Computer Security: Principles and Practice

Fourth Edition

By: William Stallings and Lawrie Brown

Chapter 2

Cryptographic Tools

Outline (1/4)

- Classification of the Field of Cryptology
- Symmetric Encryption
- Cryptanalysis and Brute-Force Attack
- Encryption Scheme Security
- Attacking Conventional Encryption
- Categorizing Cryptographic Systems
- Substitution Cipher
- Transpositional Encryption
- Why need bit-oriented ciphers
- Block Cipher vs Stream Cipher
- Padding Bits

Outline (2/4)

- Data Encryption Standard (DES)
- Double-DES
- Meet in the Middle Attack
- Triple DES (3DES)
- Advanced Encryption Standard (AES)
- Block Cipher Modes of Operation
  1. Electronic Codebook Book (ECB)
  2. Cipher Block Chaining (CBC)
  3. Cipher FeedBack (CFB)
  4. Output Feedback (OFB)
  5. Counter (CTR)

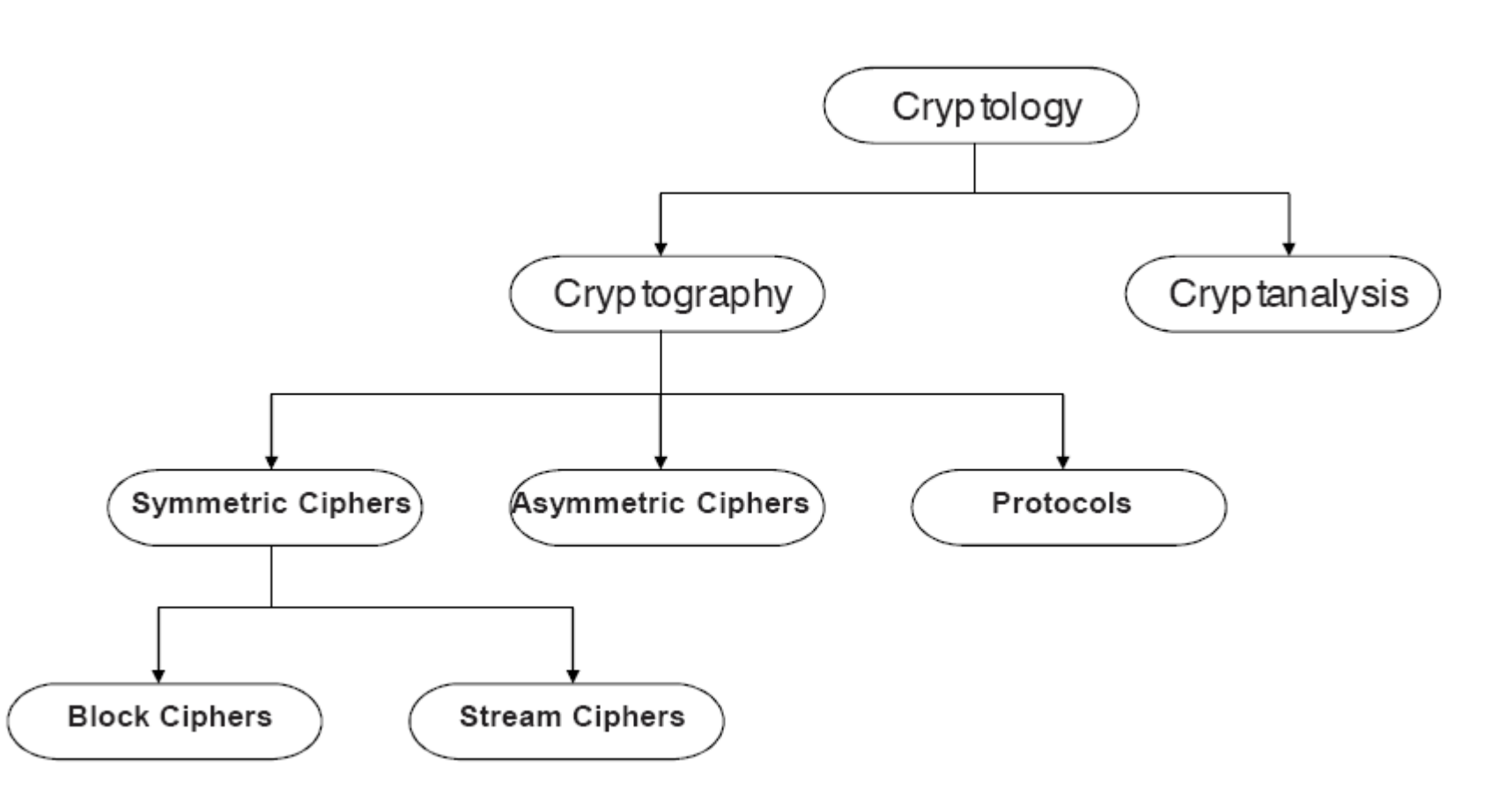
Outline (3/4)

- Modern Stream Cipher
- Random Numbers
- Randomness
- Unpredictability
- Pseudorandom Numbers
- True Random Number Generator (TRNG)
- Pseudorandom Number Generator (PRNG)
- PRNG Requirements
- Link versus end-to-end Encryption
- Key Distribution

Outline (4/4)

- Key-Distribution Center: KDC
- Public-Key Encryption Structure
- Rivest-Shamir-Adleman (RSA) Algorithm
- Diffie-Hellman Example
- Message Authentication
- Message Authentication Code (MAC)
- Digital Signatures
- Certificate Authority (CA) vs. Key Distribution Center (KDC)
- Practical Application: Encryption of Stored Data

Classification of the Field of Cryptology



Symmetric Encryption

- The universal technique for providing confidentiality for transmitted or stored data
- Also referred to as conventional encryption or single-key encryption
- Two requirements for secure use:
  - Need a strong encryption algorithm
  - Sender and receiver must have obtained copies of the secret key in a secure fashion and must keep the key secure

Some Basic Terminology (1/2)

- **Plaintext** - original message
- **Ciphertext** - coded message
- **Cipher** - algorithm for transforming plaintext to ciphertext
- **Key** - info used in cipher known only to sender/receiver
- **Encipher (encrypt)** - converting plaintext to ciphertext

Some Basic Terminology (2/2)

- **Decipher (Decrypt)** - Recovering Plaintext From Ciphertext
- **Cryptography** - Study Of Encryption Principles/Methods
- **Cryptanalysis (Codebreaking)** - Study Of Principles/ Methods Of Deciphering Ciphertext *Without* Knowing Key
- **Cryptology** - Field Of Both Cryptography And Cryptanalysis

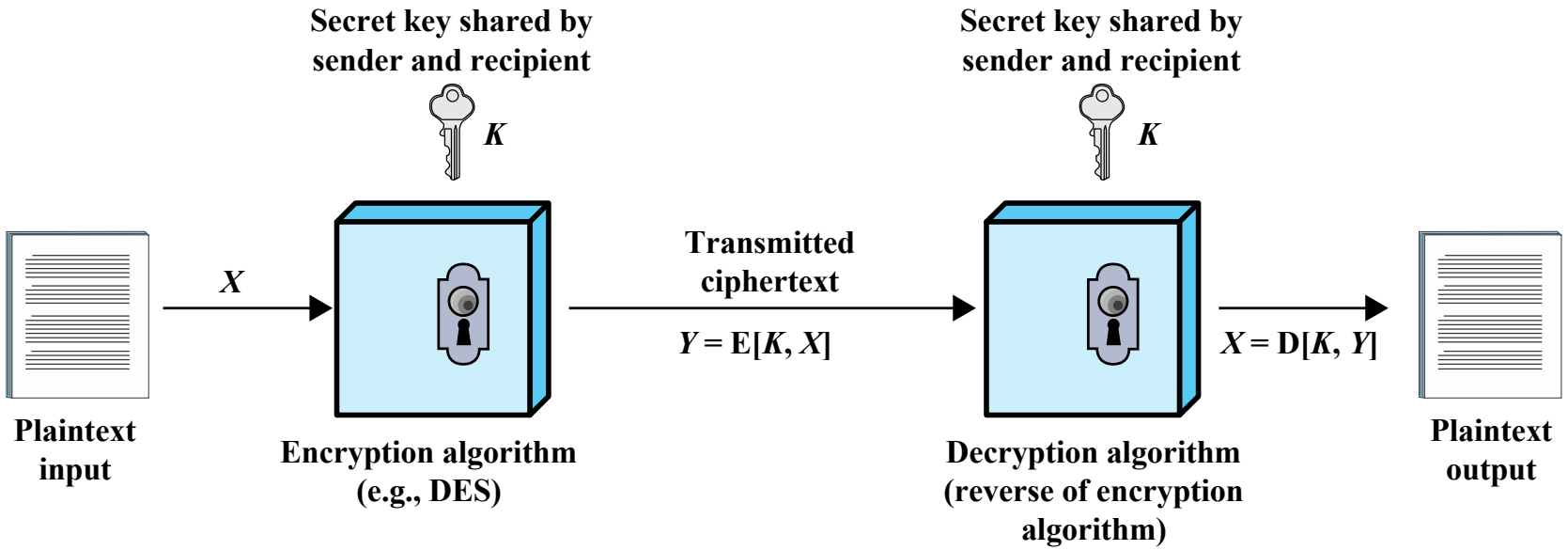


Figure 2.1 Simplified Model of Symmetric Encryption

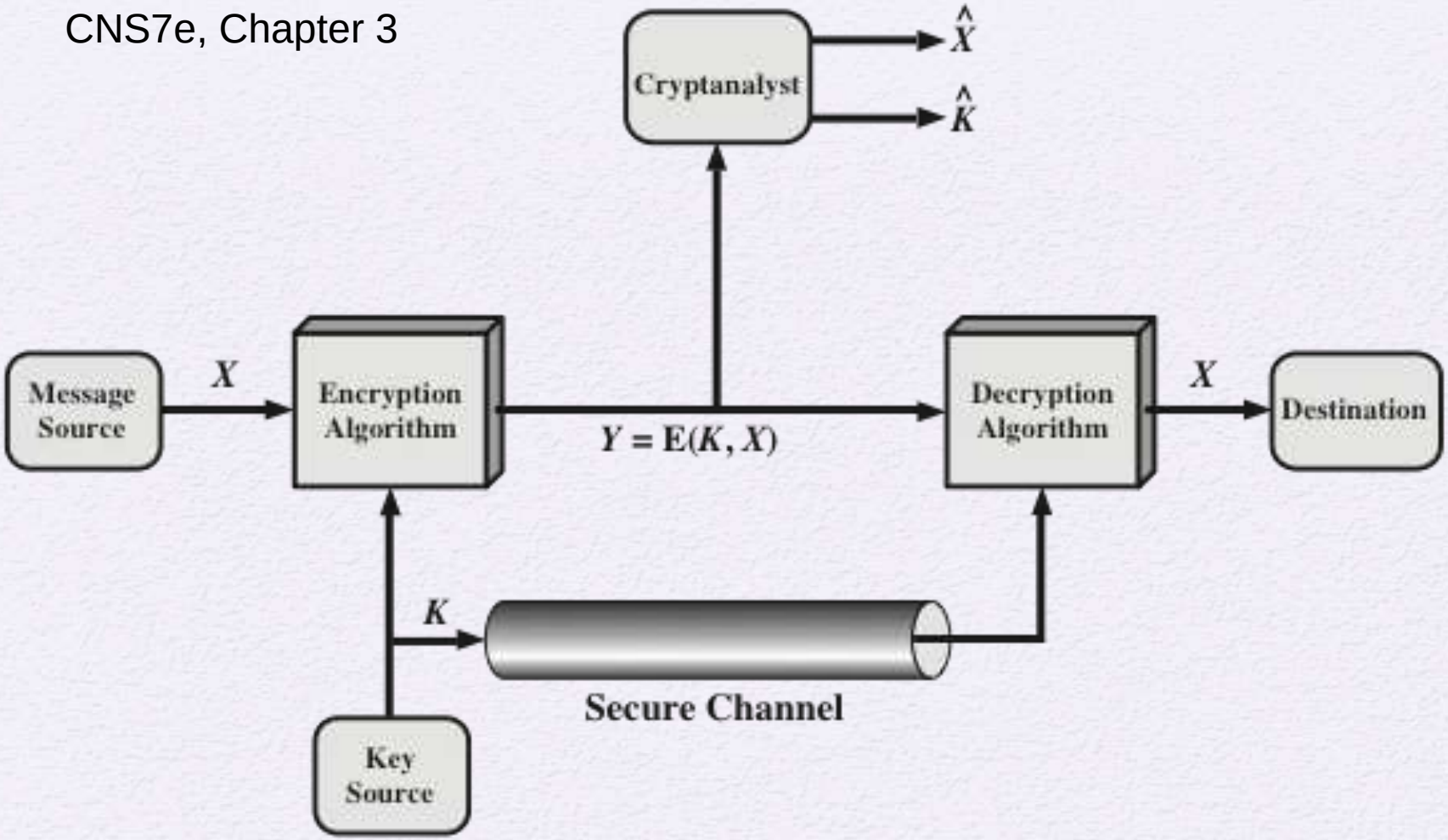


Figure 3.2 Model of Symmetric Cryptosystem

Cryptanalysis and Brute-Force Attack

Cryptanalysis

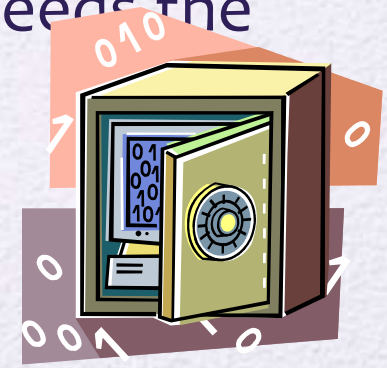
- Attack relies on the nature of the algorithm plus some knowledge of the general characteristics of the plaintext
- Attack exploits the characteristics of the algorithm to attempt to deduce a specific plaintext or to deduce the key being used

Brute-force attack

- Attacker tries every possible key on a piece of ciphertext until an intelligible translation into plaintext is obtained
- On average, half of all possible keys must be tried to achieve success

Encryption Scheme Security

- Unconditionally secure
  - No matter how much time an opponent has, it is impossible for him or her to decrypt the ciphertext simply because the required information is not there
- Computationally secure
  - The cost of breaking the cipher exceeds the value of the encrypted information
  - The time required to break the cipher exceeds the useful lifetime of the information



Attacking Conventional Encryption

- Objective to recover key not just message
- General approaches:
  - Cryptanalytic attack
    - Relay on nature of algorithm plus some knowledge of general characteristics of plain text
    - Attempt to deduce plain text or key
  - Brute-force attack
    - Try every possible key until plain text is achieved

Brute-Force Attack

Involves trying every possible key until an intelligible translation of the ciphertext into plaintext is obtained

On average, half of all possible keys must be tried to achieve success

To supplement the brute-force approach, some degree of knowledge about the expected plaintext is needed, and some means of automatically distinguishing plaintext from garble is also needed

Categorizing Cryptographic Systems

- Characterized along three independent dimensions:

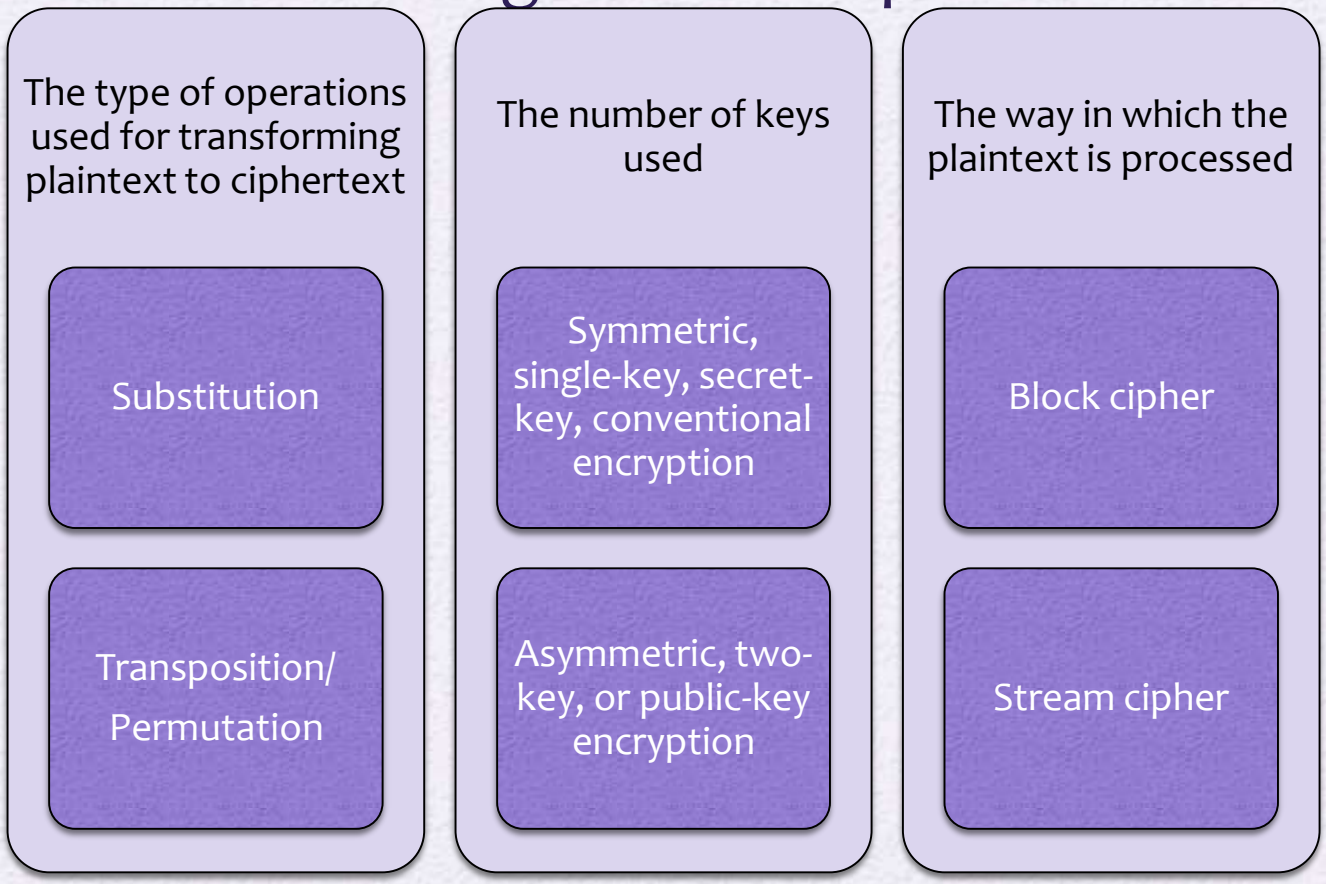


Figure 23-9

Substitution Cipher

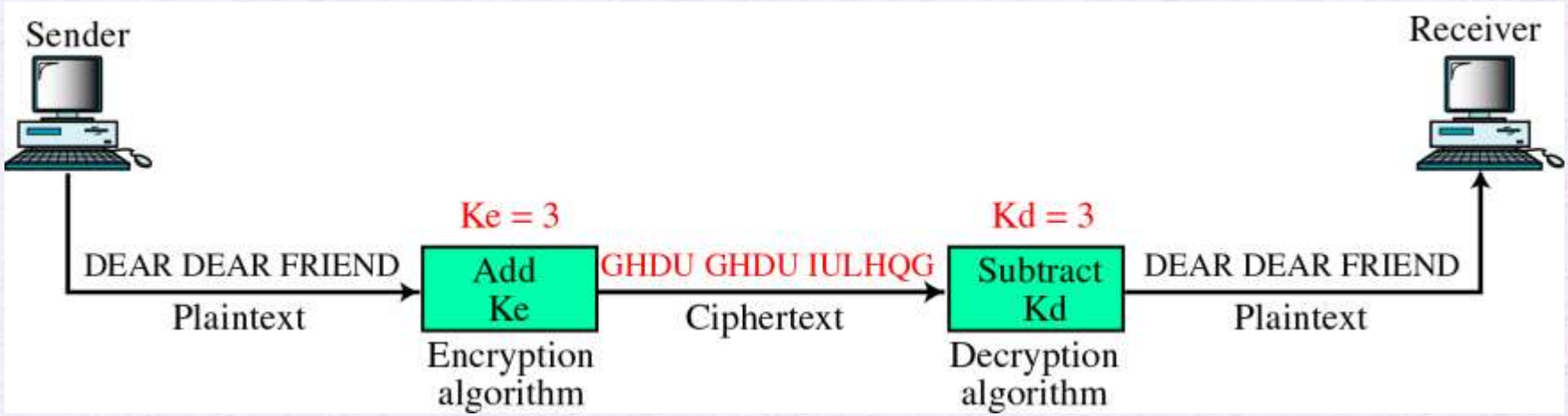
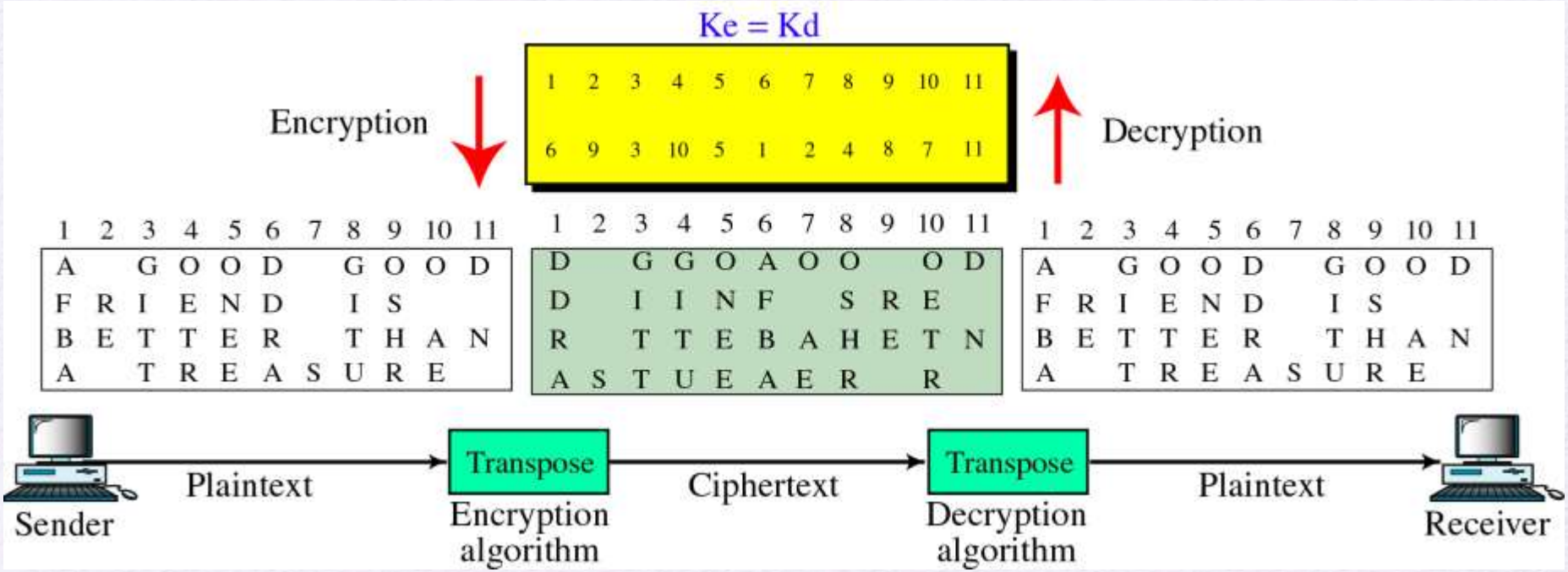
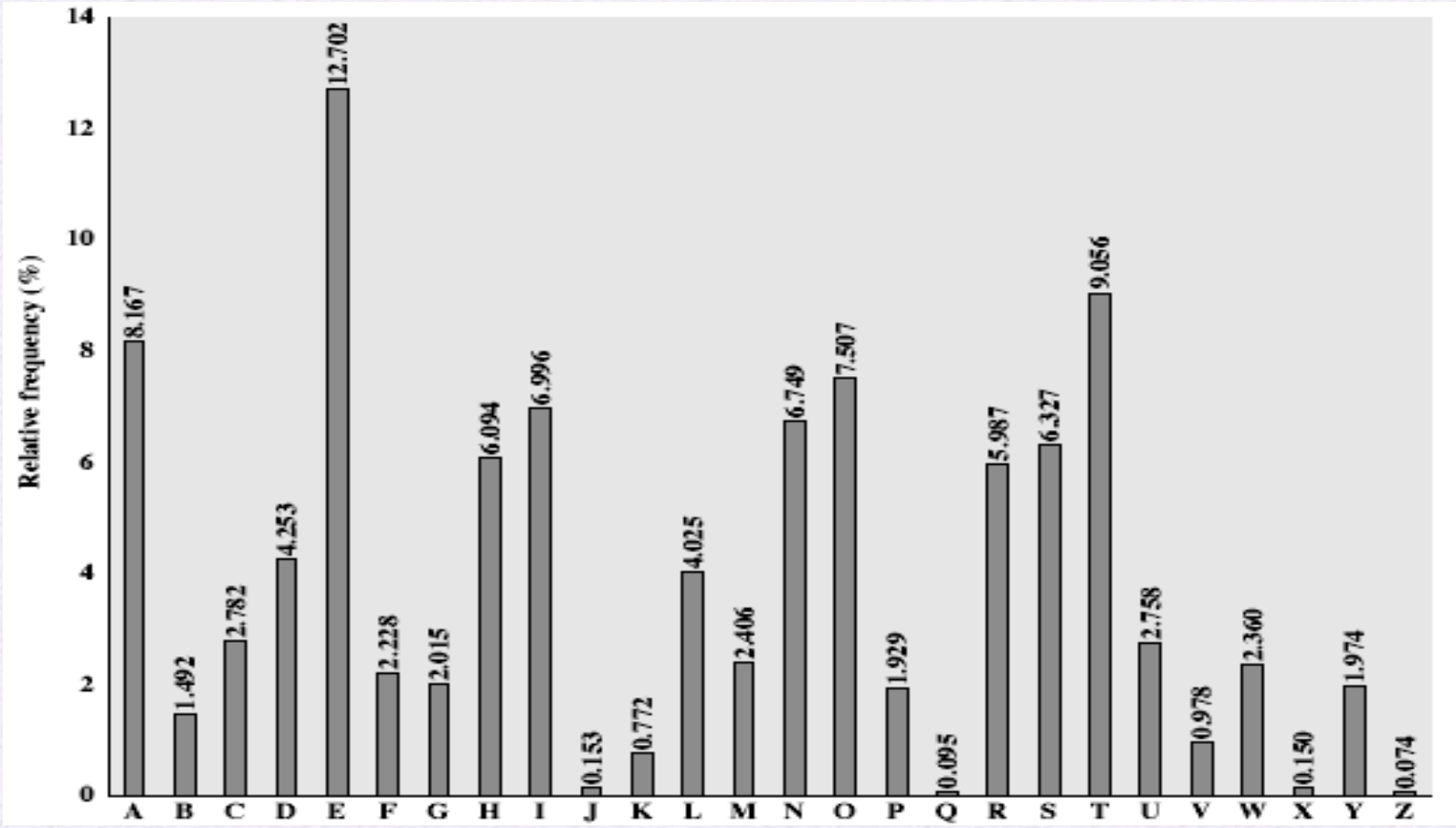


Figure 23-11

Transpositional Encryption



English Letter Frequencies



Why need bit-oriented ciphers

FRCNS1e, ch5, P-123

FRCNS2e, ch5, P-104

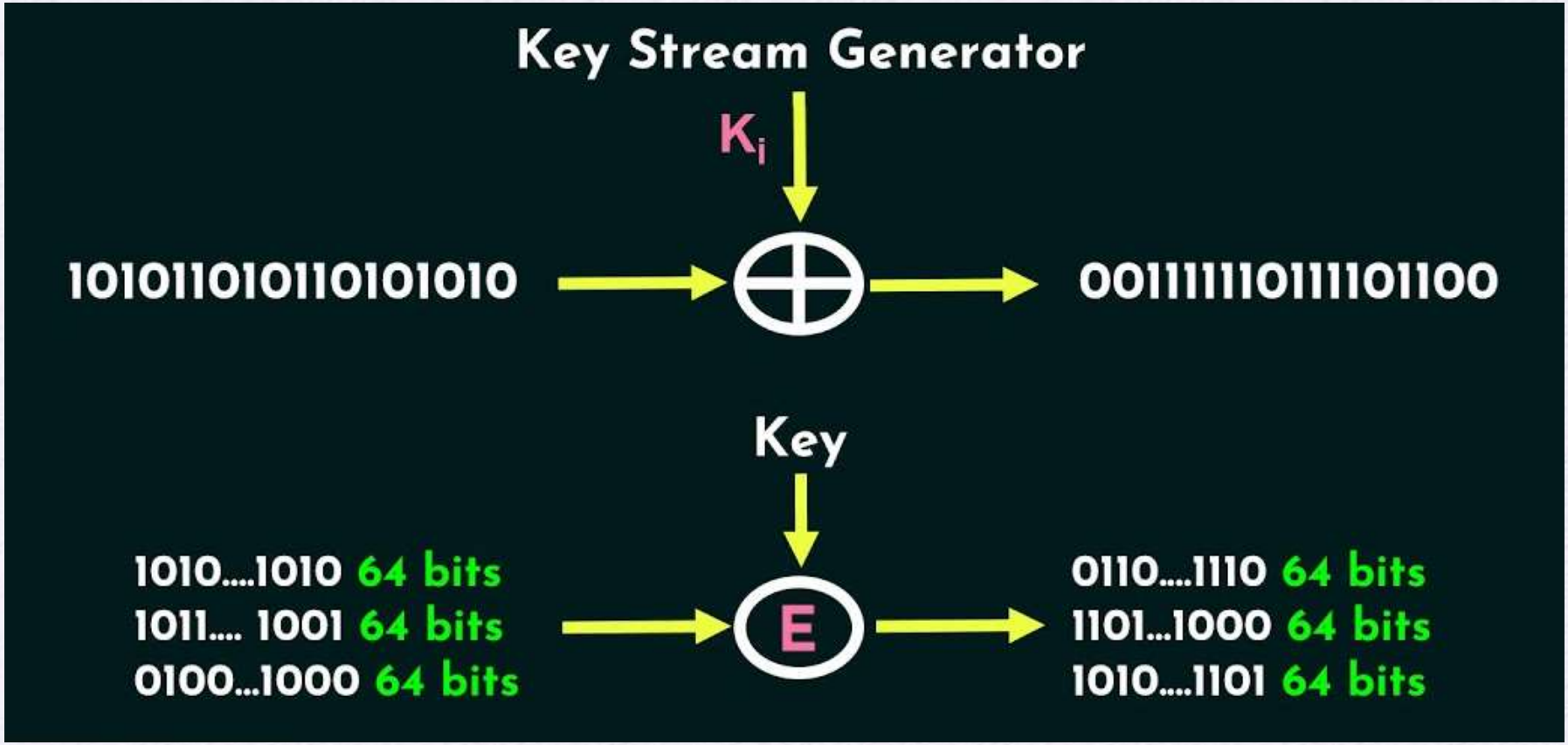
- In previous chapters we discussed *character-oriented ciphers the problem is*
  - Keep language statistics
- Need bit-oriented ciphers
  - Numbers, graphics, audio, and video data
  - Mixing at larger number of symbols increases the security

Block Cipher vs Stream Cipher (1/3)

- Both block and stream ciphers are symmetric key ciphers (like DES, RC4, Blowfish, and AES).
- Block ciphers convert plaintext to ciphertext block by block, while stream ciphers convert one byte at a time.
- Most modern symmetric algorithms are block ciphers, though the block sizes vary (such as DES (64 bits), AES (128 bits), and so on).

https://www.freecodecamp.org/news/what-is-a-block-cipher/

Block Cipher vs Stream Cipher (2/3)

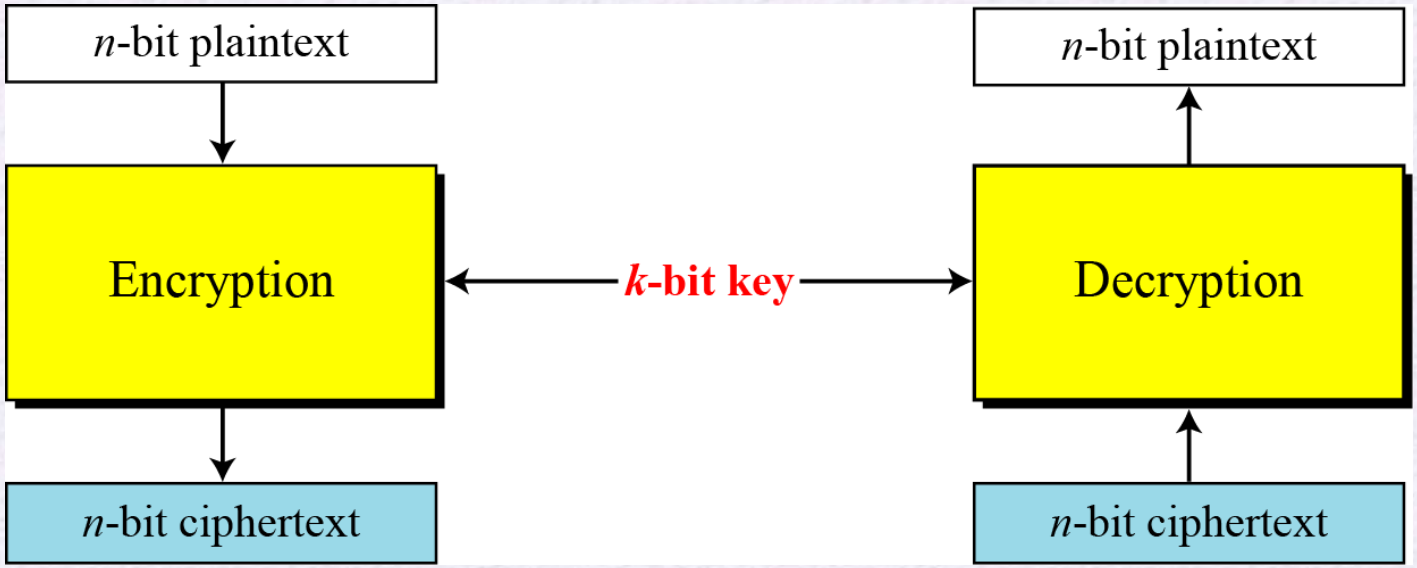


Block Cipher vs Stream Cipher (3/3)

Feature	Block Cipher	Stream Cipher
Processing	Encrypts fixed-size blocks	Encrypts bit-by-bit or byte-by-byte
Speed	Slower but stronger	Faster and efficient for real-time data
Use Case	File encryption, databases, disk encryption	Live streaming, VoIP, real-time encryption
Examples	AES, DES, Blowfish, 3DES	RC4, ChaCha20, Salsa20
Security	More secure	Can be weak if keystream is reused

Modern Block Ciphers

- A symmetric-key modern block cipher encrypts an *n-bit* block of plaintext or decrypts an *n-bit* block of ciphertext. The encryption or decryption algorithm uses a *k-bit* key.



A modern block cipher

Padding Bits

Example 5.1

How many padding bits must be added to a message of 100 characters if 8-bit ASCII is used for encoding and the block cipher accepts blocks of 64 bits?

Solution

Encoding 100 characters using 8-bit ASCII results in an 800-bit message. The plaintext must be divisible by 64. If | M | and |Pad| are the length of the message and the length of the padding,

$$|M| + |Pad| = 0 \bmod 64 \quad \rightarrow \quad |Pad| = -800 \bmod 64 \quad \rightarrow \quad 32 \bmod 64$$

Table 2.1

	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

Table 2.2

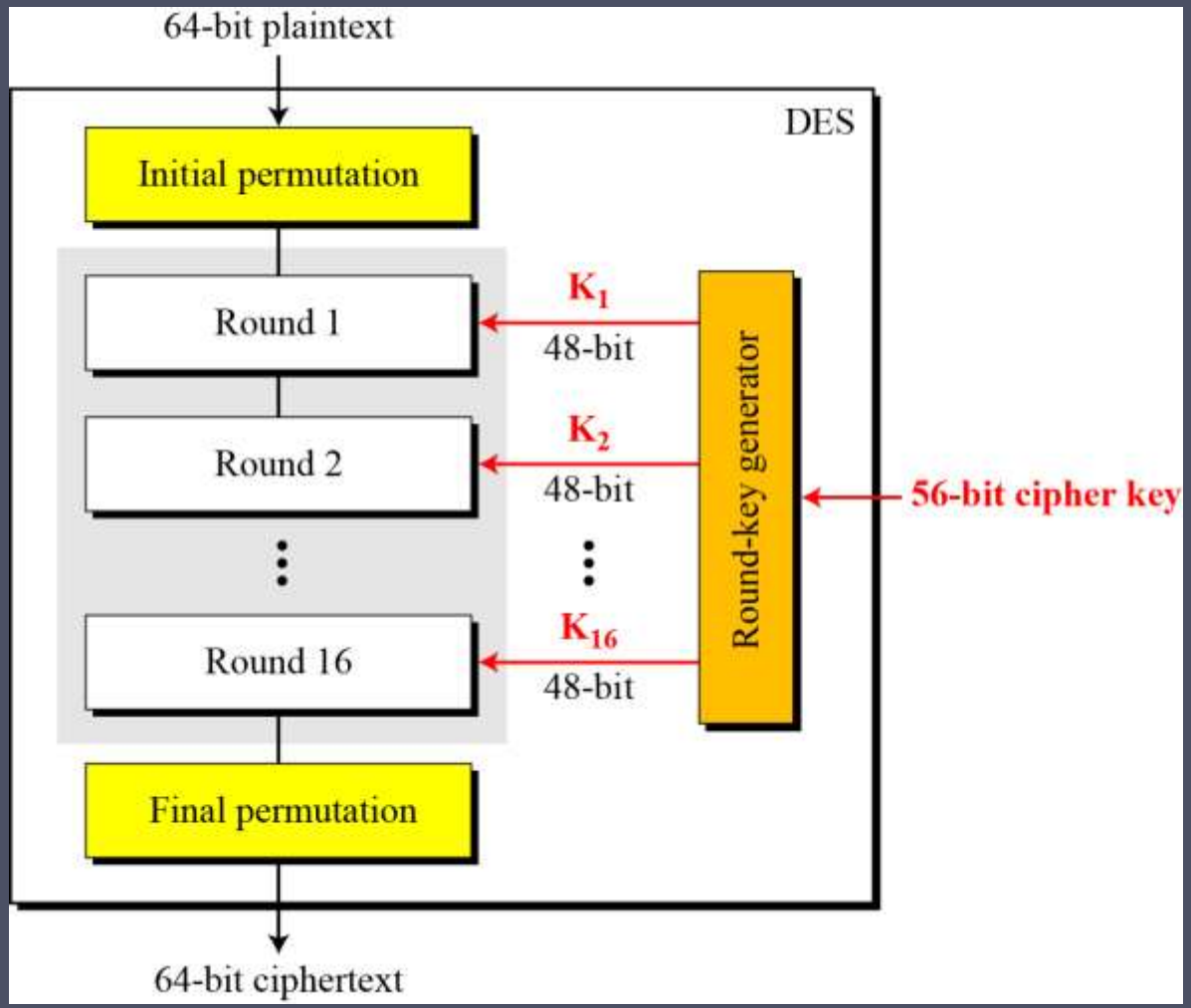
Key size (bits)	Cipher	Number of Alternative Keys	Time Required at 10^9 decryptions/s	Time Required at 10^{13} decryptions/s
56	DES	$2^{56} \approx 7.2 \times 10^{16}$	2^{55} ns = 1.125 years	1 hour
128	AES	$2^{128} \approx 3.4 \times 10^{38}$	2^{127} ns = 5.3×10^{21} years	5.3×10^{17} years
168	Triple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

Average Time Required for Exhaustive Key Search

Data Encryption Standard (DES) (1/3)

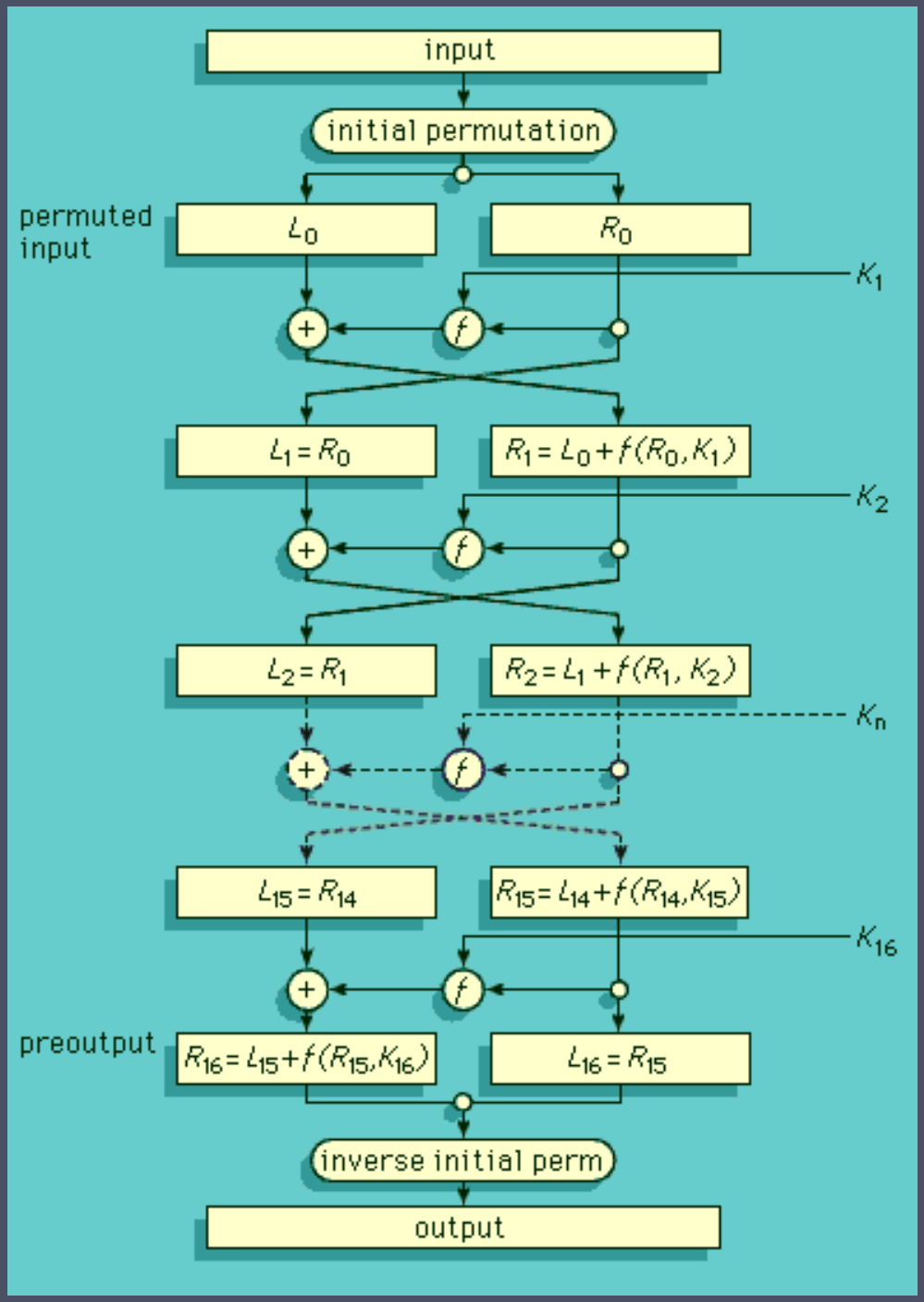
- Until recently was the most widely used encryption scheme
  - FIPS PUB 46
  - Referred to as the Data Encryption Algorithm (DEA)
  - Uses 64 bit plaintext block and 56 bit key to produce a 64 bit ciphertext block
- Strength concerns:
  - Concerns about the algorithm itself
    - DES is the most studied encryption algorithm in existence
  - Concerns about the use of a 56-bit key
    - The speed of commercial off-the-shelf processors makes this key length woefully inadequate

DES (2/3)


64-bit plaintext
DES
Initial permutation
Round 1
Round 2
Round 16
Final permutation
K_1
48-bit
K_2
48-bit
K_{16}
48-bit
Round-key generator
56-bit cipher key
64-bit ciphertext

DES (3/3)

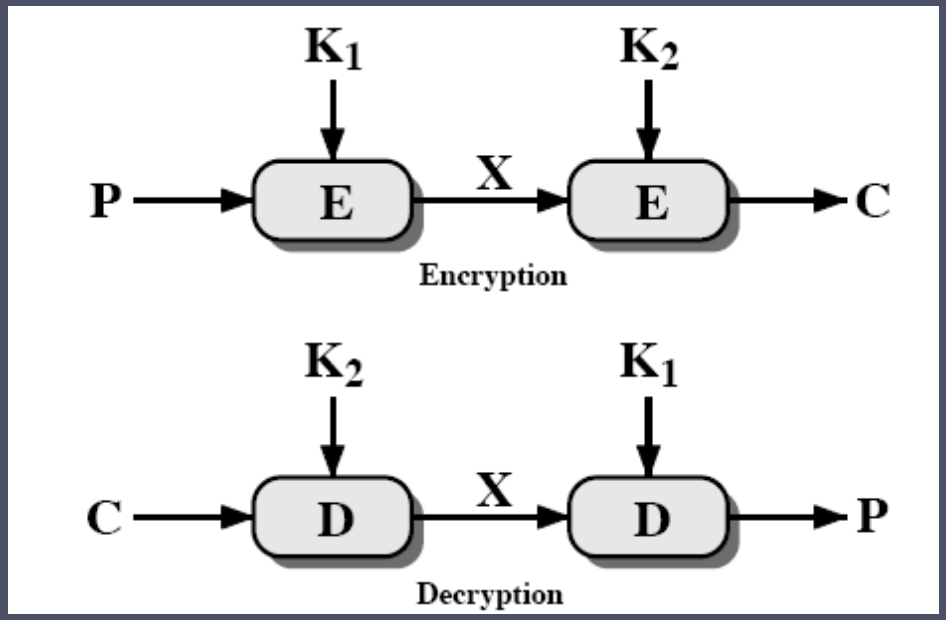
https://www.britannica.com/topic/Data-Encryption-Standard



Double-DES (1/2)

- Two encryption stages, two keys
- Given plaintext P, keys K_1, K_2
- encryption $C = E(K_2, E(K_1, P))$
- decryption $P = D(K_1, D(K_2, C))$
- for decryption, keys applied in reverse order
- Apparently, key length 56×2 = 112 bits

Double-DES (2/2)



$X=E(K_1, P)$

Meet in the Middle Attack (1/2)

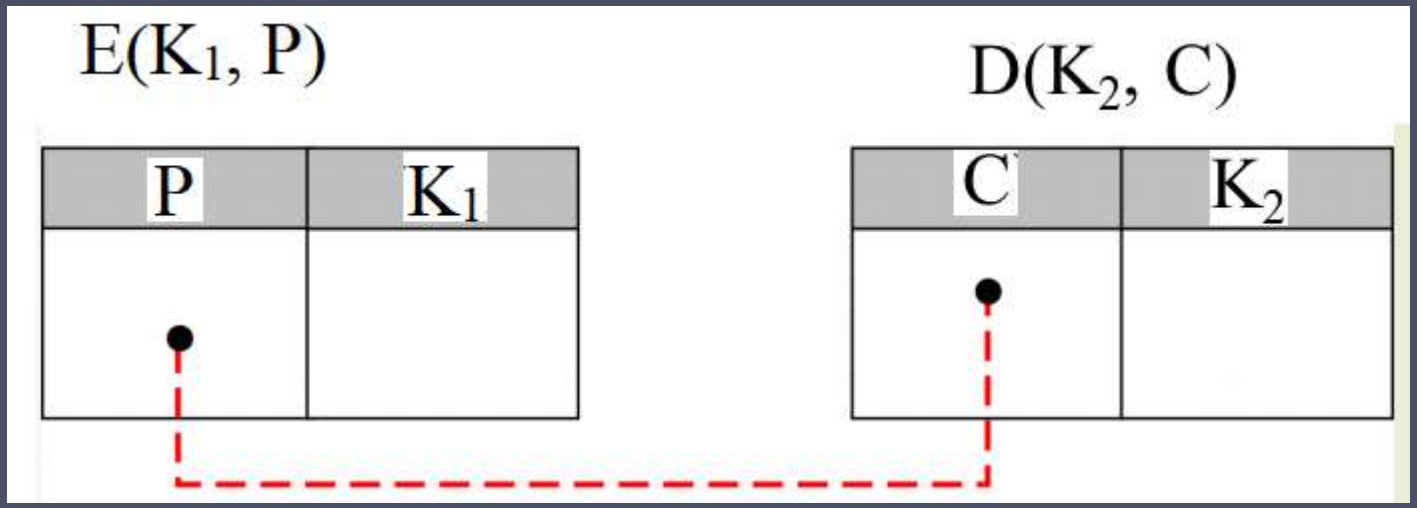
Ref sec-chap-4-1.ppt

- $C = E(K_2, E(K_1, P))$
- $X = E(K_1,P) = D(K_2, C)$

1. Encrypt P for all possible 2^{56} values of K_1
2. Decrypt C for all possible 2^{56} values of K_2
3. Comparing the matching pairs between the outputs from 1 and 2. If a match occurs on the pair (K_1, K_2), check whether this pair can be used for other plain-text and cipher-text pairs.

Meet in the Middle Attack (2/2)

Double-DES

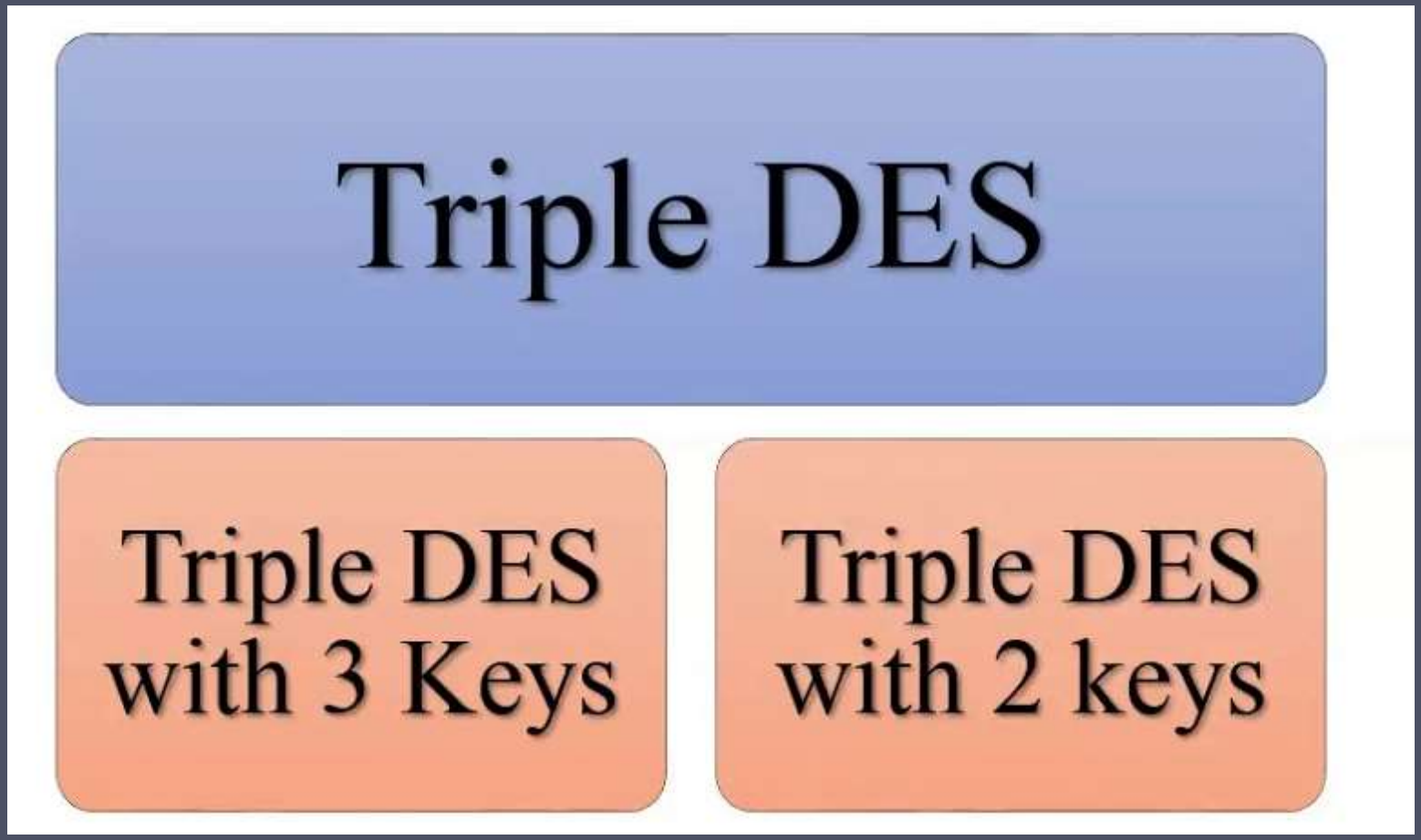


Now compares the values for P and C until we find those pairs of K1 & K2 for which the value of P =C is same in both tables

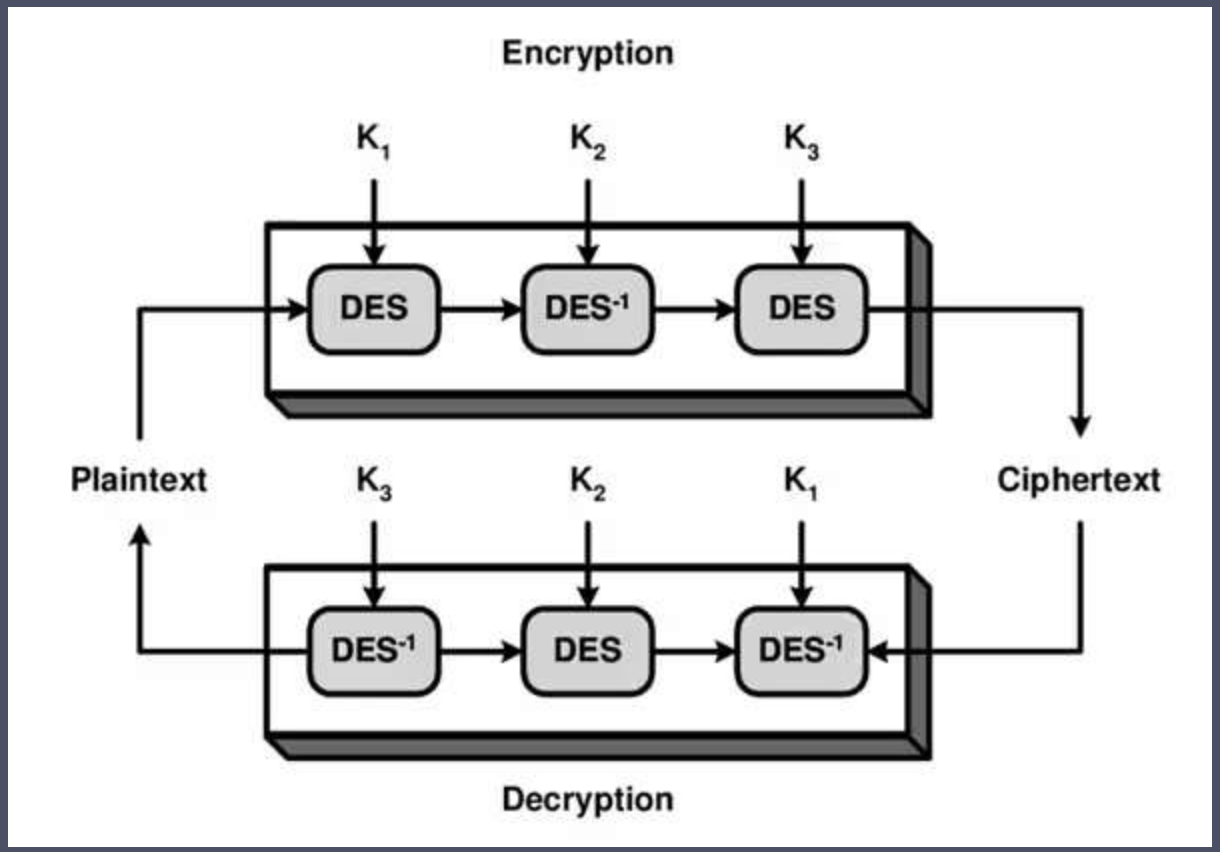
Triple DES (3DES)

- Repeats basic DES algorithm three times using either two or three unique keys
- First standardized for use in financial applications in ANSI standard X9.17 in 1985
- Attractions:
  - 168-bit key length overcomes the vulnerability to brute-force attack of DES
  - Underlying encryption algorithm is the same as in DES
- Drawbacks:
  - Algorithm is sluggish in software
  - Uses a 64-bit block size

Triple DES (3DES) (2/3)



Triple DES with Three Keys

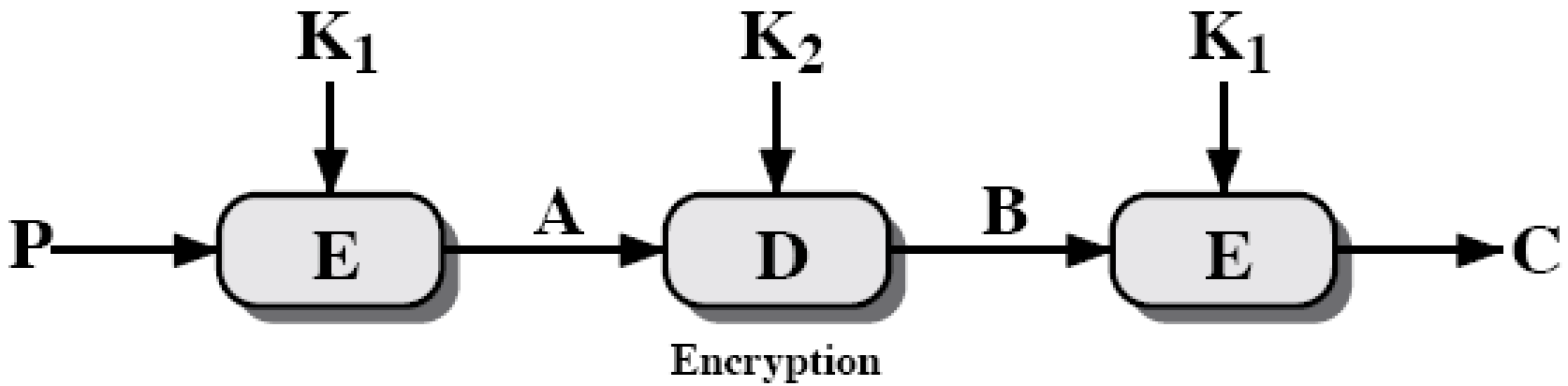


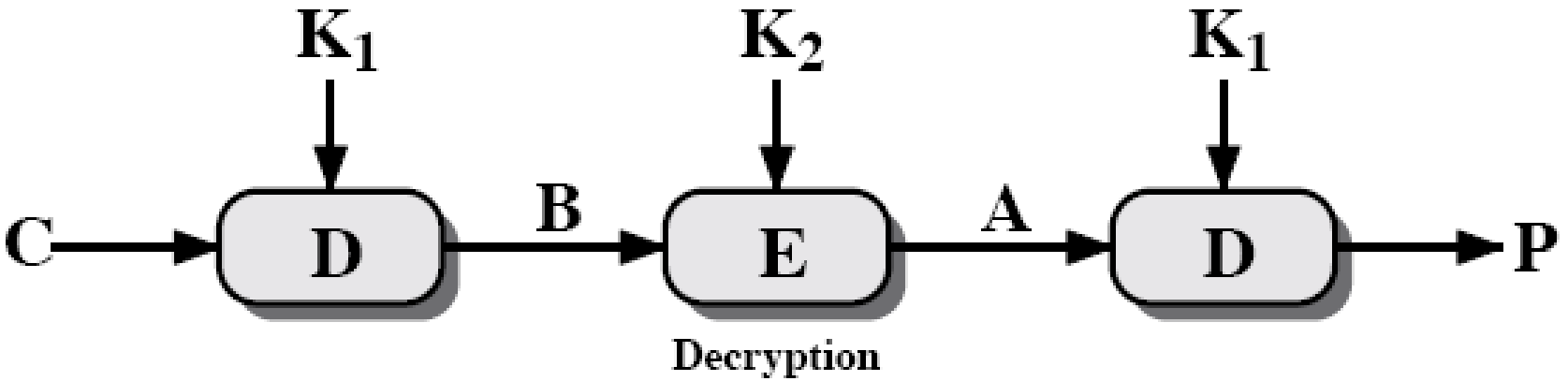
- Why is the middle portion of 3DES a decryption rather than an encryption?
- There is no cryptographic significance to the use of decryption for the second stage.
- Its only advantage is that it allows users of 3DES to decrypt data encrypted by users of the older single DES by repeating the key.

Triple-DES with Two-Keys (1/2)

- To counter meet-in-the-middle attack
- Encrypt-decrypt-encrypt EDE sequence
- $C = E(K_1, D(K_2, E(K_1, P)))$
- Key length 112 bits
- No current cryptanalysis attack on 3DES

Triple DES with Two Keys (1/2)





Triple-DES with Three-Keys

- $C = E(K_3, D(K_2, E(K_1, P)))$
- Key length 168 bit
- EDE → backward compatible with DES
- put $K_3 = K_2 = K_1$

- has been adopted by some Internet applications, e.g. PGP (Pretty Good Privacy) , S/MIME((Secure / Multipurpose Internet Mail Extensions)

A Short History of the Advanced Encryption Standard (AES)

Background – The Need for a New Standard

- In the 1990s, DES became insecure due to advances in computing power.
- Needed a replacement for 3DES
- 3DES was not reasonable for long term use
- Should have a security strength equal to or better than 3DES
- Significantly improved efficiency
- Symmetric block cipher
- 128 bit data and 128/192/256 bit keys

The AES Competition (1997–2000)

- The U.S. NIST sought a stronger encryption standard.
- The goal was to find a modern, efficient, and secure symmetric cipher.
- NIST launched an open international competition in 1997.
- Requirements: strong security, efficiency, and flexibility.
- 15 algorithms were submitted, 5 finalists chosen:
    1. MARS (IBM)
    2. RC6 (RSA Labs)
    3. Rijndael (Belgium)
    4. Serpent (UK/Norway/Israel)
    5. Twofish (US)

Selection of Rijndael

- In October 2000, Rijndael was selected as the winner.
- Designed by Belgian cryptographers Joan Daemen and Vincent Rijmen.
- Officially adopted as AES in 2001 under FIPS PUB 197.
- Chosen for its strong security, speed, and simple structure.

Technical Highlights of AES

- Symmetric block cipher with 128-bit block size.
- Supports key sizes: 128, 192, and 256 bits.
- Based on a substitution–permutation network (SPN).
- Provides high efficiency and resistance to known attacks.

Global Adoption of AES

- Adopted by governments and industries worldwide.
- Used in:
  - SSL/TLS for secure internet communication.
  - Wi-Fi encryption (WPA2/WPA3).
  - Disk and file encryption (BitLocker, VeraCrypt).

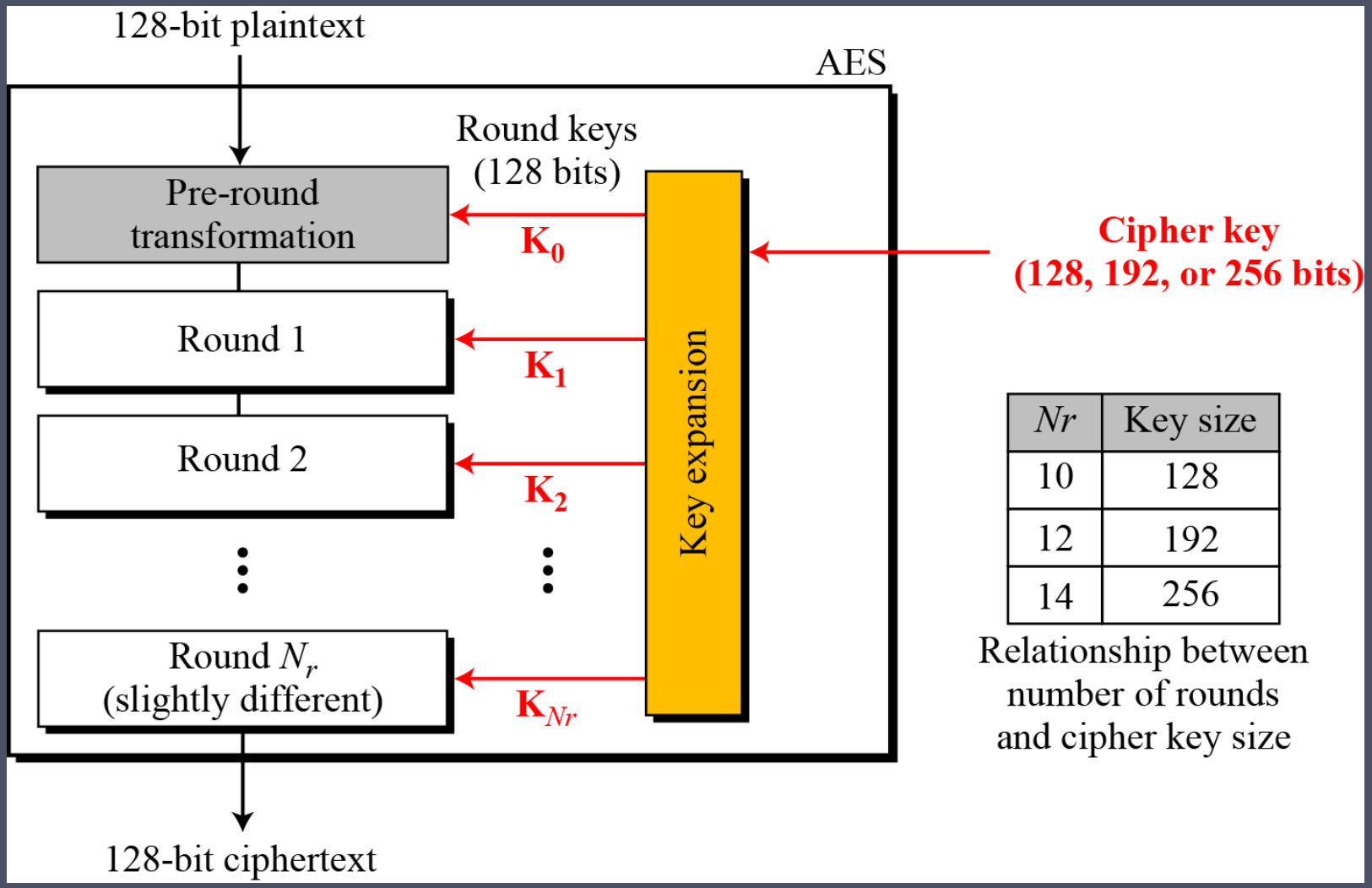
- Recognized as the global standard for encryption.

Legacy of AES

- AES remains the most widely used encryption algorithm today.
- Has resisted cryptanalytic attacks for over two decades.
- Continues to secure sensitive data across the world.

7.1.3 Continue

Figure 7.1 *General design of AES encryption cipher*



Nr	Key size
10	128
12	192
14	256

Relationship between number of rounds and cipher key size

Block Cipher Modes of Operation

- Understanding their Purpose and Applications

https://secgroup.dais.unive.it/teaching/cryptography/block-cipher-modes/

Block cipher modes

- When using block ciphers we have to face the problem of encrypting plaintexts that are longer than the block size.
- We then adopt a mode of operation, i.e., a scheme that repeatedly applies the block cipher and allows for encrypting a plaintext of arbitrary size.
- Electronic codebook (ECB) mode is the simplest approach to multiple-block encryption
  - Each block of plaintext is encrypted using the same key
  - Cryptanalysts may be able to exploit regularities in the plaintext

Introduction to Block Cipher Modes

- Block ciphers encrypt data in fixed-size blocks (e.g., 128 bits for AES).
- To encrypt large data, modes of operation define how multiple blocks are processed.
- Modes ensure security, randomness, and efficiency across encrypted data.

Purpose of Block Cipher Modes

1. Extend encryption to large data sizes.
2. Add randomness using Initialization Vectors (IVs).
3. Control error propagation.
4. Enable parallelism and efficiency.
5. Support stream-like encryption.
6. Provide authenticated encryption (e.g., GCM).

Common Block Cipher Modes

- ECB (Electronic Codebook): Simple but insecure for patterns.
- CBC (Cipher Block Chaining): Secure, sequential processing.
- CFB (Cipher Feedback): Stream-like, self-synchronizing.
- OFB (Output Feedback): Stream-like, no error propagation.
- CTR (Counter Mode): Parallelizable and fast.
- GCM (Galois/Counter Mode): Provides confidentiality and integrity.

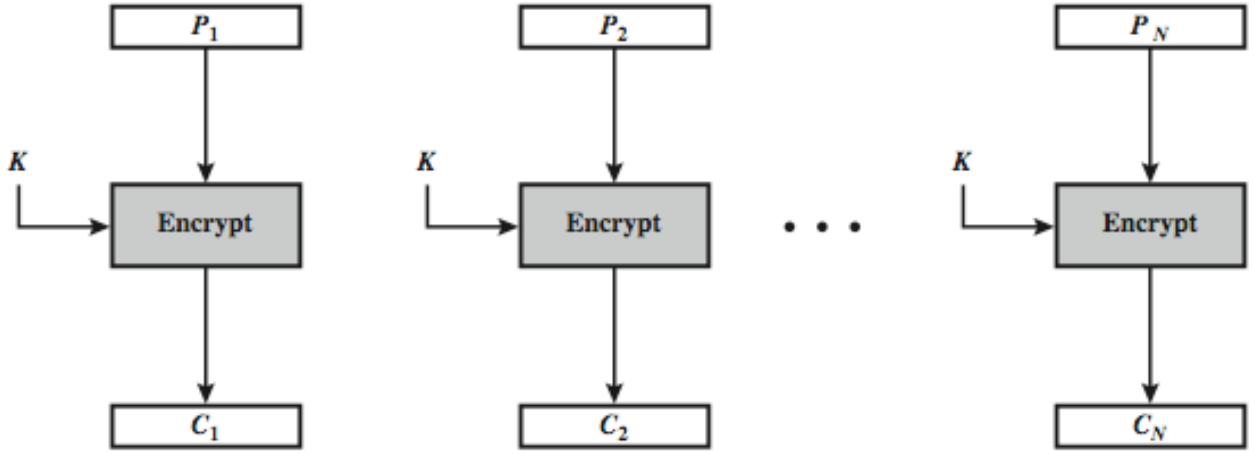
Comparison of Modes

Mode	Type	Key Property	Typical Use
ECB	Deterministic	Simple, insecure for patterns	Not recommended
CBC	Chained	Good security, sequential	File encryption
CFB	Stream-like	Self-synchronizing	Data streams
OFB	Stream-like	No error propagation	Wireless comms
CTR	Stream-like	Parallelizable	High-speed encryption
GCM	Authenticated	Encryption + Integrity	Network protocols

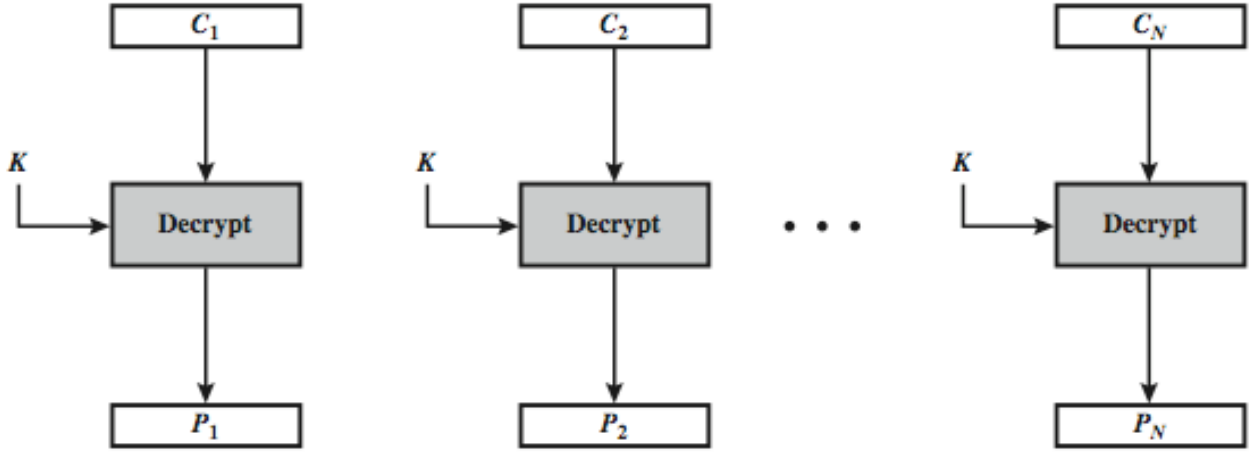
Summary

- Block cipher modes make encryption secure and practical.
- They define how blocks are linked, adding randomness and efficiency.
- Modern applications use advanced modes like GCM for confidentiality and integrity.

Electronic Codebook Book (ECB)



(a) Encryption

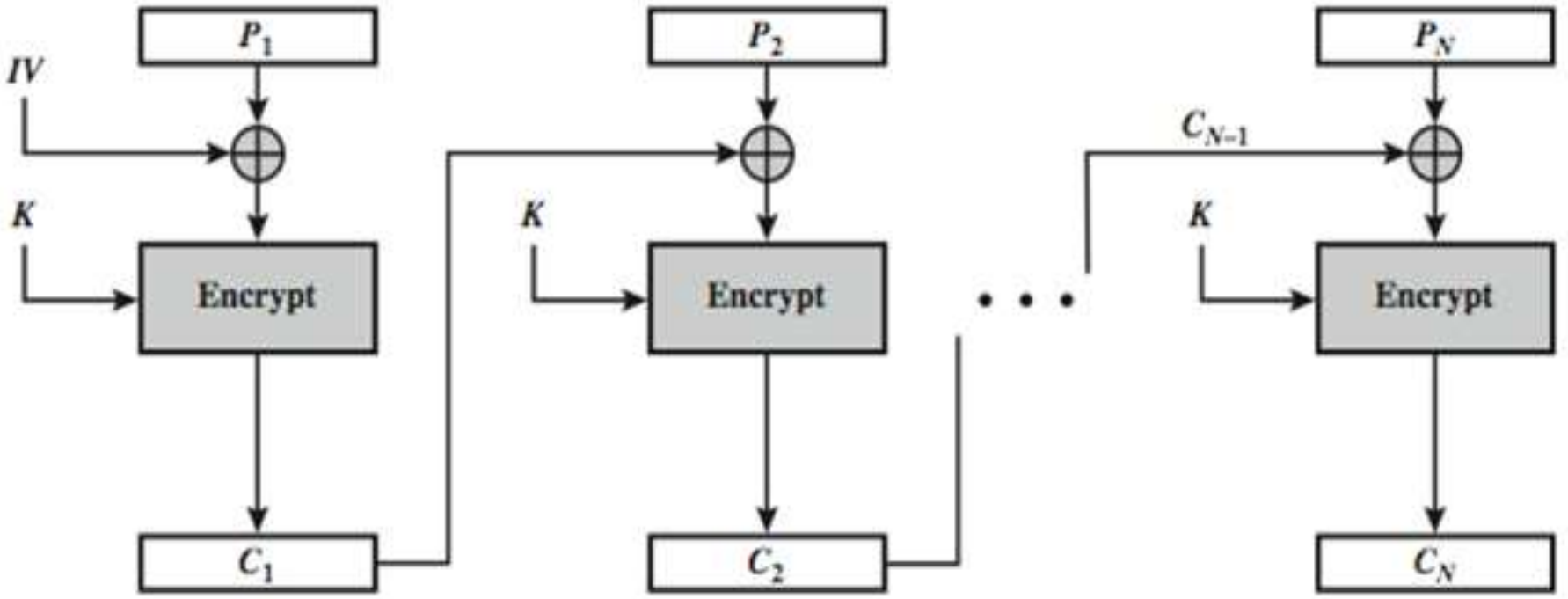


(b) Decryption

Practical Security Issues

- Typically symmetric encryption is applied to a unit of data larger than a single 64-bit or 128-bit block
- Modes of operation
  - Alternative techniques developed to increase the security of symmetric block encryption for large sequences
  - Overcomes the weaknesses of ECB

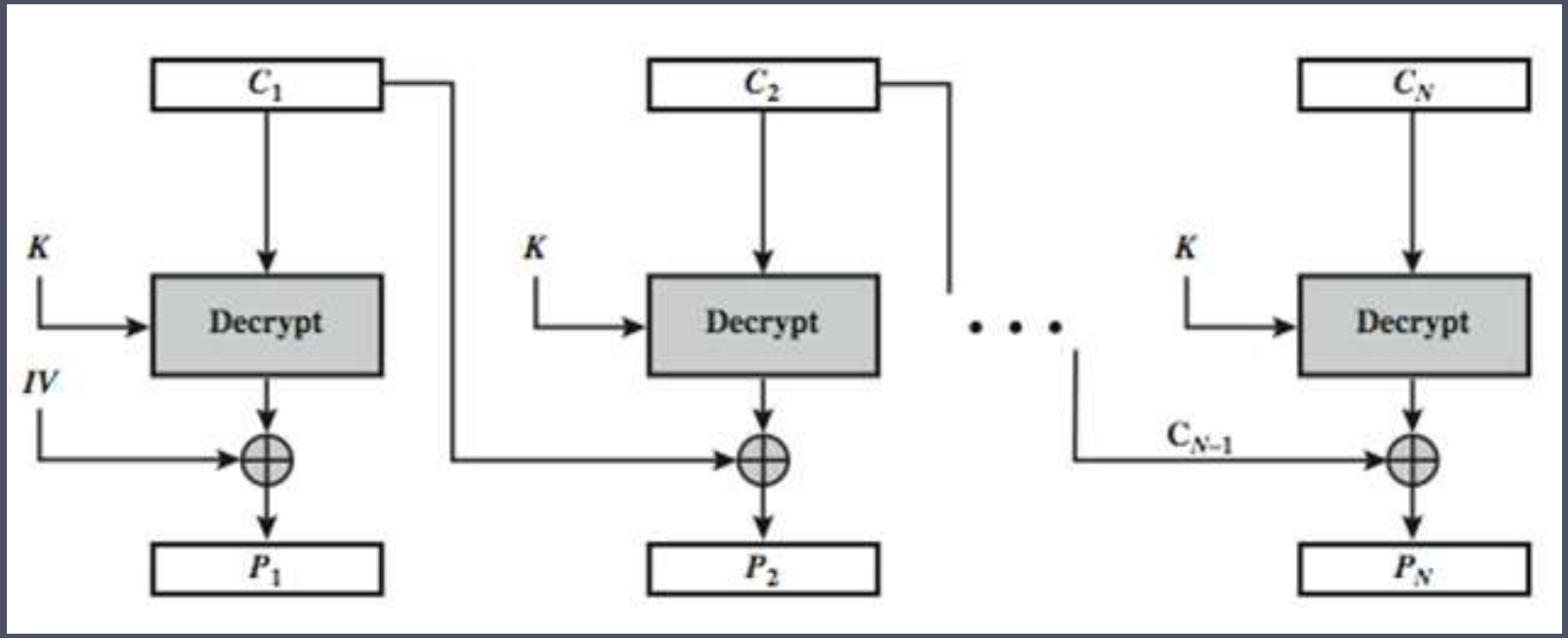
Cipher Block Chaining (CBC)



Encryption (1/3)

Dr. Hanif Durad

Cipher Block Chaining (CBC)



Decryption (2/3)

Cipher Block Chaining (CBC) (3/3)

- Encryption

  $C_1 = E(K, IV \otimes P_1)$

  $C_i = E(K, C_{i-1} \otimes P_i)$

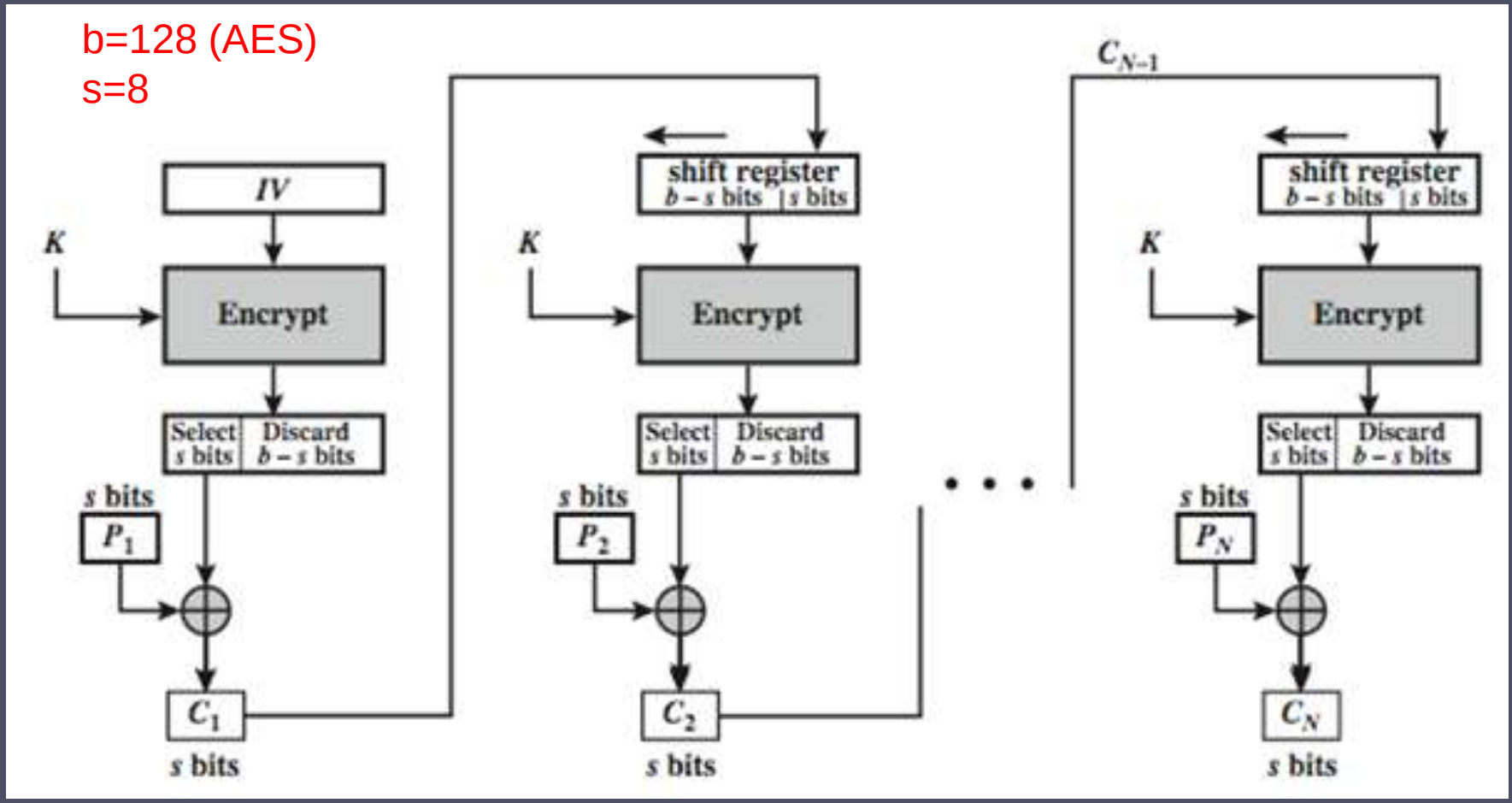
- Decryption

  $P_1 = IV \otimes D(K, C_1)$

  $P_i = C_{i-1} \otimes D(K, C_i)$

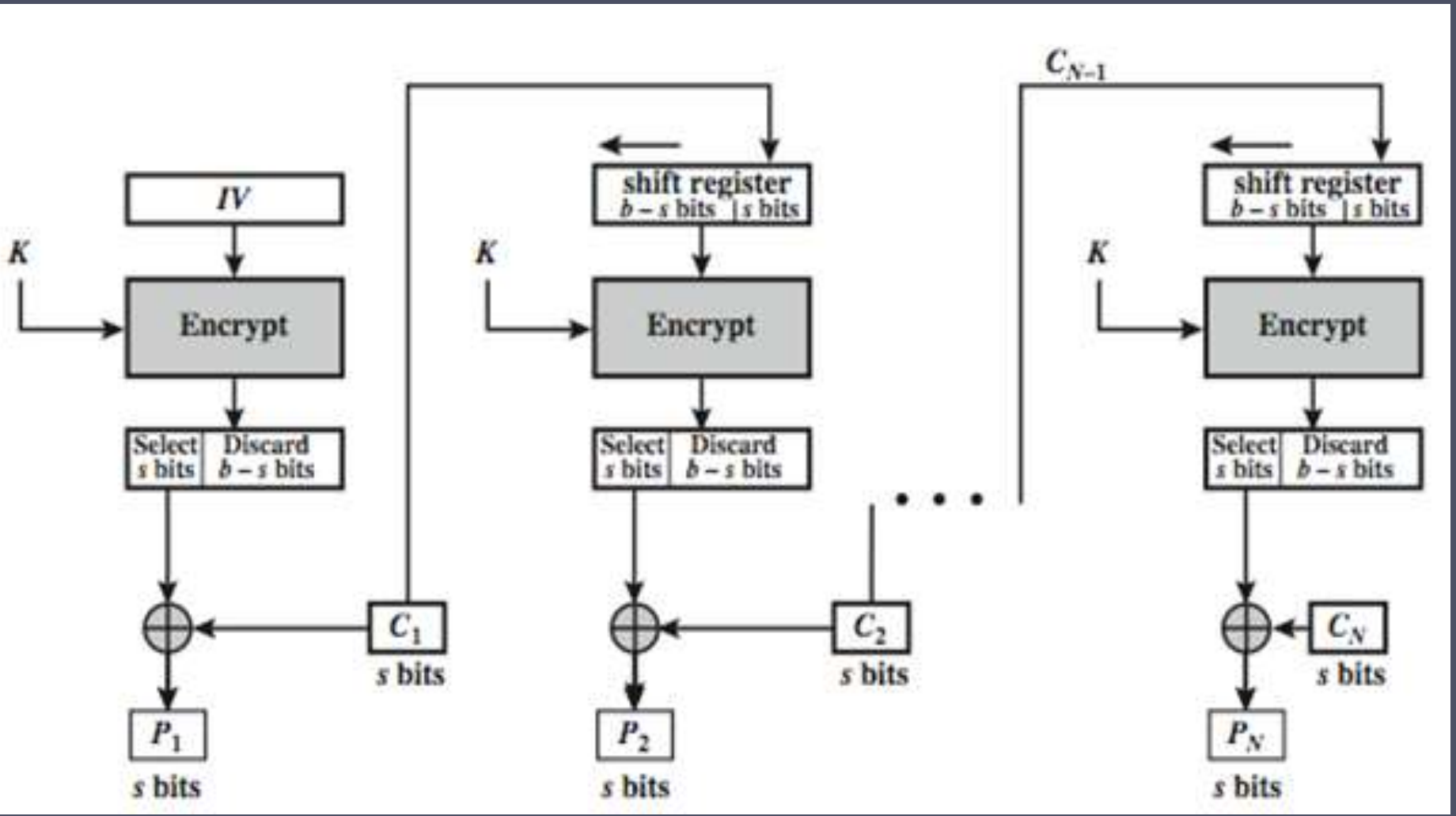
(3/3)

Cipher FeedBack (CFB)



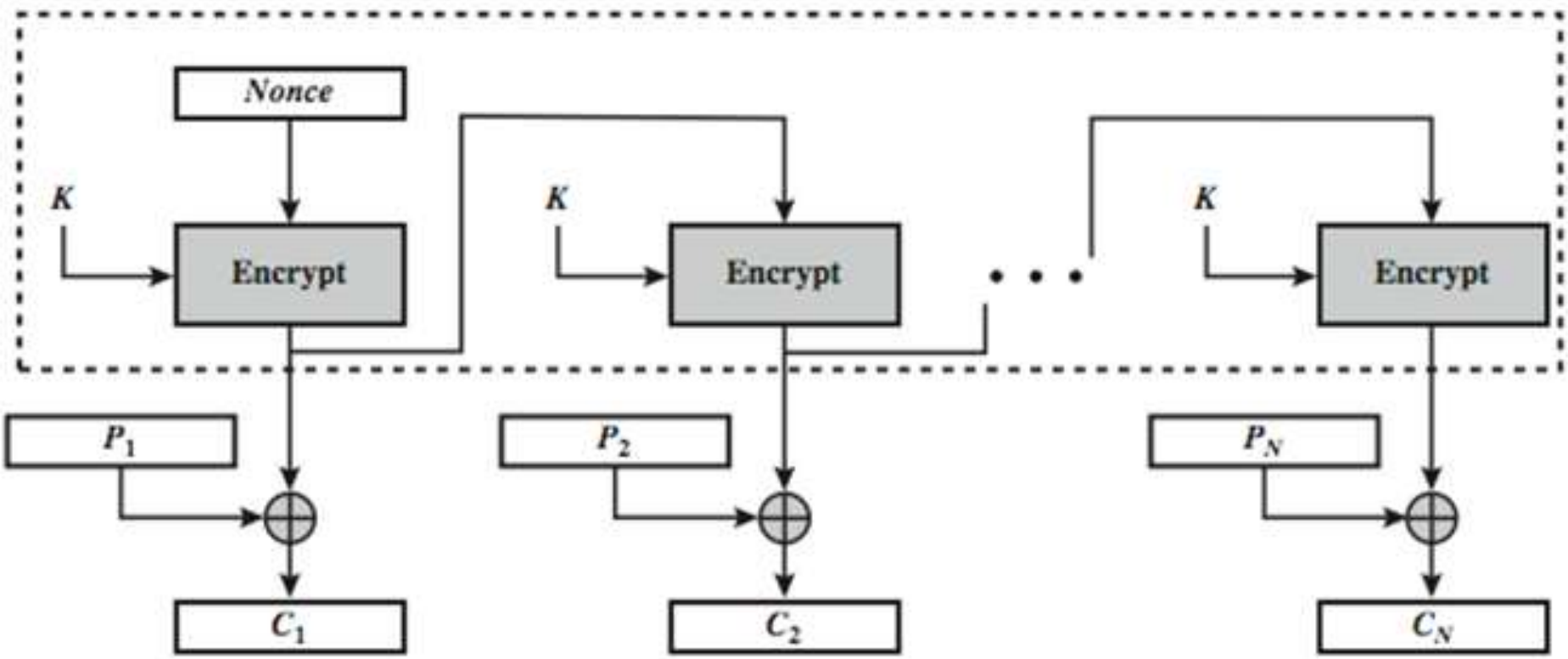
Encryption (4/7)

Cipher Feedback (CFB)



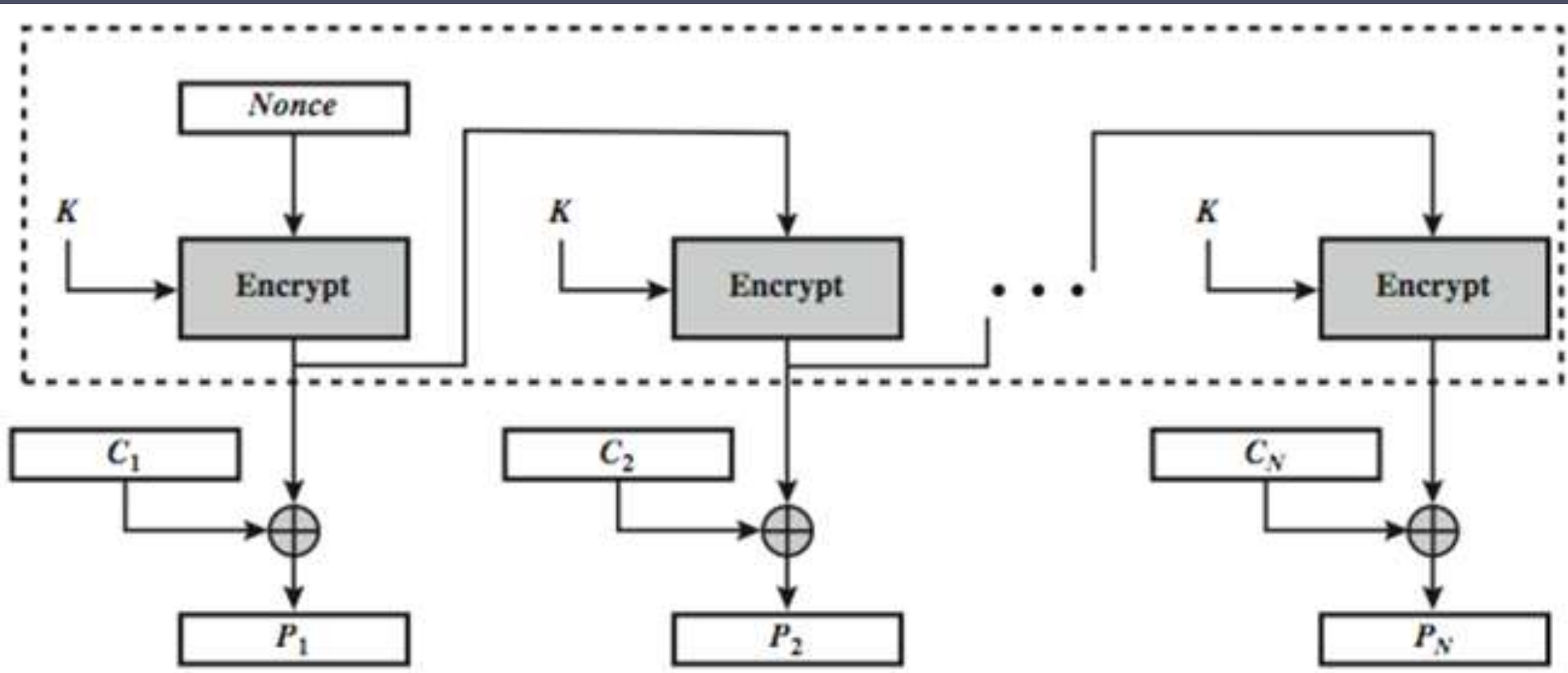
Decryption (6/7)

Output Feedback (OFB)



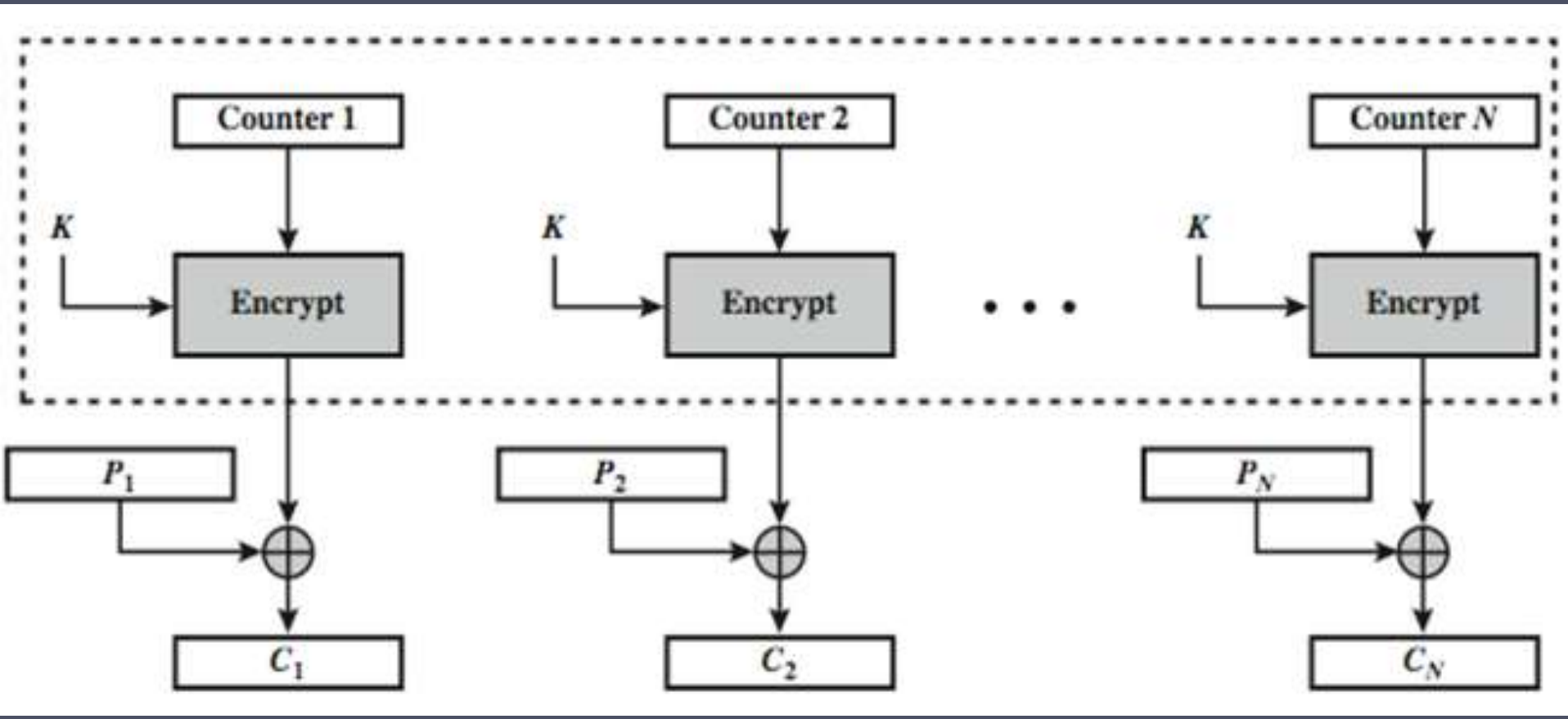
Encryption (1/2)

Output Feedback (OFB)



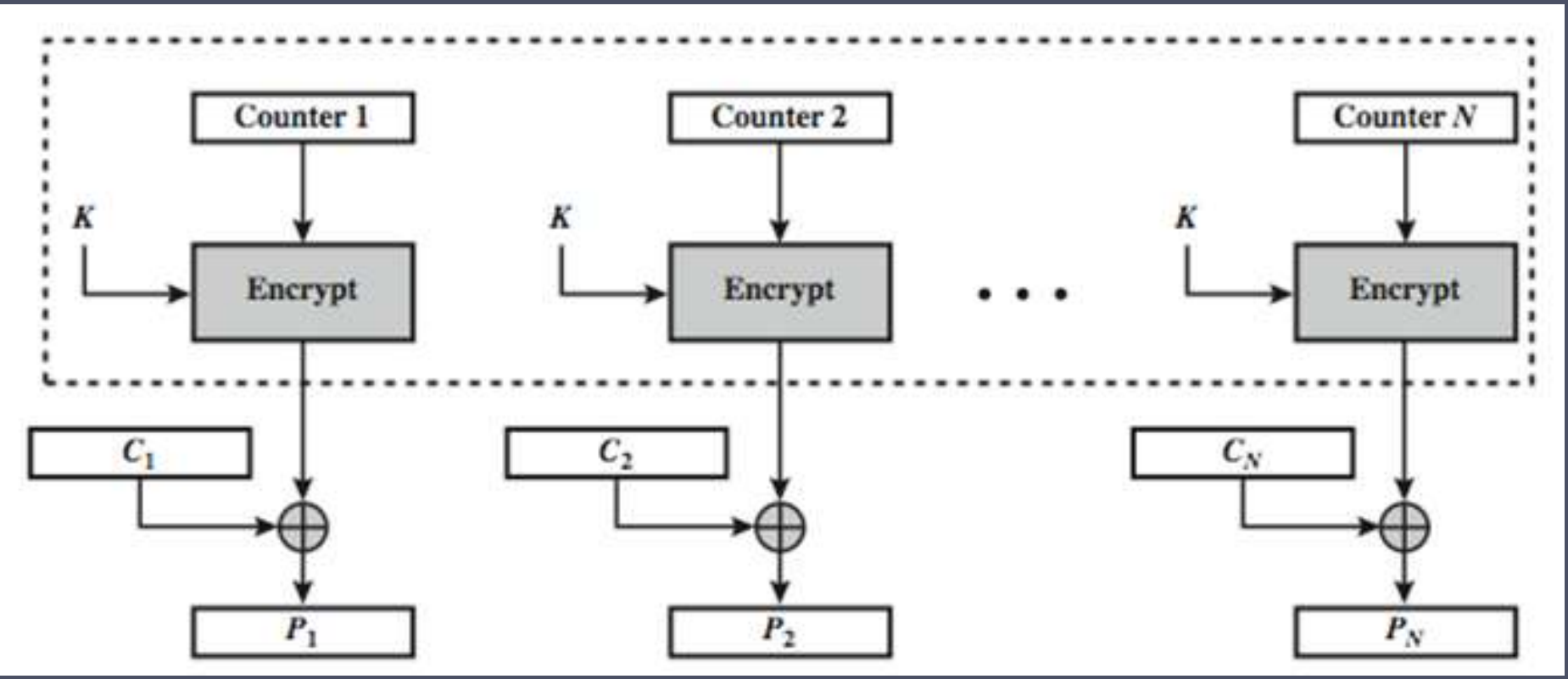
Decryption (2/2)

Counter (CTR)



Encryption (1/2)

Counter (CTR)



Decryption (2/2)

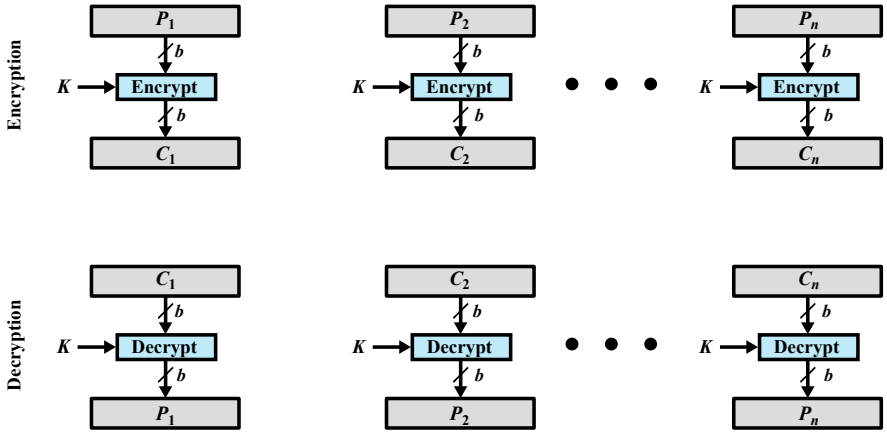
Block & Stream Ciphers

Block Cipher

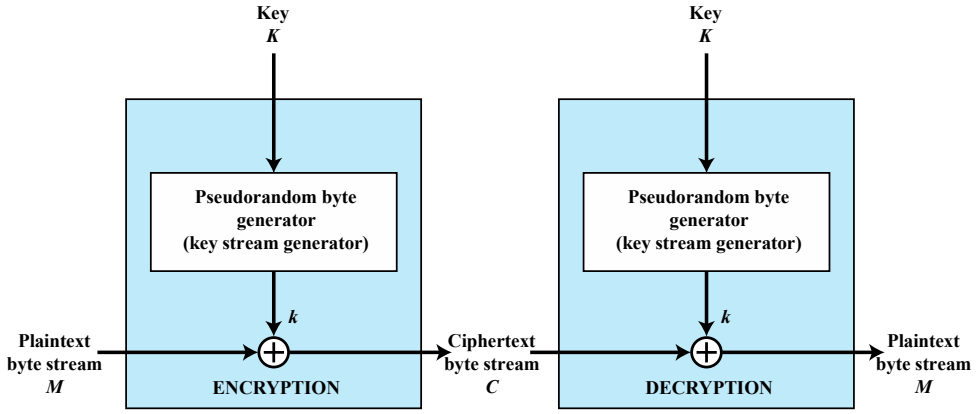
- Processes the input one block of elements at a time
- Produces an output block for each input block
- Can reuse keys
- More common

Stream Cipher

- Processes the input elements continuously
- Produces output one element at a time
- Primary advantage is that they are almost always faster and use far less code
- Encrypts plaintext one byte at a time
- Pseudorandom stream is one that is unpredictable without knowledge of the input key



(a) Block cipher encryption (electronic codebook mode)

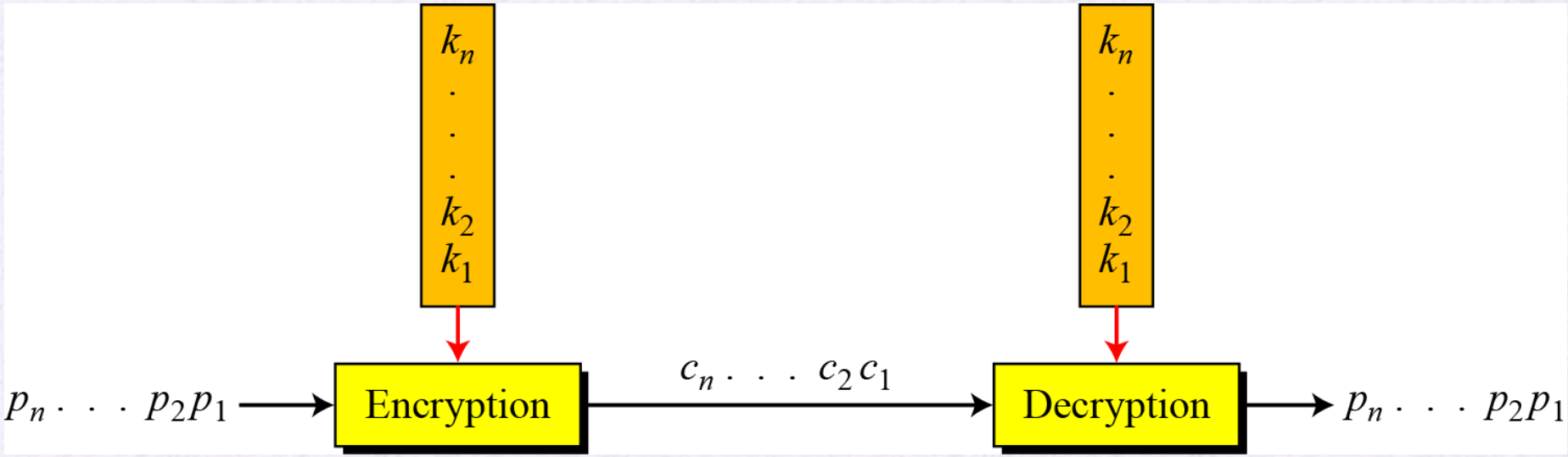


(b) Stream encryption

Figure 2.2 Types of Symmetric Encryption

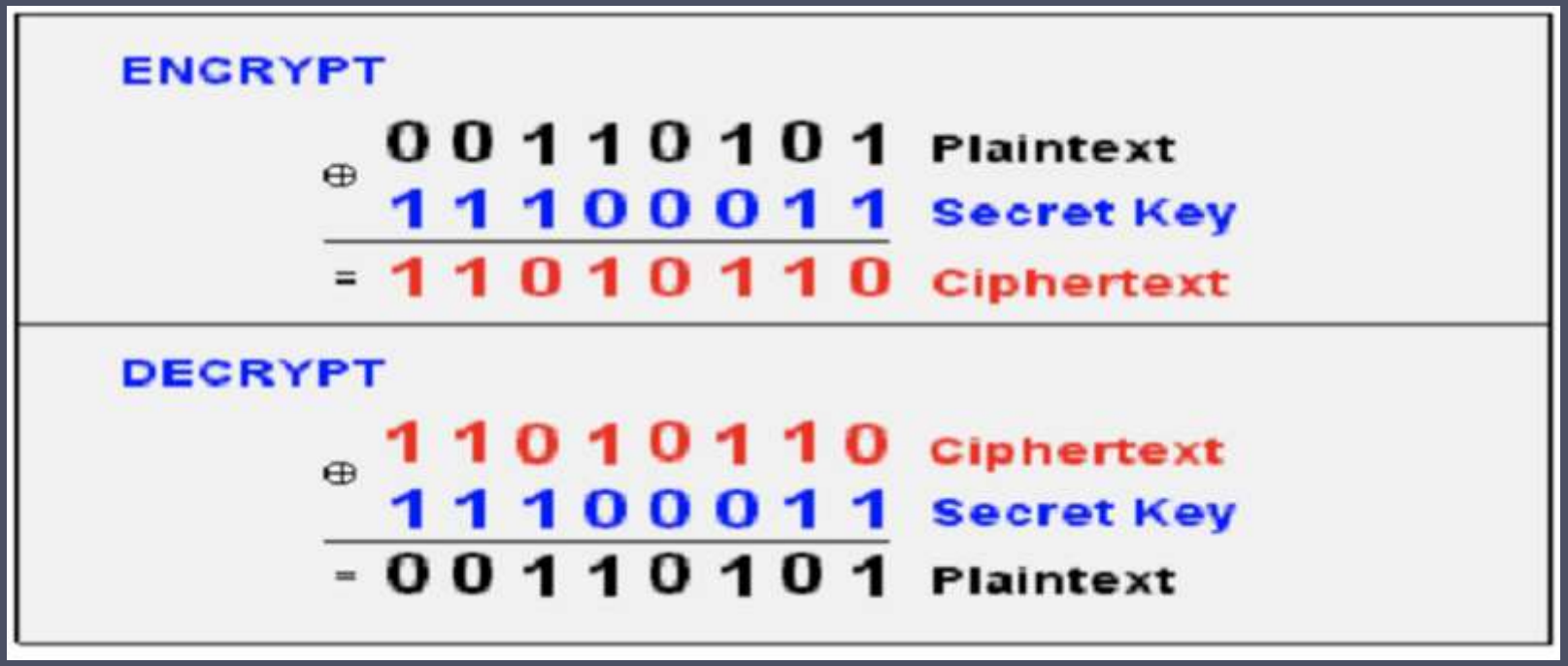
Modern Stream Cipher

- In a modern stream cipher, encryption and decryption are done r bits at a time. We have a plaintext bit stream P = $p_n \ldots p_2\ p_1$, a ciphertext bit stream C = $c_n \ldots c_2\ c_1$, and a key bit stream K = $k_n \ldots k_2\ k_1$, in which p_i , c_i , and k_i are r-bit words.



A modern stream cipher

Stream Cipher using XOR



Stream Ciphers Algorithms

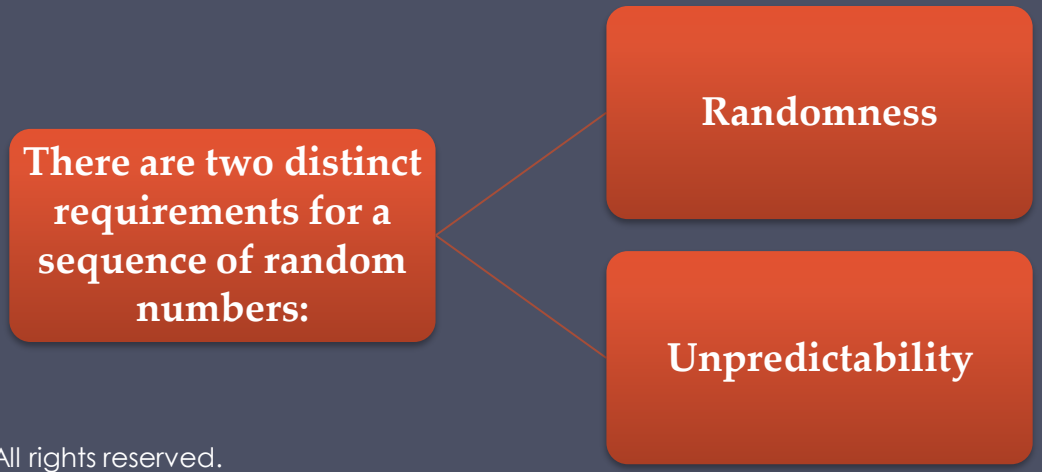
- Rivest Cipher 4 (RC4) initially used for Wifi (WEP)
- Wi-Fi Protected Access (WPA), which replaced WEP in WLAN
- WPA2/WPA3
- **A5/1** is a stream cipher used to provide over-the-air communication privacy in the GSM cellular telephone standard.

Differences between stream and block ciphers

Stream Cipher	Block Cipher
Takes one byte of plaintext at a time	Takes one block of plaintext at a time.
Need less time hence simple	Need more time hence complex
Uses exactly 8 bits	Uses 64 or more bits
Utilizes substitution methods	Utilizes transposition methods
No probability of redundency	Redundency might occure
Requires less code for implementation	Requires more code for implementation
Uses one key for one time	One key can be used multiple times
Suitable for implementation in hardwares	Suitable for implementation in softwares
Faster than block cipher	Slower than stream cipher
Vernam cipher is the main implementation	Feistel cipher is the main implementation
Easy to reverse encrypted text	Difficult to reverse encrypted text
Some examples are RC4, A5/1	Some examples are DES, AES

Random Numbers

- A number of network security algorithms and protocols based on cryptography make use of random binary numbers:
  - Key distribution and reciprocal authentication schemes
  - Session key generation
  - Generation of keys for the RSA public-key encryption algorithm
  - Generation of a bit stream for symmetric stream encryption



Randomness (1/2)

- The generation of a sequence of allegedly random numbers being random in some well-defined statistical sense has been a concern

Uniform distribution

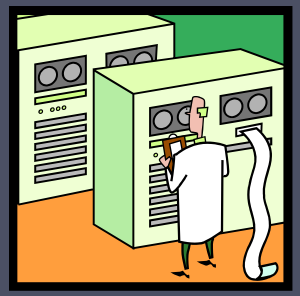
- The frequency of occurrence of ones and zeros should be approximately equal

Independence

- No one subsequence in the sequence can be inferred from the others

Randomness (2/2)

- The generated bit stream needs to appear random even though it is deterministic
- There is no single test that can determine if a PRNG generates numbers that have the characteristic of randomness
  - If the PRNG exhibits randomness on the basis of multiple tests, then it can be assumed to satisfy the randomness requirement
- NIST SP 800-22 specifies that the tests should seek to establish three characteristics:
  - Uniformity
  - Scalability
  - Consistency



Unpredictability

- The requirement is not just that the sequence of numbers be statistically random, but that the successive members of the sequence are unpredictable
- With "true" random sequences each number is statistically independent of other numbers in the sequence and therefore unpredictable
  - True random numbers have their limitations, such as inefficiency, so it is more common to implement algorithms that generate sequences of numbers that appear to be random
  - Care must be taken that an opponent not be able to predict future elements of the sequence on the basis of earlier elements

Pseudorandom Numbers

- Cryptographic applications typically make use of algorithmic techniques for random number generation
- These algorithms are deterministic and therefore produce sequences of numbers that are not statistically random
- If the algorithm is good, the resulting sequences will pass many tests of randomness and are referred to as *pseudorandom numbers*

True Random Number Generator (TRNG)

- Takes as input a source that is effectively random
- The source is referred to as an *entropy source* and is drawn from the physical environment of the computer
  - Includes things such as keystroke timing patterns, disk electrical activity, mouse movements, and instantaneous values of the system clock
  - The source, or combination of sources, serve as input to an algorithm that produces random binary output

- The TRNG may simply involve conversion of an analog source to a binary output

- The TRNG may involve additional processing to overcome any bias in the source

Pseudorandom Number Generator (PRNG)

- Takes as input a fixed value, called the *seed*, and produces a sequence of output bits using a deterministic algorithm
  - Quite often the seed is generated by a TRNG
- The output bit stream is determined solely by the input value or values, so an adversary who knows the algorithm and the seed can reproduce the entire bit stream
- Other than the number of bits produced there is no difference between a PRNG and a PRF

Two different forms of PRNG

Pseudorandom number generator

- An algorithm that is used to produce an open-ended sequence of bits
- Input to a symmetric stream cipher is a common application for an open-ended sequence of bits

Pseudorandom function (PRF)

- Used to produce a pseudorandom string of bits of some fixed length
- Examples are symmetric encryption keys and nonces

PRNG Requirements

- The basic requirement when a PRNG or PRF is used for a cryptographic application is that an adversary who does not know the seed is unable to determine the pseudorandom string
- The requirement for secrecy of the output of a PRNG or PRF leads to specific requirements in the areas of:
  - Randomness
  - Unpredictability
  - Characteristics of the seed

Random Numbers

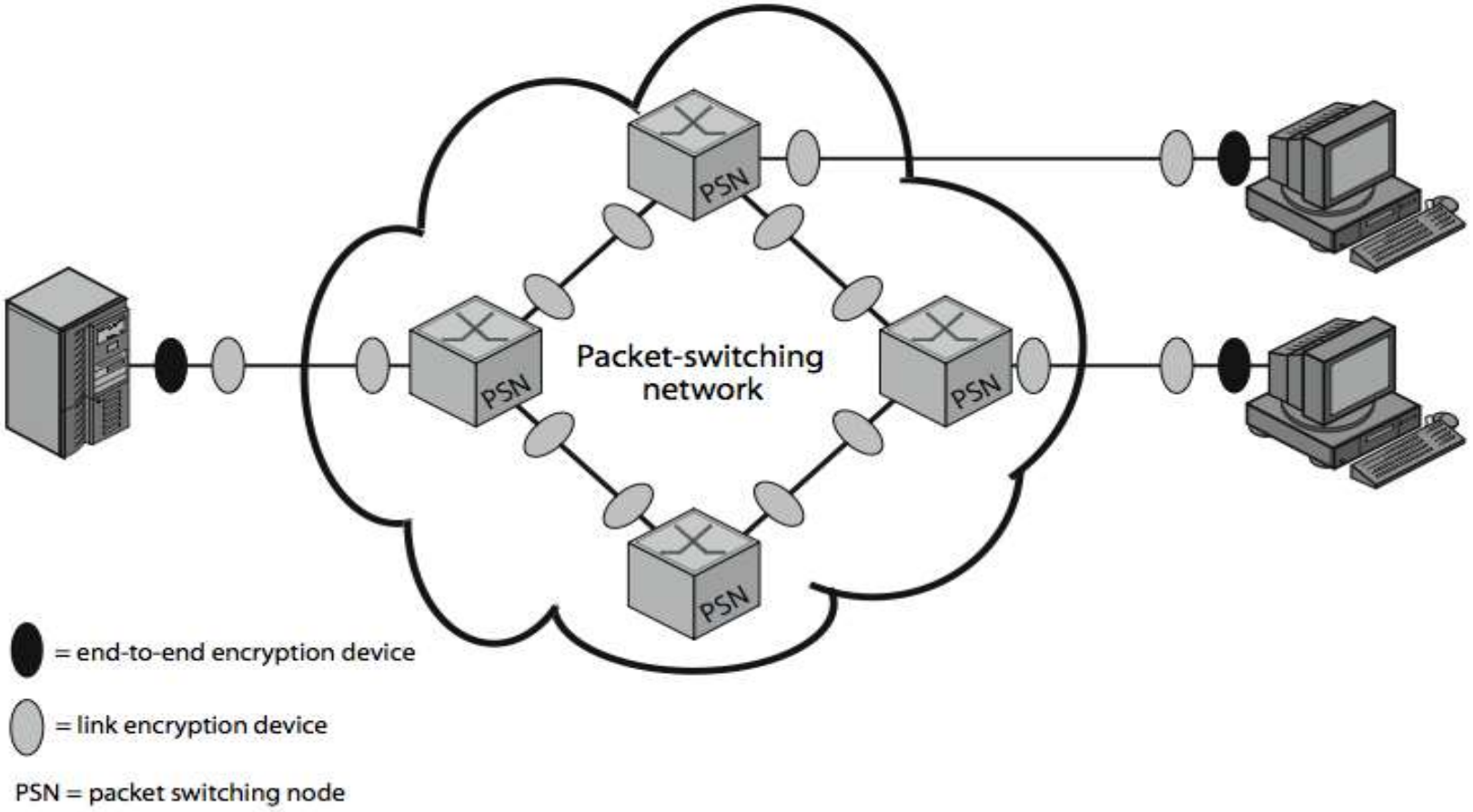
Uses include generation of:

- Keys for public-key algorithms
- Stream key for symmetric stream cipher
- Symmetric key for use as a temporary session key or in creating a digital envelope
- Handshaking to prevent replay attacks
- Session key

Link versus end-to-end Encryption (1/2)

- Traditionally symmetric encryption is used to provide message confidentiality
- have two major placement alternatives
  - link encryption
  - end-to-end encryption

Placement of Encryption Function



Link versus end-to-end Encryption (2/2)

- link encryption
  - encryption occurs independently on every link
  - implies must decrypt traffic between links
  - requires many devices, but paired keys
- end-to-end encryption
  - encryption occurs between original source and final destination
  - need devices at each end with shared keys

Placement of Encryption Function (1/2)

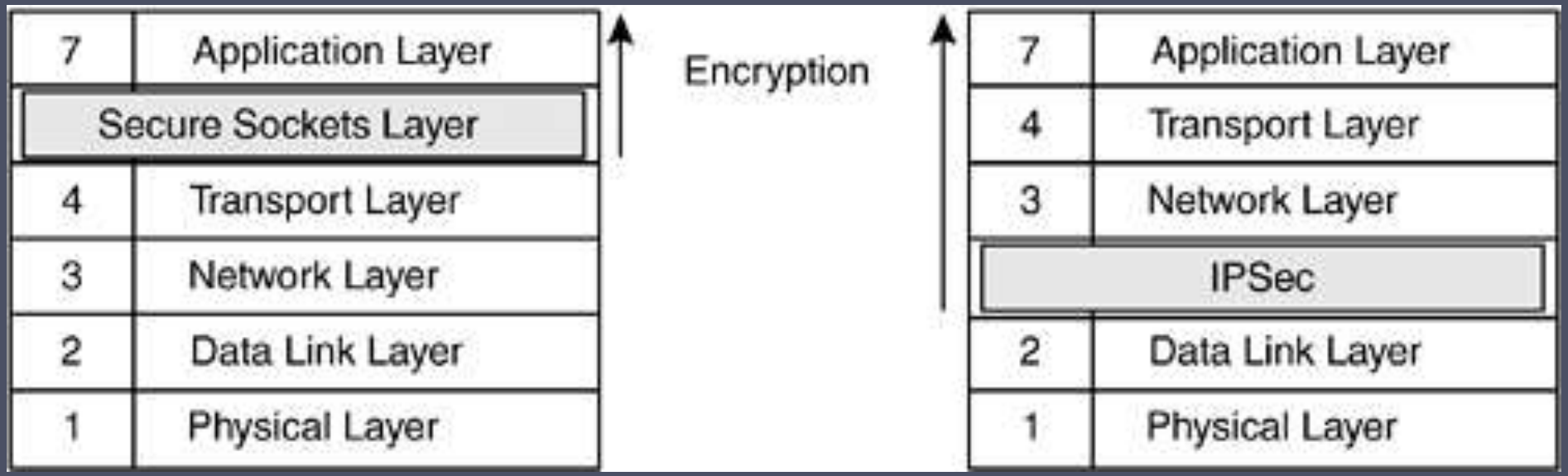
- When using end-to-end encryption must leave headers in clear
  - So network can correctly route information
- Hence although contents protected, traffic pattern flows are not
- Ideally want both at once
  - end-to-end protects data contents over entire path and provides authentication
  - link protects traffic flows from monitoring

Placement of Encryption Function (2/2)

- Can place encryption function at various layers in OSI Reference Model
  - Link encryption occurs at layers 1 or 2
  - End-to-end can occur at layers 3, 4, 6, 7
  - As move higher less information is encrypted but it is more secure though more complex with more entities and keys

I disagree in little sense

Encryption with reference to OSI Model



Key Distribution (1/6)

- Symmetric schemes require both parties to share a common secret key
- Issue is how to securely distribute this key?
- key distribution scheme refers to the means of delivering a key to two parties who wish to exchange data, without allowing others to see the key
- Often secure system failure due to a break in the key distribution scheme

Key Distribution (2/6)

- given parties A and B have various **key distribution** alternatives:
  1. A can select key and physically deliver to B
  2. third party can select & deliver key to A & B
  3. if A & B have communicated previously can use previous key to encrypt a new key
  4. if A & B have secure communications with a third party C, C can relay key between A & B

Key Distribution (3/6)

- Option 1 & 2 ok for link encryption
- Impossible for end-to-end encryption
  - *At network level using node level encryption with 1000 nodes* possible keys for communication 1/2 million ([*N*(*N*-1)]/2 =)
  - If the same network supports 10,000 applications then 50 million keys may be required for application level encryption

Key Distribution (4/6)

- Option 3 can be used in both link and end-to-end communication
  - again dangerous gaining access to one key will get access to all subsequent keys.

Key Distribution (5/6)

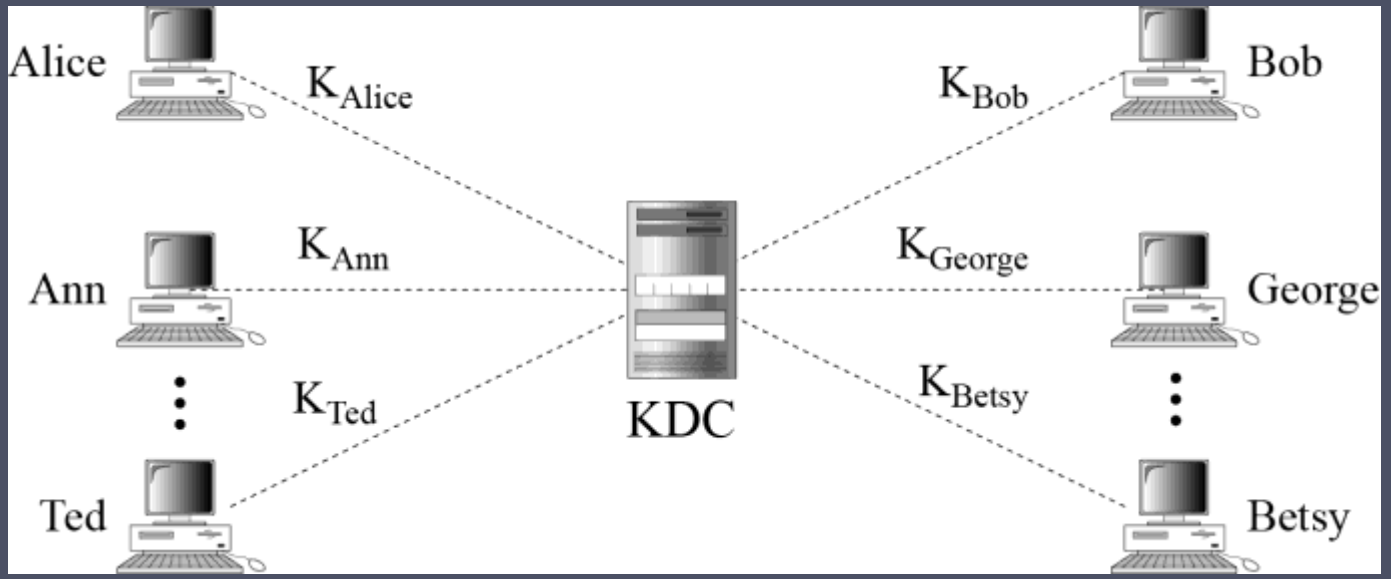
- Option 4 is the most commonly used for end-to-end encryption
  - a key distribution center (KDC) is used
  - a hierarchy of keys, at least two levels
    - a unique *master key* for each host for communicating with KDC
    - *session keys* for communicating between the end hosts

Key Hierarchy

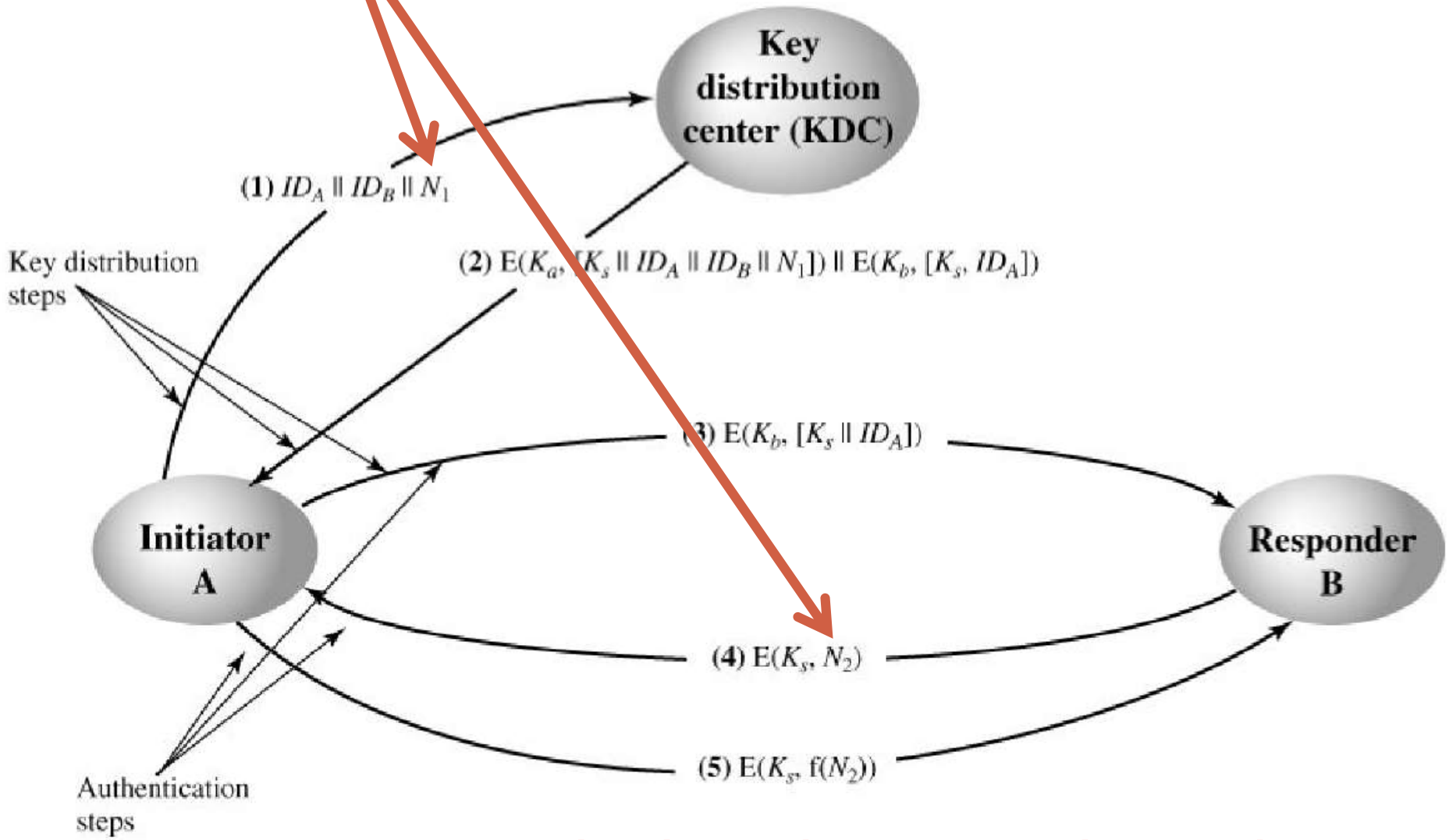
- typically have a hierarchy of keys
- session key
  - temporary key
  - used for encryption of data between users
  - for one logical session then discarded
- master key
  - used to encrypt session keys
  - shared by user & key distribution center

Key-Distribution Center: KDC

FRCNS, ch15



Nonce – Fresh randomness coming from Cryptographically Strong (pseudo)Random Source



Kizza ch11

Key-Distribution Center

- A Key Distribution Center (KDC) is a single, trusted network entity with which all network communicating elements must establish a shared secret key. It requires all communicating elements to have a shared secret key with which they can communicate with the KDC confidentially
- This requirement still presents a problem of distributing this shared key.
- The KDC does not create or generate keys for the communicating elements; it only stores and distributes keys. The creation of keys must be done somewhere else.

Key-Distribution Center

- Since all network communicating elements confidentially share their secret keys with the KDC, it distributes these keys secretly to the corresponding partners in the communication upon request.
- Any network element that wants to communicate with any other element in the network using symmetric encryption schemes uses the KDC to obtain the shared keys needed for that communication. Figure 11.7 shows the working of the KDC.

Public-Key Encryption Structure

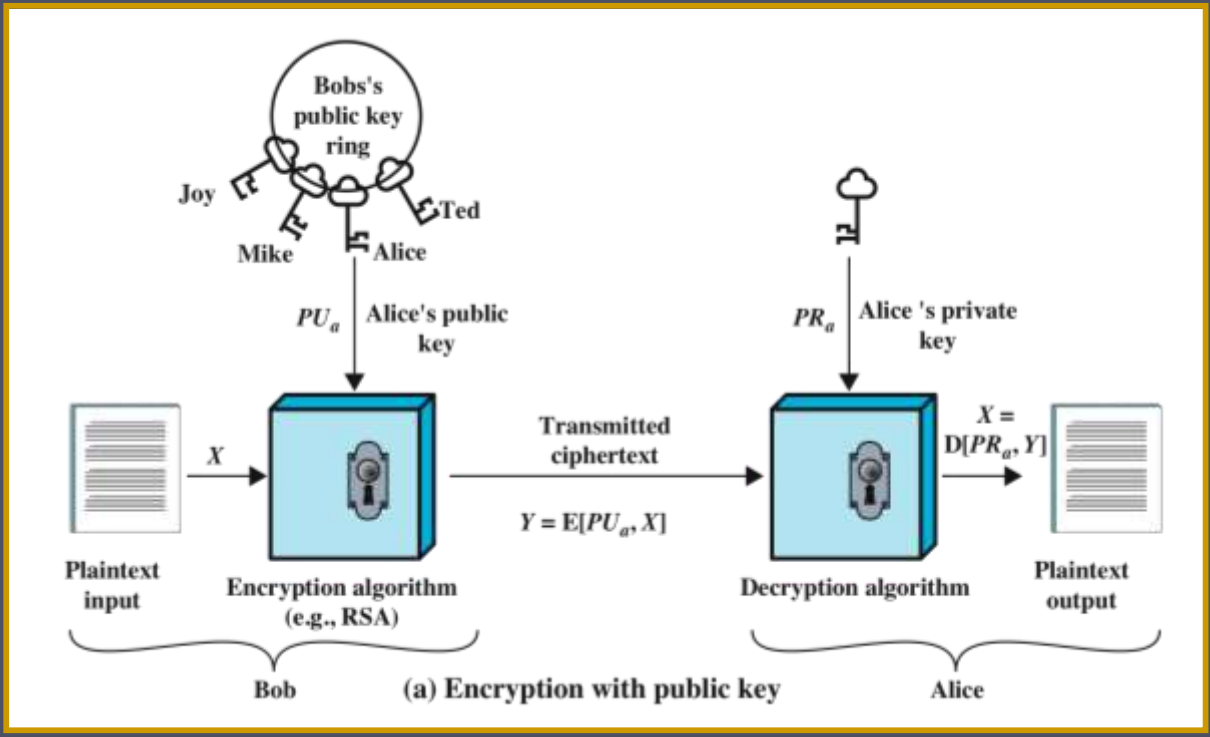
Publicly proposed by Diffie and Hellman in 1976

Based on mathematical functions

Asymmetric

- Uses two separate keys
- Public key and private key
- Public key is made public for others to use

Some form of protocol is needed for distribution



- **Plaintext**
  - Readable message or data that is fed into the algorithm as input
- **Encryption algorithm**
  - Performs transformations on the plaintext
- **Public and private key**
  - Pair of keys, one for encryption, one for decryption
- **Ciphertext**
  - Scrambled message produced as output
- **Decryption key**
  - Produces the original plaintext

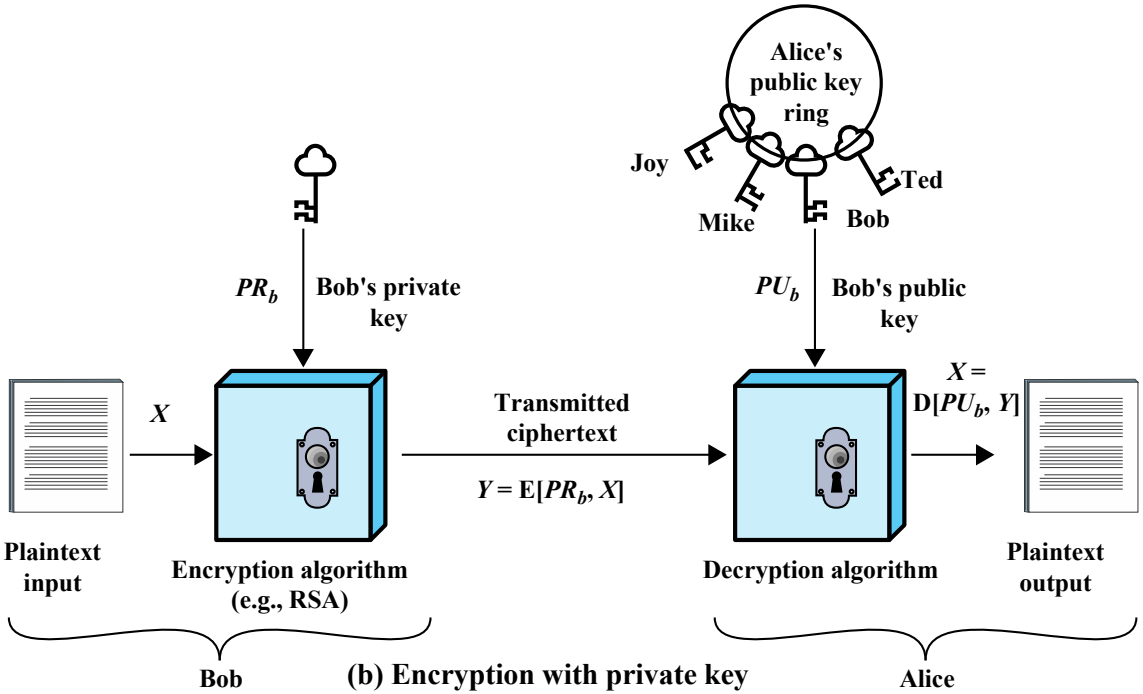


Figure 2.6 Public-Key Cryptography

- User encrypts data using his or her own private key
- Anyone who knows the corresponding public key will be able to decrypt the message

Table 9.2
Conventional and Public-Key Encryption

Conventional Encryption	Public-Key Encryption
Needed to Work: 1. The same algorithm with the same key is used for encryption and decryption. 2. The sender and receiver must share the algorithm and the key. *Needed for Security:* 1. The key must be kept secret. 2. It must be impossible or at least impractical to decipher a message if the key is kept secret. 3. Knowledge of the algorithm plus samples of ciphertext must be insufficient to determine the key.	*Needed to Work:* 1. One algorithm is used for encryption and a related algorithm for decryption with a pair of keys, one for encryption and one for decryption. 2. The sender and receiver must each have one of the matched pair of keys (not the same one). *Needed for Security:* 1. One of the two keys must be kept secret. 2. It must be impossible or at least impractical to decipher a message if one of the keys is kept secret. 3. Knowledge of the algorithm plus one of the keys plus samples of ciphertext must be insufficient to determine the other key.

Rivest-Shamir-Adleman (RSA) Algorithm

- Developed in 1977 at MIT by Ron Rivest, Adi Shamir & Len Adleman
- Most widely used general-purpose approach to public-key encryption
- Is a cipher in which the plaintext and ciphertext are integers between 0 and $n - 1$ for some n
  - A typical size for n is 1024 bits, or 309 decimal digits

RSA Algorithm

- RSA makes use of an expression with exponentials
- Plaintext is encrypted in blocks with each block having a binary value less than some number *n*
- Encryption and decryption are of the following form, for some plaintext block *M* and ciphertext block C

  $C = M^e \bmod n$

  $M = C^d \bmod n = (M^e)^d \bmod n = M^{ed} \bmod n$
- Both sender and receiver must know the value of *n*
- The sender knows the value of *e*, and only the receiver knows the value of *d*
- This is a public-key encryption algorithm with a public key of *PU={e,n}* and a private key of *PR={d,n}*

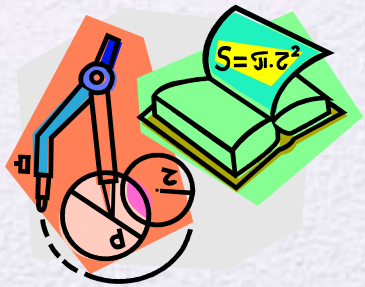
Algorithm Requirements

- For this algorithm to be satisfactory for public-key encryption, the following requirements must be met:

1. It is possible to find values of e, d, n such that $M^{ed} \bmod n = M$ for all $M < n$

2. It is relatively easy to calculate $M^e \bmod n$ and $C^d \bmod n$ for all values of $M < n$

3. It is infeasible to determine d given e and n



Φ (n)
is called Euler totient function

e^{-1}
Multiplicative inverse of e

Key Generation by Alice

Select p, q	p and q both prime, $p \neq q$
Calculate $n = p \times q$	
Calculate $\phi(n) = (p-1)(q-1)$	
Select integer e	$\gcd(\phi(n), e) = 1;\ 1 < e < \phi(n)$
Calculate d	$d = e^{-1} \pmod{\phi(n)}$
Public key	$PU = \{e, n\}$
Private key	$PR = \{d, n\}$

Encryption by Bob with Alice's Public Key

Plaintext:	$M < n$
Ciphertext:	$C = M^e \bmod n$

Decryption by Alice with Alice's Private Key

Ciphertext:	C
Plaintext:	$M = C^d \bmod n$

p=7;
q=19;
n=133;
Φ (n)=108;
Let e=5;
d=65;
M=34;
C=97

Figure 9.5 The RSA Algorithm

Example of RSA Algorithm

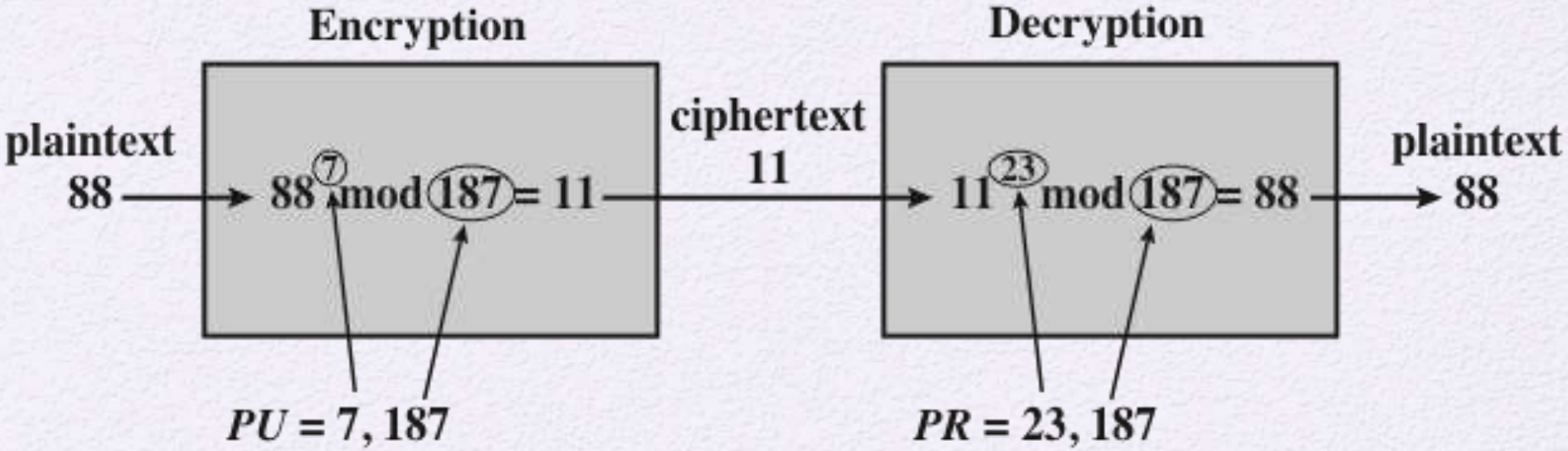


Figure 9.6 Example of RSA Algorithm

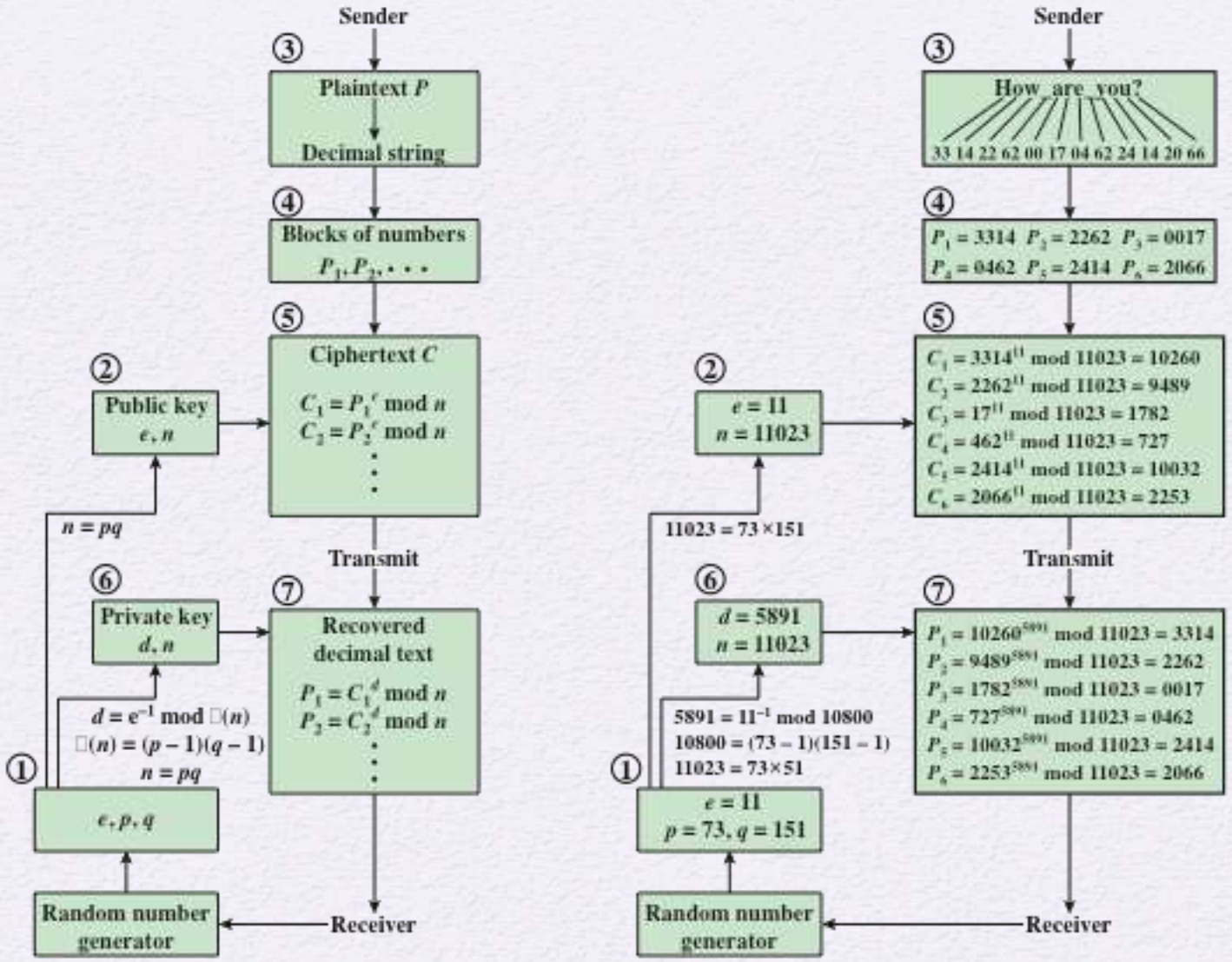


Figure 9.7 RSA Processing of Multiple Blocks

Asymmetric Encryption Example

- For example. make $p = 7$ and $q = 13$;
- We then calculate $n = 7*13 = 91$ and $(p-1)(q-1) = 72$
- We next select e relatively prime to 72 and < 72, yielding 5
- Finally, we calculate d such that de mod 72 = 1, yielding 29
- We how have our keys
  - Public key, $e, n = 5, 91$
  - Private key, $d, n = 29, 91$
- Encrypting the message 69 with the public key results in the cyphertext 62
- Ciphertext can be decoded with the private key
  - Public key can be distributed in clear text to anyone who wants to communicate with holder of public key

Diffie-Hellman Key Exchange

- First published public-key algorithm
- A number of commercial products employ this key exchange technique
- Purpose is to enable two users to securely exchange a key that can then be used for subsequent symmetric encryption of messages
- The algorithm itself is limited to the exchange of secret values
- Its effectiveness depends on the difficulty of computing discrete logarithms

Primitive Roots

Definition: A primitive root modulo a prime p is an integer r in $\mathbf{Z}_p$ such that every nonzero element of $\mathbf{Z}_p$ is a power of r.

Example: Since every element of $\mathbf{Z}_{11}$ is a power of 2, 2 is a primitive root of 11.

Example: Since not all elements of $\mathbf{Z}_{11}$ are powers of 3, 3 is not a primitive root of 11.

Important Fact: There is a primitive root modulo p for every prime number p.

5 is a primitive root mod 23, here is a list of the powers of 5 mod 23:

n	5^n	n	5^n	n	5^n	n	5^n	n	5^n
1	5	6	8	11	22	16	3	21	14
2	2	7	17	12	18	17	15	22	1
3	10	8	16	13	21	18	6		
4	4	9	11	14	13	19	7		
5	20	10	9	15	19	20	12		

Alice

Bob

Alice		Bob
Alice and Bob share a prime q and α, such that $\alpha < q$ and α is a primitive root of q		Alice and Bob share a prime q and α, such that $\alpha < q$ and α is a primitive root of q
Alice generates a private key X_A such that $X_A < q$		Bob generates a private key X_B such that $X_B < q$
Alice calculates a public key $Y_A = \alpha^{X_A} \bmod q$	Y_A →	Bob calculates a public key $Y_B = \alpha^{X_B} \bmod q$
Alice receives Bob's public key Y_B in plaintext	← Y_B	Bob receives Alice's public key Y_A in plaintext
Alice calculates shared secret key $K = (Y_B)^{X_A} \bmod q$		Bob calculates shared secret key $K = (Y_A)^{X_B} \bmod q$

Figure 10.1 Diffie-Hellman Key Exchange

Diffie-Hellman Example

Exercise $q = 23, \alpha = 5, X_A = 6, X_B = 15.$

Determine ***Public Key*** and ***Shared key*** for both users using Diffie-Hellman key exchange algorithm.

Solution:

1. Here $q = 23, \alpha = 5$
2. Calculate *Public key for user A*. $X_A = 6$,

   $Y_A = \alpha^{X_A} \bmod q = 5^6 \bmod 23 = 8$ **$Y_A = 8$**
3. Calculate *Public key for user B*. $X_B = 15$,

   $Y_B = \alpha^{X_B} \bmod q = 5^{15} \bmod 23 = 19$ **$Y_B = 19$**
4. Calculation of *secret key by user A*,

   $K = (Y_B)^{X_A} \bmod q = 19^6 \bmod 23 = 2$ **$K = 2$**
5. Calculation of *secret key by user B*,

   $K = (Y_A)^{X_B} \bmod q = 8^{15} \bmod 23 = 2$ **$K = 2$**

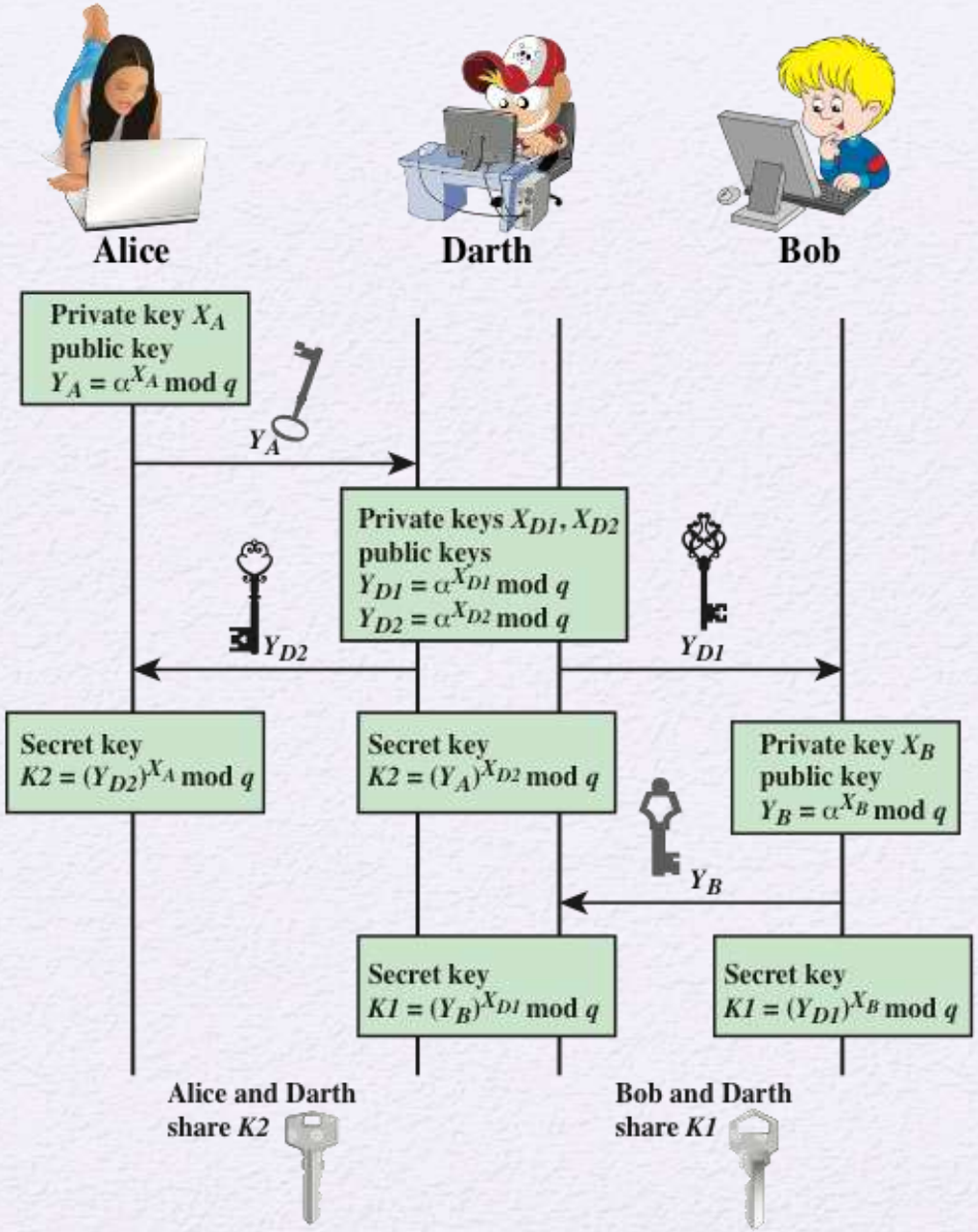


Figure 10.2 Man-in-the-Middle Attack

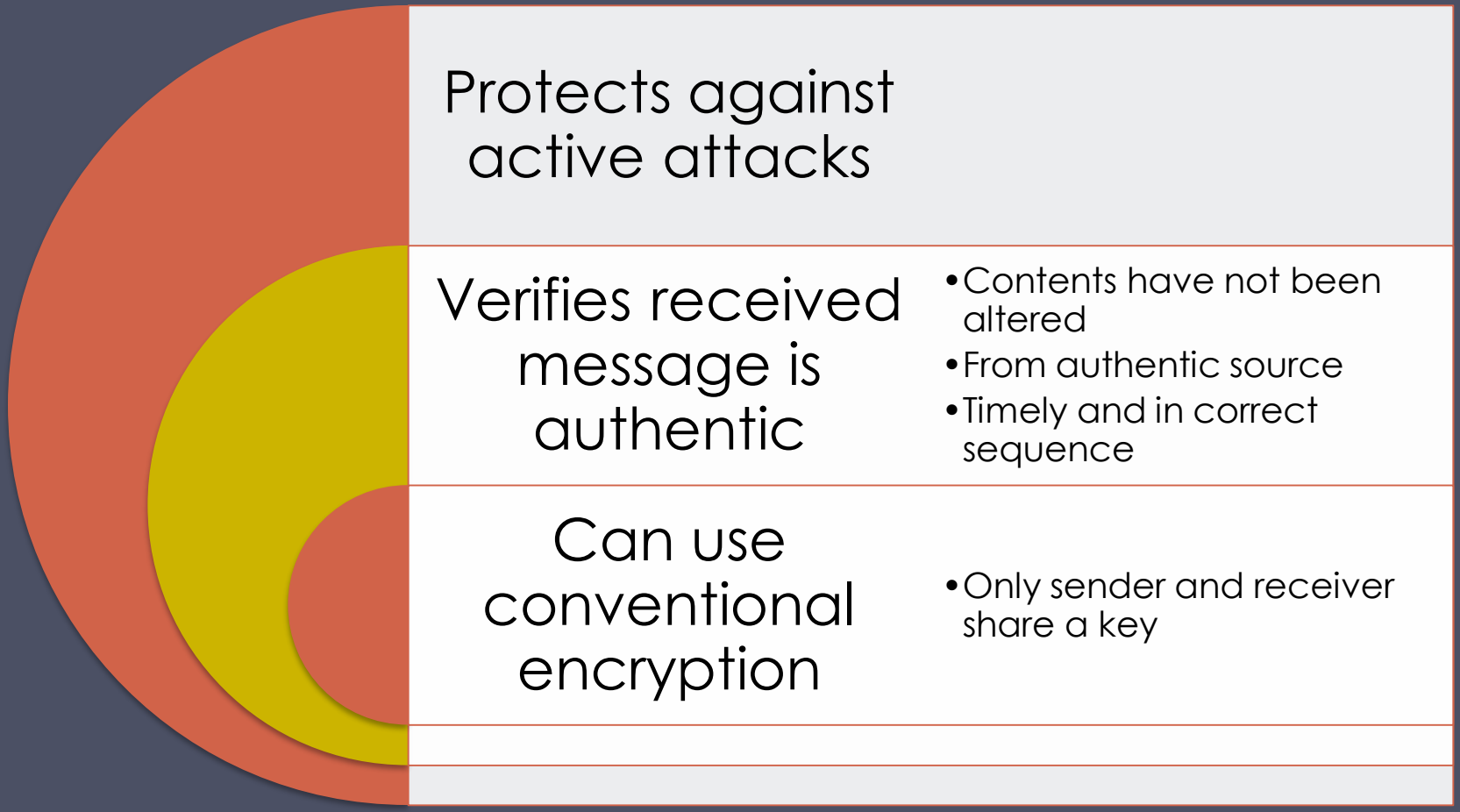
Table 2.3

Applications for Public-Key Cryptosystems

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

Message Authentication

Message Authentication



Authentication Techniques

1. Hash Functions
2. Message Authentication Code (MAC)
3. Digital Signature

Message Authentication Without Confidentiality

- Message encryption by itself does not provide a secure form of authentication
- It is possible to combine authentication and confidentiality in a single algorithm by encrypting a message plus its authentication tag
- Typically message authentication is provided as a separate function from message encryption
- Situations in which message authentication without confidentiality may be preferable include:
  - There are a number of applications in which the same message is broadcast to a number of destinations
  - An exchange in which one side has a heavy load and cannot afford the time to decrypt all incoming messages
  - Authentication of a computer program in plaintext is an attractive service
- Thus, there is a place for both authentication and encryption in meeting security requirements

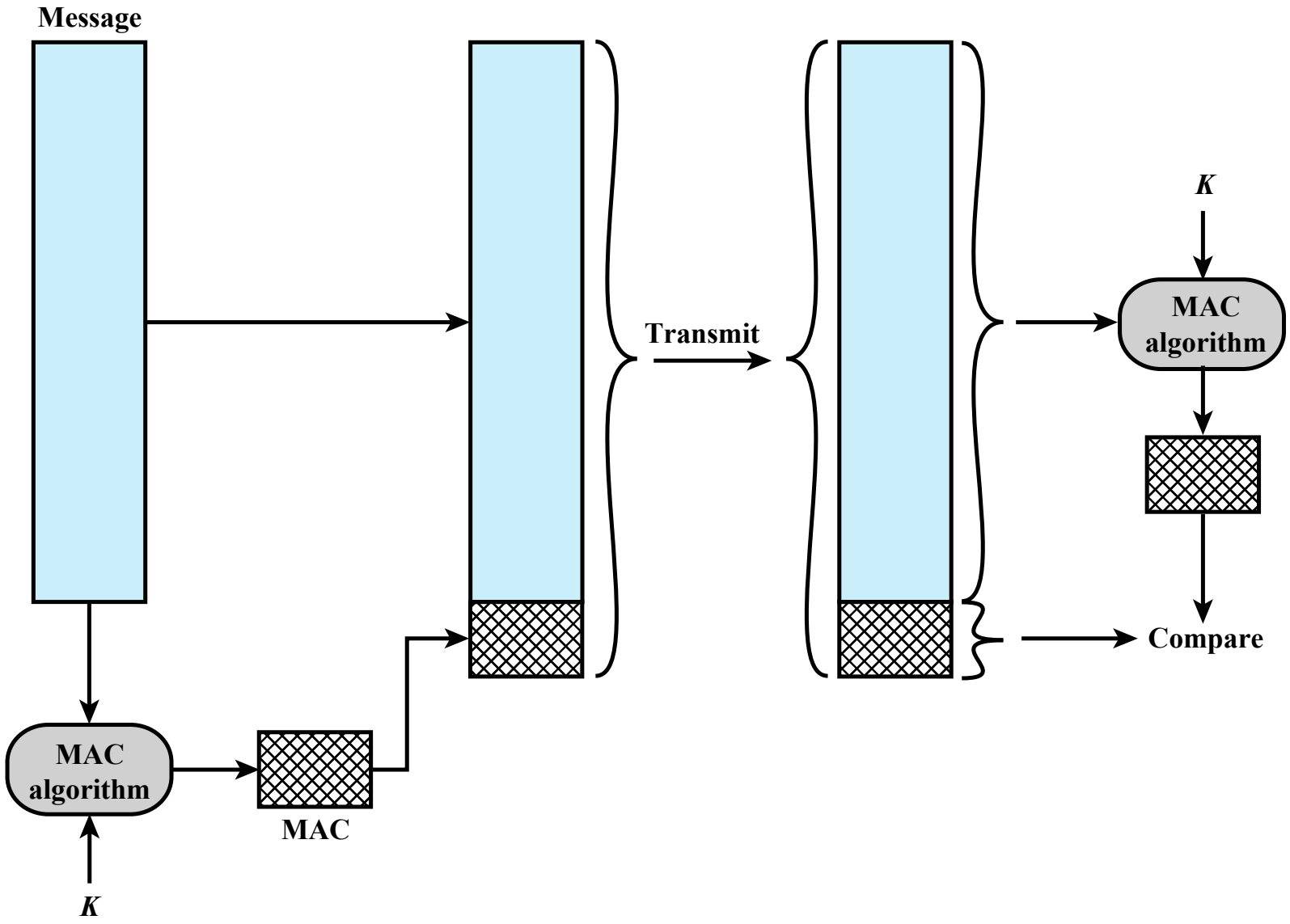
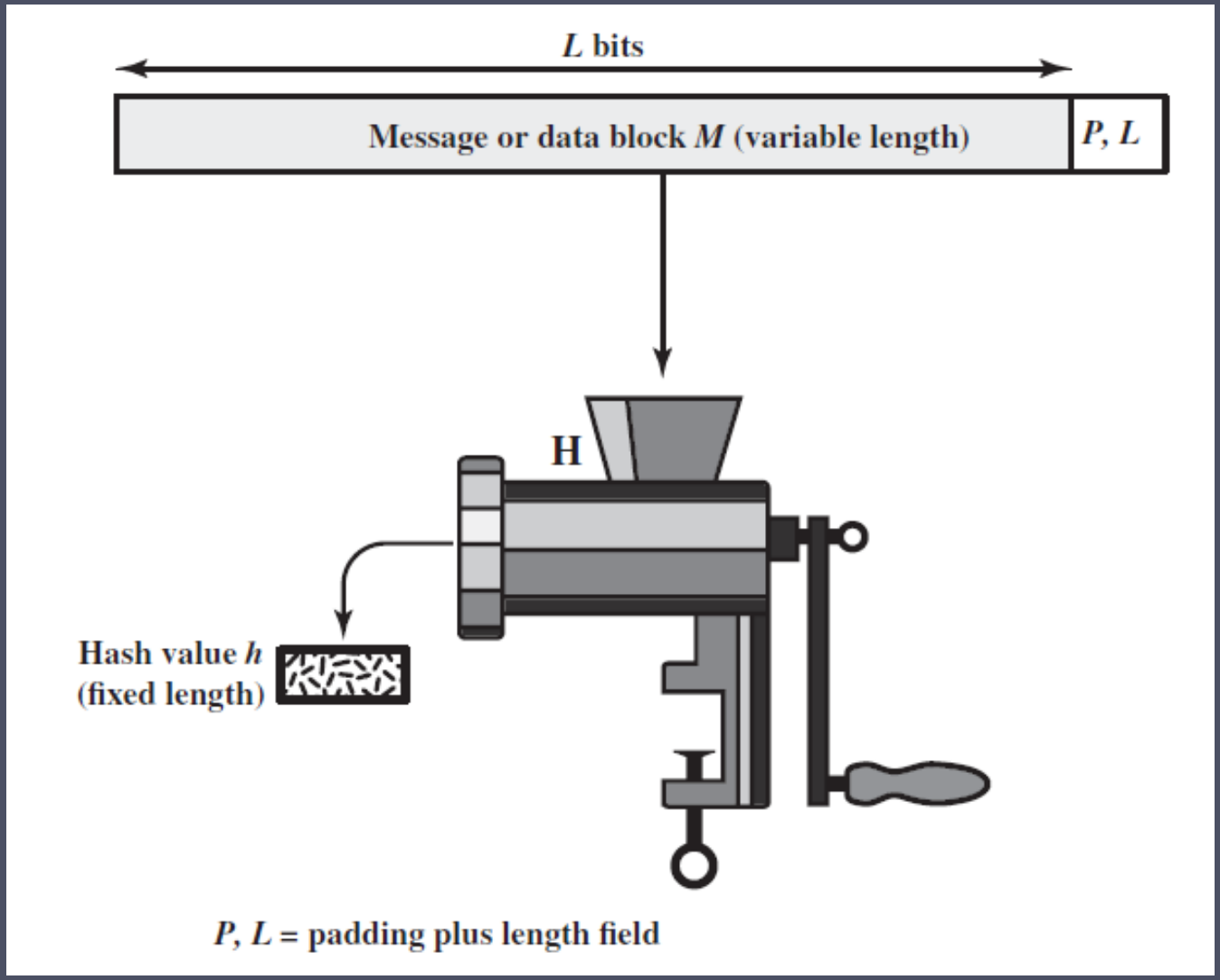


Figure 2.3 Message Authentication Using a Message Authentication Code (MAC).



Cryptographic Hash Function; h =H(M)

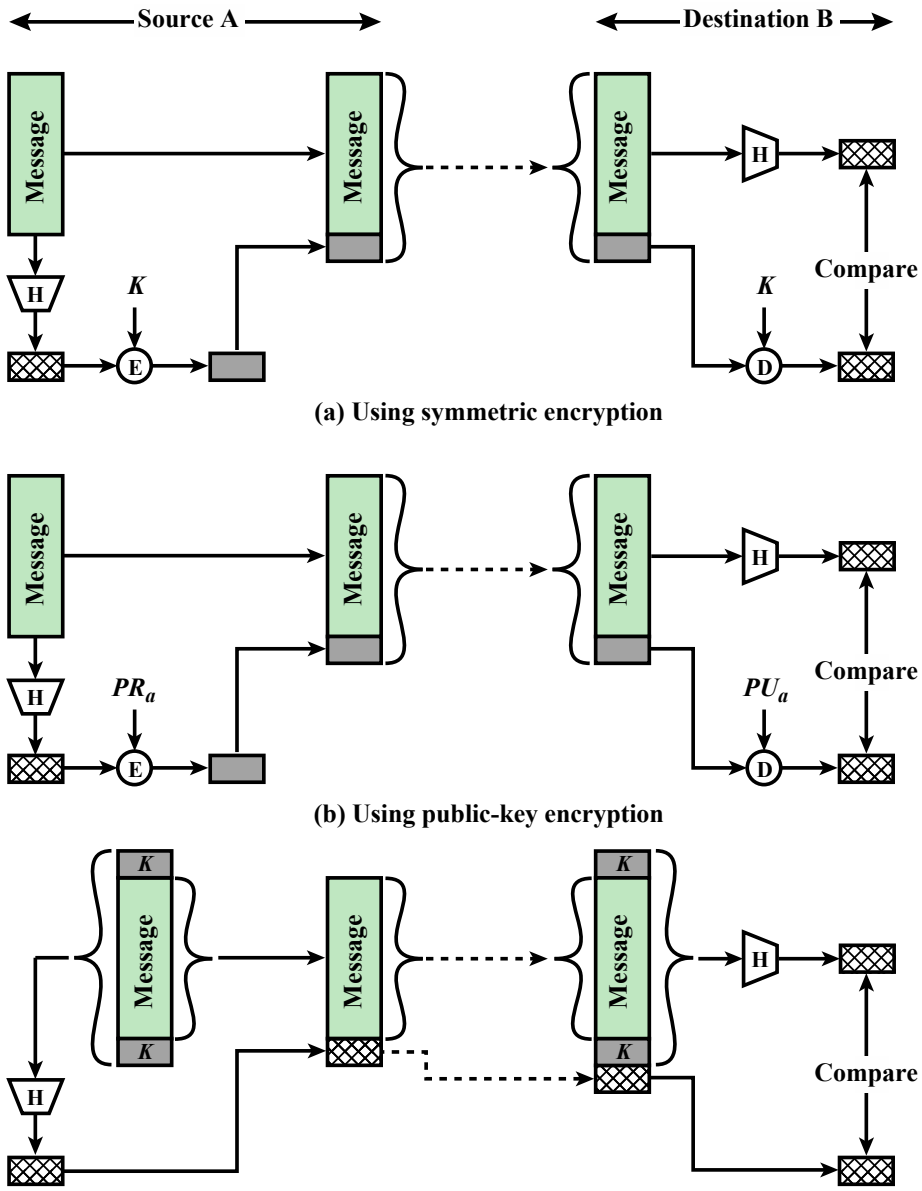


Figure 2.5 Message Authentication Using a One-Way Hash Function.

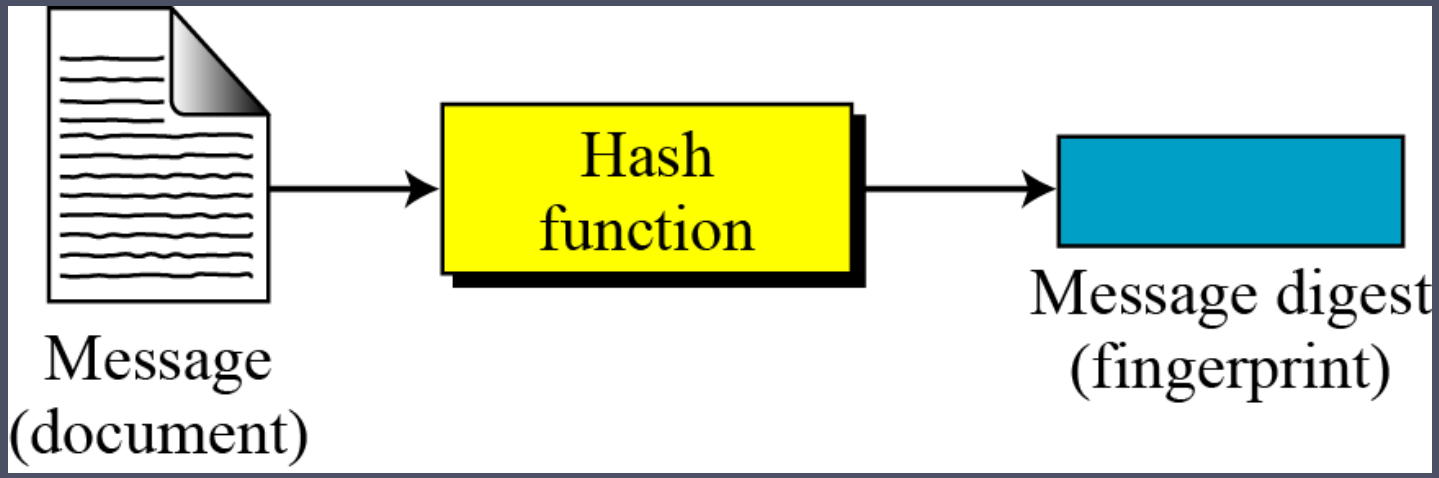
Hash Functions

1. Hash Functions (1/4)

- Basis of authentication
- Creates small, fixed-size block of data **message digest** (**hash value)** from *m*
- Hash Function *H* must be collision resistant on *m*
  - Must be infeasible to find an $m' \neq m$ such that $H(m) = H(m')$
- If $H(m) = H(m')$, then $m = m'$
  - The message has not been modified

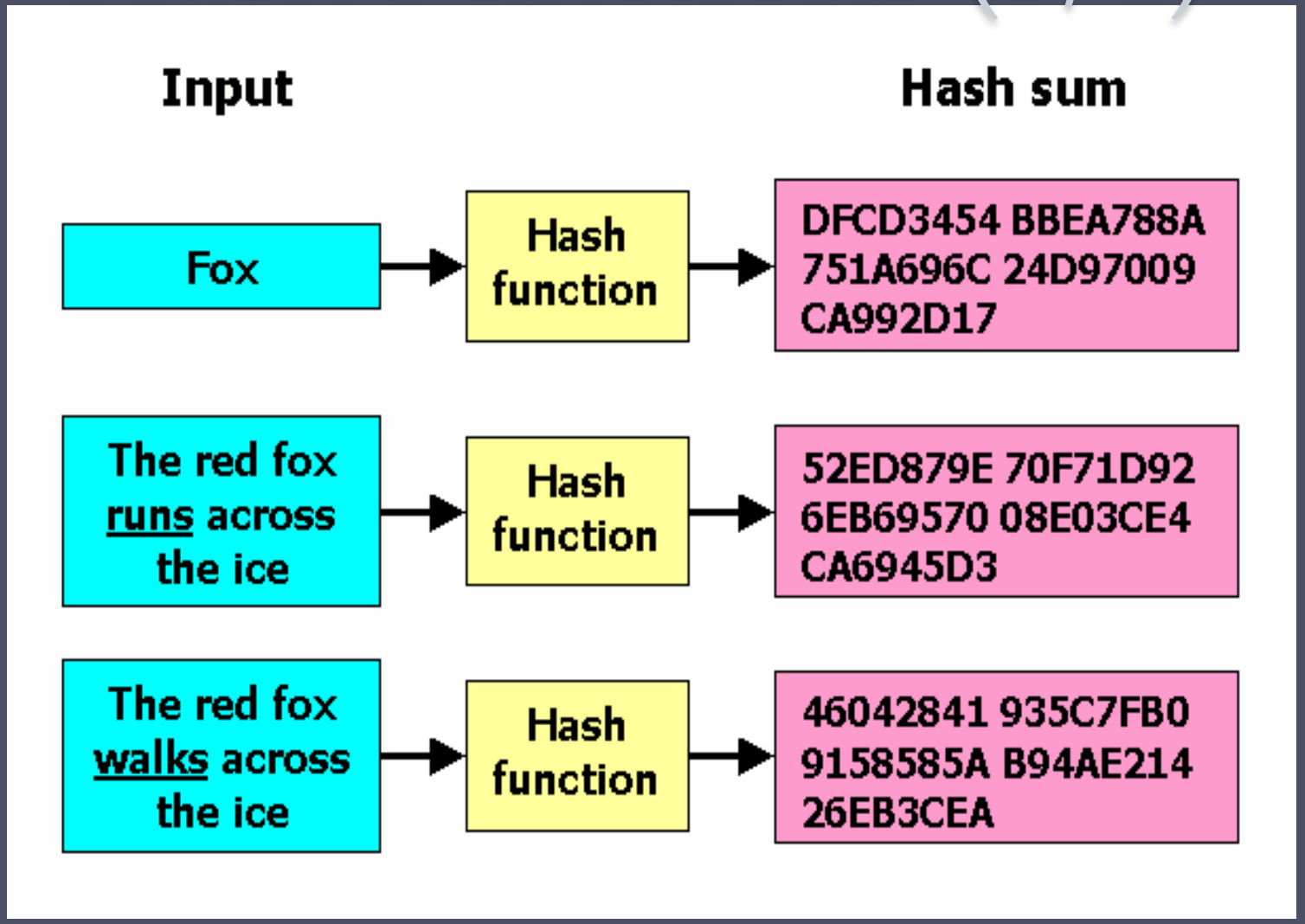
Hash Functions (2/4)

- The electronic equivalent of the document and fingerprint pair is the message and digest pair.



Message and Message Digest

Hash Functions (3/4)



 http://en.wikipedia.org/wiki/Image:Hash_function_long.png

Hash Functions (4/4)

- Common message-digest functions include **MD5**, which produces a 128-bit hash, and **SHA-1**, which outputs a 160-bit hash
- Not useful as authenticators
  - For example *H(m)* can be sent with a message
    - But if *H* is known someone could modify *m* to *m'* and recompute *H(m')* and modification not detected
    - So must authenticate *H(m)*

To be useful for message authentication, a hash function H must have the following properties:

- Can be applied to a block of data of any size
- Produces a fixed-length output
- H(x) is relatively easy to compute for any given x
- One-way or pre-image resistant
  - Computationally infeasible to find x such that H(x) = h
- Computationally infeasible to find $y \neq x$ such that H(y) = H(x)
- Collision resistant or strong collision resistance
  - Computationally infeasible to find any pair (x,y) such that H(x) = H(y)

Security of Hash Functions

There are two approaches to attacking a secure hash function:

- **Cryptanalysis**
  - Exploit logical weaknesses in the algorithm
- **Brute-force attack**
  - Strength of hash function depends solely on the length of the hash code produced by the algorithm

SHA most widely used hash algorithm

Additional secure hash function applications:

- **Passwords**
  - Hash of a password is stored by an operating system
- **Intrusion detection**
  - Store H(F) for each file on a system and secure the hash values

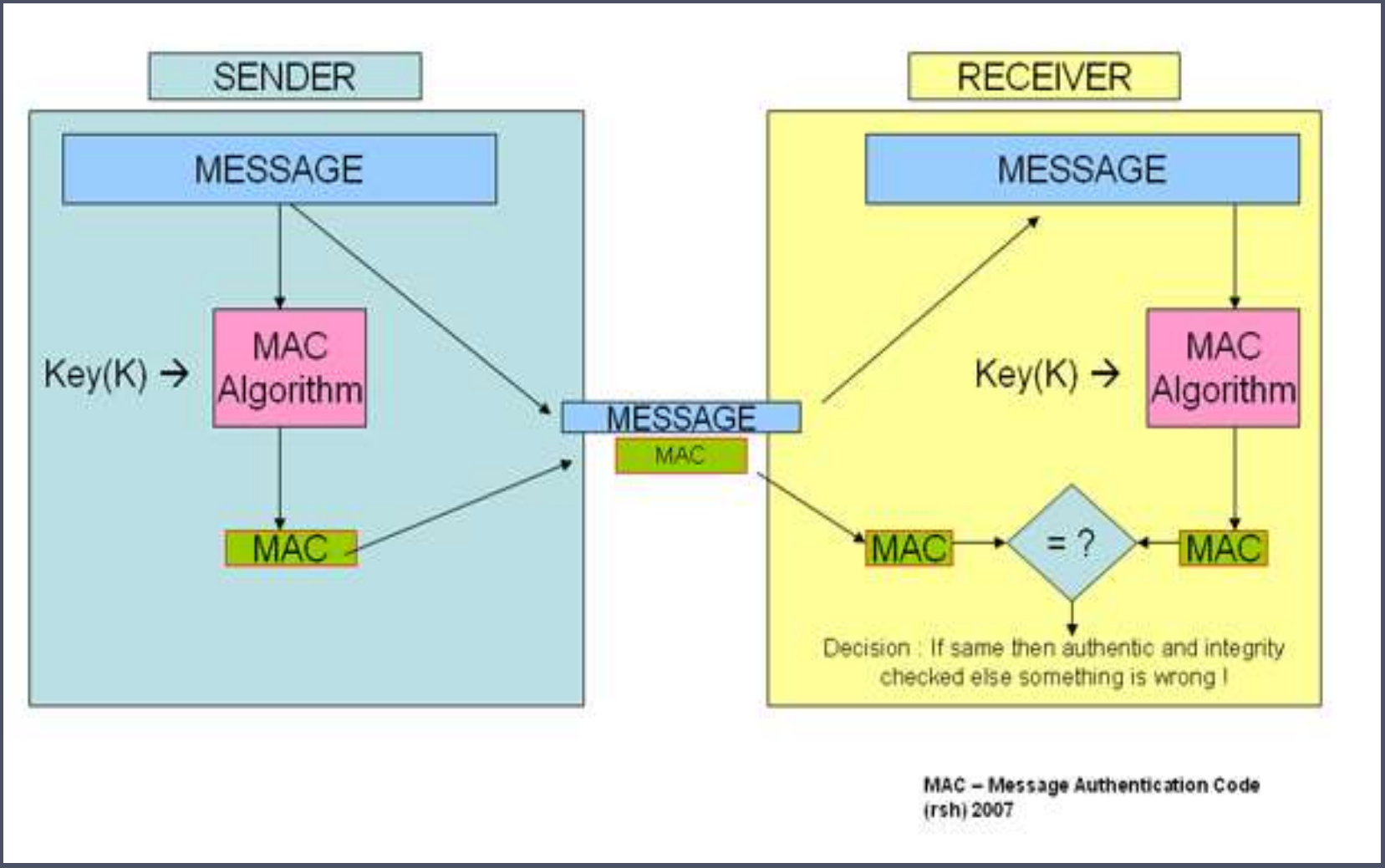
Name	Release information	Other information	Output sizes
MD5 (Message Digest)	Designed by Ron Rivest in 1991	Suitable for non-cryptographic uses, such as basic data integrity. Collisions against MD5 can be calculated within seconds	128 bits (16 bytes)
SHA-1 (Secure Hash Algorithm)	Developed as part of the U.S. Government's Capstone project in 1993.	Collisions against SHA-1 have been produced	160 bits (20 bytes)
SHA-2 (Secure Hash Algorithm)	Designed by the United States National Security Agency (NSA) in 2001.	Consists of two hash algorithms: SHA-256 and SHA-512. SHA-512 is more secure than SHA-256.	SHA-256 : 256 bits (32 bytes) SHA-512 : 512 bits (64 bytes)
SHA-3 (Secure Hash Algorithm)	Released by National Institute of Standards and Technology (NIST) in 2015	subset of the broader cryptographic primitive family Keccak	Same as SHA-2: 224, 256, 384 and 512 bits

Message Authentication Code (MAC)

2. Message Authentication Code (MAC) (1/2)

- Symmetric encryption used in **message-authentication code (MAC)** authentication algorithm
- Cryptographic checksum generated from message using secret key
  - Can securely authenticate short values
- If used to authenticate *H(m)* for an *H* that is collision resistant, then obtain a way to securely authenticate long message by hashing them first
- Note that *k* is needed to compute both *MACs*, so anyone able to compute one can compute the other

MAC (2/2)



http://en.wikipedia.org/wiki/Message_authentication_code

Digital Signatures

Digital Signatures

- NIST FIPS PUB 186-4 defines a digital signature as:

  "The result of a cryptographic transformation of data that, when properly implemented, provides a mechanism for verifying origin authentication, data integrity and signatory non-repudiation."

- Thus, a digital signature is a data-dependent bit pattern, generated by an agent as a function of a file, message, or other form of data block

- FIPS 186-4 specifies the use of one of three digital signature algorithms:
  - Digital Signature Algorithm (DSA)
  - RSA Digital Signature Algorithm
  - Elliptic Curve Digital Signature Algorithm (ECDSA)

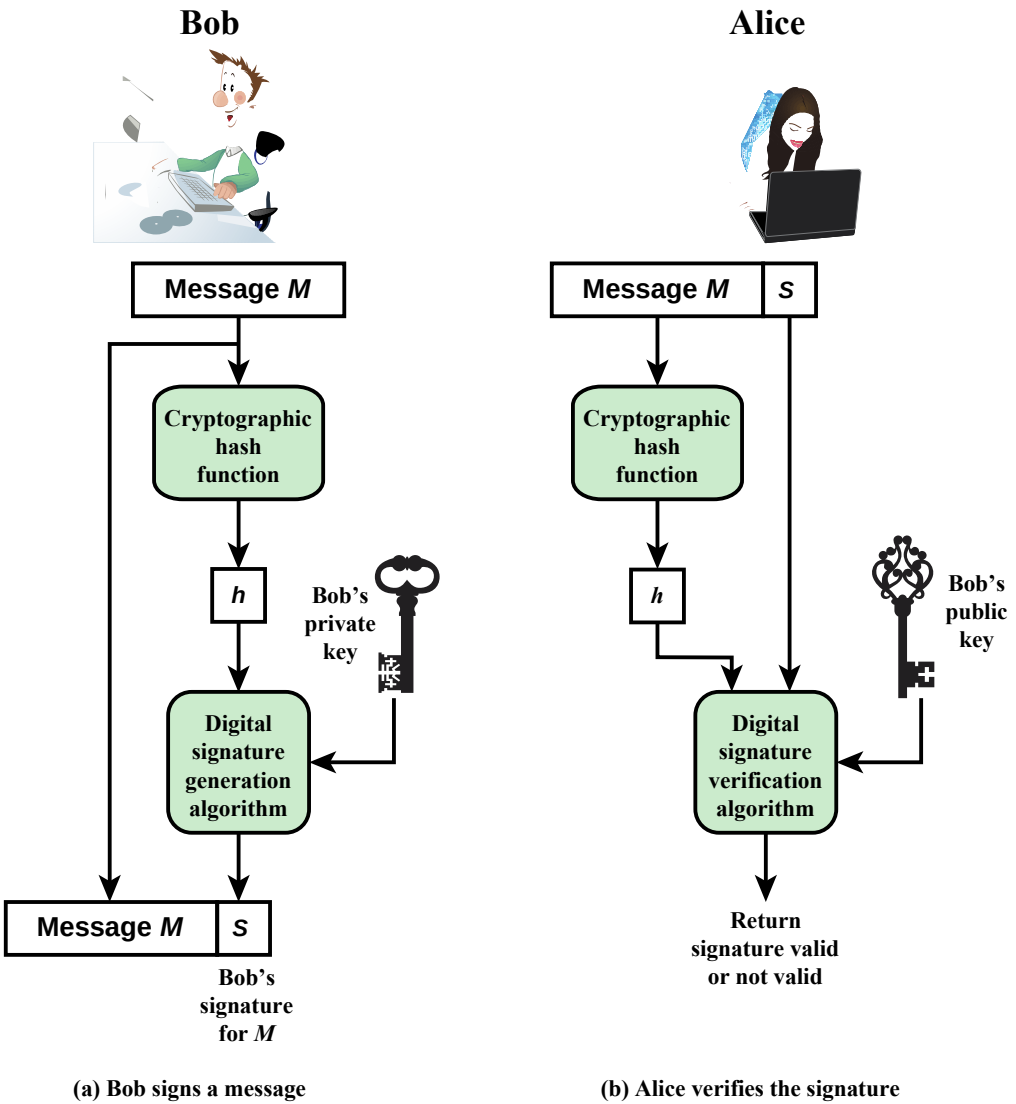
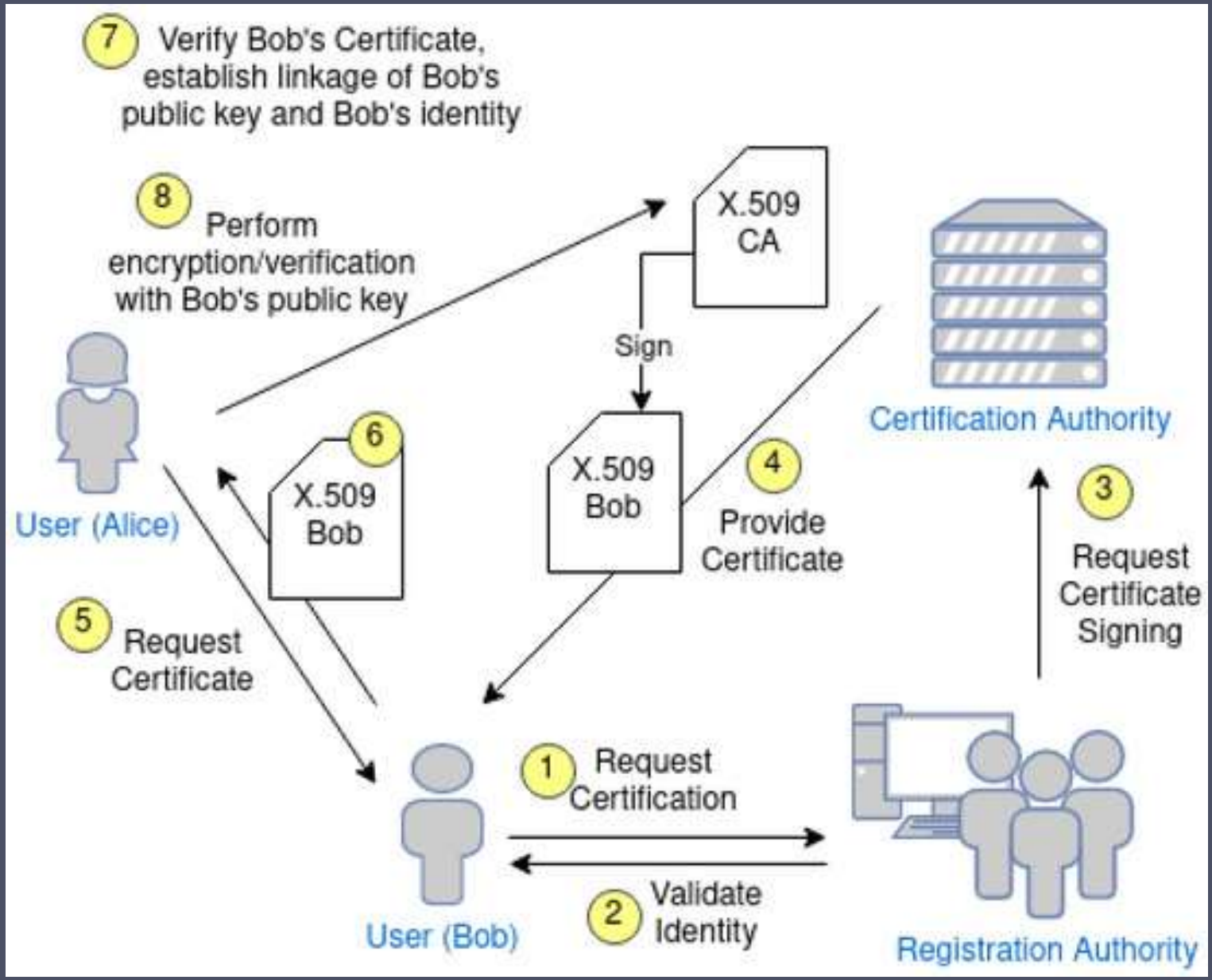


Figure 2.7 Simplified Depiction of Essential Elements of Digital Signature Process

Public Key Infrastructure (PKI)



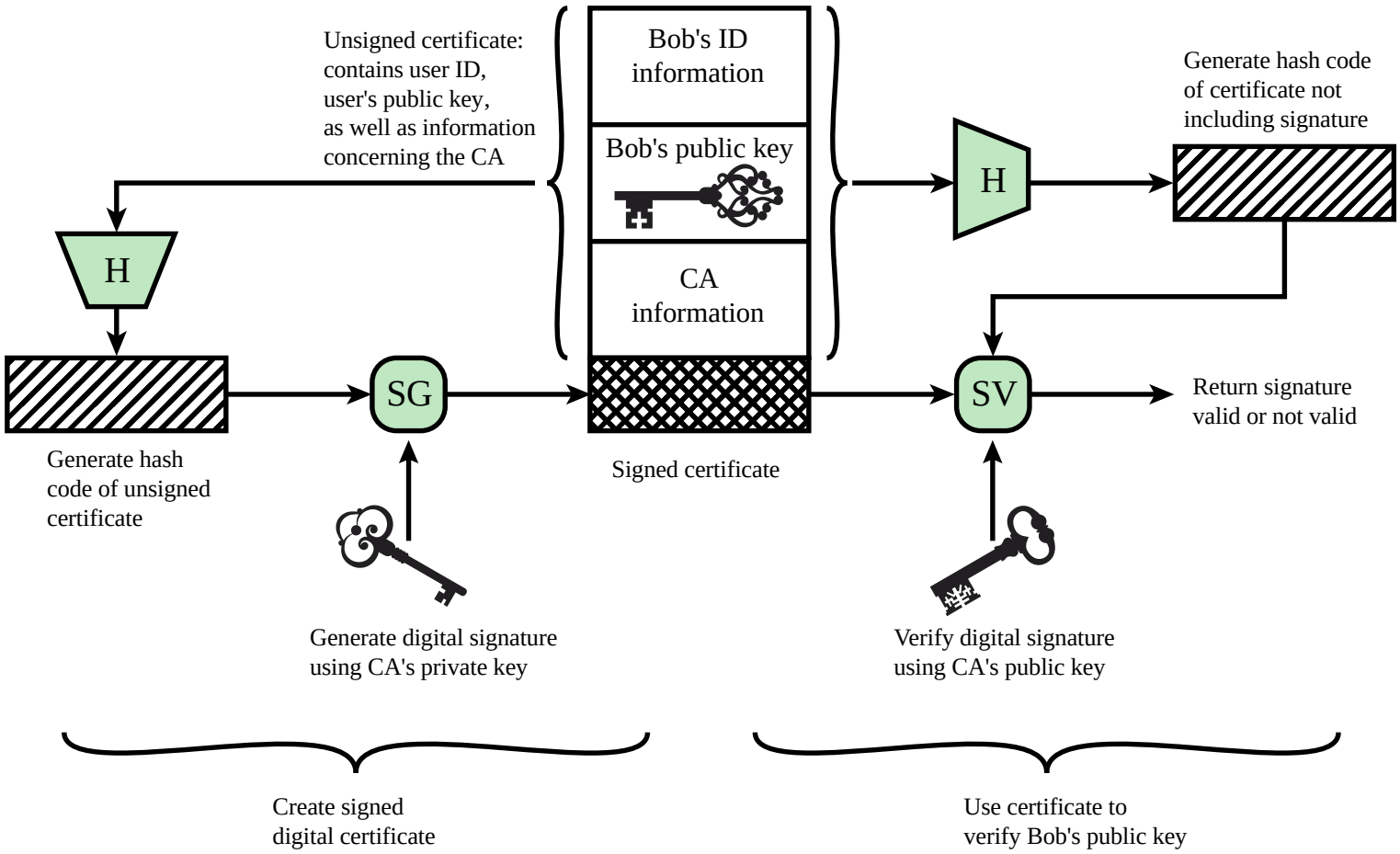
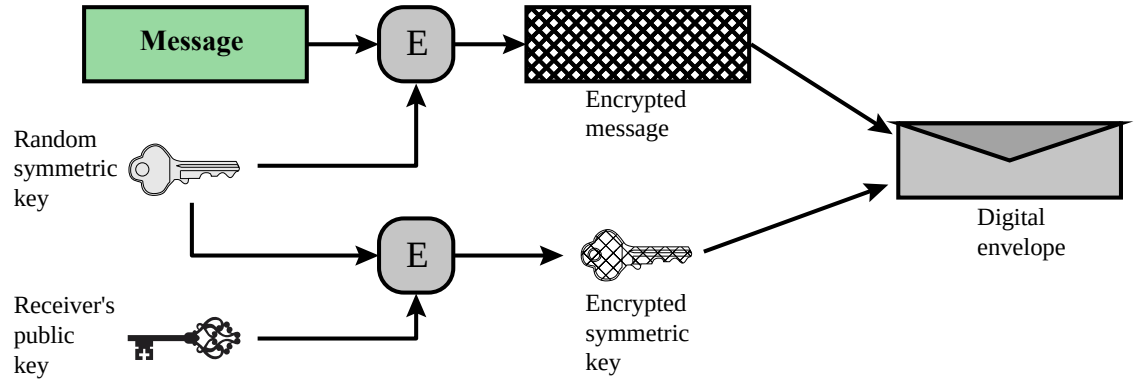
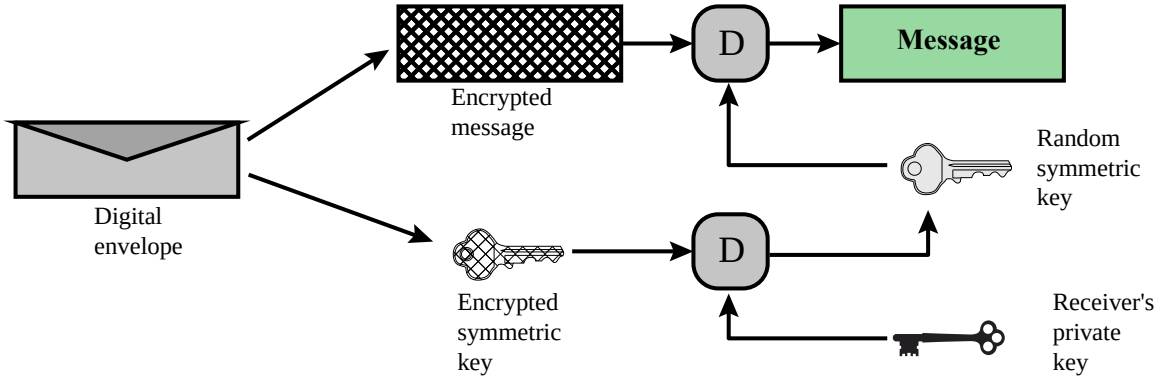


Figure 2.8 Public-Key Certificate Use



(a) Creation of a digital envelope



(b) Opening a digital envelope

Figure 2.9 Digital Envelopes

Certificate Authority (CA) vs. Key Distribution Center (KDC)

- CA in contrast to KDC:
  - CA does not need to be online.
  - CA not a distributed computing entity.
    - Simpler, hence more secure.
  - CA crash merely prevents setting up new users.
  - Certificates are not security sensitive. They can be stored anywhere with universal read privileges.
    - Deleting a certificate would disable the use of the public key.
  - A compromised CA cannot read conversations, fake conversations, …
    - However, it can issue bogus certificates.
- CA more secure, more convenient than KDC.

KDC vs CA

- KDC features
  - All user secret keys in one database
  - Must be available all time
  - Need replication for performance, availability
  - Application: Kerberos
  - Connection: on-line
  - Load: load connection
  - Encryption: single key cryptography
- CA features
  - Simple, need not be online all the time
  - High processing complexity
  - Application: PEM, SSL, S-HTTP
  - Connection: off-line
  - Load: no load connection
  - Encryption: public key cryptography

Practical Application: Encryption of Stored Data

Common to encrypt transmitted data

Much less common for stored data

- There is often little protection beyond domain authentication and operating system access controls
- Data are archived for indefinite periods
- Even though erased, until disk sectors are reused data are recoverable

Approaches to encrypt stored data:

- Use a commercially available encryption package e.g. PGP
- Back-end appliance e.g. VPN using Firewall
- Library based tape encryption
- Background laptop/PC data encryption

Summary

- Confidentiality with symmetric encryption
  - Symmetric encryption
  - Symmetric block encryption algorithms
  - Stream ciphers
- Message authentication and hash functions
  - Authentication using symmetric encryption
  - Message authentication without message encryption
  - Secure hash functions
  - Other applications of hash functions
- Random and pseudorandom numbers
  - The use of random numbers
  - Random versus pseudorandom
- Public-key encryption
  - Structure
  - Applications for public-key cryptosystems
  - Requirements for public-key cryptography
  - Asymmetric encryption algorithms
- Digital signatures and key management
  - Digital signature
  - Public-key certificates
  - Symmetric key exchange using public-key encryption
  - Digital envelopes
- Practical Application: Encryption of Stored Data