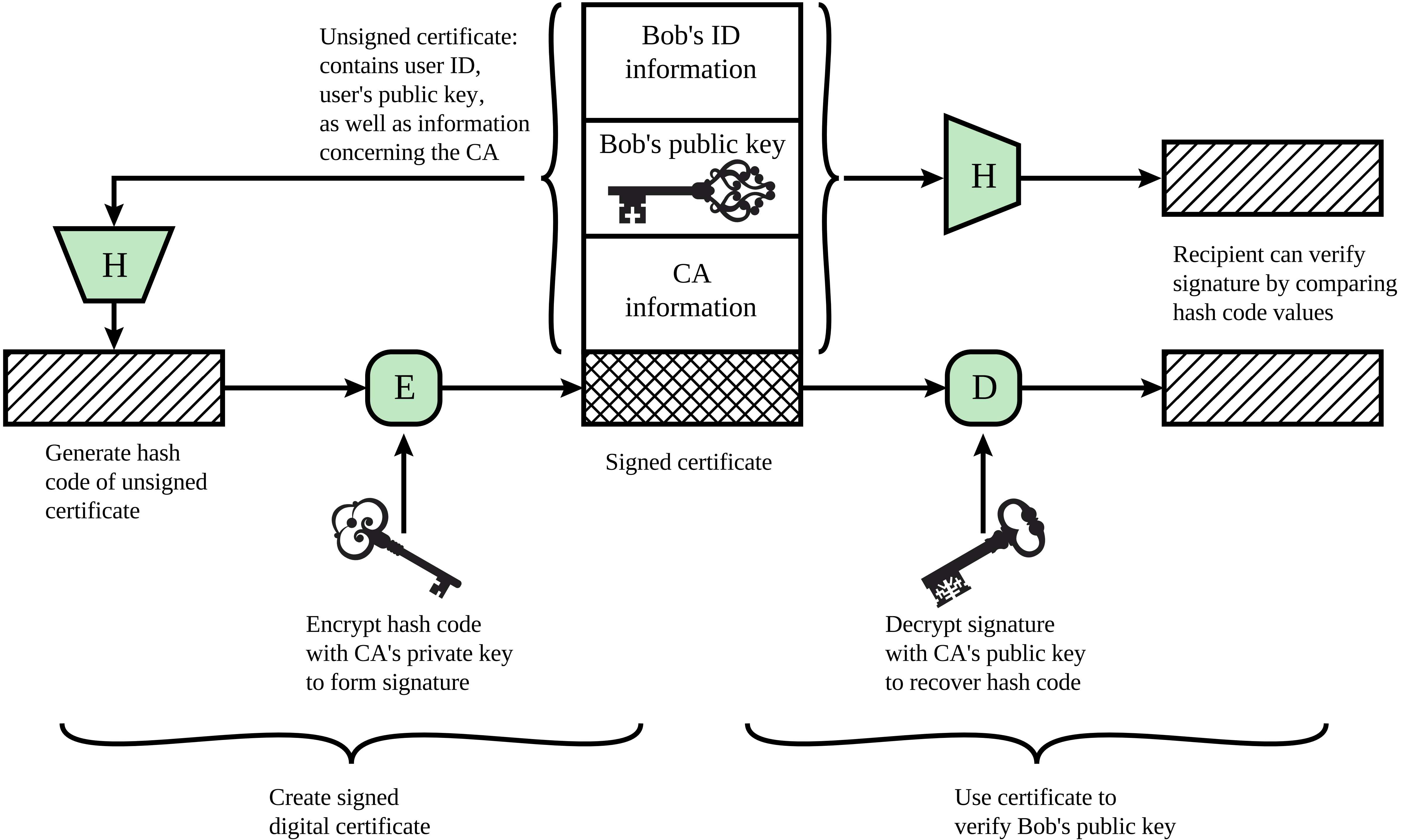
# Digital Signatures

- Encrypts hash code with private key
- What security objectives can be achieved?
- Do they provide confidentiality?



# Public-Key Certificates

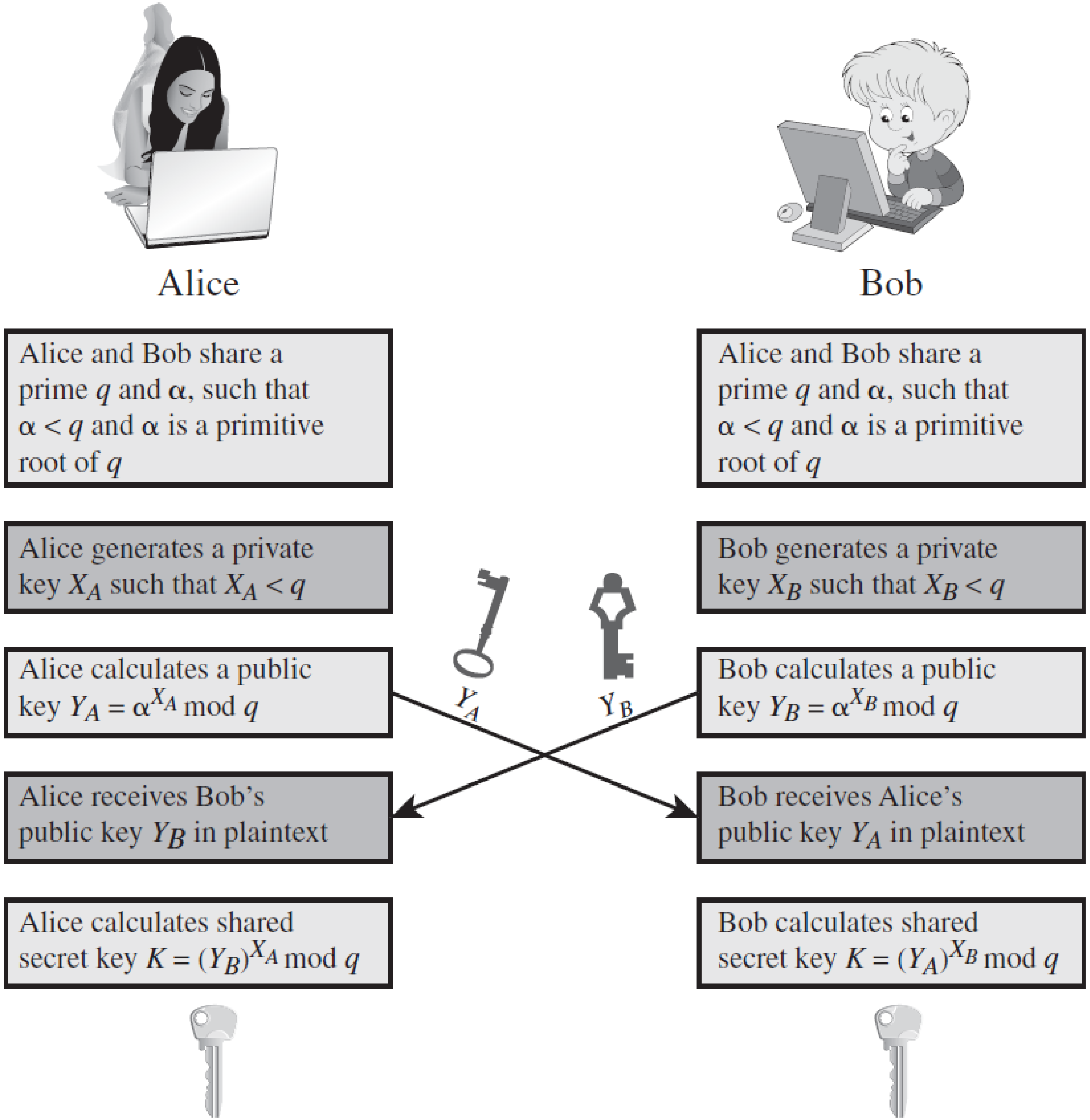
- Public key distribution
  - Any person can release his or her public key
  - But, anyone can forge such a public announcement
- Solution: public-key certificate
  - Certificate: a public key + a user ID of the key owner
  - The whole block signed by a trusted third party, CA (Certificate Authority)
    - CA has to be trusted by the user community (e.g., government)
  - Certificate also includes CA information and validity period
- X.509 standard: universally accepted certificates
  - Used in most network security apps: IPSec, TLS, SSH, S/MIME



Unsigned certificate:
contains user ID,
user's public key,
as well as information
concerning the CA
Bob's ID
information
Bob's public key
CA
information
H
H
Recipient can verify
signature by comparing
hash code values
E
D
Generate hash
code of unsigned
certificate
Signed certificate
Encrypt hash code
with CA's private key
to form signature
Decrypt signature
with CA's public key
to recover hash code
Create signed
digital certificate
Use certificate to
verify Bob's public key

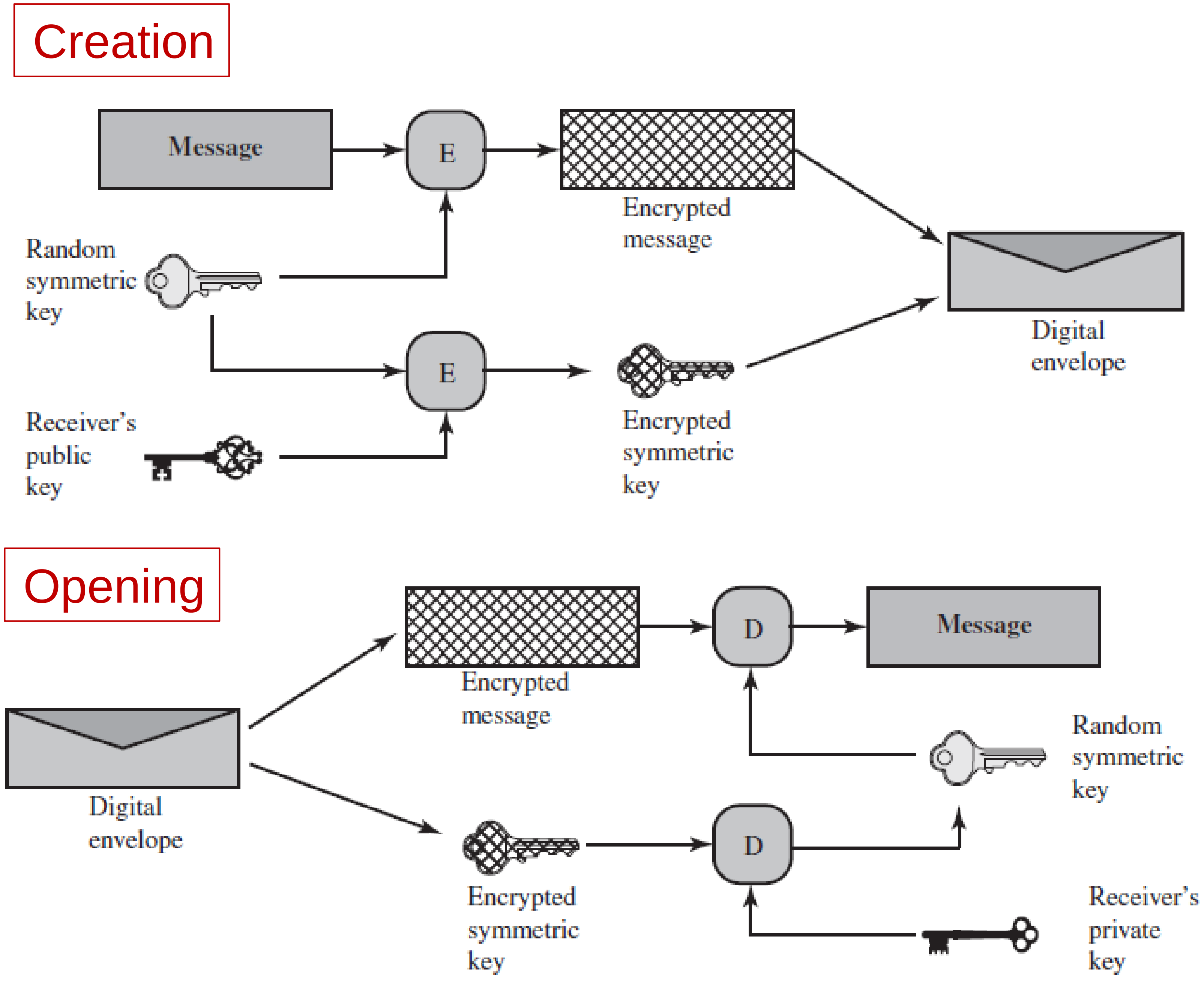
# Symmetric Key Exchange

- Diffie-Hellman key exchange
  - Major drawback: no authentication of two communicating partners
  - Many variations proposed to overcome this problem



# Digital Envelopes

- Protect a message without having the same secret key between sender/receiver
  - Equates to the same thing as a sealed envelope containing a letter
- Can also be used to deliver a symmetric key only



# Random Numbers

- Used for generation of:
  - Keys for public-key algorithms, stream keys for symmetric stream cipher, symmetric keys, etc.
- Two distinct requirements
  - Randomness
    - Uniform distribution: frequency of each number should be roughly the same
    - Independence: no one value can be inferred from the others
  - Unpredictability
    - Not so much that the sequence of numbers be statistically random
    - But, the successive members of the sequence are unpredictable

# Random vs. Pseudorandom

- Pseudorandom: its use is widely accepted
  - Apps typically use deterministic algorithms
  - Sequences: not statistically random, but can satisfy statistical randomness tests

- Random: true random number generator (TRNG)
  - Uses a nondeterministic source to produce randomness
  - Most operate by measuring unpredictable natural processes
    - E.g., radiation, gas discharge, leaky capacitors, thermal noise
  - Increasing provided on modern chips

# Questions?