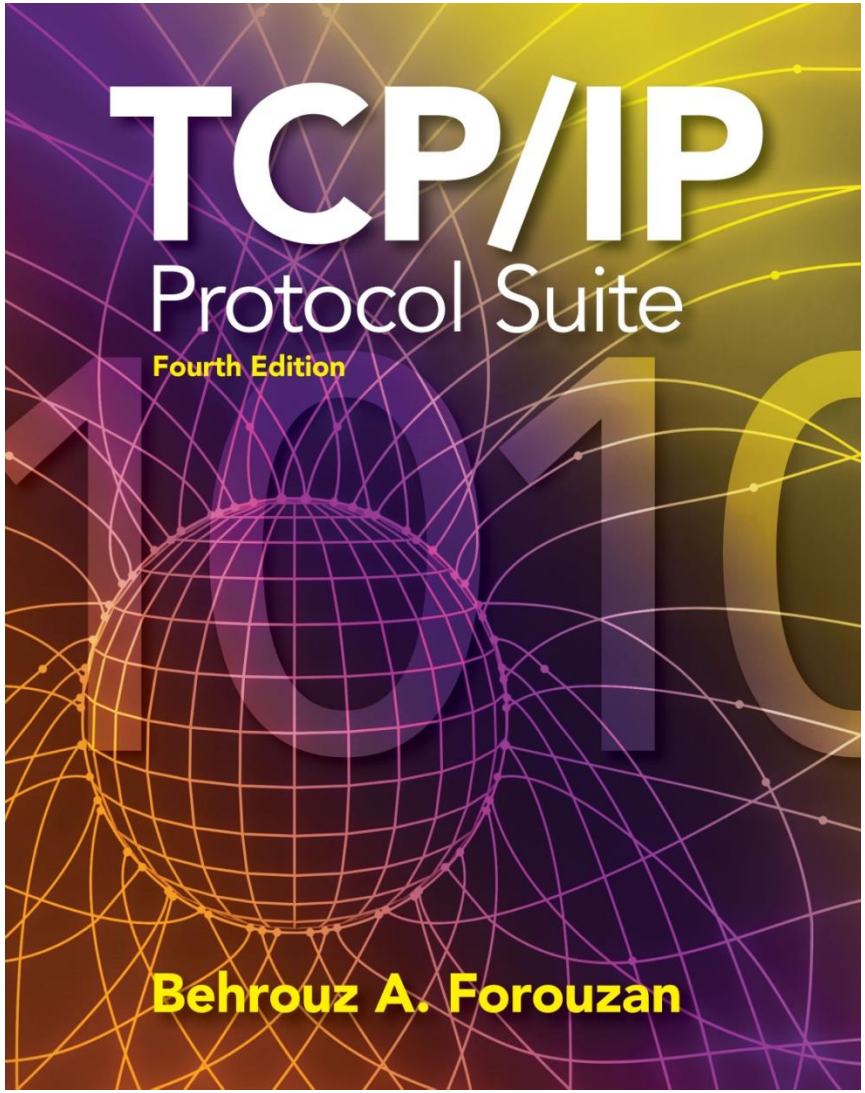
Chapter 29

Cryptography and Network Security



OBJECTIVES:

- To introduce security goals and to discuss the types of attacks that threaten these goals.
- To introduce traditional ciphers as symmetric-key ciphers to create the background for understanding modern symmetric-key ciphers.
- To introduce the elements of modern block ciphers and show an example of a modern block cipher in which these elements are used.
- To discuss the general idea behind asymmetric-key ciphers and introduce one common cipher in this category.
- To discuss message integrity and show how to use a cryptographic hash function to create a message digest.

OBJECTIVES (*continued*):

- To introduce the idea of message authentication and show how a message digest combined with a secret can authenticate the sender.
- To show how the idea of digital signatures can be used to authenticate a message using a pair of private-public keys.
- To introduce the idea of entity authentication and show some simple schemes using either a secret key or a pair of private-public keys.
- To show how secret keys in symmetric-key cryptography and how public keys in asymmetric-key cryptography can be distributed and managed using KDCs or certificate authorities (CAs).

Chapter Outline

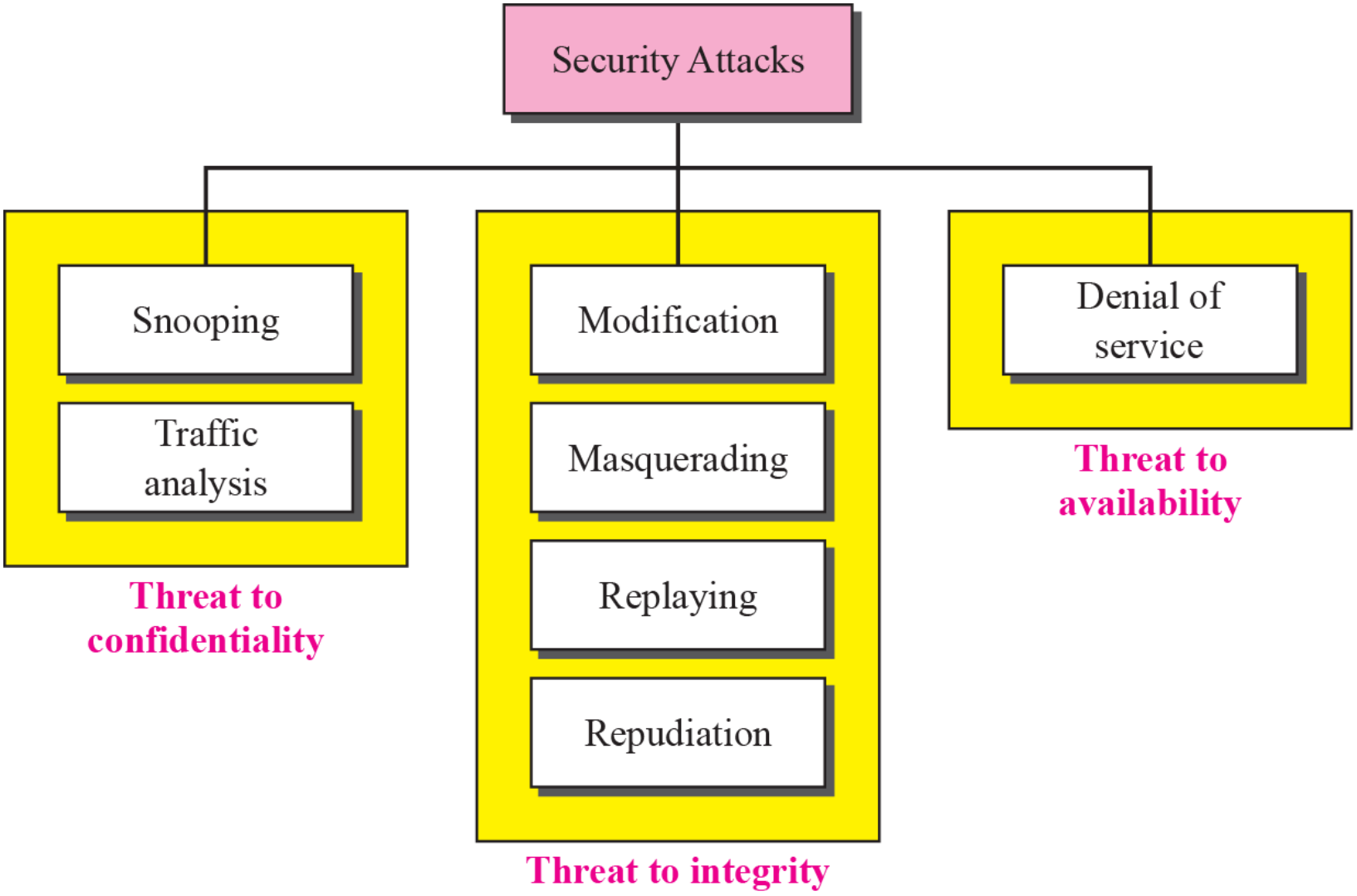
29-1 INTRODUCTION

We are living in the information age. We need to keep information about every aspect of our lives. In other words, information is an asset that has a value like any other asset. As an asset, information needs to be secured from attacks. To be secured, information needs to be hidden from unauthorized access (*confidentiality*), protected from unauthorized change (*integrity*), and available to an authorized entity when it is needed (*availability*).

Topics Discussed in the Section

- ✓ **Security Goals**
- ✓ **Attacks**
- ✓ **Services**
- ✓ **Techniques**

Figure 29.1 *Taxonomy of attacks with relation to security goals*



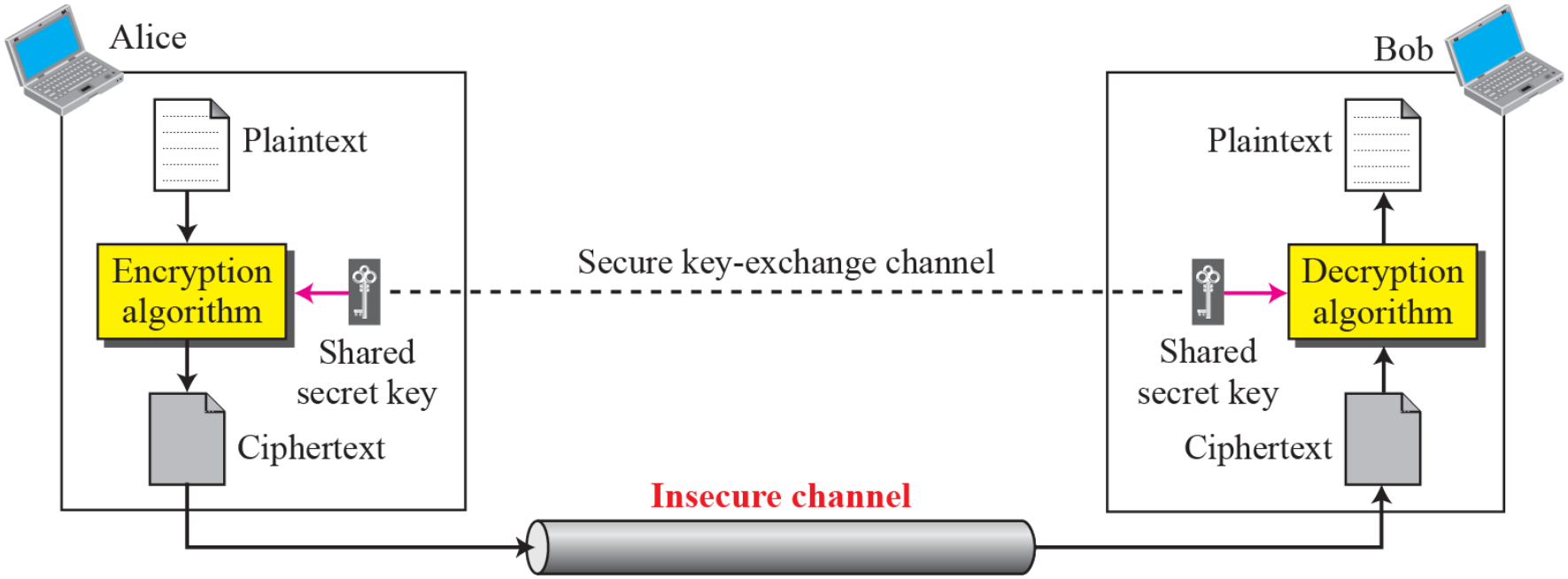
29-2 TRADITIONAL CIPHERS

We now look at the first goal of security, confidentiality. Confidentiality can be achieved using ciphers. Traditional ciphers are called symmetric-key ciphers (or secret-key ciphers) because the same key is used for encryption and decryption and the key can be used for bidirectional communication. Figure 29.2 shows the general idea behind a symmetric-key cipher.

Topics Discussed in the Section

- ✓ **Key**
- ✓ **Substitution Ciphers**
- ✓ **Transposition Ciphers**
- ✓ **Stream and Block Ciphers**

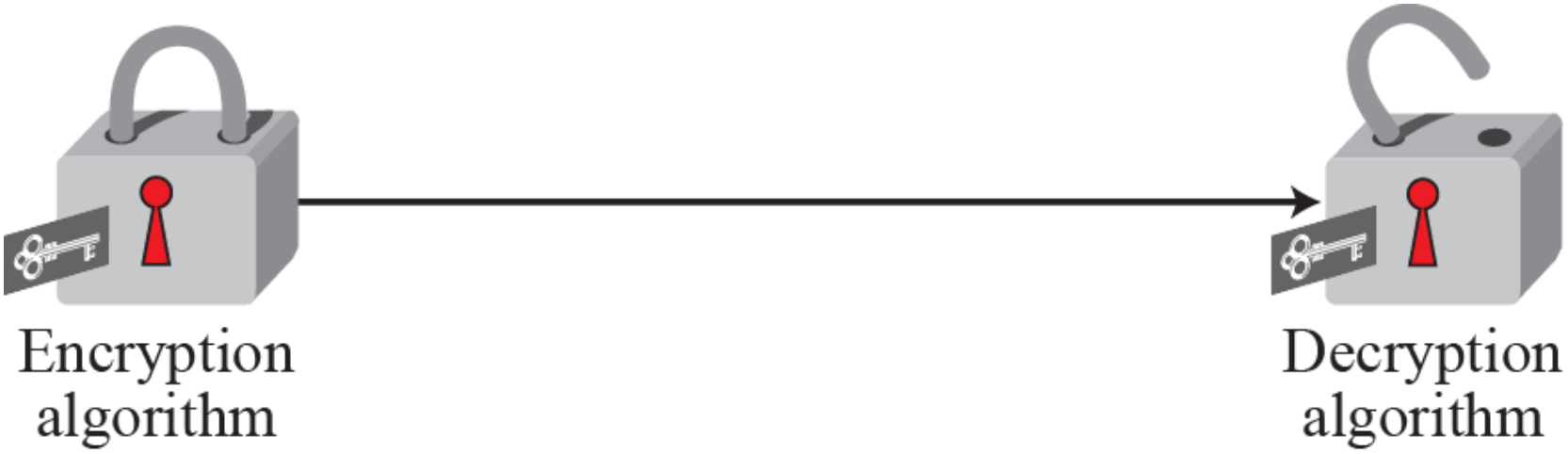
Figure 29.2 *General idea of traditional cipher*



Note

A substitution cipher replaces one symbol with another.

Figure 29.3 *Symmetric-key: locking and unlocking with the same key*



Note

A substitution cipher replaces one symbol with another.

Figure 29.4 *Representation of characters in modulo 26*

Plaintext →	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
Ciphertext →	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
Value →	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Note

In additive cipher, the plaintext, ciphertext, and key are integers in modulo 26.

Example 29.1

Use the additive cipher with key = 15 to encrypt the message "hello".

Solution

We apply the encryption algorithm to the plaintext, character by character. The result is "WTAAD". Note that the cipher is monoalphabetic because two instances of the same plaintext character (ls) are encrypted as the same character (A).

Plaintext: h → 07	Encryption: (07 + 15) mod 26	Ciphertext: 22 → W
Plaintext: e → 04	Encryption: (04 + 15) mod 26	Ciphertext: 19 → T
Plaintext: l → 11	Encryption: (11 + 15) mod 26	Ciphertext: 00 → A
Plaintext: l → 11	Encryption: (11 + 15) mod 26	Ciphertext: 00 → A
Plaintext: o → 14	Encryption: (14 + 15) mod 26	Ciphertext: 03 → D

Example 29.2

Use the additive cipher with key = 15 to decrypt the message "WTAAD".

Solution

We apply the decryption algorithm to the plaintext character by character. The result is "hello". Note that the operation is in modulo 26, which means that we need to add 26 to a negative result (for example –15 becomes 11).

Ciphertext: W → 22	Decryption: (22 – 15) mod 26	Plaintext: 07 → h
Ciphertext: T → 19	Decryption: (19 – 15) mod 26	Plaintext: 04 → e
Ciphertext: A → 00	Decryption: (00 – 15) mod 26	Plaintext: 11 → l
Ciphertext: A → 00	Decryption: (00 – 15) mod 26	Plaintext: 11 → l
Ciphertext: D → 03	Decryption: (03 – 15) mod 26	Plaintext: 14 → o

Figure 29.5 *An example key for mono-alphabetic substitution cipher*

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

Example 29.3

We can use the key in Figure 29.5 to encrypt the message

this message is easy to encrypt but hard to find the key

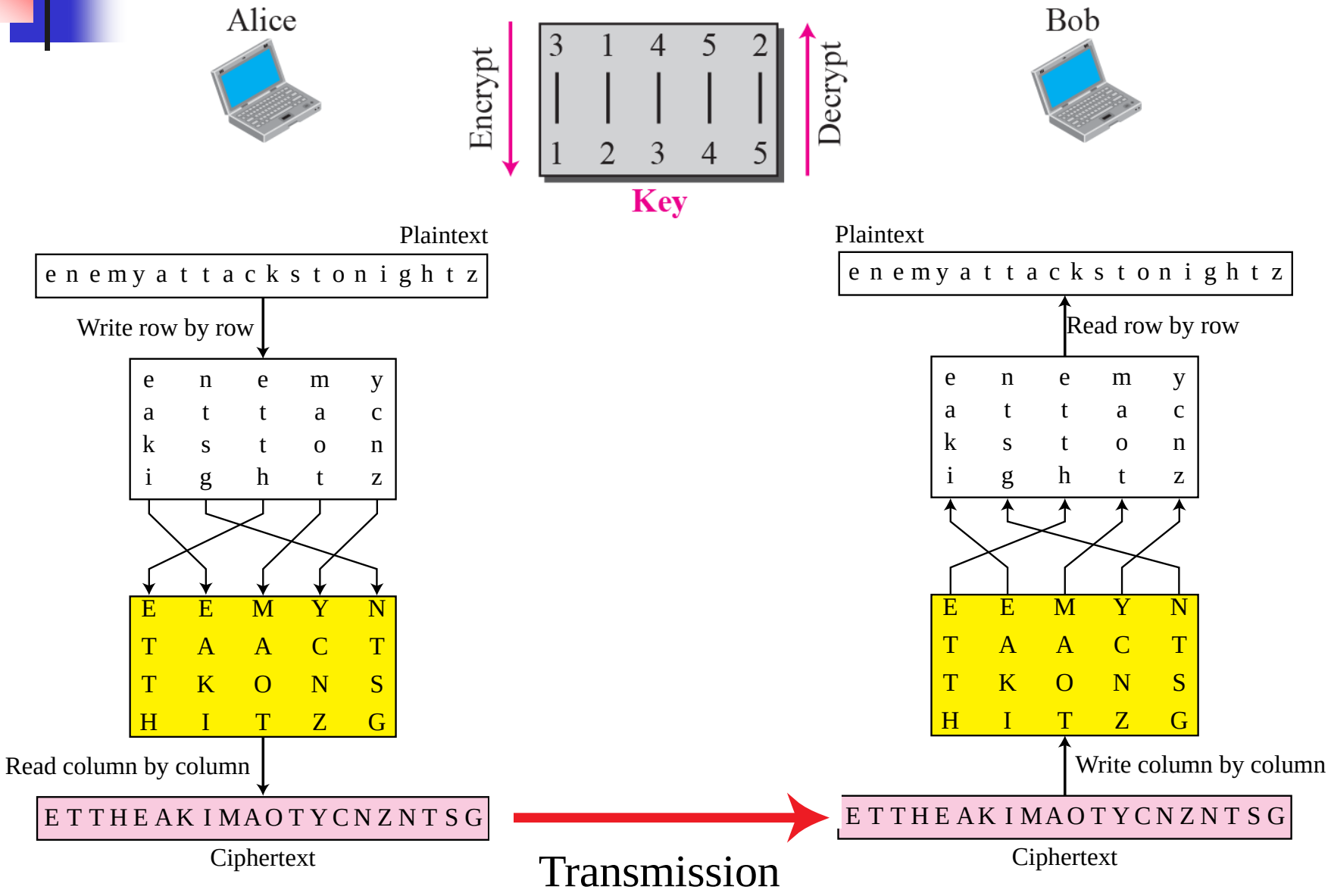
The ciphertext is

ICFVQRVVNEFVRNVSIYRGAHSLIOJICNHTIYBFGTICRXRS

Note

A transposition cipher reorders symbols.

Figure 29.6 *Transposition cipher*



29-3 MODERN CIPHERS

The traditional symmetric-key ciphers that we have studied so far are character-oriented ciphers. With the advent of the computer, we need bit-oriented ciphers. This is because the information to be encrypted is not just text; it can also consist of numbers, graphics, audio, and video data. It is convenient to convert these types of data into a stream of bits, to encrypt the stream, and then to send the encrypted stream. A modern block cipher can be either a block cipher or a stream cipher.

Topics Discussed in the Section

- ✓ **Modern Block Ciphers**
- ✓ **Data Encryption Standard (DES)**
- ✓ **Modern Stream Ciphers**

Figure 29.7 *A modern block cipher*

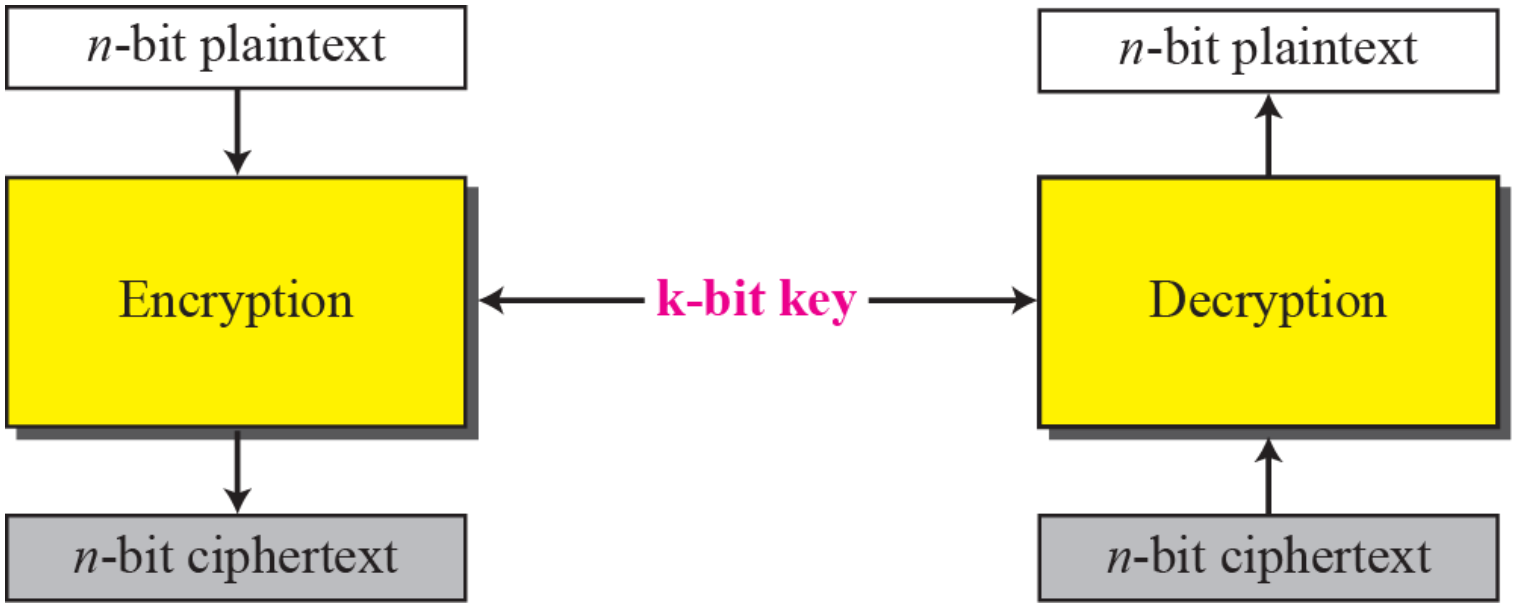


Figure 29.8 *Components of a modern block cipher*

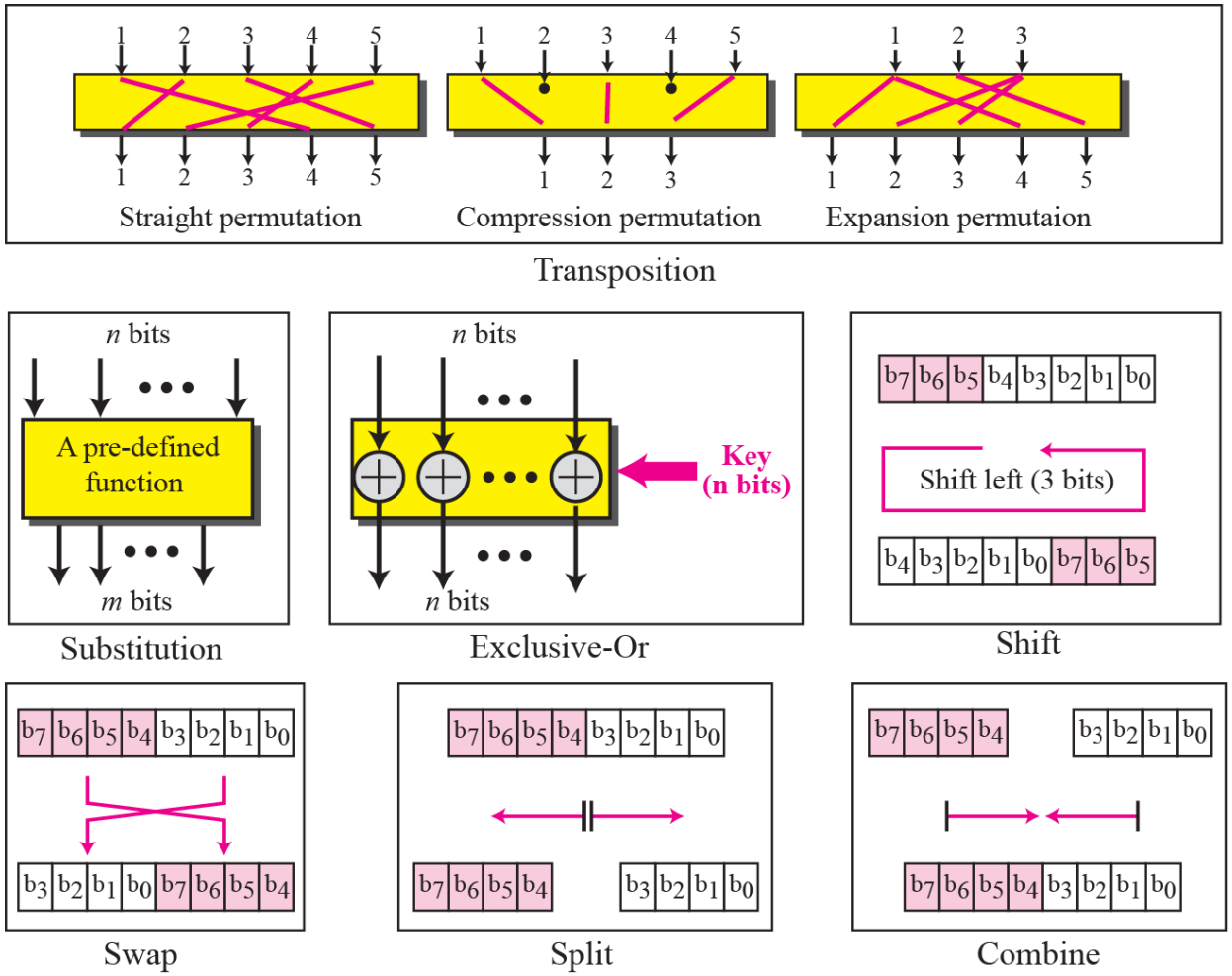


Figure 29.9 *General structure of DES*

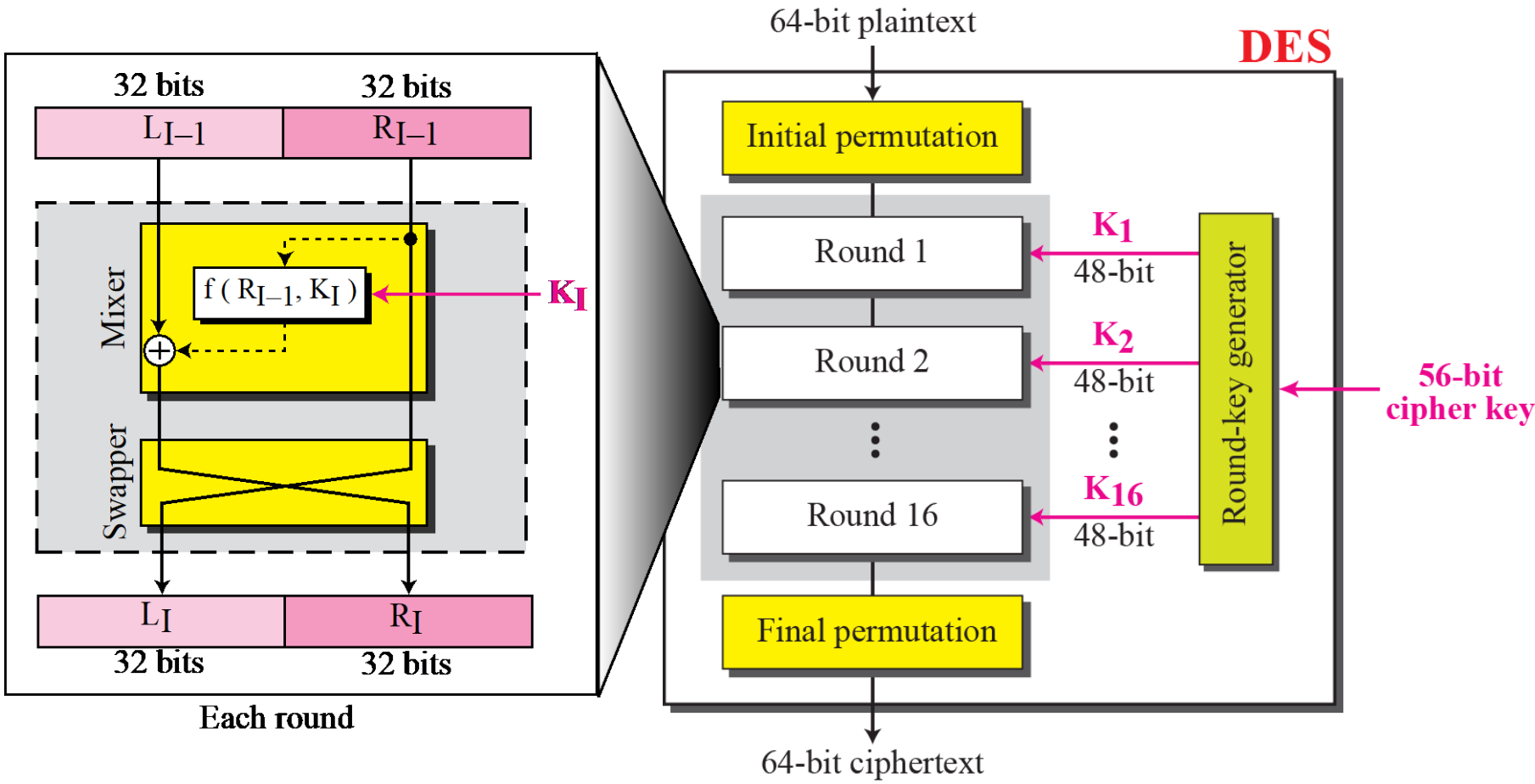


Figure 29.10 *DES function*

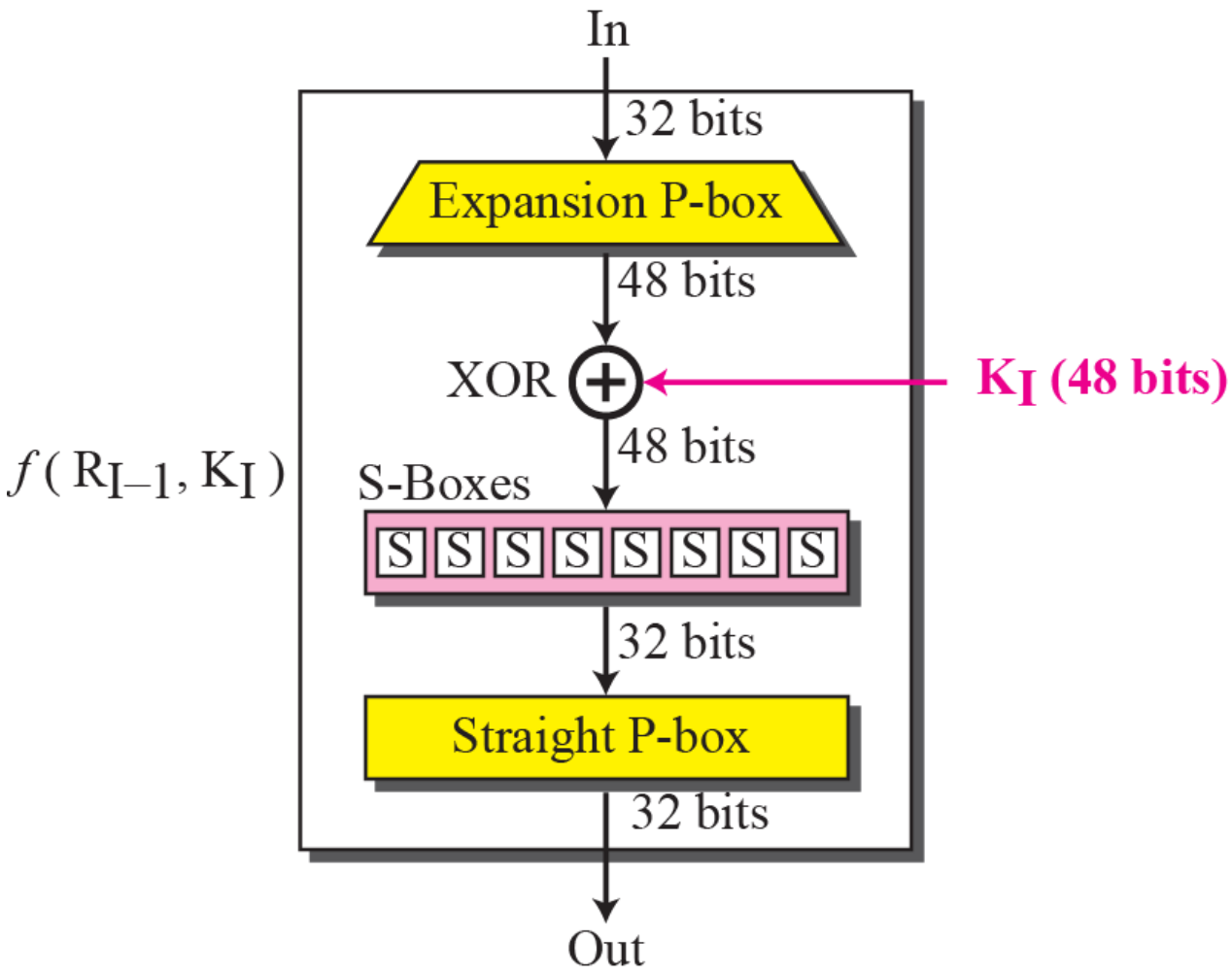
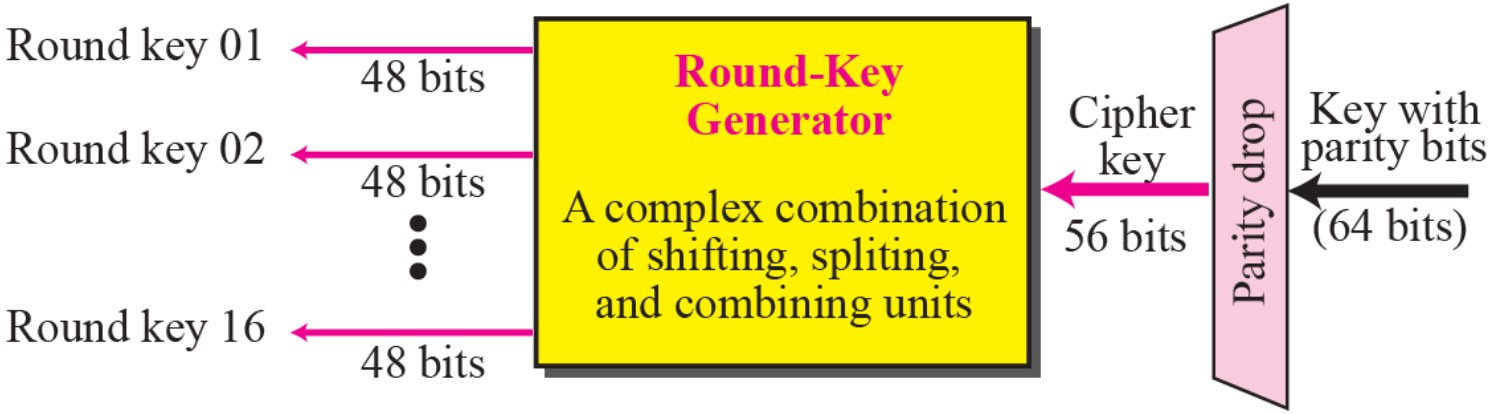


Figure 29.11 *Key generation*



Example 29.4

We choose a random plaintext block, a random key, and a computer program to determine what the ciphertext block would be (all in hexadecimal):

Plaintext:	Key:	CipherText:
123456ABCD132536	AABB09182736CCDD	C0B7A8D05F3A829C

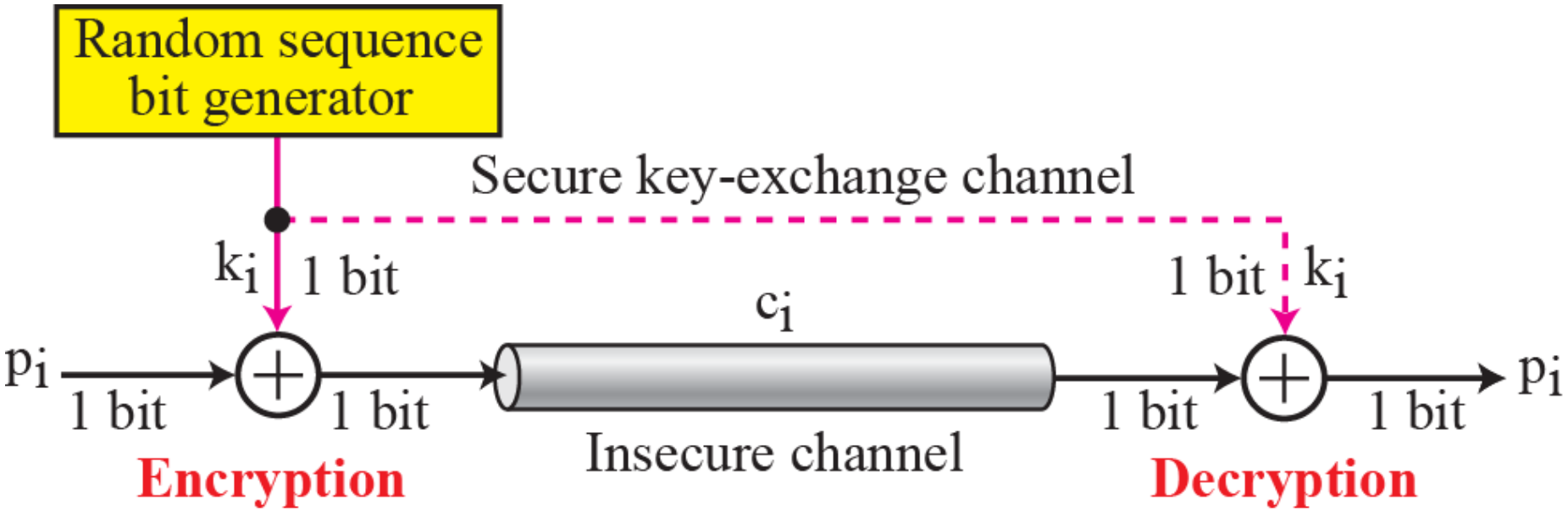
Example 29.5

To check the effectiveness of DES, when a single bit is changed in the input, let us use two different plaintexts with only one single bit difference. The two ciphertexts are completely different without even changing the key:

Plaintext:	Key:	Ciphertext:
0000000000000000	22234512987ABB23	4789FD476E82A5F1
Plaintext:	Key:	Ciphertext:
0000000000000001	22234512987ABB23	0A4ED5C15A63FEA3

Although the two plaintext blocks differ only in the rightmost bit, the ciphertext blocks differ in 29 bits.

Figure 29.12 *One-time pad*



29-4 ASYMMETRIC-KEY CIPHERS

In previous sections we discussed symmetric-key ciphers. In this chapter, we start the discussion of asymmetric-key ciphers. Symmetric-key and asymmetric-key ciphers will exist in parallel and continue to serve the community. We actually believe that they are complements of each other; the advantages of one can compensate for the disadvantages of the other.

Topics Discussed in the Section

- ✓ **Keys**
- ✓ **General Idea**
- ✓ **RSA Cryptosystem**
- ✓ **Applications**

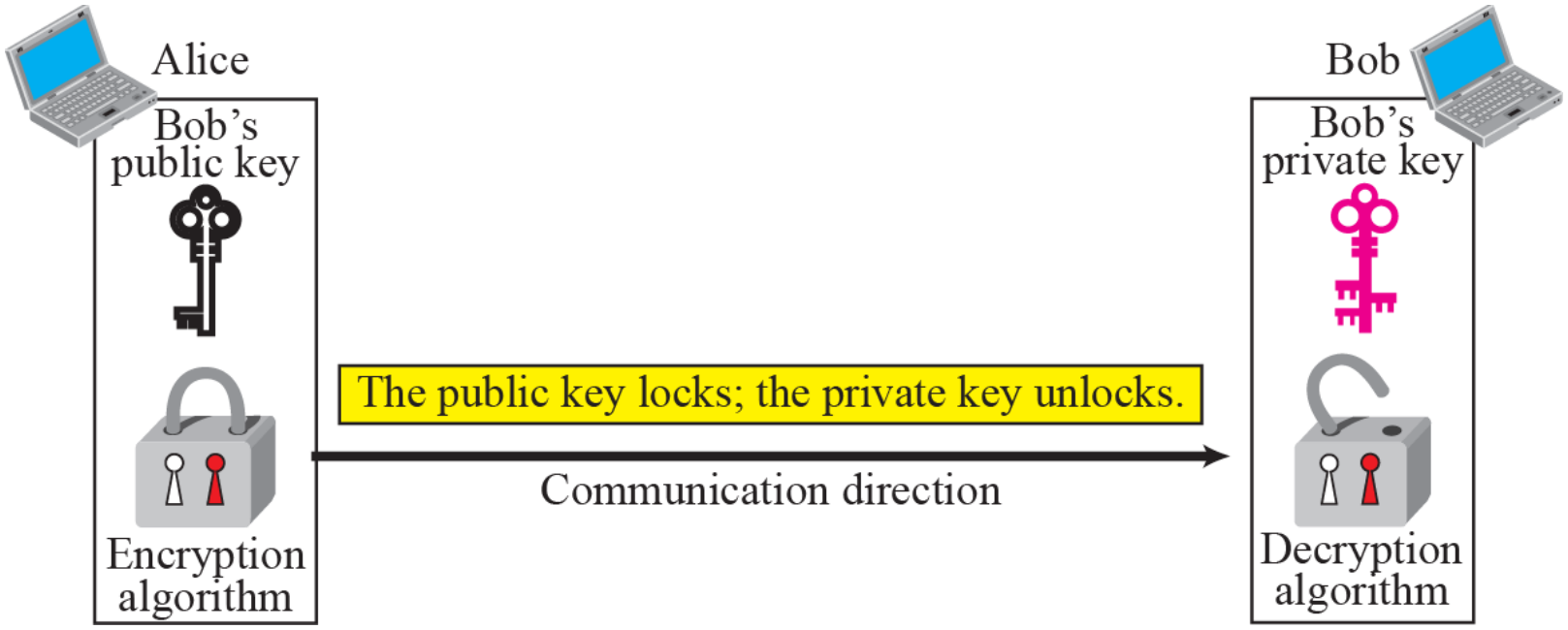
Note

Symmetric-key cryptography is based on sharing secrecy; asymmetric-key cryptography is based on personal secrecy.

Note

In symmetric-key cryptography, symbols are permuted or substituted; in asymmetric-key cryptography, numbers are manipulated.

Figure 29.13 *Locking and unlocking in asymmetric-key cryptosystem*



Note

Asymmetric-key ciphers are sometimes called public-key ciphers.

Figure 29.14 *General idea of asymmetric-key cryptosystem*

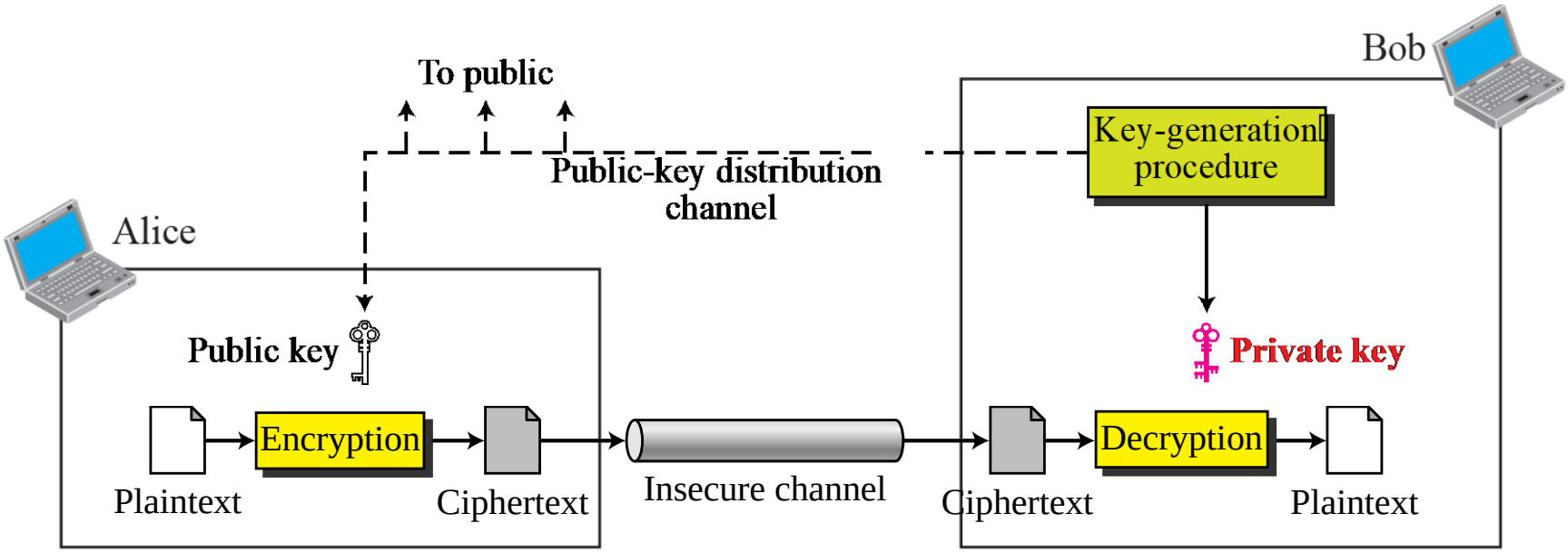
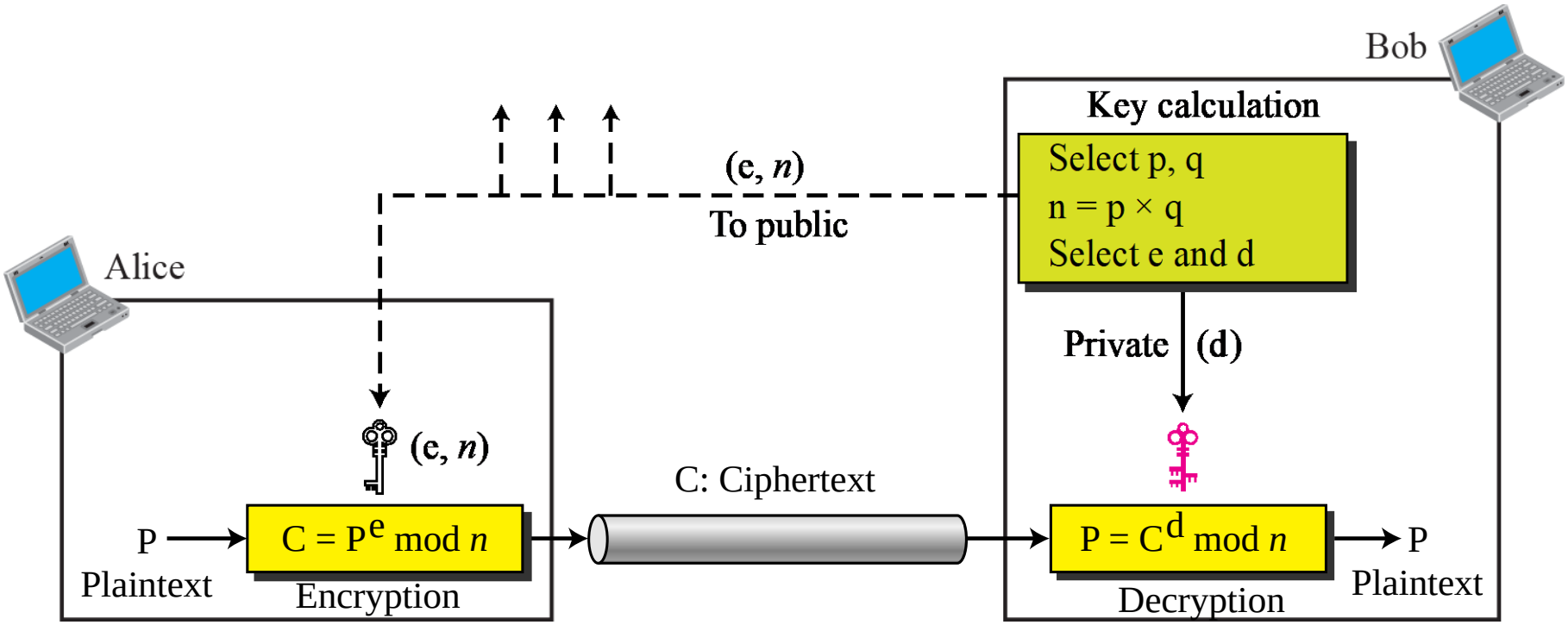


Figure 29.15 *Encryption, decryption, and key Generation in RSA*



Example 29.6

For the sake of demonstration, let Bob choose 7 and 11 as *p* and *q* and calculate $n = 7 \times 11 = 77$. The value of $\phi(n) = (7 - 1)(11 - 1)$, or 60. If he *chooses e* to be 13, then *d* is 37. Note that $e \times d \bmod 60 = 1$. Now imagine that Alice wants to send the plaintext 5 to Bob. She uses the public exponent 13 to encrypt 5. This system is not safe because p and q are small.

Plaintext: 5

$C = 5^{13} = 26 \bmod 77$

Ciphertext: 26

Ciphertext: 26

$P = 26^{37} = 5 \bmod 77$

Plaintext: 5

Example 29.7

Here is a more realistic example calculated with a computer. We choose a 512-bit p and q, calculate n and $\phi(n)$, We then choose e and calculate d. Finally, we show the results of encryption and decryption. The integer p is a 159-digit number.

$p =$ 96130345313583504574191581280615427909309845594996215822583150879647940455056470638491257160180347503120986666064924201918087806674210960633542199266612 09

The integer q is a 160-digit number.

$q =$ 12060191957231446918276794204450896001555925054637033936061798321731482148483764659215389453209175225273226830107120695604602513887145524969000359660045617

Example 29.7 *Continued*

The modulus $n = p \times q$. It has 309 digits.

$n =$ 115935041739676149688925098646158875237714573754541447754855261376
147885408326350817276878815968325168468849300625485764111250162414
552339182927162507656772727460097082714127730434960500556347274566
628060099924037102991424472292215772798531727033839381334692684137
327622000966676671831831088373420823444370953

$\phi(n) = (p - 1)(q - 1)$ has 309 digits.

$\phi(n) =$ 115935041739676149688925098646158875237714573754541447754855261376
147885408326350817276878815968325168468849300625485764111250162414
552339182927162507656751054233608492916752034482627988117554787657
013923444405716989581728196098226361075467211864612171359107358640
614008885170265377277264467341066243857664128

Example 29.7 *Continued*

Bob chooses $e = 35535$ (the ideal is 65537). He then finds d.

$e =$	35535
$d =$	58008302860037763936093661289677917594669062089650962180422866111380593852822358731706286910030021710859044338402170729869087600611530620252495988444804756824096624708148581713046324064407770483313401085094738529564507193677406119732655742423721761767462077637164207600337085333288532144708859551366702948 31

Alice wants to send the message "THIS IS A TEST", which can be changed to a numeric value using the 00–26 encoding scheme (26 is the space character).

P =	1907081826081826002619041819

Example 29.7 *Continued*

The ciphertext calculated by Alice is $C = P^e$, which is

$$C = 4753091236462268272063655506105451809423717960704917165232392430544529606131993285666178434183591141511974112520056829797945717360361012782188478927415660904800235071907152771859149751884658886321011483541033616578984679683867637337657774656250792805211481418440441844308127730590046928742485591664621086561418443081277305900469287424855916646210865614184430812773059004692874248559166462108656$$

Bob can recover the plaintext from the ciphertext using $P = C^d$, which is

$$P = 190708182608182600261904181919070818260818260026190418191907081826081826002619041819$$

The recovered plaintext is "THIS IS A TEST" after decoding.

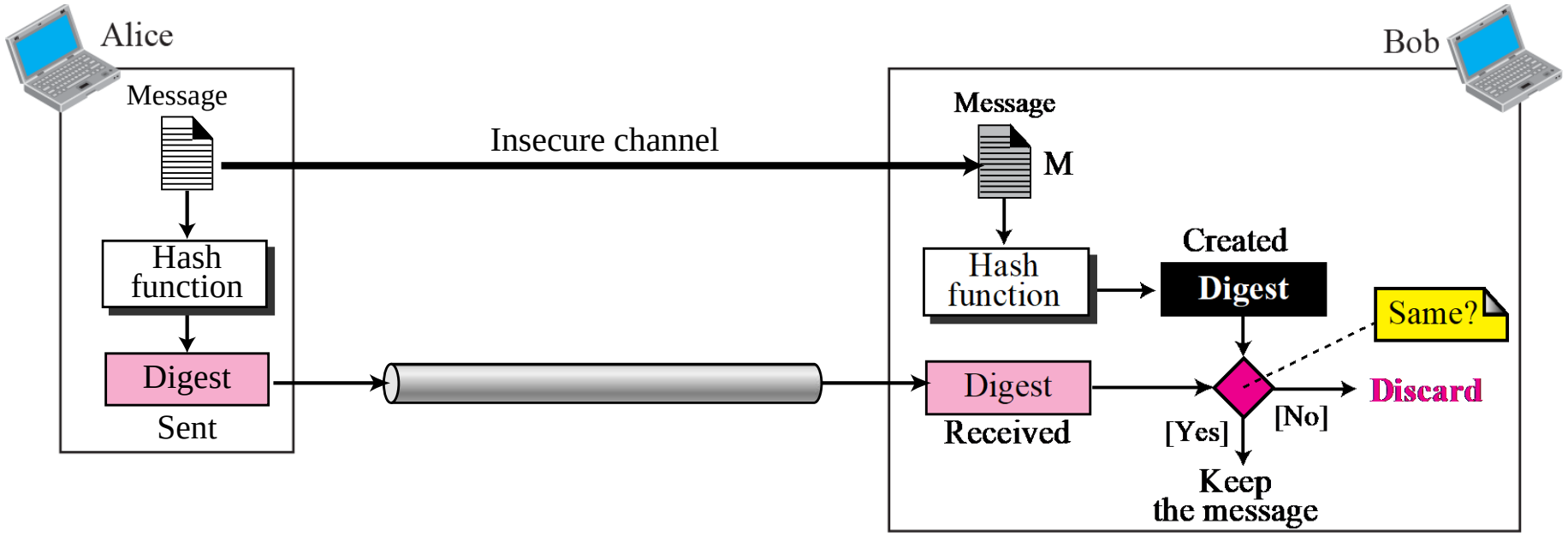
29-5 MESSAGE INTEGRITY

The cryptography systems that we have studied so far provide secrecy, or confidentiality, but not integrity. However, there are occasions where we may not even need secrecy but instead must have integrity. For example, Alice may write a will to distribute her estate upon her death. The will does not need to be encrypted. After her death, anyone can examine the will. The integrity of the will, however, needs to be preserved. Alice does not want the contents of the will to be changed.

Topics Discussed in the Section

- ✓ **Message and Message Digest**
- ✓ **Hash Functions**

Figure 29.16 *Message and digest*



Note

The message digest needs to be safe from change.

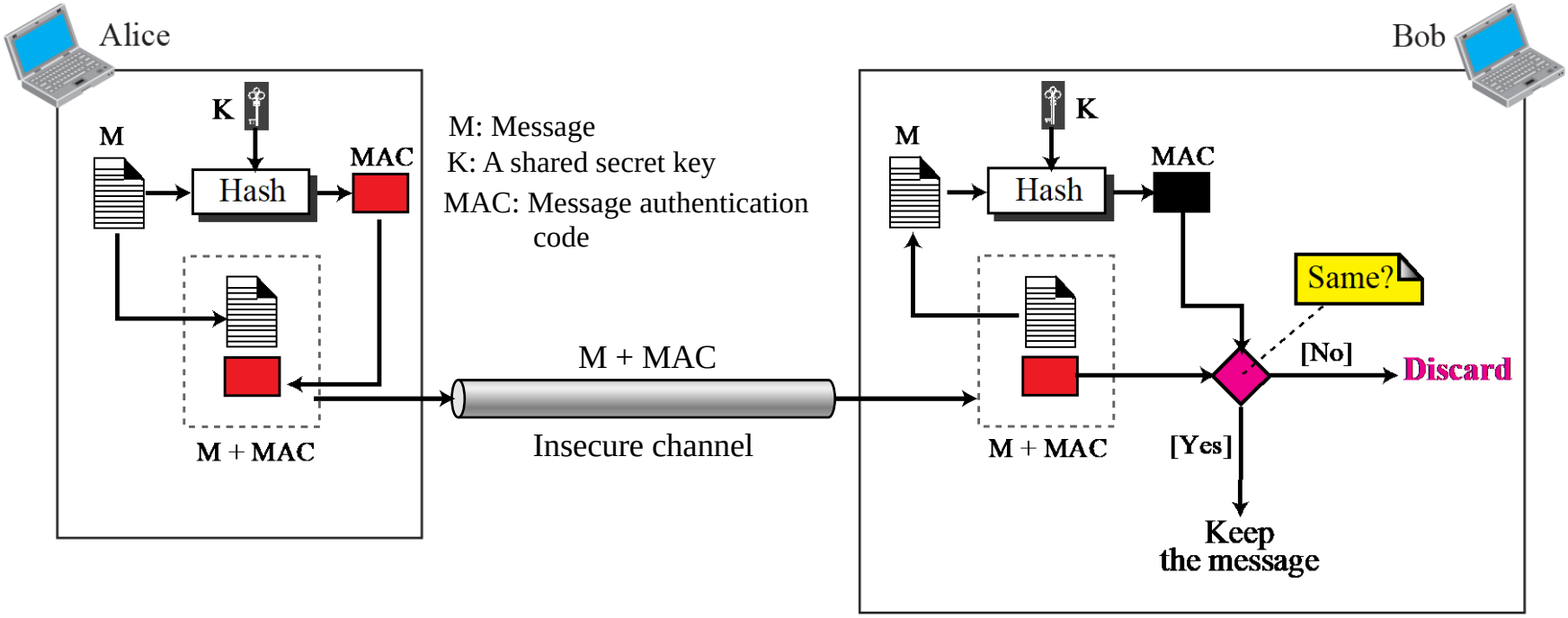
29-6 MESSAGE AUTHENTICATION

A digest can be used to check the integrity of a message: that the message has not been changed. To ensure the integrity of the message and the data origin authentication— that Alice is the originator of the message, not somebody else—we need to include a secret held by Alice (that Eve does not possess) in the process; we need to create a message authentication code (MAC). Figure 29.17 shows the idea.

Topics Discussed in the Section

- ✓ **MAC**
- ✓ **HMAC**

Figure 29.17 *Message authentication code*



Note

A MAC provides message integrity and message authentication using a combination of a hash function and a secret key.

29-7 DIGITAL SIGNATURE

Another way to provide message integrity and message authentication (and some more security services as we see shortly) is a digital signature. A MAC uses a secret key to protect the digest; a digital signature uses a pair of private–public keys.

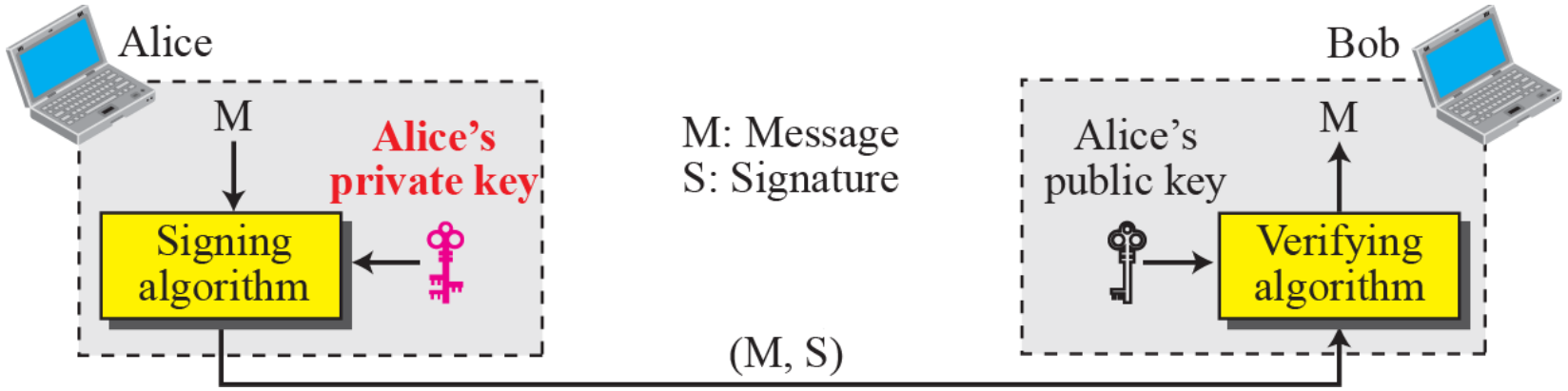
Topics Discussed in the Section

- ✓ **Comparison**
- ✓ **Process**
- ✓ **Signing the Digest**
- ✓ **Services**
- ✓ **RSA Digital Signature Scheme**
- ✓ **Digital Signature Standard (DSS)**

Note

A digital signature uses a pair of private-public keys.

Figure 29.18 *Digital signature process*



Note

A digital signature needs a public-key system.
The signer signs with her private key; the verifier verifies with the signer's public key.

Note

A cryptosystem uses the private and public keys of the receiver: a digital signature uses the private and public keys of the sender.

Figure 29.19 *Signing the digest*

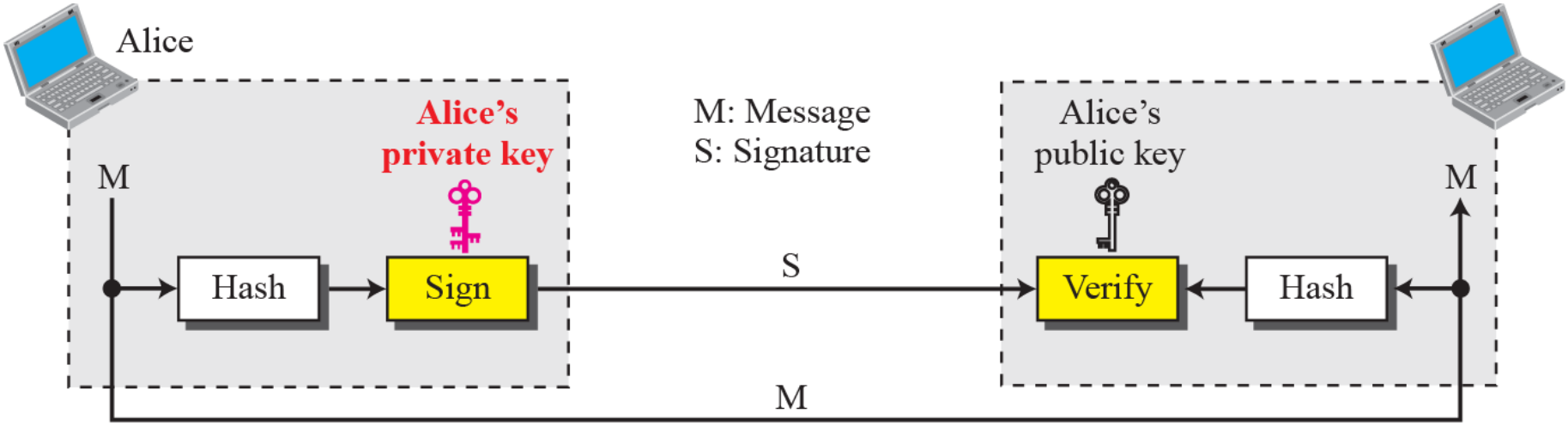


Figure 29.20 *Using a trusted center for non-repudiation*

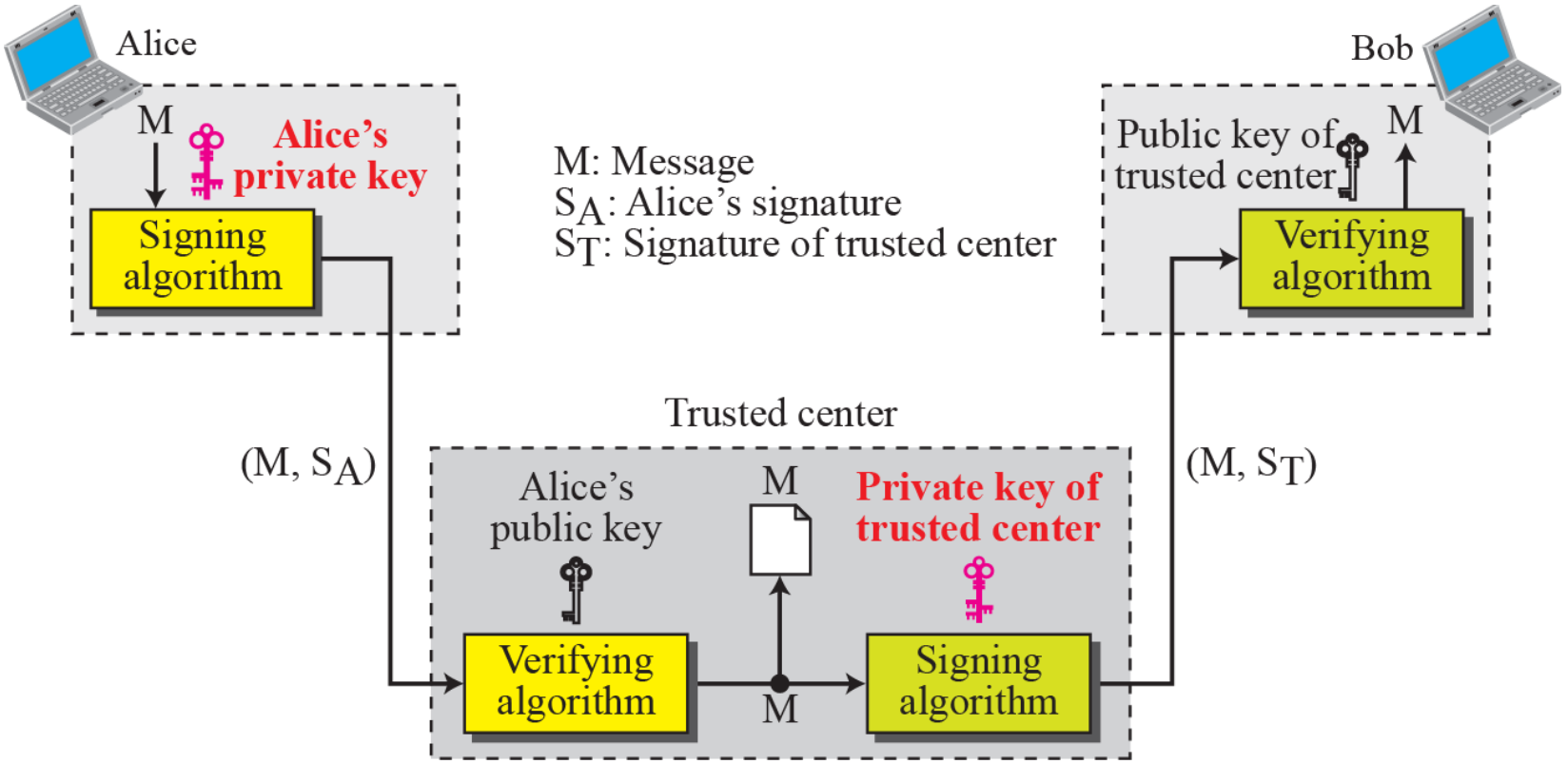
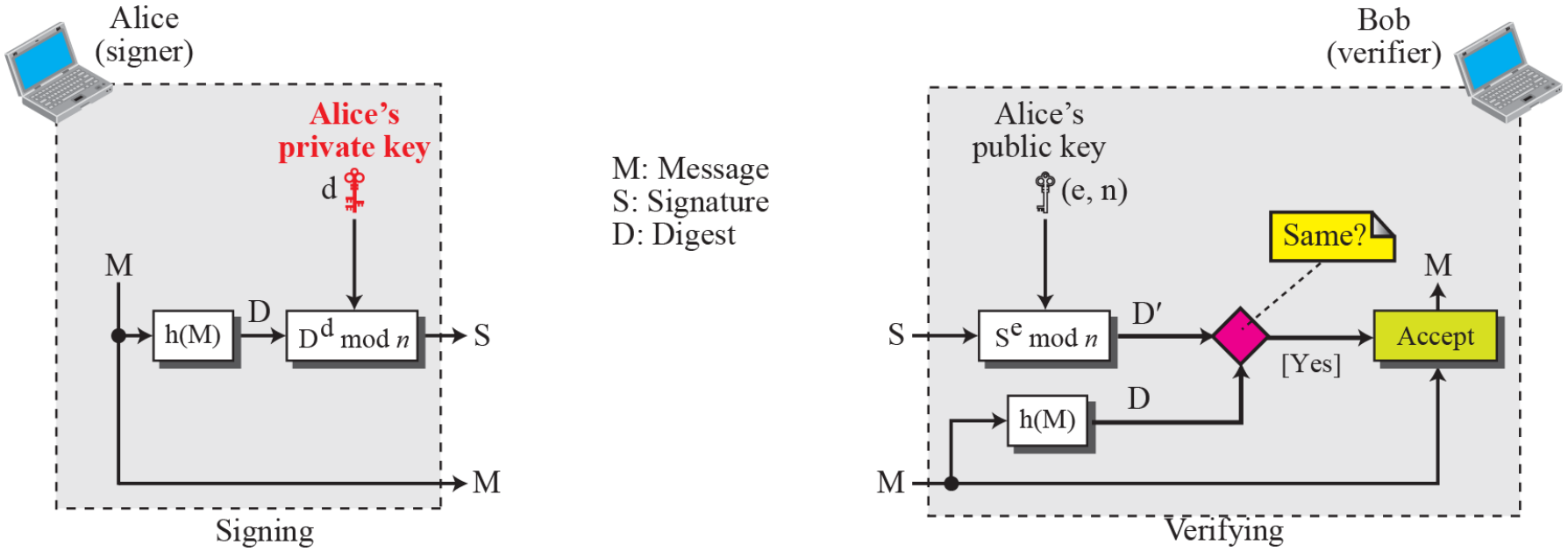


Figure 29.21 *The RSA signature one the message digest*



29-8 ENTITY AUTHENTICATION

Entity authentication is a technique designed to let one party prove the identity of another party. An entity can be a person, a process, a client, or a server. The entity whose identity needs to be proven is called the claimant; the party that tries to prove the identity of the claimant is called the verifier.

Topics Discussed in the Section

- ✓ **Entity versus Message Authentication**
- ✓ **Verification Categories**
- ✓ **Passwords**
- ✓ **Challenge-Response**

Note

In challenge-response authentication, the claimant proves that she knows a secret without sending it to the verifier.

Figure 29.22 *Unidirectional symmetric-key authentication*

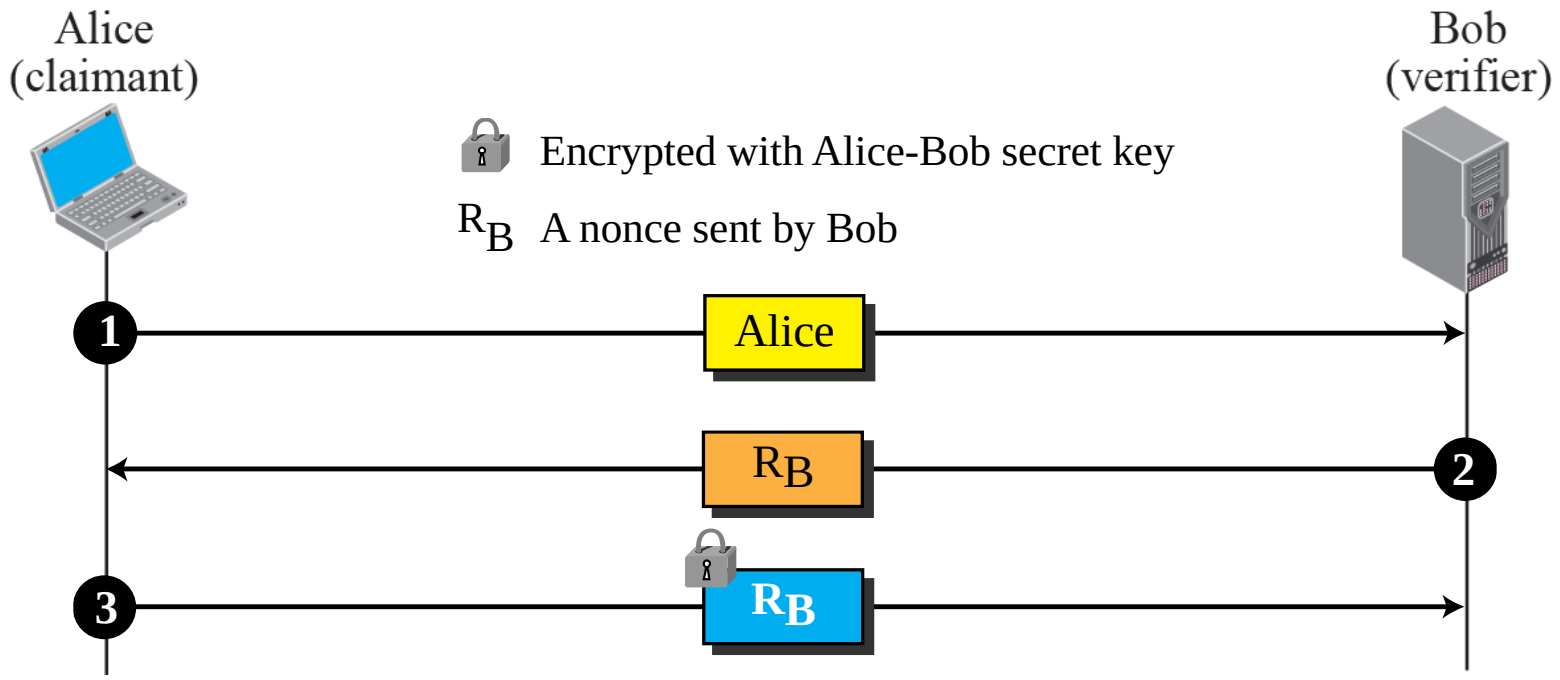


Figure 29.23 *Unidirectional asymmetric-key authentication*

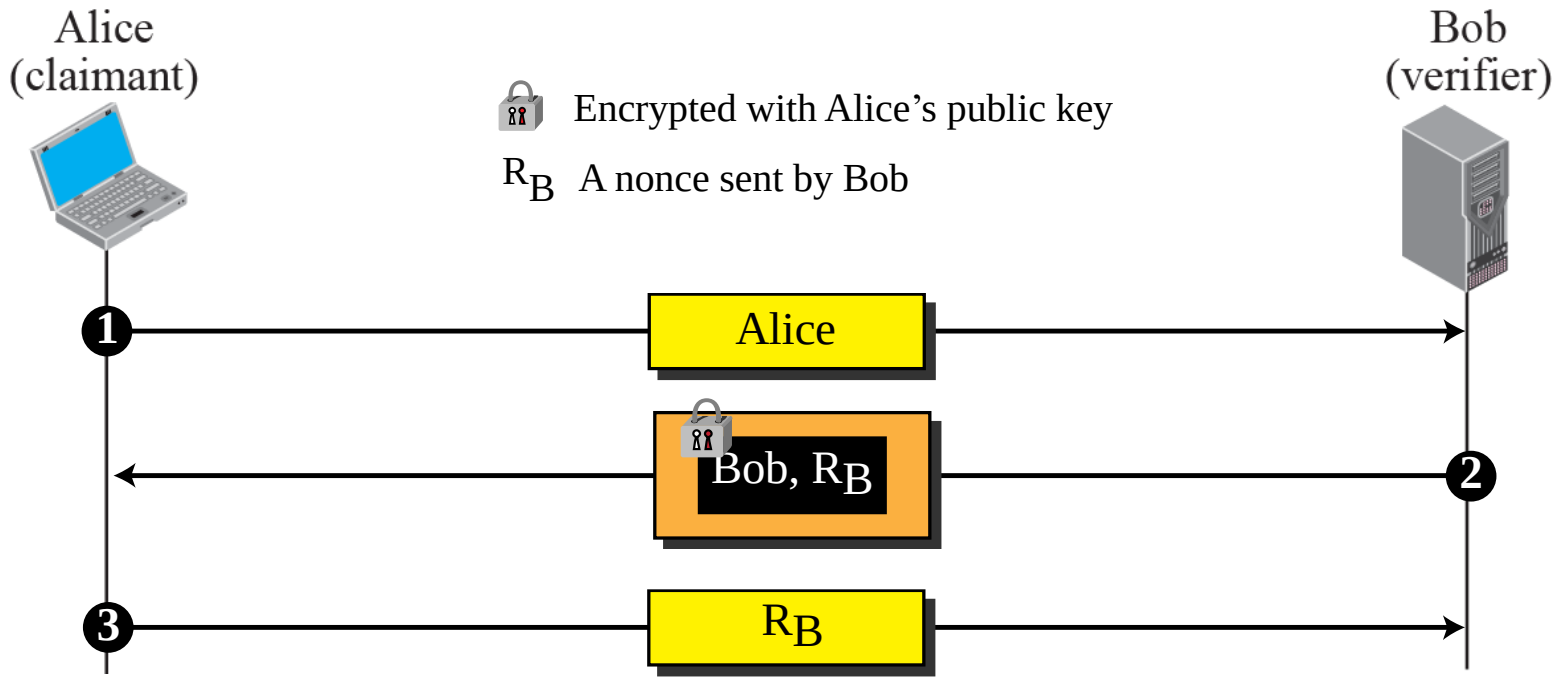
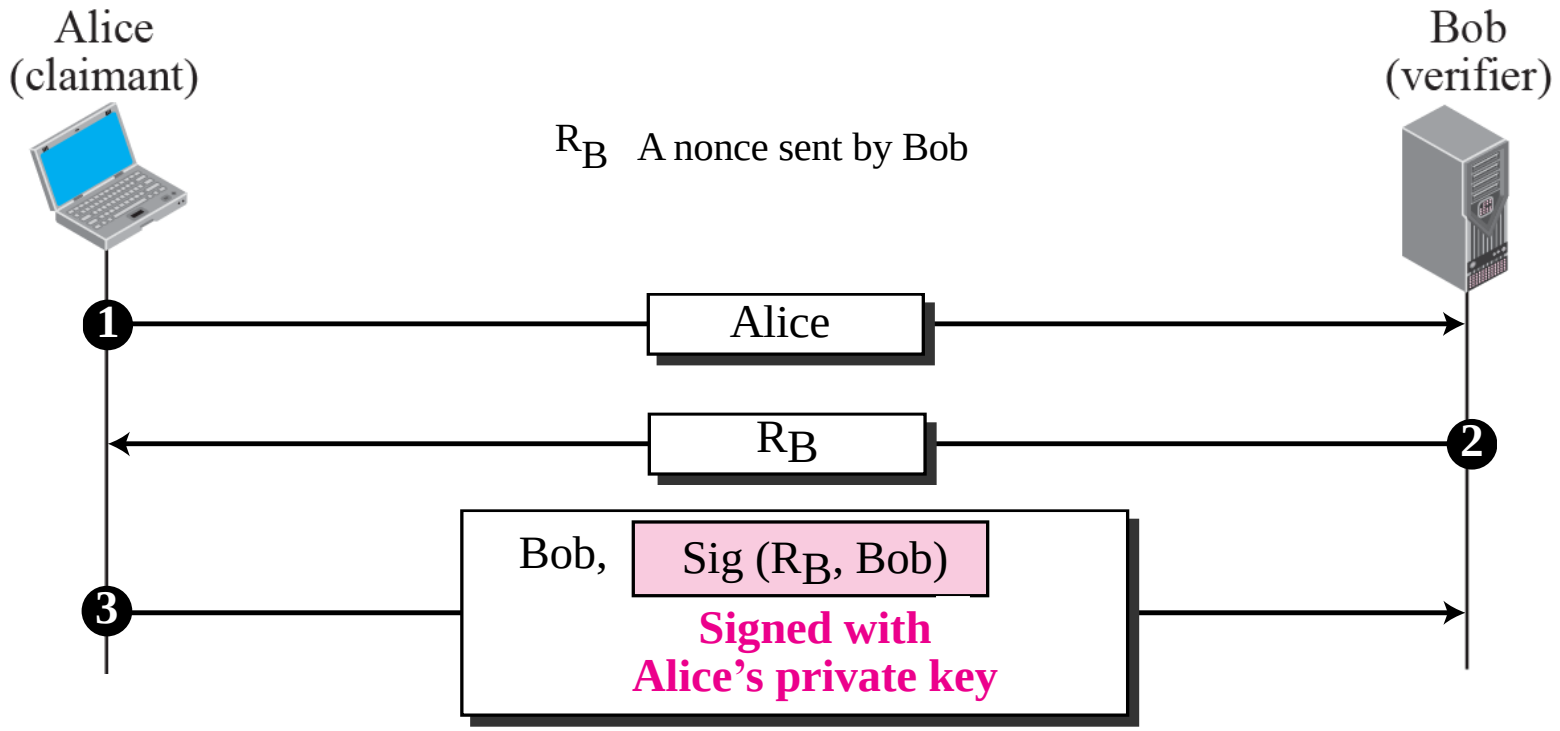


Figure 29.24 *Digital signature, unidirectional authentication*



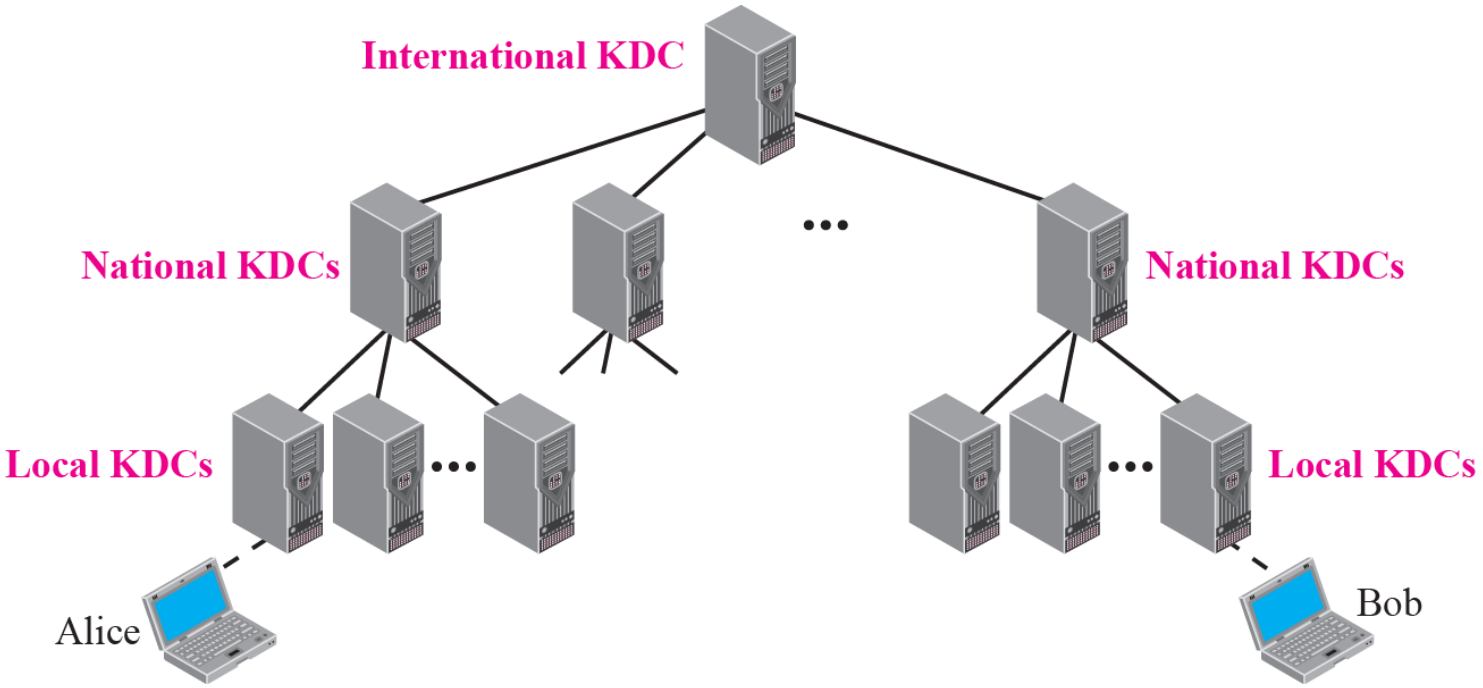
29-9 KEY MANAGEMENT

We discussed symmetric-key and asymmetric-key cryptography in the previous sections. However, we have not yet discussed how secret keys in symmetric-key cryptography, and public keys in asymmetric-key cryptography, are distributed and maintained. This section touches on these two issues.

Topics Discussed in the Section

- ✓ **Symmetric-Key Distribution**
- ✓ **Symmetric-Key Agreement**
- ✓ **Public-Key Distribution**

Figure 29.25 *Multiple KDC's*



Note

A session symmetric key between two parties is used only once.

Figure 29.26 *Creating a session key using KDC*

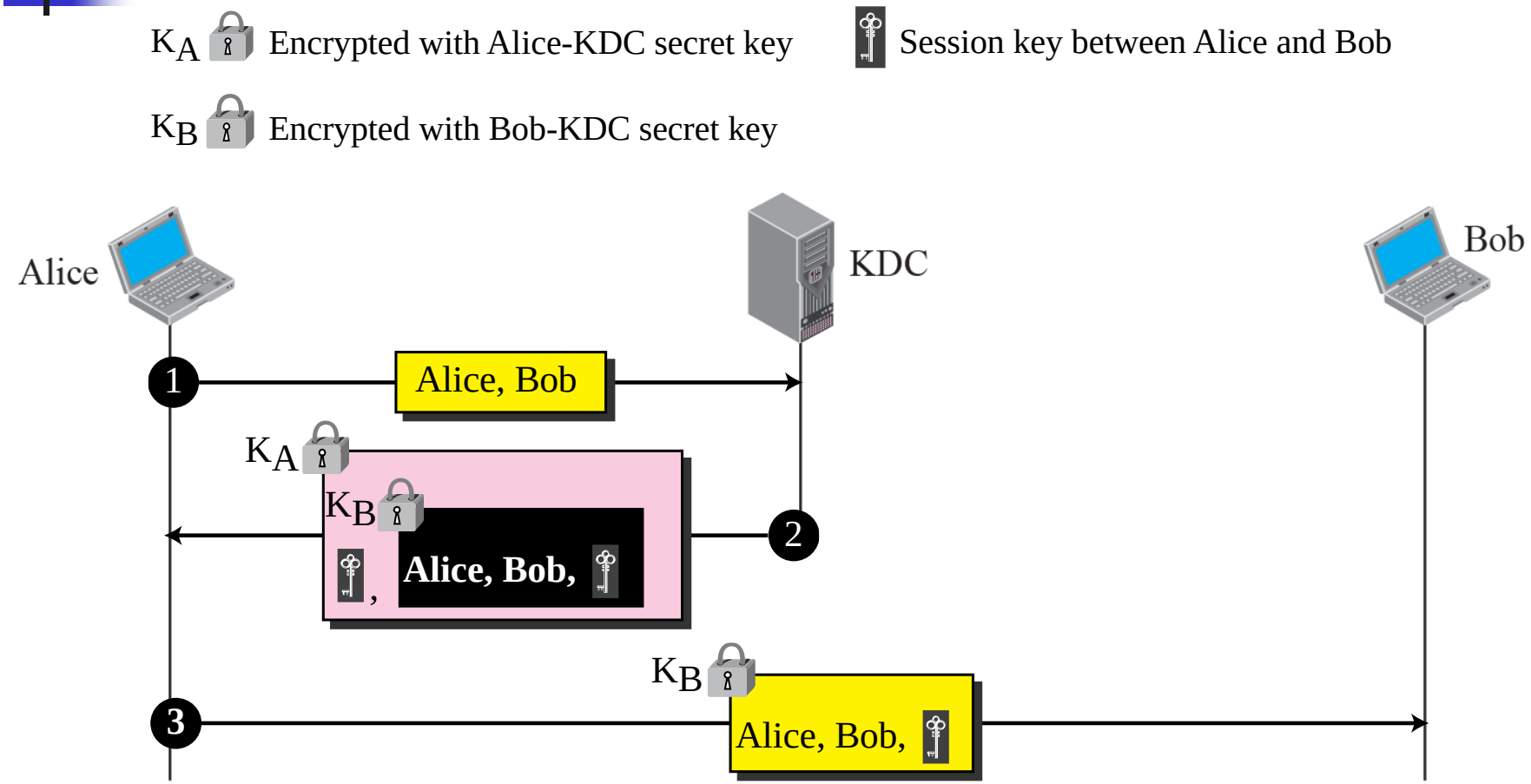
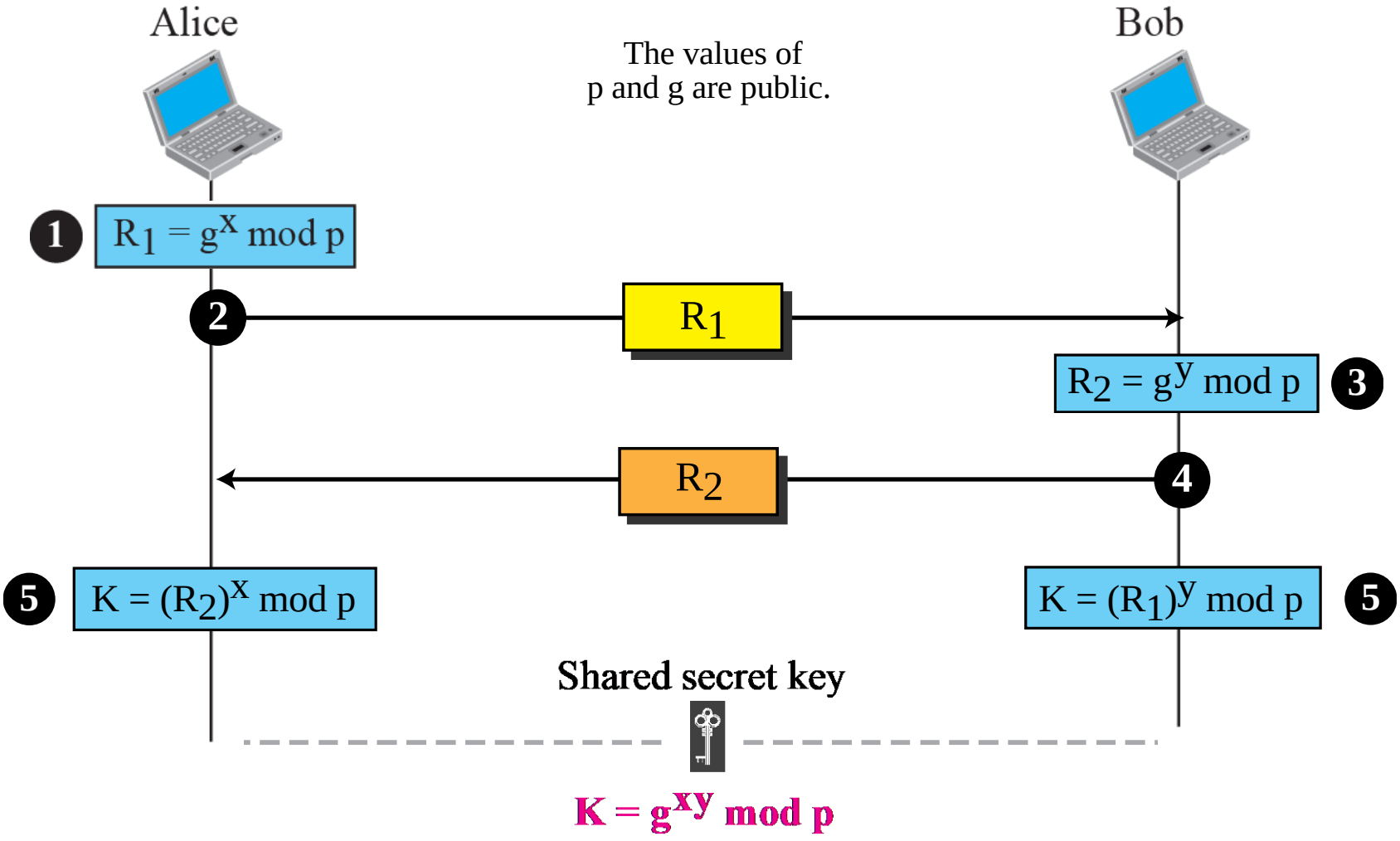


Figure 29.27 *Diffie-Hellman method*



Note

The symmetric (shared) key in the Diffie-Hellman method is $K = g^{xy} \bmod p$.

Example 29.8

Let us give a trivial example to make the procedure clear. Our example uses small numbers, but note that in a real situation, the numbers are very large. Assume that $g = 7$ and $p = 23$. The steps are as follows:

1. Alice chooses $x = 3$ and calculates $R^1 = 7^3 \bmod 23 = 21$.
2. Alice sends the number 21 to Bob.
3. Bob chooses y = 6 and calculates $R_2 = 7^6 \bmod 23 = 4$.
4. Bob sends the number 4 to Alice.
5. Alice calculates the symmetric key $K = 4^3 \bmod 23 = 18$. Bob calculates the symmetric key $K = 21^6 \bmod 23 = 18$.

The value of K is the same for both Alice and Bob; $g^{xy} \bmod p = 7^{18} \bmod 35 = 18$.

Note

In public-key cryptography, everyone has access to everyone's public key; public keys are available to the public.

Figure 29.28 *Certification authority*