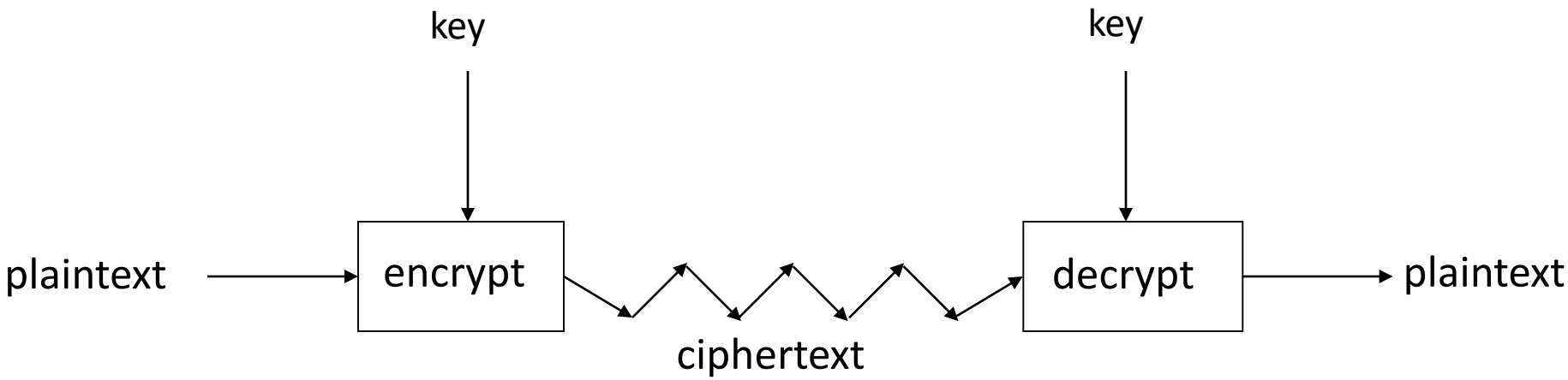
Cryptography

Crypto as Black Box



A generic view

Basic Terminology

- ***plaintext*** - the original message
- ***ciphertext*** - the 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
- ***decipher* (*decrypt*)** - converting ciphertext to plaintext → (knowing the *key*)
- ***cryptography*** - study of encryption principles/methods
- ***cryptanalysis* (*code breaking*)** - the study of principles/ methods of deciphering ciphertext *without* knowing *key*
- ***cryptology*** - the field of both cryptography and cryptanalysis

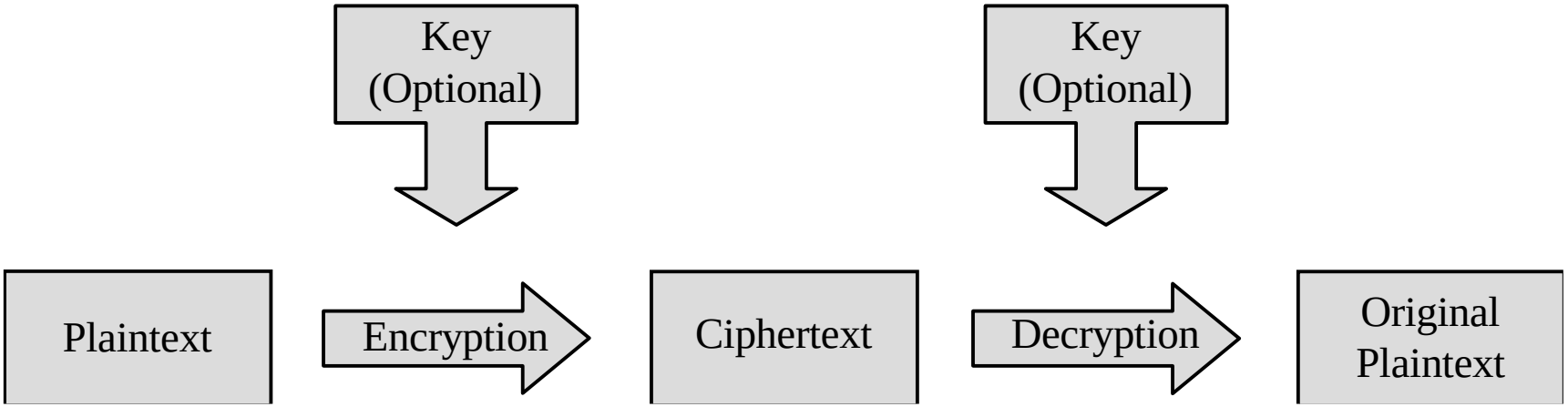
In other words:

- **Cryptology** — The art and science of making and breaking "secret codes"
- **Cryptography** — making "secret codes"
- **Cryptanalysis** — breaking "secret codes"
- **Crypto** — all of the above (and more)

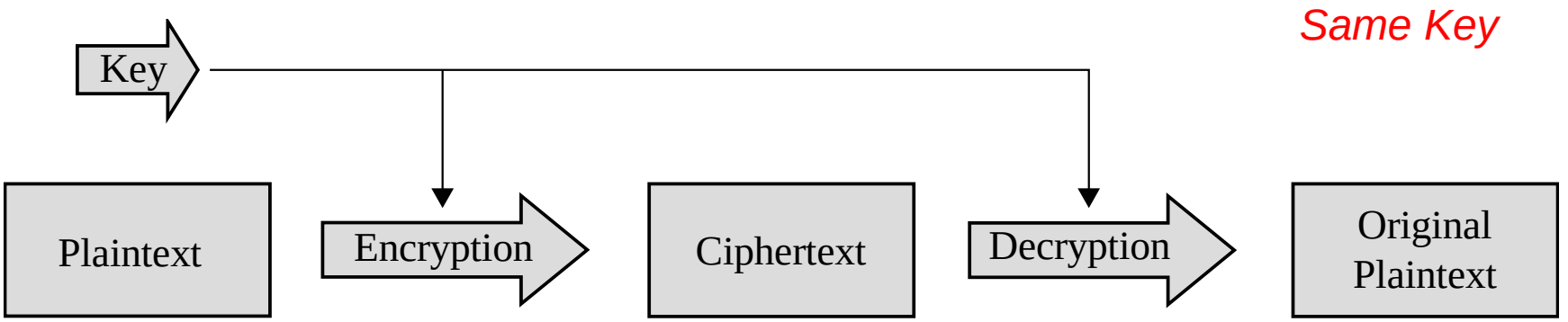
Crypto

- **Basic assumptions**
  - The system is completely known to the attacker
  - Only the key is secret
  - That is, crypto algorithms are not secret ☺
- This is known as **Kerckhoffs' Principle**
- **Why do we make such an assumption?**
  - Experience has shown that secret algorithms tend to be weak when exposed
  - Secret algorithms never remain secret
  - Better to find weaknesses beforehand

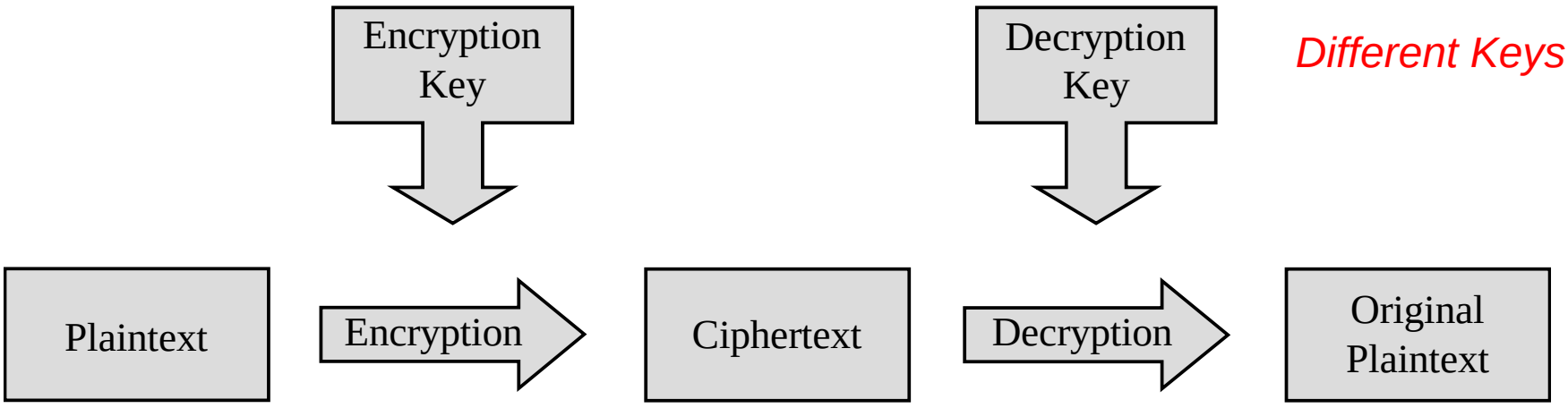
Encryption/Decryption Process



Symmetric vs. Asymmetric

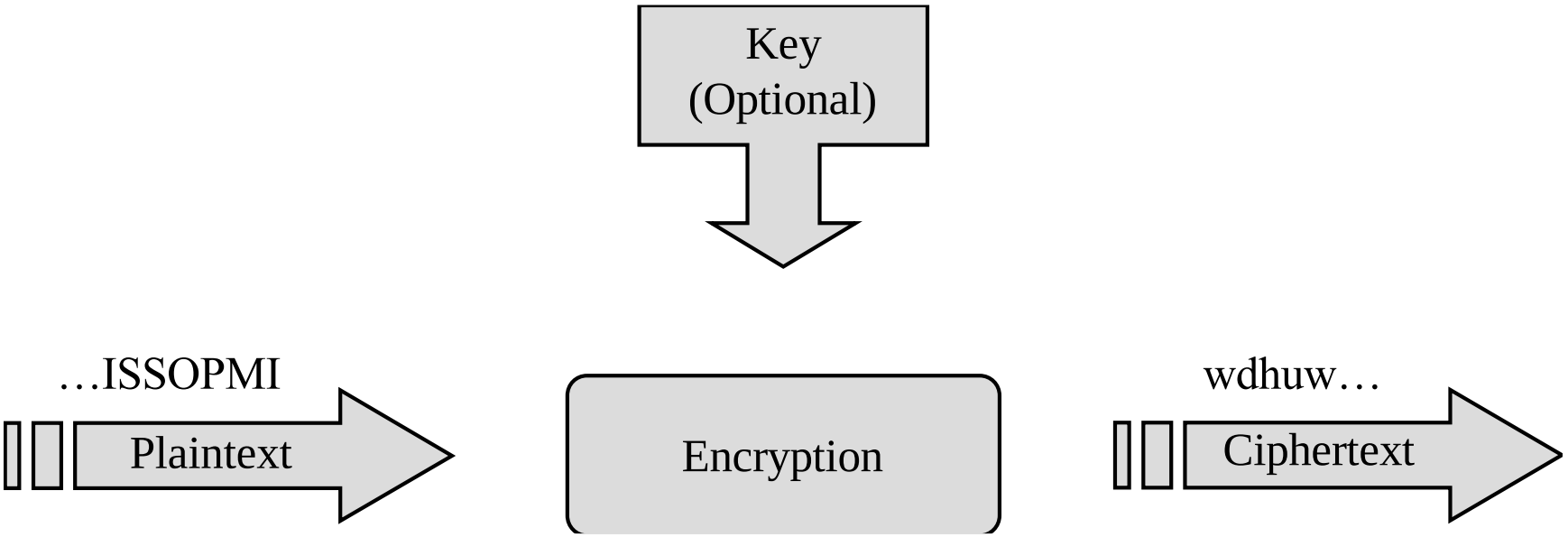


(a) Symmetric Cryptosystem

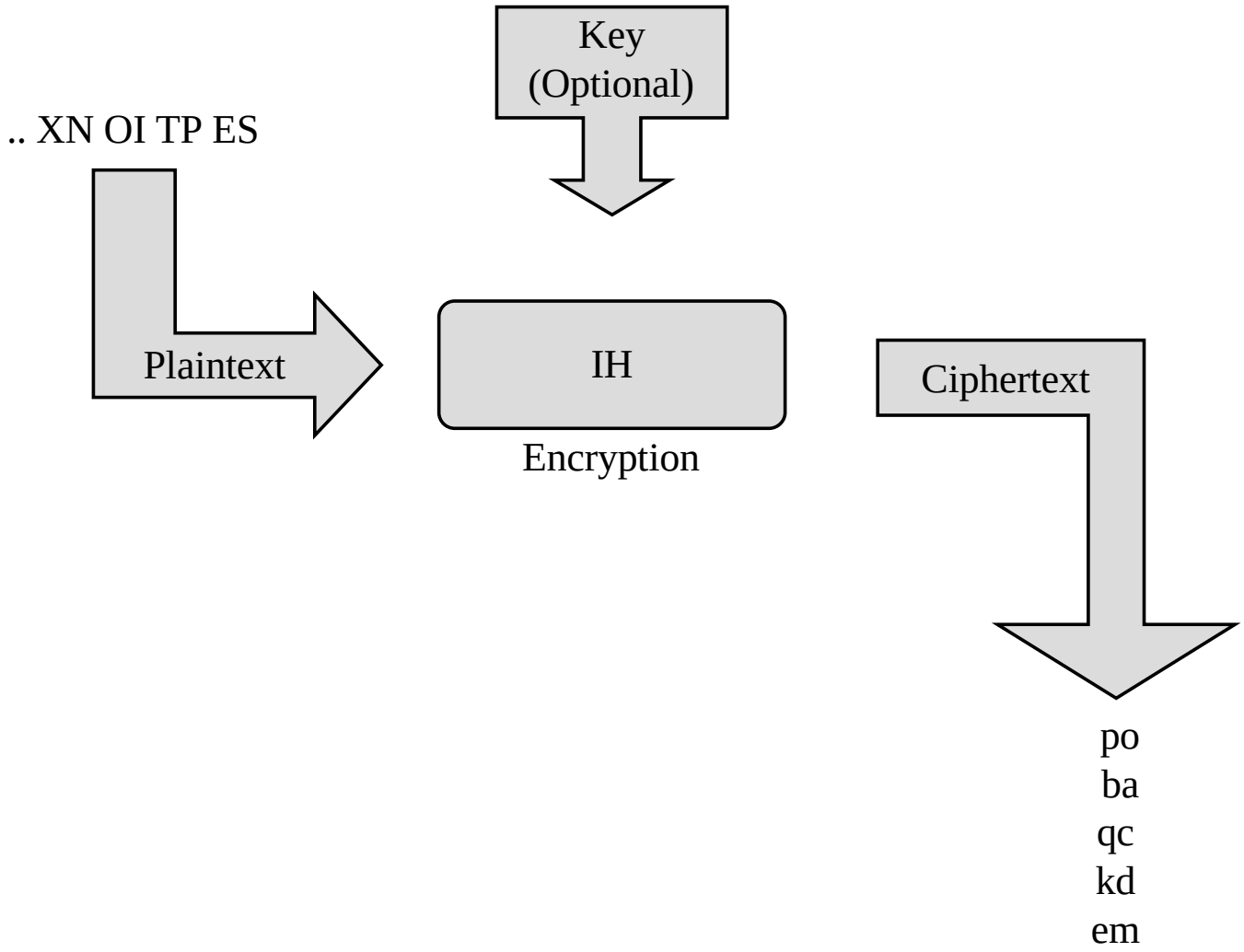


(b) Asymmetric Cryptosystem

Stream Ciphers



Block Ciphers



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

Cryptographic Primitives

- **Substitution**
  - One set of bits is exchanged for another
- **Transposition**
  - Rearranging the order of the ciphertext to break any repeating patterns in the underlying plaintext
- **Confusion**
  - An algorithm providing good confusion has a complex functional relationship between the plaintext/key pair and the ciphertext, so that changing one character in the plaintext causes unpredictable changes to the resulting ciphertext
- **Diffusion**
  - Distributes the information from single plaintext characters over the entire ciphertext output, so that even small changes to the plaintext result in broad changes to the ciphertext

Classification of Cryptography

- **Number of keys used**
  - Secret key cryptography: one key
  - Public key cryptography: two keys – (public, private)
- **Type of encryption operations used**
  - substitution / transposition / product
- **Way in which plaintext is processed**
  - block / stream

Cryptanalysis Scheme

- **Ciphertext only**

  – only know algorithm / ciphertext, statistical, can identify plaintext

- **Known plaintext**

  – know/suspect/guess plaintext & ciphertext to attack cipher

- **Chosen plaintext**

  – select plaintext and obtain ciphertext to attack cipher

- **Chosen ciphertext**

  – select ciphertext and obtain plaintext to attack cipher

- **Chosen text**

- – select either plaintext or ciphertext to en/decrypt to attack cipher

Unconditional vs. Computational Security

- **Unconditional security**
  - No matter how much computer power is available → the cipher cannot be broken
  - The ciphertext provides insufficient information to uniquely determine the corresponding plaintext
  - Only one-time pad scheme qualifies (*one-time pre-shared key*)
- **Computational security**
  - **Cost** of breaking the cipher exceeds the value of the encrypted info
  - **Time** required to break the cipher exceeds the useful lifetime of the info
    - *(either takes too long, or is too expensive)*

Brute Force Search

- Always possible to simply try every key !!!
- Most basic attack, proportional to **key size**
- Assume either know / recognise plaintext

Key Size (bits)	Number of Alternative Keys	Time required at 1 encryption/μs	Time required at 10^6 encryptions/μs
32	$2^{32} = 4.3 \times 10^9$	$2^{31}\ \mu s = 35.8$ minutes	2.15 milliseconds
56	$2^{56} = 7.2 \times 10^{16}$	$2^{55}\ \mu s = 1142$ years	10.01 hours
128	$2^{128} = 3.4 \times 10^{38}$	$2^{127}\ \mu s = 5.4 \times 10^{24}$ years	5.4×10^{18} years
168	$2^{168} = 3.7 \times 10^{50}$	$2^{167}\ \mu s = 5.9 \times 10^{36}$ years	5.9×10^{30} years
26 characters (permutation)	$26! = 4 \times 10^{26}$	$2 \times 10^{26}\ \mu s = 6.4 \times 10^{12}$ years	6.4×10^{6} years

Classical Substitution Ciphers

- Letters of plaintext are replaced by other letters or by numbers or symbols
- Plaintext is viewed as a sequence of bits, then substitution replaces plaintext bit patterns with ciphertext bit patterns

Caesar Cipher

- Simplest and earliest known use of a substitution cipher
- Used by Julius Caesar
- Involves replacing each letter of the alphabet with the letter standing three places further down the alphabet
  - For example, when *key=3*
    - Replaces each letter by 3rd letter on
    - Alphabet is wrapped around so that the letter following Z is A
- Example:

plain: meet me after the toga party

cipher: PHHW PH DIWHU WKH WRJD SDUWB

Caesar Cipher

- **Define transformation as:**

  a b c d e f g h i j k l m n o p q r s t u v w x y z

  D E F G H I J K L M N O P Q R S T U V W X Y Z A B C

- **Mathematically give each letter a number**

  a b c d e f g h i j k l m

  0 1 2 3 4 5 6 7 8 9 10 11 12

  n o p q r s t u v w x y Z

  13 14 15 16 17 18 19 20 21 22 23 24 25

- **Then have Caesar cipher as:**

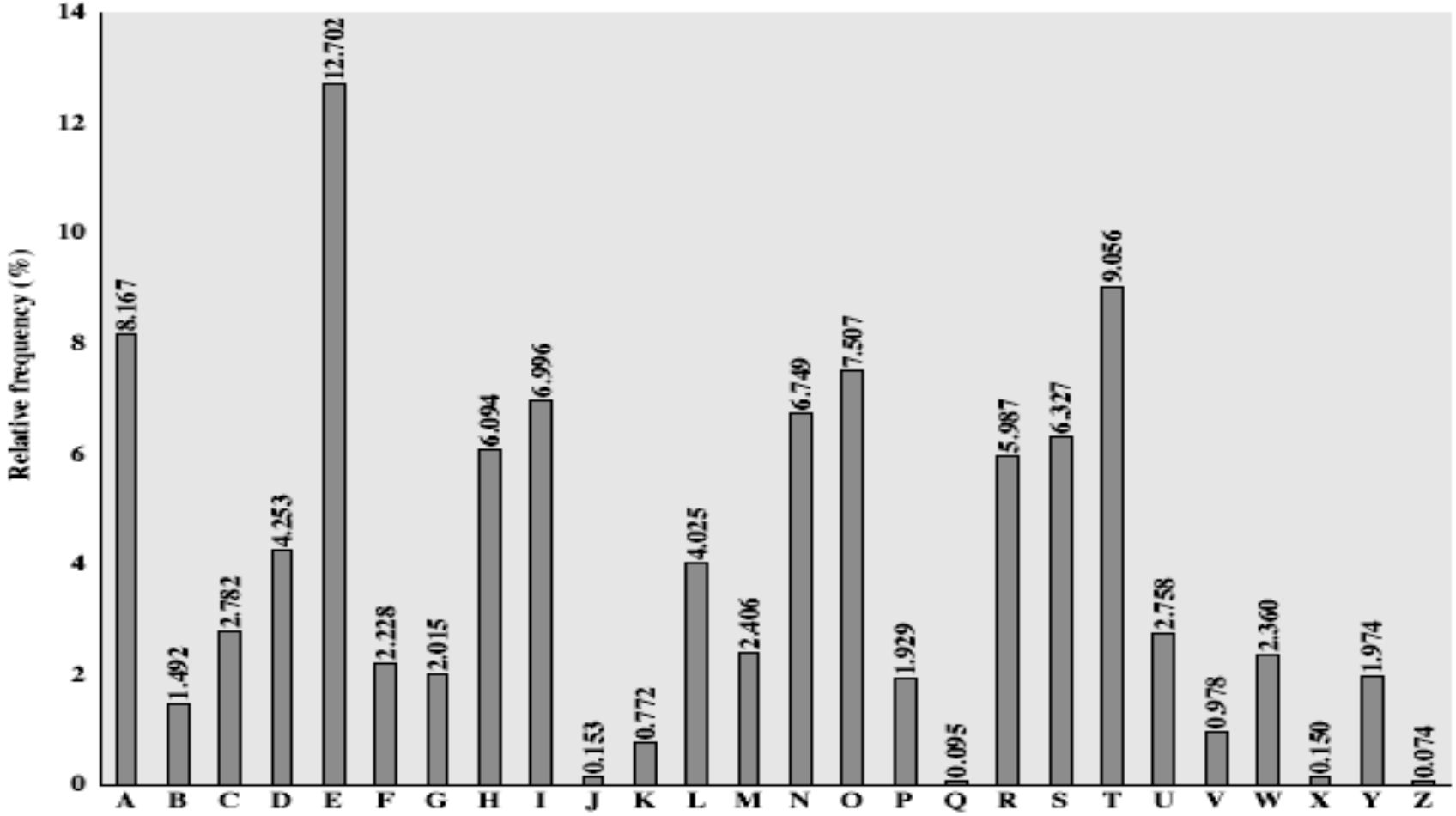
  $C = E(p) = (p + k) \bmod (26)$

  $p = D(C) = (C - k) \bmod (26)$

Brute-Force Cryptanalysis of Caesar Cipher

KEY						
	PHHW	PH	DIWHU	WKH	WRJD	SDUWB
1	oggv	og	chvgt	vjg	vqic	rctva
2	nffu	nf	bgufs	uif	uphb	qbsuz
3	meet	me	after	the	toga	party
4	ldds	ld	zesdq	sgd	snfz	ozqsx
5	kccr	kc	ydrcp	rfc	rmey	nyprw
6	jbbq	jb	xcqbo	qeb	qldx	mxoqv
7	iaap	ia	wbpan	pda	pkcw	lwnpu
8	hzzo	hz	vaozm	ocz	ojbv	kvmot
9	gyyn	gy	uznyl	nby	niau	julns
10	fxxm	fx	tymxk	max	mhzt	itkmr
11	ewwl	ew	sxlwj	lzw	lgys	hsjlq
12	dvvk	dv	rwkvi	kyv	kfxr	grikp
13	cuuj	cu	qvjuh	jxu	jewq	fqhjo
14	btti	bt	puitg	iwt	idvp	epgin
15	assh	as	othsf	hvs	hcuo	dofhm
16	zrrg	zr	nsgre	gur	gbtn	cnegl
17	yqqf	yq	mrfqd	ftq	fasm	bmdfk
18	xppe	xp	lqepc	esp	ezrl	alcej
19	wood	wo	kpdob	dro	dyqk	zkbdi
20	vnnc	vn	jocna	cqn	cxpj	yjach
21	ummb	um	inbmz	bpm	bwoi	xizbg
22	tlla	tl	hmaly	aol	avnh	whyaf
23	skkz	sk	glzkx	znk	zumg	vgxze
24	rjjy	rj	fkyjw	ymj	ytlf	ufwyd
25	qiix	qi	ejxiv	xli	xske	tevxc

English Letter Frequencies



Example Cryptanalysis

- Given ciphertext:

  UZQSOVUOHXMOPVGPOZPEVSGZWSZOPFPESXUDBMETSXAIZ
  VUEPHZHMDZSHZOWSFPAPPDTSVPQUZWYMXUZUHSX
  EPYEPOPDZSZUFPOMBZWPFUPZHMDJUDTMOHMQ

- Count relative letter frequencies (*see text*)
- Guess **P** & **Z** are **e** and **t**
- Guess **ZW** is **th** and hence **ZWP** is **the**
- Proceeding with trial and error finally get:

  it was disclosed yesterday that several informal but
  direct contacts have been made with political
  representatives of the viet cong in moscow

Transposition Ciphers

- Now consider classical **transposition** or **permutation** ciphers
- These hide the message by rearranging the letter order, *without* altering the actual letters used
- Can recognise these since have the same frequency distribution as the original text

Rail Fence cipher (zigzag cipher)

- Write message letters out diagonally over a number of rows
- Then read off cipher row by row
- E.g., write message out as:

Plaintext	T	H	I	S	I	S	A	S	E	C	R	E	T	M	E	S	S	A	G	E

- Giving ciphertext: (*key*=2)

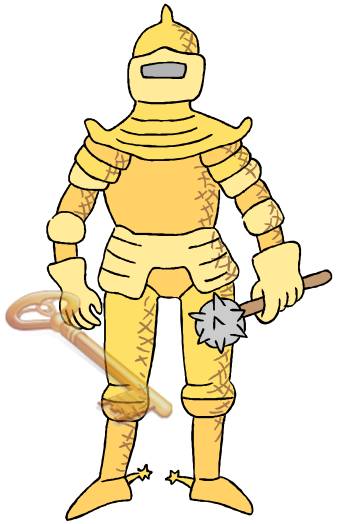
Rail Fence	T		I		I		A		E		R		T		E		S		G	
Encoding		H		S		S		S		C		E		M		S		A		E

Product Ciphers

- Ciphers using substitutions or transpositions are ***not secure*** because of *language characteristics*
- Hence consider using several ciphers in succession to make it harder, **but**:
  - Two substitutions make a more complex substitution
  - Two transpositions make more complex transposition
  - But a substitution followed by a transposition makes a new much harder cipher
- This is the bridge from classical to modern ciphers

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*



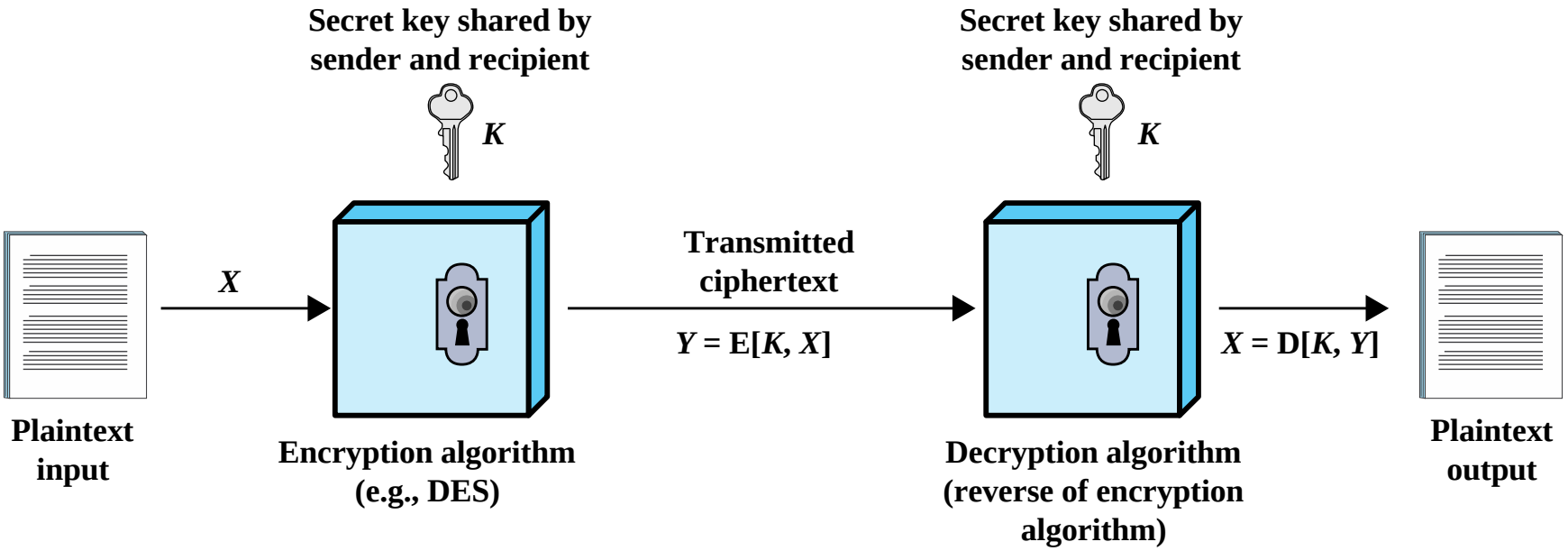


Figure 2.1 Simplified Model of Symmetric Encryption

Attacking Symmetric Encryption

Two Different Approaches

1) Cryptanalytic Attacks

- Rely on:
  - Nature of the **algorithm**
  - Some knowledge of the **general characteristics** of the ***plaintext***
  - Some sample ***plaintext-ciphertext pairs***
- Exploits the characteristics of the algorithm to attempt to deduce a specific plaintext or the key being used
  - If successful all future and past messages encrypted with that key are compromised

2) Brute-Force Attack

- **Try all possible keys** on some ***ciphertext*** until an intelligible translation into plaintext is obtained
  - On **average half of all possible** keys must be tried to achieve success

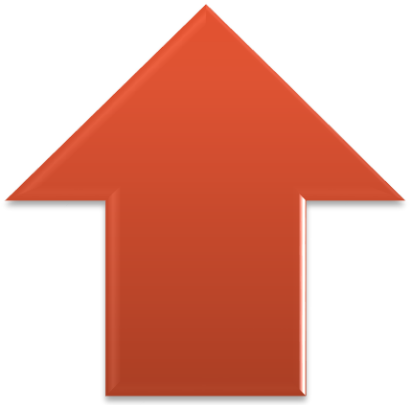
Comparison of Three Popular Symmetric Encryption Algorithms

	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

Table 2.1

Data Encryption Standard (DES)



- **The most widely used encryption scheme**
  - FIPS PUB 46 (*Federal Information Processing Standards*)
  - 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 algorithm
    - DES is the most studied encryption algorithm in existence
  - Use of 56-bit key (serious concern)
    - Electronic Frontier Foundation (EFF) announced in July 1998 that it had broken a DES encryption

Average Time Required for Exhaustive Key Search (*brute force approach*)

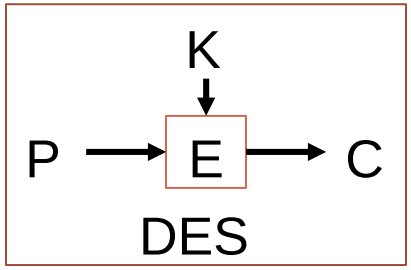
Key size (bits)	Cipher	Number of Alternative Keys	Time Required at 10^9 decryptions/s	Time Required at 10^{13} decryptions/s
56	DES	$2^{56} \approx 7.2 \times 10^{16}$	2^{55} ns = 1.125 years	1 hour
128	AES	$2^{128} \approx 3.4 \times 10^{38}$	2^{127} ns = 5.3×10^{21} years	5.3×10^{17} years
168	Triple 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

Table 2.2

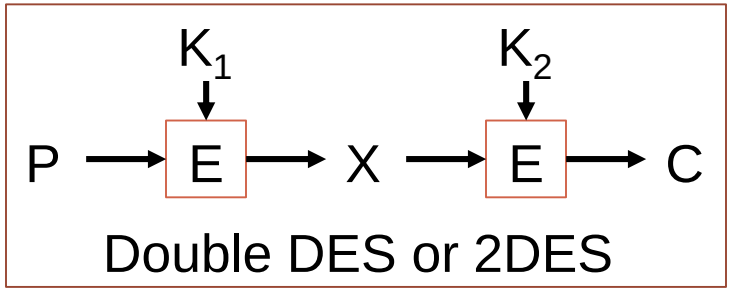
Triple DES (3DES)

- 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
    - 3DES requires three times as many calculations as DES
  - Underlying encryption algorithm is the same as in DES
- **Drawbacks:**
  - Algorithm is ***sluggish*** in software
  - Uses a ***64-bit*** block size

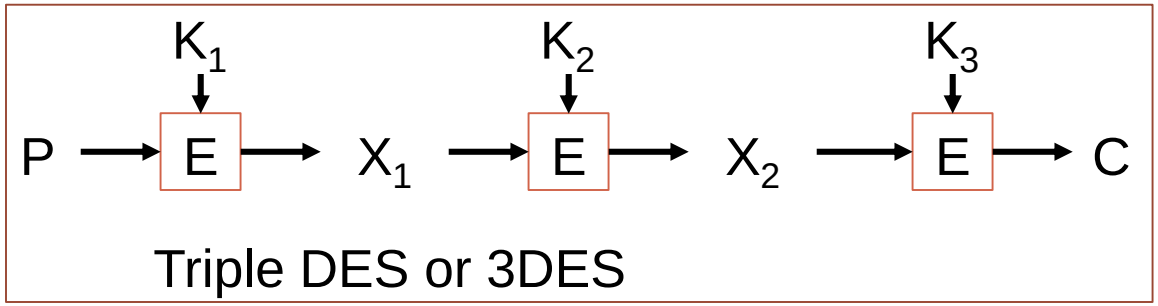
DES vs. Double DES vs. 3DES



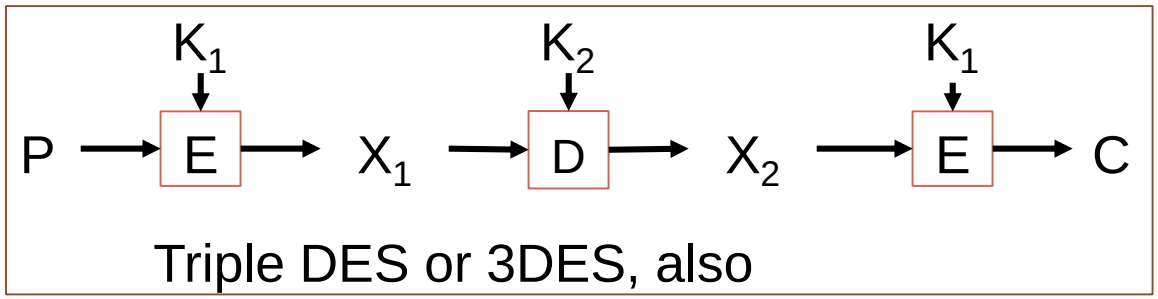
Key Space: 2^{56}



Key Space: $2^{56} * 2^{56}$



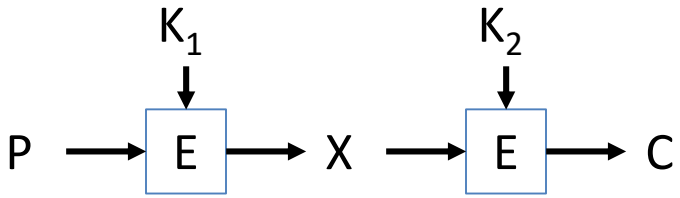
Key Space: $2^{56} * 2^{56} * 2^{56}$



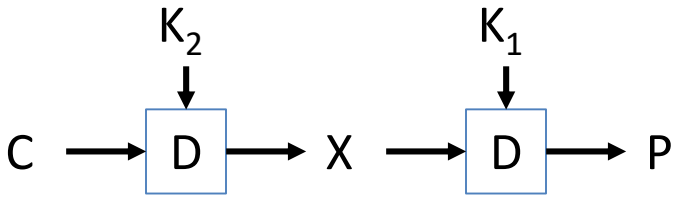
Key Space: $2^{56} * 2^{56}$

Meet In The Middle (MITM) Attack

- Double DES (2DES) is supposed to give 112-bit key length.
  - **E(E(P,K_1), K_2)** and **D(D(P,K_1), K_2)**



Double DES: **Encryption**



Double DES: **Decryption**

  - $2^{56} * 2^{56} = 2^{112}$, key space (*possible keys*)
  - Instead, attackers may end up trying 2^{57}, *how come*?
    - *Thus, double encryption is not that significant in terms of added security, but why?*

Advanced Encryption Standard (*AES*)

Needed a replacement for 3DES

3DES was not reasonable for long term use

NIST called for proposals for a new AES in 1997

- 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

Selected Rijndael in November 2001

Published as FIPS 197

Practical Security Issues

- Typically symmetric encryption is applied to a unit of data larger than a single **64-bit** or **128-bit** block
- **E**lectronic **C**ode**B**ook (**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, **but *how can we prevent such an attack?***
- **Modes of operation**
  - Alternative techniques developed to increase the security of symmetric block encryption for large sequences
  - Overcomes the weaknesses of ECB

Block Cipher Modes

- Using a **stream cipher** is easy—you generate a keystream that is the same length as the plaintext (or ciphertext) and **XOR**.
- Using a **block cipher** is also easy, provided that you have exactly one block to encrypt.
- But how should **multiple blocks be encrypted** with a block cipher?
  - It turns out that the answer is ***not as straightforward as it might seem***.

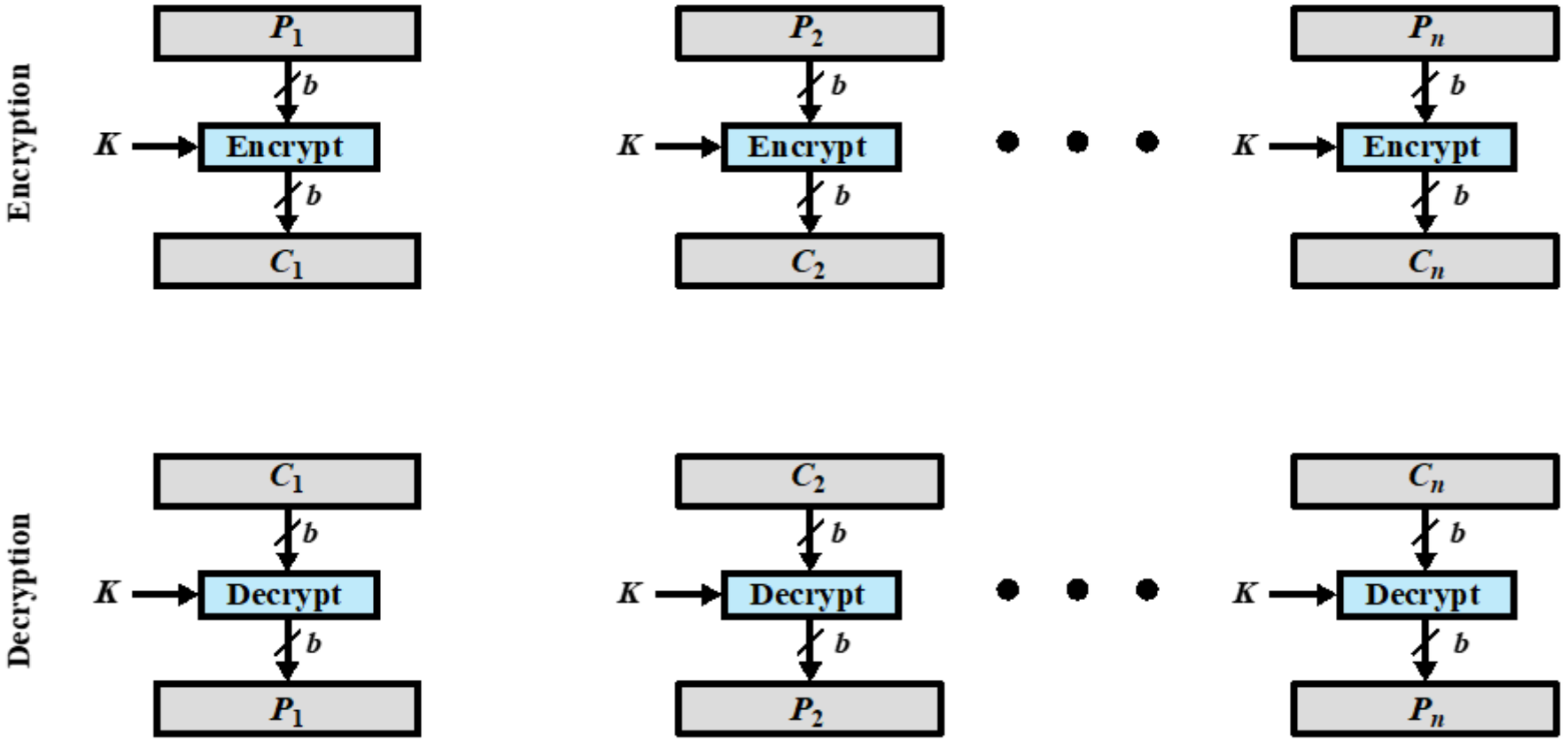
- How to encrypt multiple blocks?
- Do we need a ***new key*** for each block?
  - If so, as impractical as a One-Time Pad!
- Encrypt **each block independently**?

Block Cipher: Modes of Operation

- Many modes —— we discuss 3 most popular

1. **Electronic Codebook (ECB) mode**
   - Encrypt each block independently
   - Most obvious approach, but a bad idea, (*we will see why* ?)
2. **Cipher Block Chaining (CBC) mode**
   - Chain the blocks together
   - More secure than **ECB**, virtually no extra work
3. **Counter Mode (CTR) mode**
   - Block ciphers acts like a stream cipher
   - Popular for random access

ECB: Electronic Codebook



(a) Block cipher encryption (electronic codebook mode)

*the **same key** is used for all blocks*

ECB: Electronic Codebook Mode

- Notation: $C = E(P, K)$
- Given plaintext $P_0, P_1, \ldots, P_m, \ldots$
- Most obvious way to use a **ECB** block cipher:

Encrypt	Decrypt
$C_0 = E(P_0, K)$	$P_0 = D(C_0, K)$
$C_1 = E(P_1, K)$	$P_1 = D(C_1, K)$
$C_2 = E(P_2, K)$ …	$P_2 = D(C_2, K)$ …

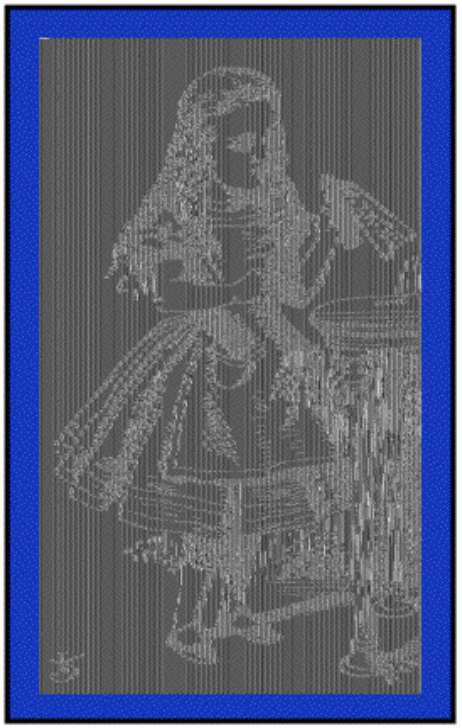
- An attacker might observes that $C_i = C_j$.
- Then the attacker knows that $P_i = P_j$.
- This gives some information, even if P_i or P_j **are unknown, and if** P_i is known then now is P_j
- Although this may seem innocent enough, there are cases where the attacker will know part of the plaintext, and any match with a known block reveals another block.

ECB: Electronic Codebook

- Simplest mode
- Plaintext is handled ***b*-bit** (block) at a time and each block is encrypted using the **same key**
- "**Codebook**" is used because there is a unique ciphertext for every ***b*-bit** block of plaintext
  - **Not** secure for long messages
    - since **repeated** plaintext is seen in **repeated** ciphertext
    - *this is a serious security issues and it should never be used in practice*.
  - To overcome security deficiencies:
    - you need a technique where the **same** plaintext block, if repeated, produces **different** ciphertext blocks ??

Alice Hates **ECB** Mode

- Alice's uncompressed image, and **ECB** encrypted



- Why does this happen?
- Same plaintext yields same ciphertext!

CBC: Cipher Block Chaining Mode

- Blocks are "**chained**" together
- A random **Initialization Vector** (**IV**) is required to start with the **CBC** mode
- **IV** is random, but not secret

Encryption

$$C_0 = E(IV \oplus P_0, K),$$
$$C_1 = E(C_0 \oplus P_1, K),$$
$$C_2 = E(C_1 \oplus P_2, K), \ldots$$

Decryption

$$P_0 = IV \oplus D(C_0, K),$$
$$P_1 = C_0 \oplus D(C_1, K),$$
$$P_2 = C_1 \oplus D(C_2, K), \ldots$$

- Analogous to classic codebook *with additive*

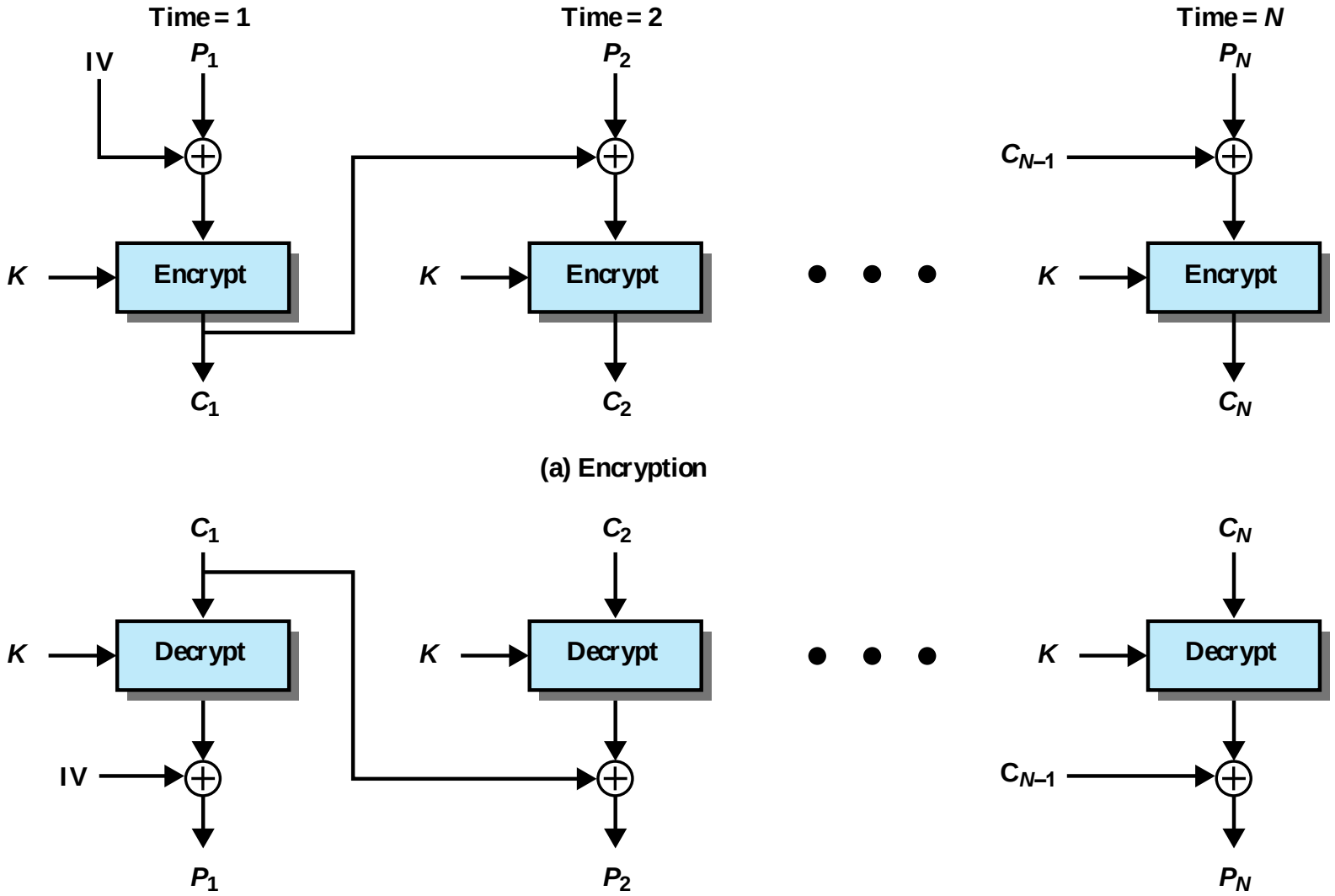


Figure 20.7 Cipher Block Chaining (CBC) Mode

CBC: Cipher Block Chaining Mode

- **Pros**: Using the **CBC**, identical plaintext blocks yield **different ciphertext** blocks —— **this is very good**!
- But what about **errors in transmission** (*error propagation*)?
  - Suppose the ciphertext block C_i is garbled to, say, $G \neq C_i$. Then

    $$P_i \neq D(G, K) \oplus C_{i-1} \text{ and } P_{i+1} \neq D(C_{i+1}, K) \oplus G$$

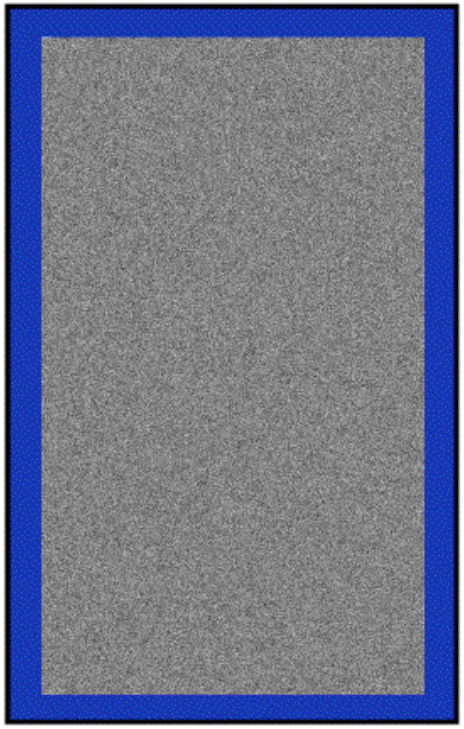
  - But

    $$P_{i+2} = D(C_{i+2}, K) \oplus C_{i+1}, \ \ldots$$

- However, all subsequent blocks are decrypted correctly.
  - That is, each plaintext block only depends on two consecutive ciphertext blocks, so errors do **not** propagate **beyond two blocks**. But, the fact that a single-bit error can cause two entire blocks to be garbled is a **serious concern** in high error-rate environments such as wireless.
  - **Stream ciphers do not have this problem**—a single garbled ciphertext bit results in a single garbled plaintext bit—and that is one reason why stream ciphers are often preferred in wireless applications

Alice Likes **CBC** Mode

- Alice's uncompressed image, Alice CBC encrypted



- Why does this happen?
- **Same** plaintext yields different ciphertext!

CTR: Counter Mode

- **CTR** is popular for random access
- Use block cipher like a stream cipher

Encryption

$C_0 = P_0 \oplus E(IV, K),$

$C_1 = P_1 \oplus E(IV+1, K),$

$C_2 = P_2 \oplus E(IV+2, K), \ldots$

Decryption

$P_0 = C_0 \oplus E(IV, K),$

$P_1 = C_1 \oplus E(IV+1, K),$

$P_2 = C_2 \oplus E(IV+2, K), \ldots$

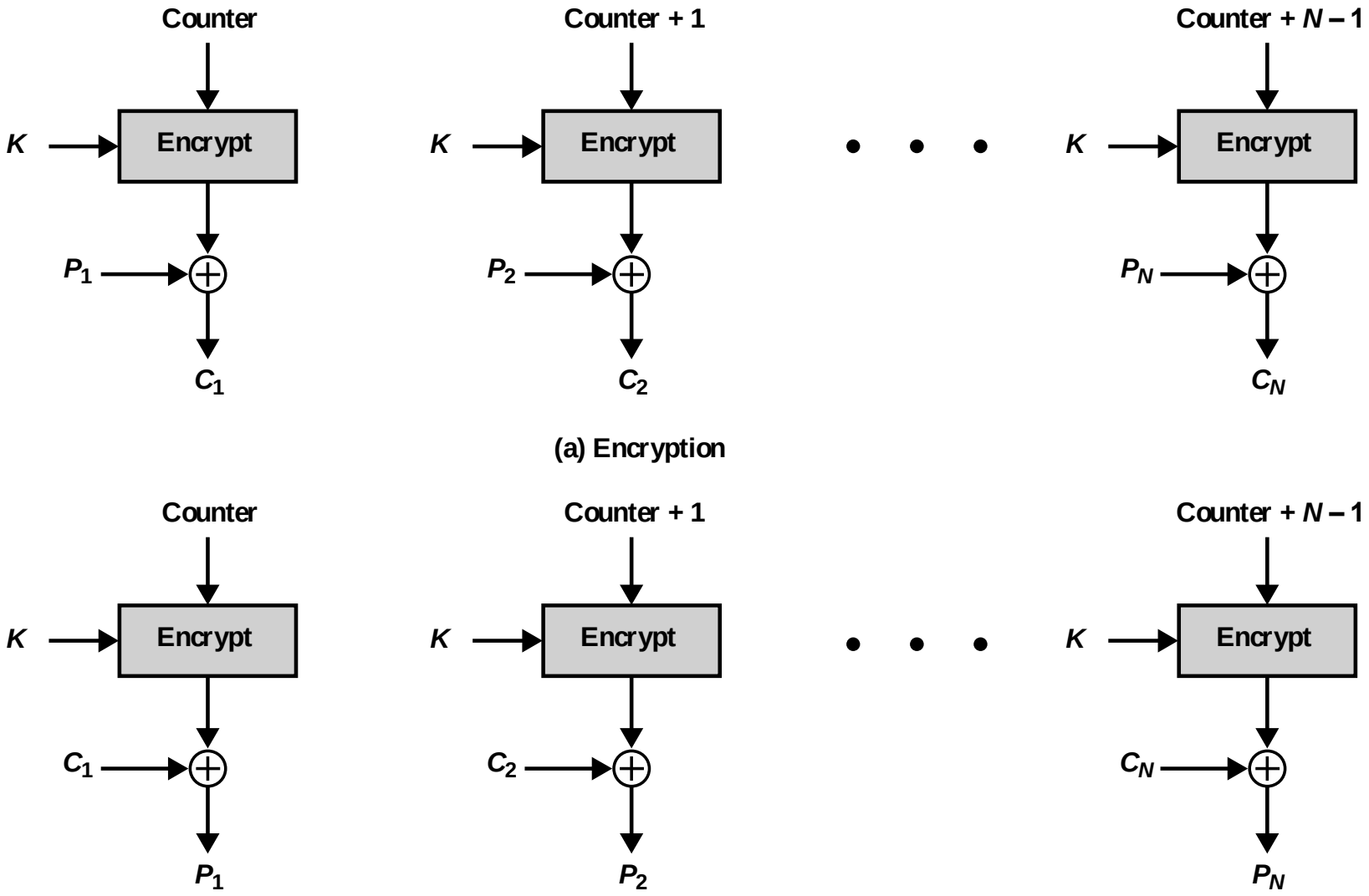


Figure 20.9 Counter (CTR) Mode

Message Authentication

- **Encryption** protects against **passive attack** (***eavesdropping***).
- A different requirement is needed to protect against **active attack** (*falsification of data and transactions*). This protection is known as **message or data authentication**.

Protects against *active attacks*

Verifies received message is authentic

- **Contents** have not been altered
- From authentic **source**
- **Timely** and in correct **sequence**

Can use conventional encryption

- Only sender & receiver share a key

MAC: Message Authentication Code

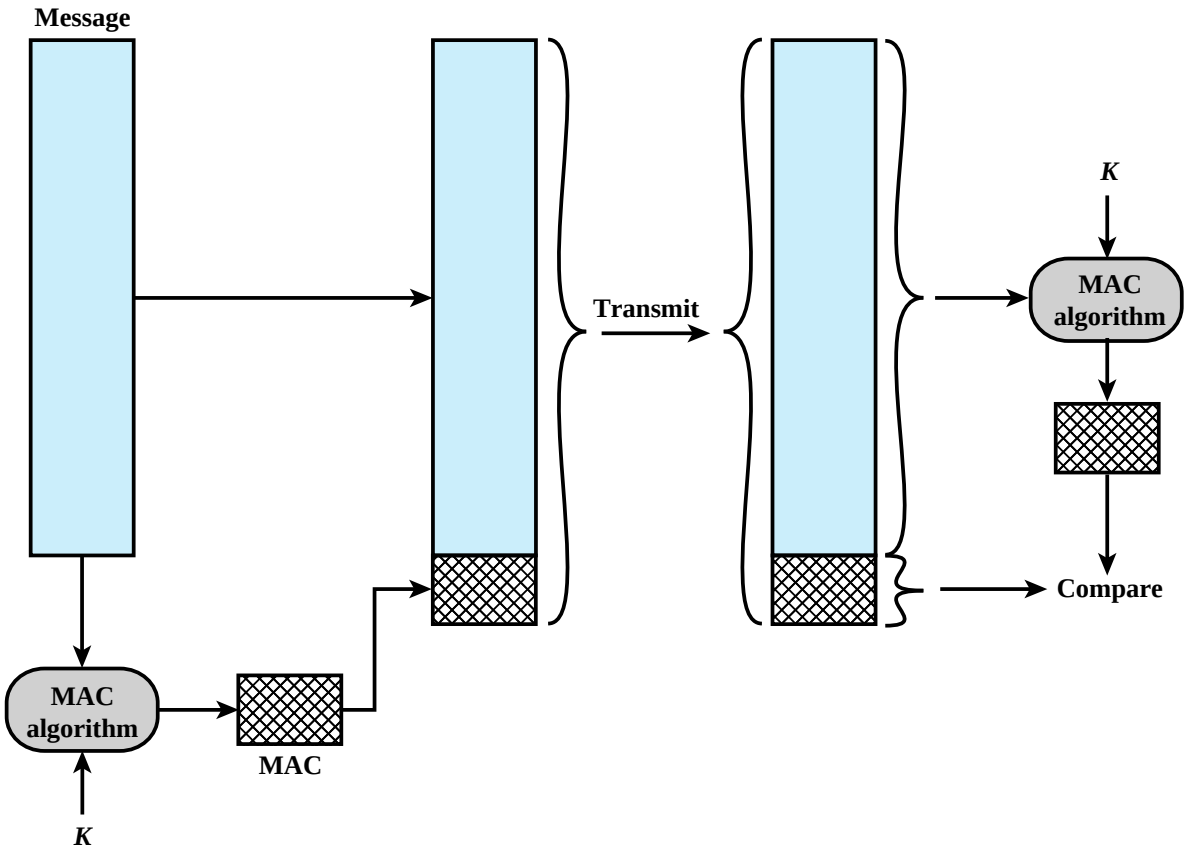


Figure 2.3 Message Authentication Using a Message Authentication Code (MAC).

Assumes that two parties, say **A** and **B**, share a common secret key K_{AB}

A calculates a **MAC** as a function $MAC_M = F(K_{AB}, M)$.

Then, both M *&* MAC_M *are sent to* **B**. *Then,* **B** *recalculates* MAC_M *and compares them*

MAC: Message Authentication Code

1. **The receiver is assured that the message has not been altered.**
   - **If an attacker** alters the message but does not alter the code, then the receiver's calculation of the code will differ from the received code. Because the attacker is *assumed not to know the secret key*, the attacker cannot alter the code to correspond to the alterations in the message.

2. **The receiver is assured that the message is from the alleged sender.**
   - ***Because*** *no one else knows the secret key, no one else could prepare a message with a proper code.*

3. **If the message includes a sequence number**, then the receiver can be assured of the **proper sequence**,
   - ***because*** *an attacker cannot successfully alter the sequence number.*

- An **alternative** to the **MAC** is the **one-way hash function**.
  - *(i.e. SHA-1, SHA-2, SHA-256, SHA-384)*
- As with the MAC, a **hash function** accepts a **variable-size message M** as input and produces a **fixed-size value (message digest) H(M)** as output

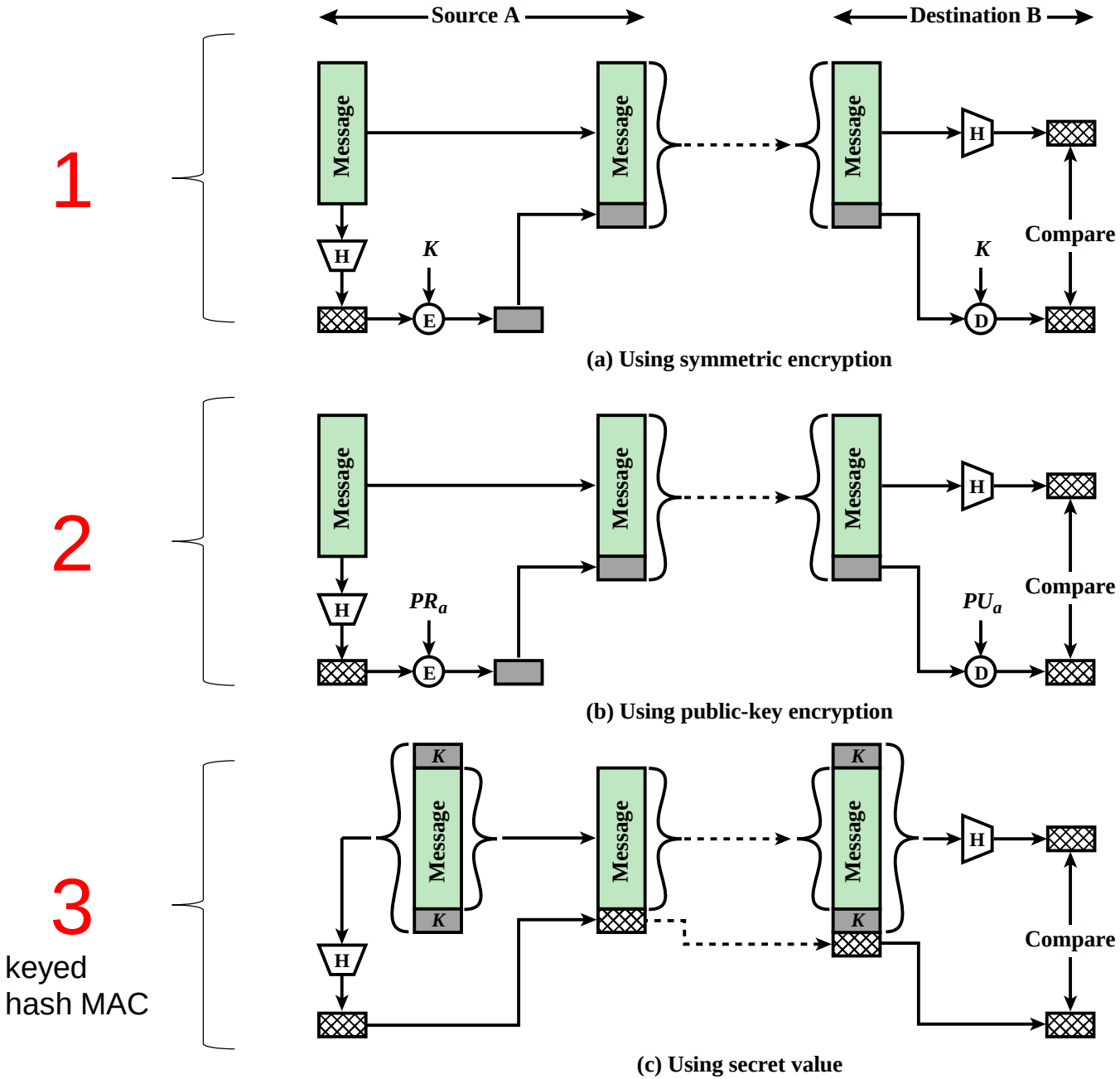


Figure 2.5 Message Authentication Using a One-Way Hash Function.

Hash Function Requirements

- A hash function can be used to produce a "***fingerprint***" of a file, message, or block of data.
- 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 ≠ 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

* **SHA-1** (Secure Hash Algorithm 1) (160-bit); *Not secure any more*

* **SHA-2** family consists of six hash functions
(SHA-256, → 256 bits
SHA-384→ 384 bits
SHA-512→ 512 bits)

***SHA-3** is the latest member of the Secure Hash Algorithm family of standards (released by NIST on August 2015)

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

Public-Key Encryption Structure

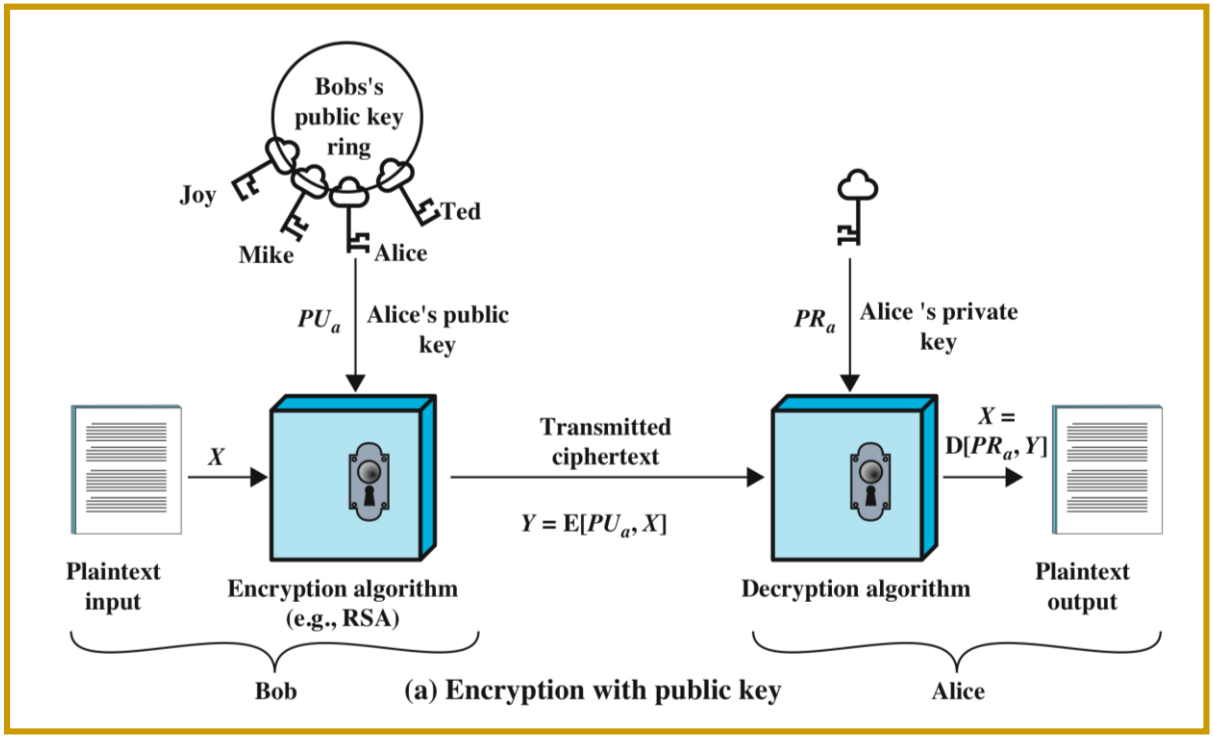
Publicly proposed by Diffie and Hellman in 1976

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



A public-key encryption scheme has six ingredients

- **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 algorithm**
  - Produces the original plaintext

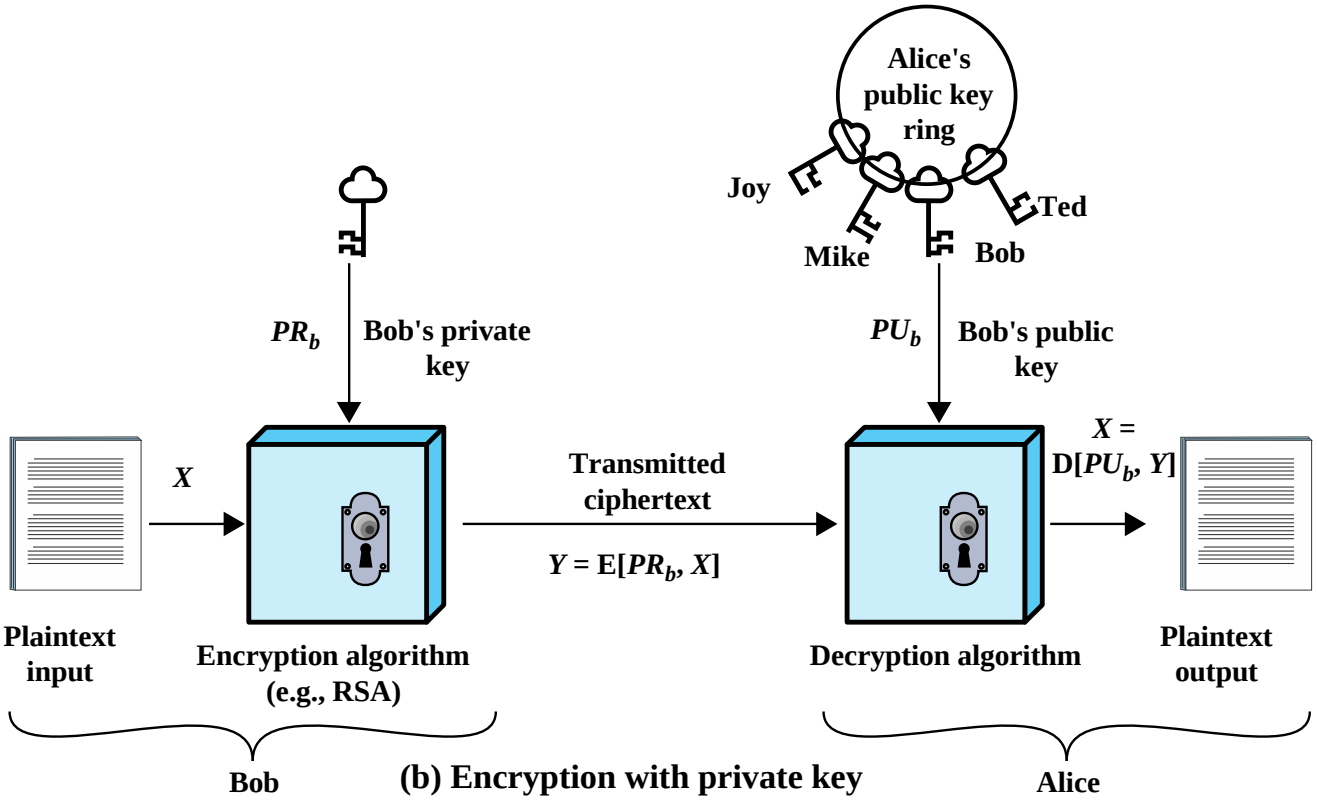


Figure 2.6 Public-Key Cryptography

- 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 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

Requirements for Public-Key Cryptosystems

Computationally easy
to create key pairs

Useful
if either key can
be used for each role

Computationally easy
for sender knowing
public key to encrypt
messages

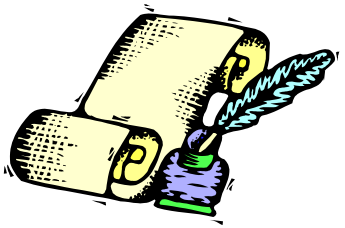
Computationally infeasible
for opponent to otherwise
recover original message

Computationally easy
for receiver knowing
private key to decrypt
ciphertext

Computationally infeasible
for opponent to determine
private key from public key

Digital Signatures

- Used for authenticating both **source** and **data integrity**
- Created by encrypting **hash code** with **private key**
- Does not provide confidentiality
  - Even in the case of complete encryption
  - Message is safe from alteration but not eavesdropping
  - because the rest of the message is transmitted in the clear
  - Even in the case of complete encryption, there is no protection of confidentiality because any observer can decrypt the message by using the sender's public key.



Digital Signatures

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

  "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)

Properties of Digital Signatures: Example

- Suppose Alice sends her bank a message authorizing it to transfer $100 to Bob.
- Alice's bank must be able to verify and prove that the message really came from Alice.
  - if she should later deny sending the message. (This property is called non-repudiation.)
- The bank also wants to know that the message is entirely Alice's, that it has not been altered along the way.
- For her part, Alice wants to be certain that her bank cannot forge such messages. (This property is called authenticity.)
- Both parties want to be sure that the message is new, not a reuse of a previous message, and that it has not been altered during transmission.
- Thus, a digital signature is a protocol that produces the same effect as a real signature:
  - It is a mark that only the sender can make but that other people can easily recognize as belonging to the sender.
  - Just like a real signature, a digital signature confirms agreement to a message.

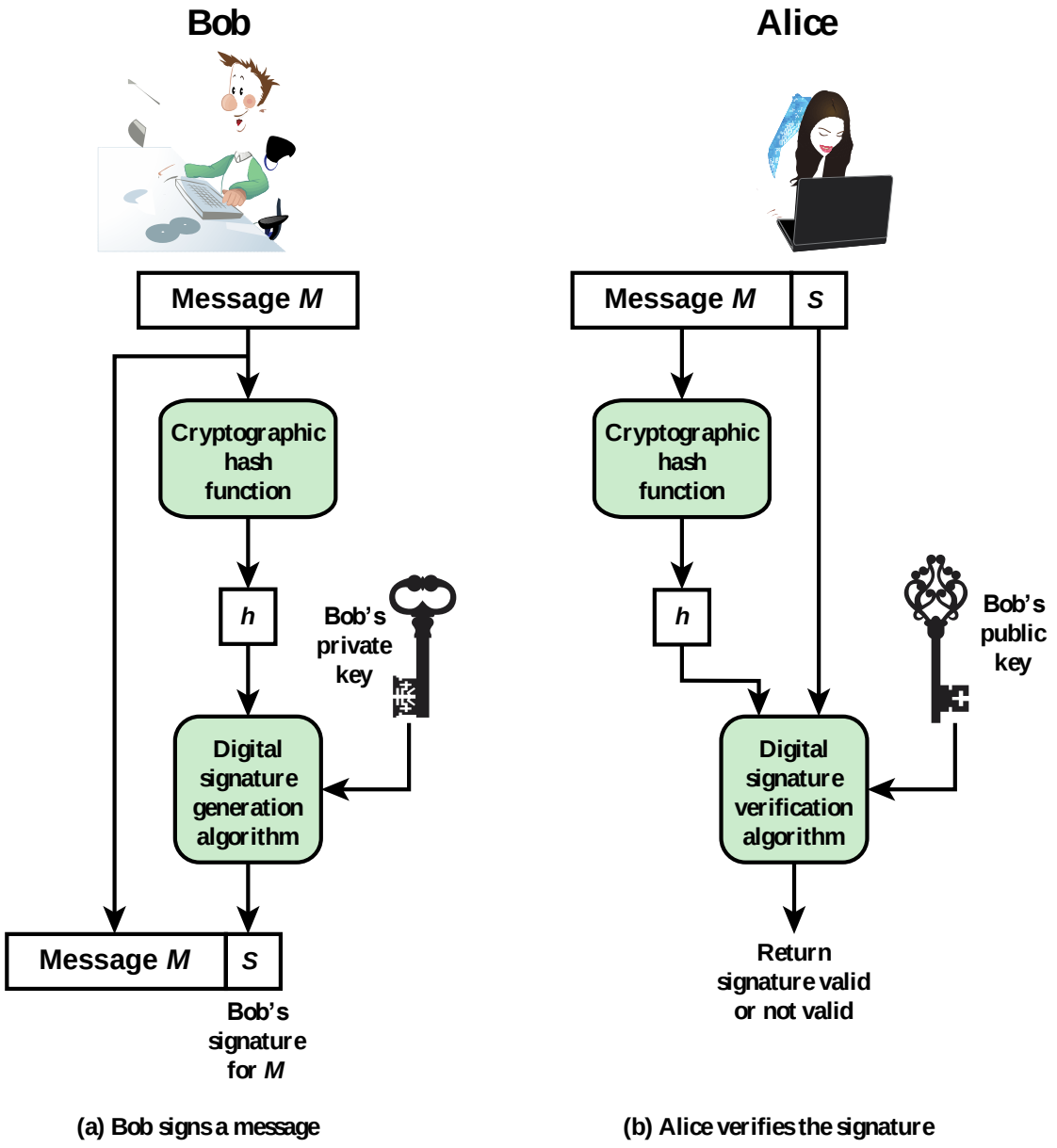


Figure 2.7 Simplified Depiction of Essential Elements of Digital Signature Process

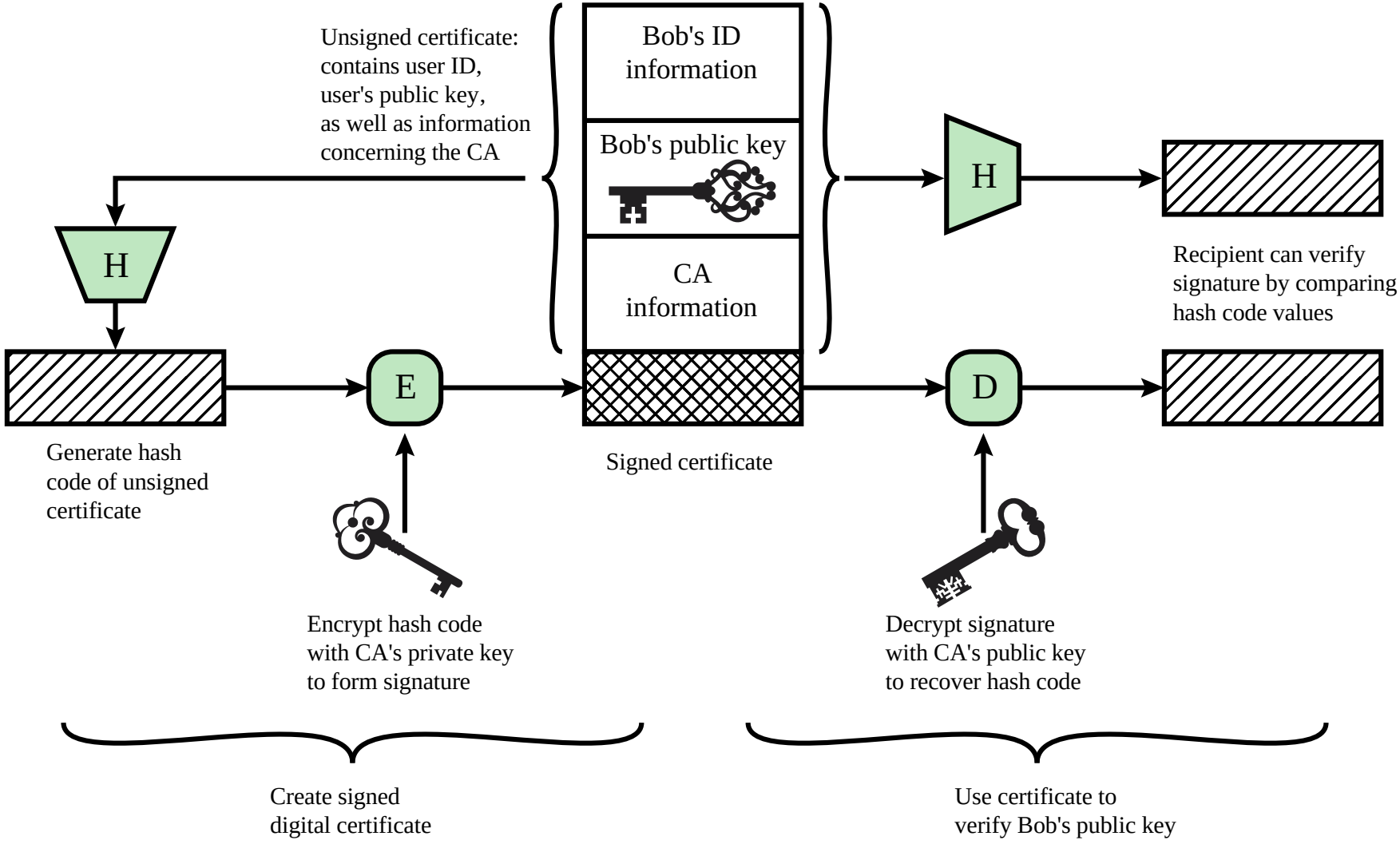
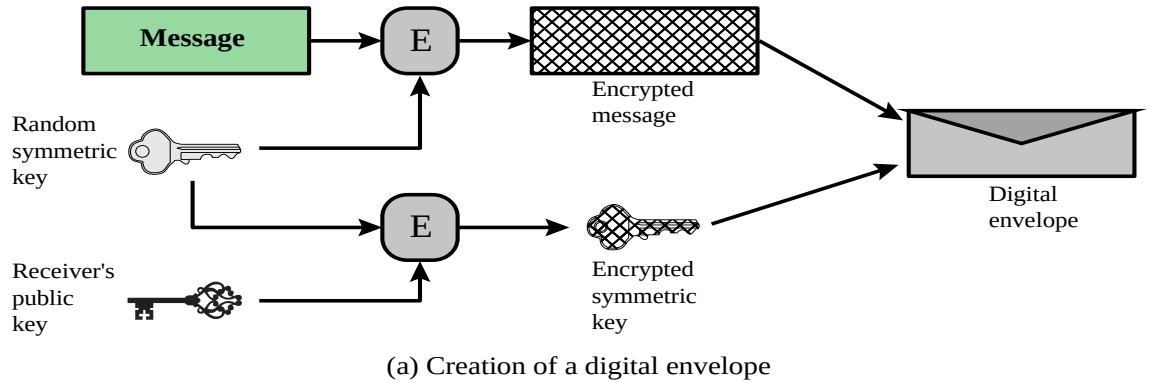
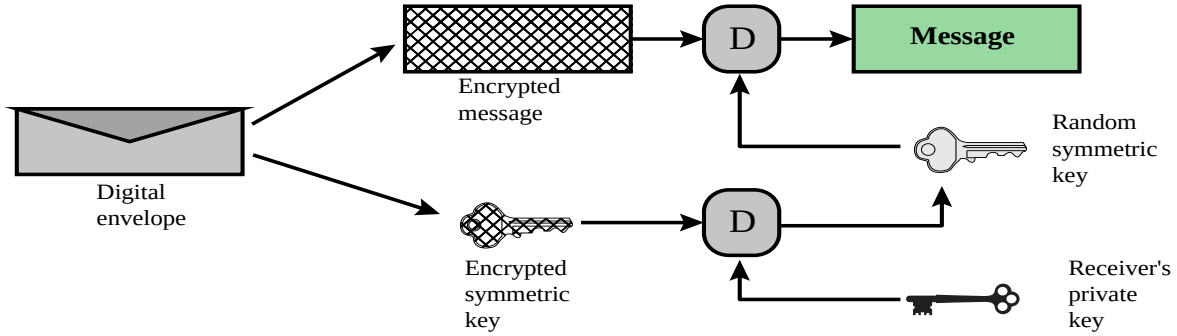


Figure 2.7 Public-Key Certificate Use

Digital Envelopes



(a) Creation of a digital envelope



(b) Opening a digital envelope

Figure 2.8 Digital Envelopes

- Protects a message without needing to first arrange for sender and receiver to have the same secret key
- Equates to the same thing as a sealed envelope containing an unsigned letter

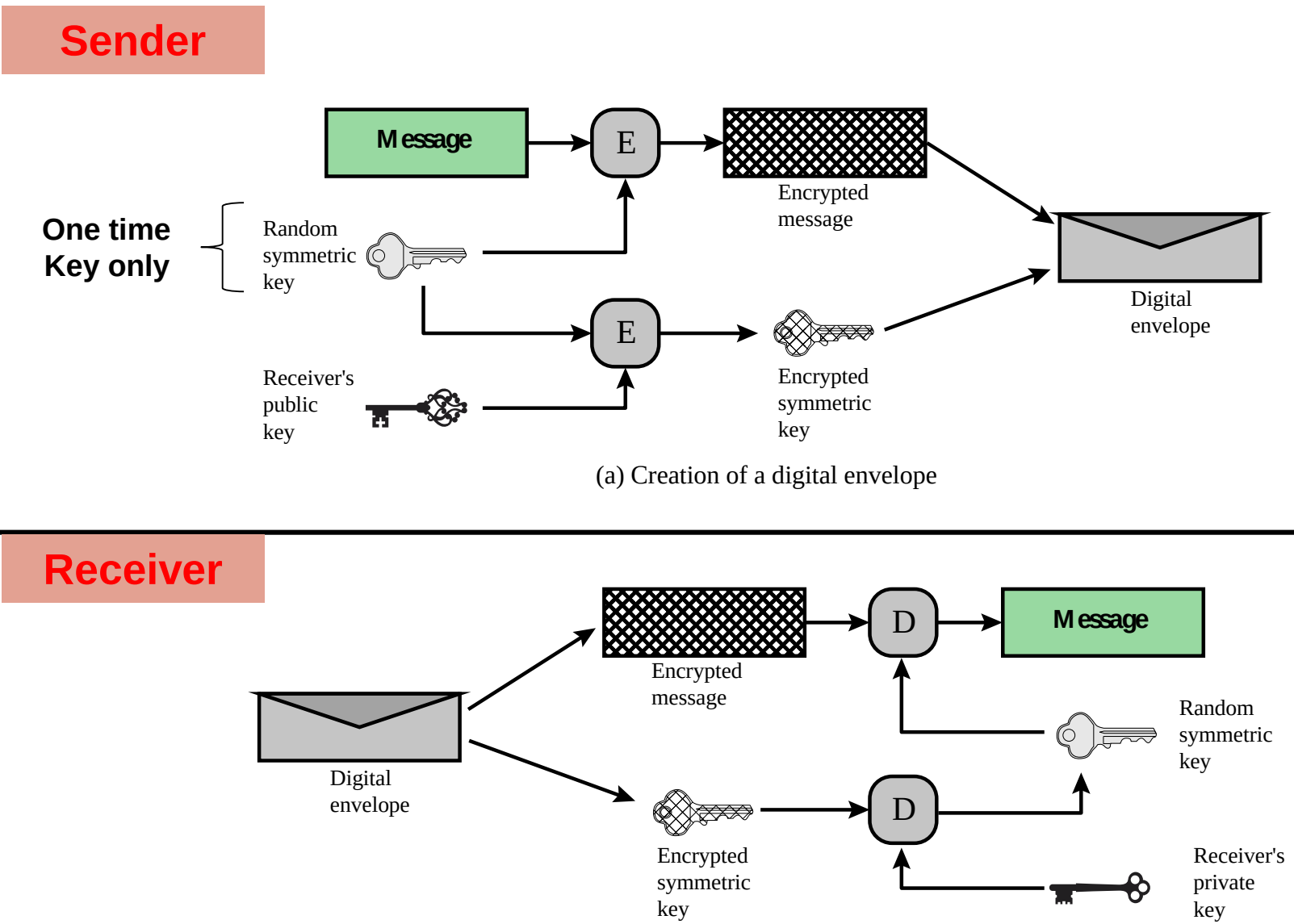


Figure 2.9 Digital Envelopes

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

Random Number Requirements

Randomness

- Criteria:
  - Uniform distribution
    - Frequency of occurrence of each of the numbers should be approximately the same
  - Independence
    - No one value in the sequence can be inferred from the others

Unpredictability

- Each number is statistically independent of other numbers in the sequence
- Opponent should not be able to predict future elements of the sequence on the basis of earlier elements

Random versus Pseudorandom

Cryptographic applications typically make use of algorithmic techniques for random number generation

- Algorithms are deterministic and therefore produce sequences of numbers that are not statistically random

Pseudorandom numbers are:

- Sequences produced that satisfy statistical randomness tests
- Likely to be predictable

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
- Increasingly provided on modern processors

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