UNIT - II

Traditional Symmetric-Key Ciphers

Objectives

- To define the terms and the concepts of symmetric key ciphers
- To emphasize the two categories of traditional ciphers: substitution and transposition ciphers
- To describe the categories of cryptanalysis used to break the symmetric ciphers
- To introduce the concepts of the stream ciphers and block ciphers
- To discuss some very dominant ciphers used in the past, such as the Enigma machine

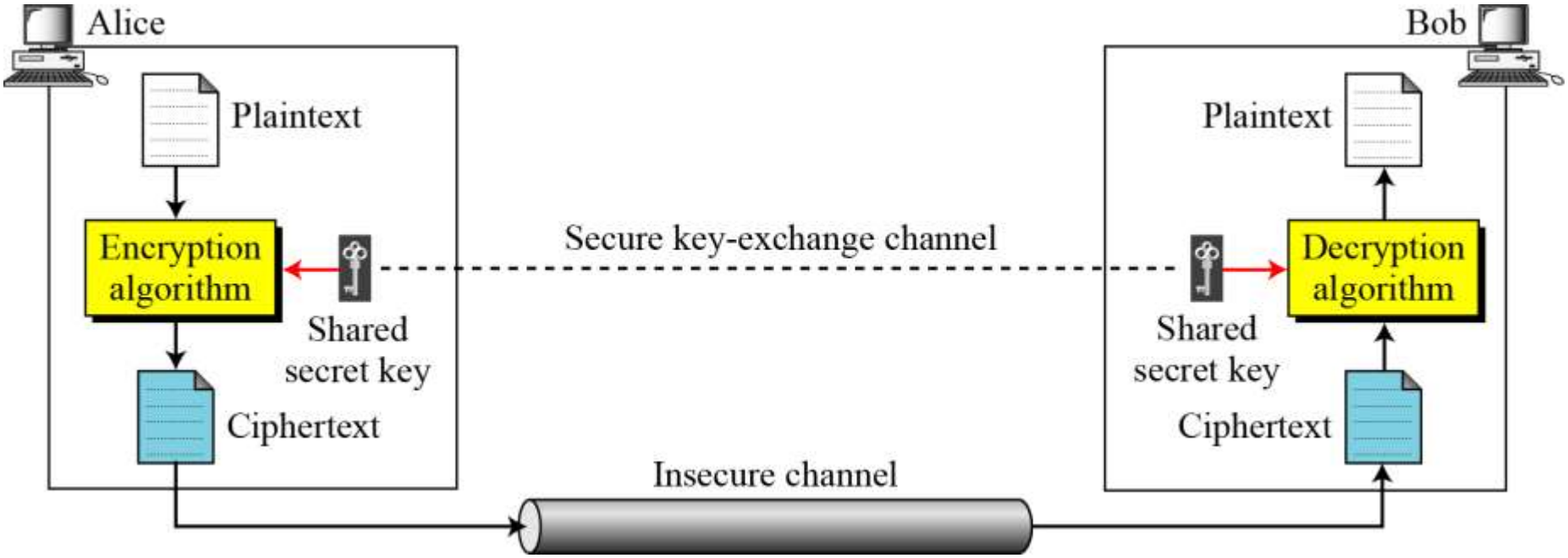
3-1 INTRODUCTION- symmetric key

Figure 3.1 shows the general idea behind a symmetric-key cipher.

- The original message from Alice to Bob is called plaintext
- The message that is sent through the channel is called the cipher text.
- To create the cipher text from the plaintext, Alice uses an encryption algorithm and a shared secret key.
- To create the plaintext from cipher text, Bob uses a decryption algorithm and the same secret key.

3.1 Continued

Figure 3.1 General idea of symmetric-key cipher



3.1 Continued

If P is the plaintext, C is the ciphertext, and K is the key,

$$\text{Encryption: } C = E_k(P) \qquad \text{Decryption: } P = D_k(C)$$

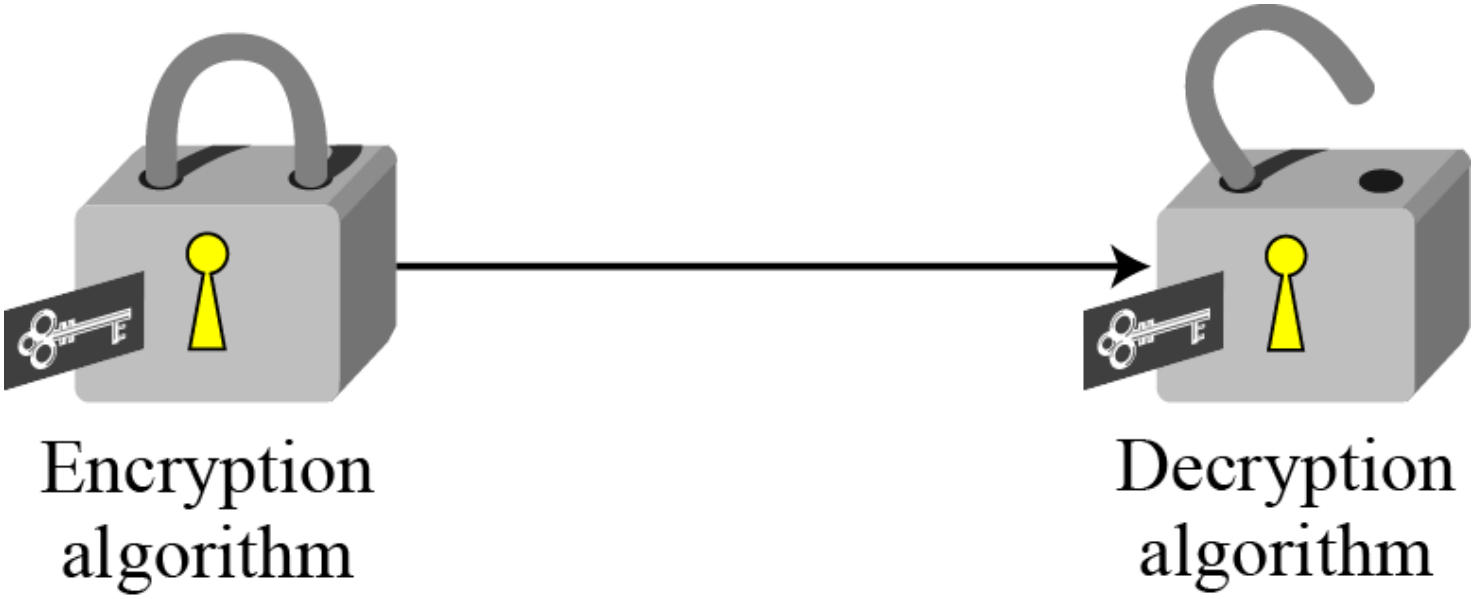
$$\text{In which, } D_k(E_k(x)) = E_k(D_k(x)) = x$$

We assume that Bob creates P_1; we prove that $P_1 = P$:

$$\textbf{Alice: } C = E_k(P)$$

$$\textbf{Bob: } P_1 = D_k(C) = D_k(E_k(P)) = P$$

Figure 3.2 Locking and unlocking with the same key



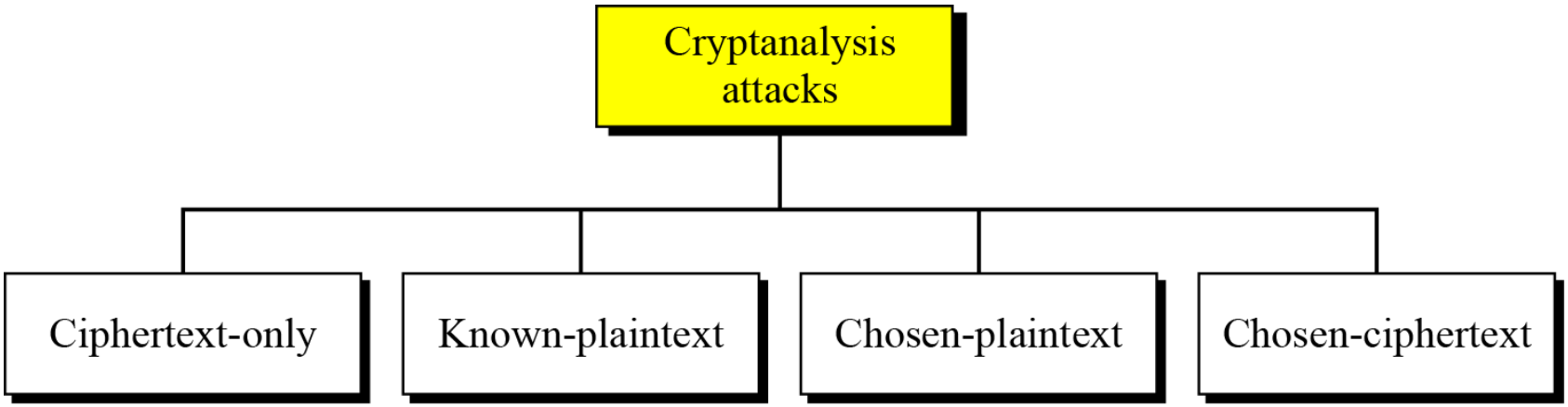
3.1.1 Kerckhoff's Principle

Based on Kerckhoff's principle, one should always assume that the opponent, Eve, knows the encryption/decryption algorithm. The resistance of the cipher to attack must be based only on the secrecy of the key.

3.1.2 Cryptanalysis

As cryptography is the science and art of creating secret codes, cryptanalysis is the science and art of breaking those codes.

Figure 3.3 Cryptanalysis attacks



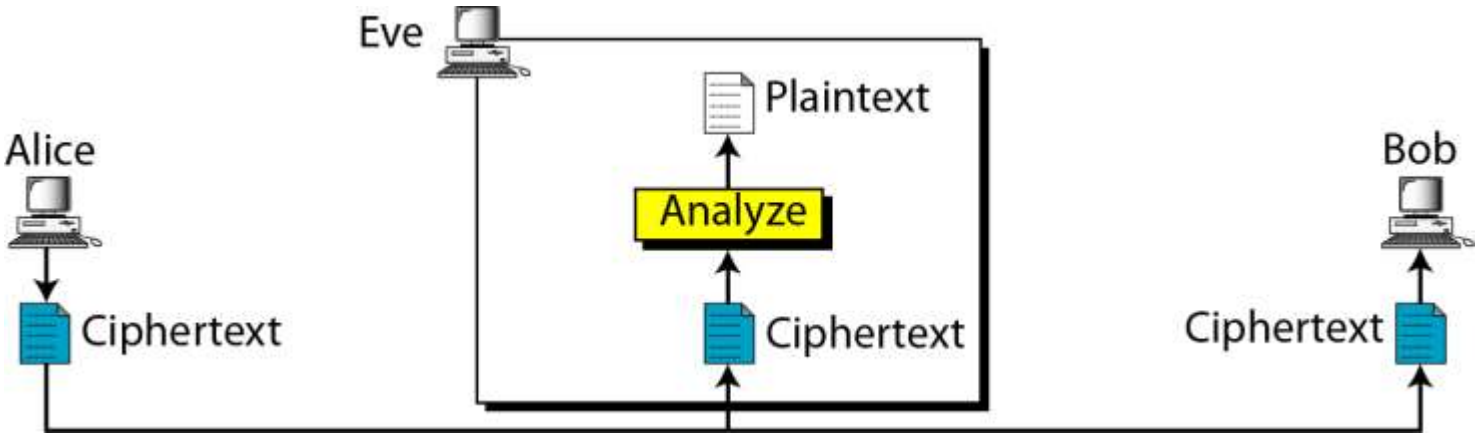
Type of Attack	Known to Cryptanalyst
Ciphertext Only	• Encryption algorithm • Ciphertext
Known Plaintext	• Encryption algorithm • Ciphertext • One or more plaintext-ciphertext pairs formed with the secret key
Chosen Plaintext	Encryption algorithm • Ciphertext • Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret key

Type of Attack	Known to Cryptanalyst
Chosen Ciphertext	Encryption algorithm • Ciphertext • Ciphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key
Chosen Text	• Encryption algorithm • Ciphertext • Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret key • Ciphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key

3.1.2 Continued

Ciphertext-Only Attack

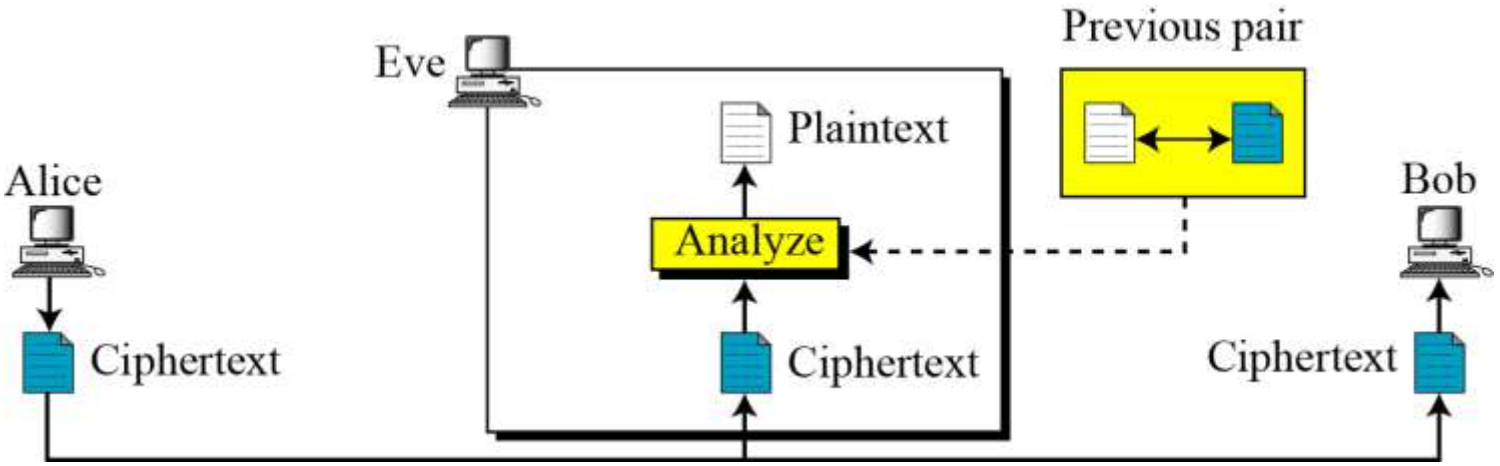
Figure 3.4 Ciphertext-only attack



3.1.2 Continued

Known-Plaintext Attack

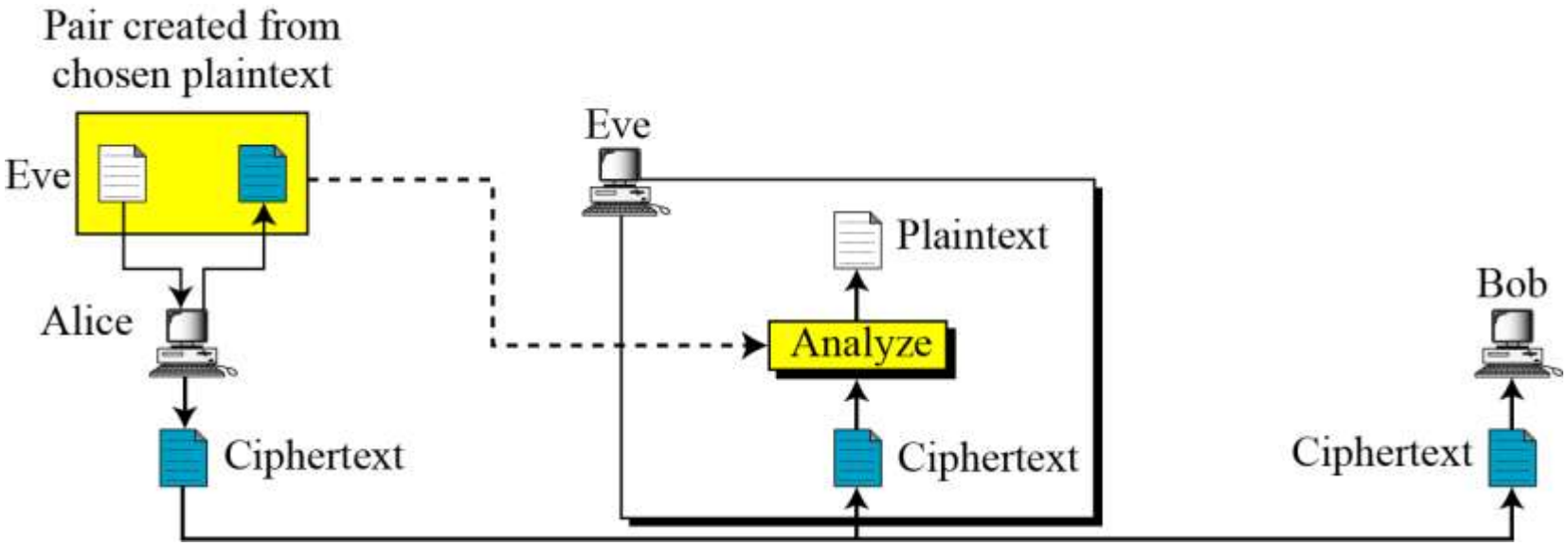
Figure 3.5 Known-plaintext attack



3.1.2 Continued

Chosen-Plaintext Attack

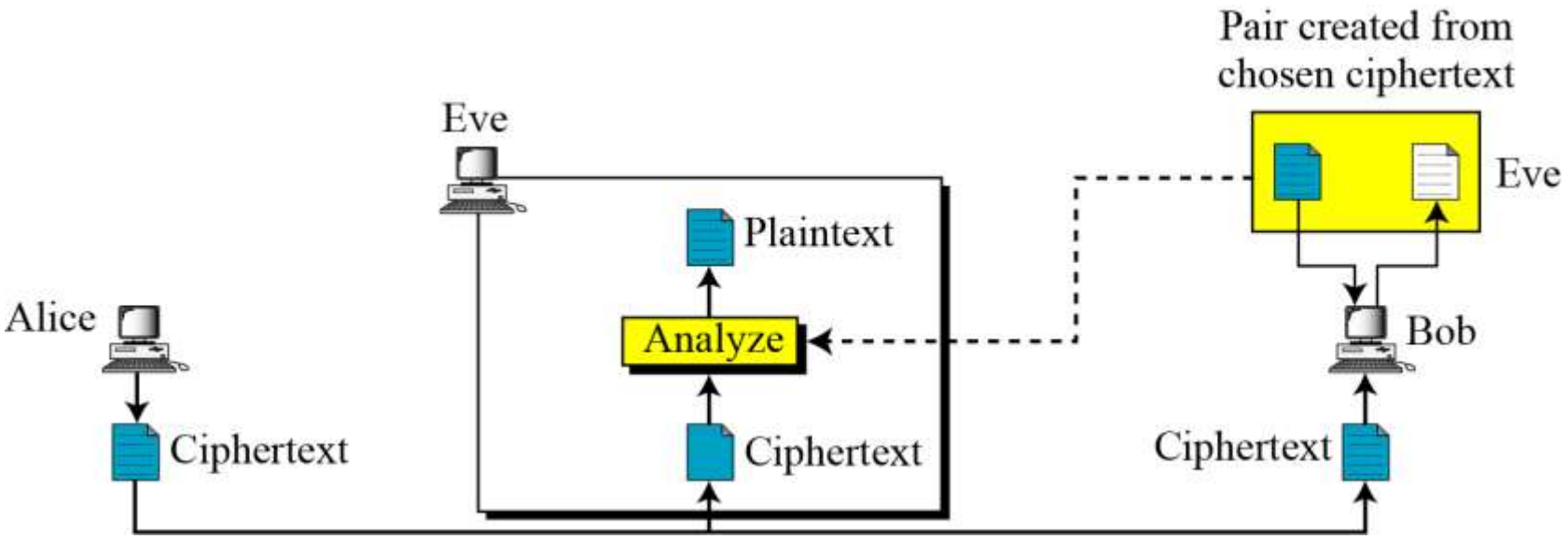
Figure 3.6 Chosen-plaintext attack



3.1.2 Continued

Chosen-Ciphertext Attack

Figure 3.7 Chosen-ciphertext attack



3-2 SUBSTITUTION CIPHERS

A substitution cipher replaces one symbol with another. Substitution ciphers can be categorized as either monoalphabetic ciphers or polyalphabetic ciphers.

Note

A substitution cipher replaces one symbol with another.

Topics discussed in this section:

3.2.1 Monoalphabetic Ciphers
3.2.2 Polyalphabetic Ciphers

3.2.1 Monoalphabetic Ciphers

Note

In monoalphabetic substitution, the relationship between a symbol in the plaintext to a symbol in the ciphertext is always one-to-one.

3.2.1 Continued

Example 3.1

The following shows a plaintext and its corresponding ciphertext. The cipher is probably monoalphabetic because both l's (els) are encrypted as O's.

Plaintext: hello **Ciphertext:** KHOOR

3.2.1 Continued

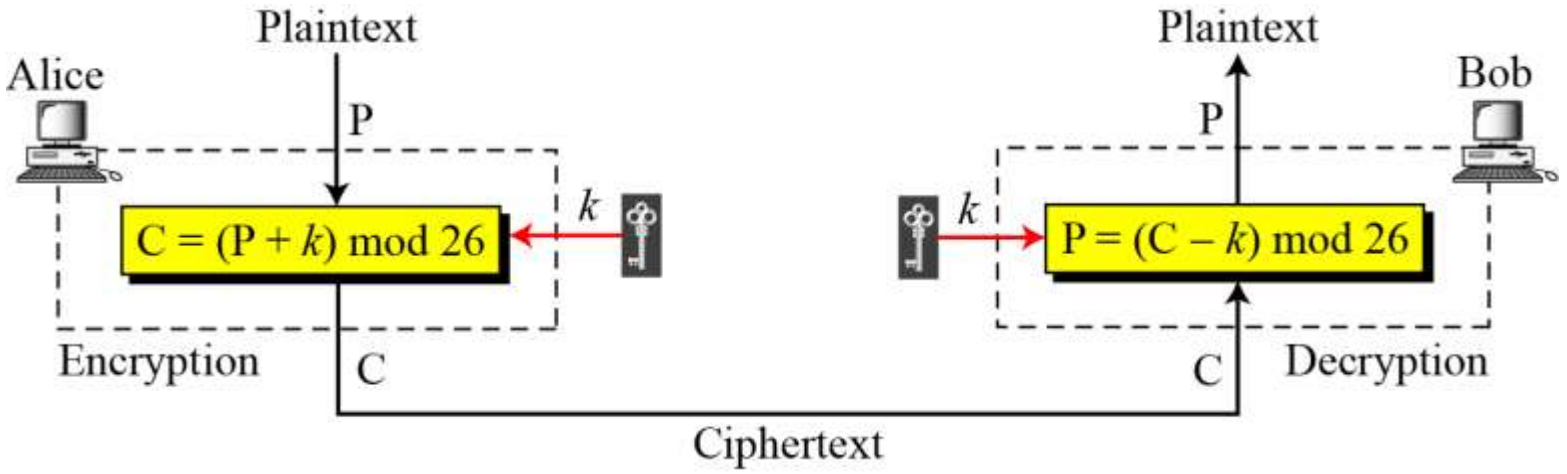
Additive Cipher

The simplest monoalphabetic cipher is the additive cipher. This cipher is sometimes called a shift cipher and sometimes a Caesar cipher, but the term additive cipher better reveals its mathematical nature.

Figure 3.8 Plaintext and ciphertext in Z_{26}

3.2.1 Continued

Figure 3.9 Additive cipher



Note

When the cipher is additive, the plaintext, ciphertext, and key are integers in Z_{26}.

3.2.1 Continued

Example 3.3

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

Solution

We apply the encryption algorithm to the plaintext, character by character:

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

3.2.1 Continued

Example 3.4

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

Solution

We apply the decryption algorithm to the plaintext character by character:

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

3.2.1 Continued

Shift Cipher and Caesar Cipher

Historically, additive ciphers are called shift ciphers. Julius Caesar used an additive cipher to communicate with his officers. For this reason, additive ciphers are sometimes referred to as the Caesar cipher. Caesar used a key of 3 for his communications.

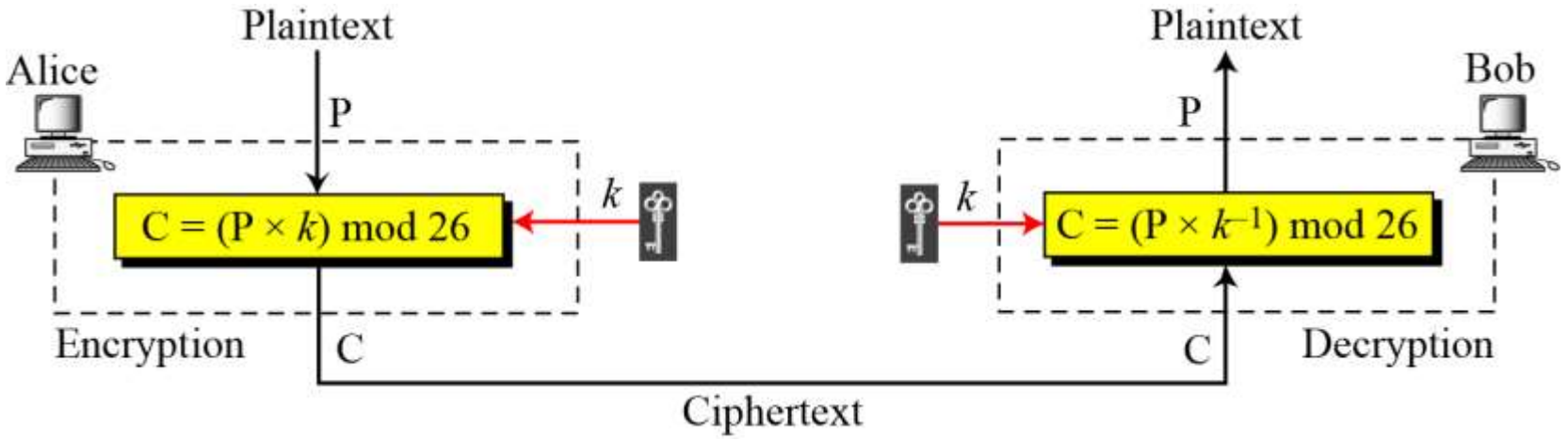
Note

Additive ciphers are sometimes referred to as shift ciphers or Caesar cipher.

3.2.1 Continued

Multiplicative Ciphers

Figure 3.10 Multiplicative cipher



Note

In a multiplicative cipher, the plaintext and ciphertext are integers in Z_{26}; the key is an integer in Z26*.

3.2.1 Continued

Example 3.7

What is the key domain for any multiplicative cipher?

Solution

The key needs to be in Z_{26}^*. This set has only 12 members: 1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, 25.

Example 3.8

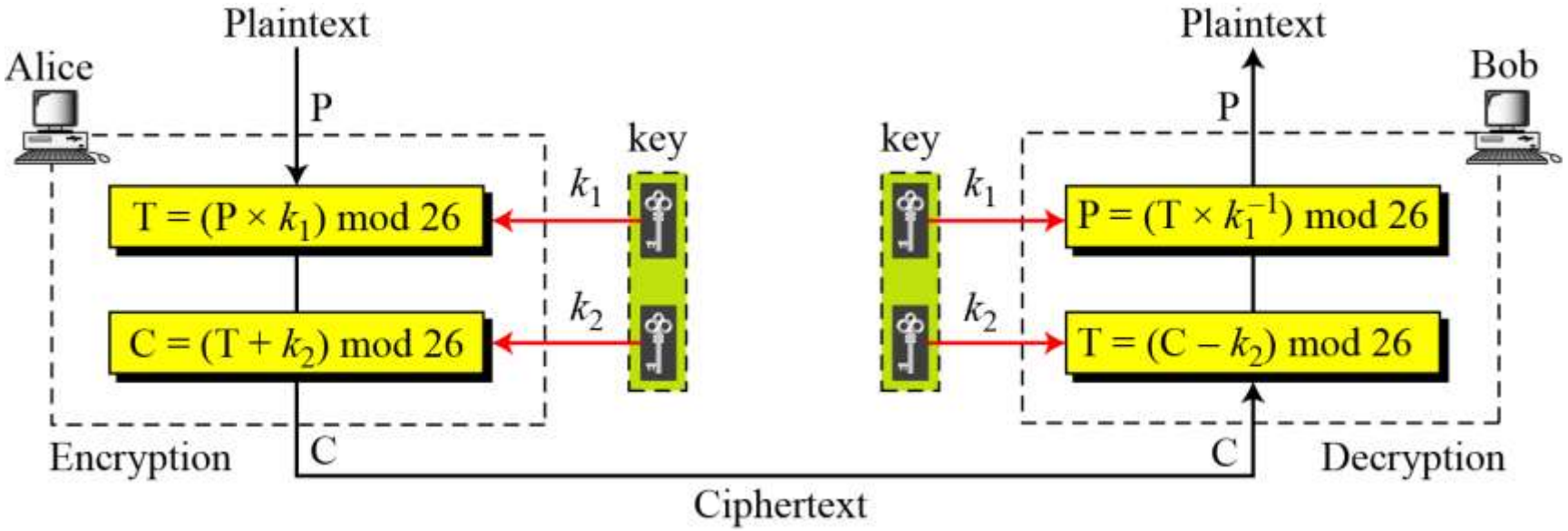
We use a multiplicative cipher to encrypt the message "hello" with a key of 7. The ciphertext is "XCZZU".

Plaintext: h $\rightarrow$ 07	Encryption: $(07 \times 07) \bmod 26$	ciphertext: 23 $\rightarrow$ X
Plaintext: e $\rightarrow$ 04	Encryption: $(04 \times 07) \bmod 26$	ciphertext: 02 $\rightarrow$ C
Plaintext: l $\rightarrow$ 11	Encryption: $(11 \times 07) \bmod 26$	ciphertext: 25 $\rightarrow$ Z
Plaintext: l $\rightarrow$ 11	Encryption: $(11 \times 07) \bmod 26$	ciphertext: 25 $\rightarrow$ Z
Plaintext: o $\rightarrow$ 14	Encryption: $(14 \times 07) \bmod 26$	ciphertext: 20 $\rightarrow$ U

3.2.1 Continued

Affine Ciphers

Figure 3.11 Affine cipher



$$C = (P \times k_1 + k_2) \bmod 26 \qquad P = ((C - k_2) \times k_1^{-1}) \bmod 26$$

where k_1^{-1} is the multiplicative inverse of k_1 and $-k_2$ is the additive inverse of k_2

3.2.1 Continued

Example 3.09

The affine cipher uses a pair of keys in which the first key is from Z_{26}^* and the second is from Z_{26}. The size of the key domain is $26 \times 12 = 312$.

Example 3.10

Use an affine cipher to encrypt the message "hello" with the key pair (7, 2).

P: h $\rightarrow$ 07	Encryption: $(07 \times 7 + 2) \bmod 26$	C: 25 $\rightarrow$ Z
P: e $\rightarrow$ 04	Encryption: $(04 \times 7 + 2) \bmod 26$	C: 04 $\rightarrow$ E
P: l $\rightarrow$ 11	Encryption: $(11 \times 7 + 2) \bmod 26$	C: 01 $\rightarrow$ B
P: l $\rightarrow$ 11	Encryption: $(11 \times 7 + 2) \bmod 26$	C: 01 $\rightarrow$ B
P: o $\rightarrow$ 14	Encryption: $(14 \times 7 + 2) \bmod 26$	C: 22 $\rightarrow$ W

3.2.1 Continued

Example 3.11

Use the affine cipher to decrypt the message "ZEBBW" with the key pair (7, 2) in modulus 26.

Solution

C: Z → 25	Decryption: $((25 - 2) \times 7^{-1}) \bmod 26$	P:07 → h
C: E → 04	Decryption: $((04 - 2) \times 7^{-1}) \bmod 26$	P:04 → e
C: B → 01	Decryption: $((01 - 2) \times 7^{-1}) \bmod 26$	P:11 → l
C: B → 01	Decryption: $((01 - 2) \times 7^{-1}) \bmod 26$	P:11 → l
C: W → 22	Decryption: $((22 - 2) \times 7^{-1}) \bmod 26$	P:14 → o

Example 3.12

The additive cipher is a special case of an affine cipher in which $k_1 = 1$. The multiplicative cipher is a special case of affine cipher in which $k_2 = 0$.

3.2.1 Continued

Monoalphabetic Substitution Cipher

Because additive, multiplicative, and affine ciphers have small key domains, they are very vulnerable to brute-force attack.

A better solution is to create a mapping between each plaintext character and the corresponding ciphertext character. Alice and Bob can agree on a table showing the mapping for each character.

Figure 3.12 An example key for monoalphabetic 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

3.2.1 Continued

Example 3.13

We can use the key in Figure 3.12 to encrypt the message

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

The ciphertext is

ICFVQRVVNEFVRNVSIYRGAHSLIOJICNHTIYBFGTICRXRS

3.2.2 Polyalphabetic Ciphers

In polyalphabetic substitution, each occurrence of a character may have a different substitute. The relationship between a character in the plaintext to a character in the ciphertext is one-to-many.

Autokey Cipher

$$P = P_1 P_2 P_3 \ldots \qquad C = C_1 C_2 C_3 \ldots \qquad k = (k_1, P_1, P_2, \ldots)$$

$$\text{Encryption: } C_i = (P_i + k_i) \bmod 26 \qquad \text{Decryption: } P_i = (C_i - k_i) \bmod 26$$

3.2.2 Continued

Example 3.14

Assume that Alice and Bob agreed to use an autokey cipher with initial key value k1 = 12. Now Alice wants to send Bob the message "Attack is today". Enciphering is done character by character.

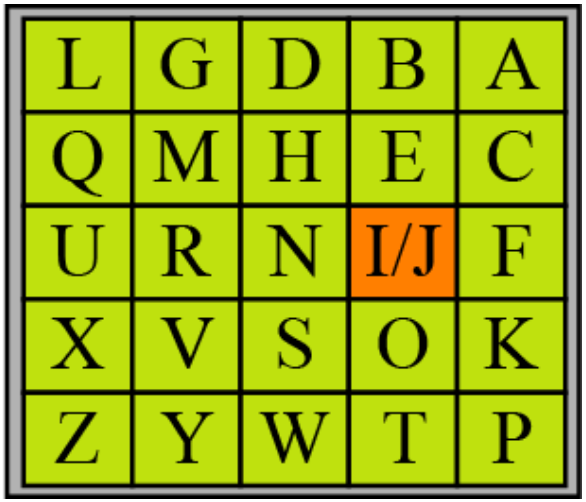
Plaintext:	a	t	t	a	c	k	i	s	t	o	d	a	y
P's Values:	00	19	19	00	02	10	08	18	19	14	03	00	24
Key stream:	*12*	*00*	*19*	*19*	*00*	*02*	*10*	*08*	*18*	*19*	*14*	*03*	*00*
C's Values:	12	19	12	19	02	12	18	00	11	7	17	03	24
Ciphertext:	**M**	**T**	**M**	**T**	**C**	**M**	**S**	**A**	**L**	**H**	**R**	**D**	**Y**

3.2.2 Continued

Playfair Cipher

Figure 3.13 An example of a secret key in the Playfair cipher

Secret Key =



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

Example 3.15

Let us encrypt the plaintext "hello" using the key in Figure 3.13.

he → EC lx → QZ lo → BX

Plaintext: hello Ciphertext: ECQZBX

3.2.2 Continued

Vigenere Cipher

$$P = P_1P_2P_3 \ldots \qquad C = C_1C_2C_3 \ldots \qquad K = [(k_1, k_2, \ldots, k_m), (k_1, k_2, \ldots, k_m), \ldots]$$

$$\text{Encryption: } C_i = P_i + k_i \qquad \text{Decryption: } P_i = C_i - k_i$$

Example 3.16

We can encrypt the message "She is listening" using the 6-character keyword "PASCAL".

Plaintext:	s	h	e	i	s	l	i	s	t	e	n	i	n	g
P's values:	18	07	04	08	18	11	08	18	19	04	13	08	13	06
Key stream:	***15***	***00***	***18***	***02***	***00***	***11***	***15***	***00***	***18***	***02***	***00***	***11***	***15***	***00***
C's values:	**07**	**07**	**22**	**10**	**18**	**22**	**23**	**18**	**11**	**6**	**13**	**19**	**02**	**06**
Ciphertext:	**H**	**H**	**W**	**K**	**S**	**W**	**X**	**S**	**L**	**G**	**N**	**T**	**C**	**G**

3.2.2 Continued

Example 3.16

Let us see how we can encrypt the message "She is listening" using the 6-character keyword "PASCAL". The initial key stream is (15, 0, 18, 2, 0, 11). The key stream is the repetition of this initial key stream (as many times as needed).

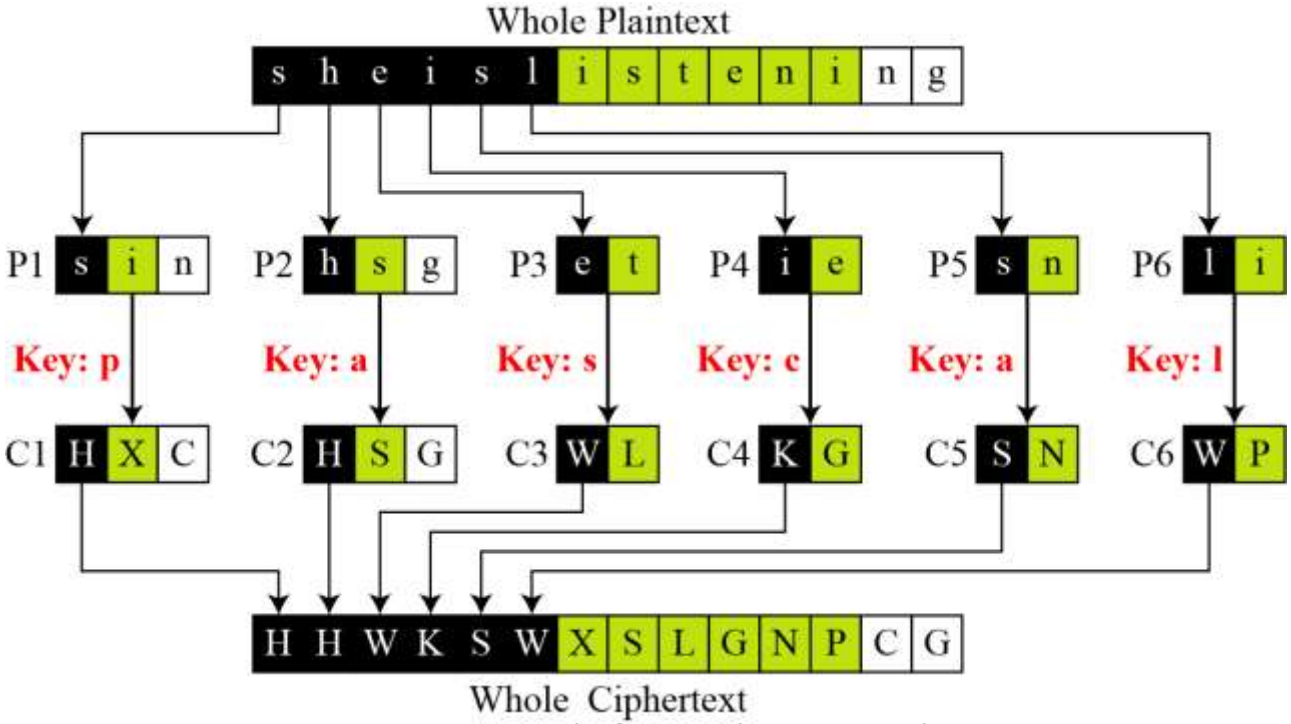
Plaintext:	s	h	e	i	s	l	i	s	t	e	n	i	n	g
P's values:	18	07	04	08	18	11	08	18	19	04	13	08	13	06
Key stream:	***15***	***00***	***18***	***02***	***00***	***11***	***15***	***00***	***18***	***02***	***00***	***11***	***15***	***00***
C's values:	**07**	**07**	**22**	**10**	**18**	**22**	**23**	**18**	**11**	**6**	**13**	**19**	**02**	**06**
Ciphertext:	**H**	**H**	**W**	**K**	**S**	**W**	**X**	**S**	**L**	**G**	**N**	**T**	**C**	**G**

3.2.2 Continued

Example 3.17

Vigenere cipher can be seen as combinations of m additive ciphers.

Figure 3.14 A Vigenere cipher as a combination of m additive ciphers



3.2.2 Continued

Example 3.18

Using Example 3.18, we can say that the additive cipher is a special case of Vigenere cipher in which m = 1.

Table 3.3

A Vigenere Tableau

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

3.2.2 Continued

Vigenere Cipher (Crypanalysis)

Example 3.19

Let us assume we have intercepted the following ciphertext:

```
LIOMWGFEGGDVWGHHCQUCRHRWAGWIOWQLKGZETKKMEVLWPCZVGTH-
VTSGXQOVGCSVETQLTJSUMVWVEUVLXEWSLGFZMVVWLGYHCUSWXQH-
KVGSHEEVFLCFDGVSUMPHKIRZDMPHHBVWVWJWIXGFWLTSHGJOUEEHH-
VUCFVGOWICQLTJSUXGLW
```

The Kasiski test for repetition of three-character segments yields the results shown in Table 3.4.

String	*First Index*	*Second Index*	*Difference*
JSU	68	168	100
SUM	69	117	48
VWV	72	132	60
MPH	119	127	8

3.2.2 Continued

Example 3.19

Let us assume we have intercepted the following ciphertext:

```
LIOMWGFEGGDVWGHHCQUCRHRWAGWIOWQLKGZETKKMEVLWPCZVGTH-
VTSGXQOVGCSVETQLTJSUMVWVEUVLXEWSLGFZMVVWLGYHCUSWXQH-
KVGSHEEVFLCFDGVSUMPHKIRZDMPHHBVWVWJWIXGFWLTSHGJOUEEHH-
VUCFVGOWICQLTJSUXGLW
```

The Kasiski test for repetition of three-character segments yields the results shown in Table 3.4.

String	*First Index*	*Second Index*	*Difference*
JSU	68	168	100
SUM	69	117	48
VWV	72	132	60
MPH	119	127	8

3.2.2 Continued

Example 3.19 (Continued)

The greatest common divisor of differences is 4, which means that the key length is multiple of 4. First try m = 4.

```
C1: LWGWCRAOKTEPGTQCTJVUEGVGUQGECVPRPVJGTJEUGCJG
P1: jueuapymircneroarhtsthihytrahcieixsthcarrehe
C2: IGGGQHGWGKVCTSOSQSWVWFVYSHSVFSHZHWWFSOHCOQSL
P2: usssctsiswhofeaeceihcetesoecatnpntherhctecex
C3: OFDHURWQZKLZHGVVLUVLSZWHWKHFDUKDHVIWHUHFWLUW
P3: lcaerotnwhiwedssirsiirhketehretltiideatrairt
C4: MEVHCWILEMWVVXGETMEXLMLCXVELGMIMBWXLGEVVITX
P4: iardysehaisrrtcapiafpwtethecarhaesfterectpt
```

In this case, the plaintext makes sense.

Julius Caesar used a cryptosystem in his wars, which is now referred to as Caesar cipher. It is an additive cipher with the key set to three. Each character in the plaintext is shifted three characters to create ciphertext.

Hill Cipher

Figure 3.15 Key in the Hill cipher

$$K = \begin{bmatrix} k_{11} & k_{12} & \cdots & k_{1m} \\ k_{21} & k_{22} & \cdots & k_{2m} \\ \vdots & \vdots & & \vdots \\ k_{m1} & k_{m2} & \cdots & k_{mm} \end{bmatrix}$$

$$C_1 = P_1 k_{11} + P_2 k_{21} + \cdots + P_m k_{m1}$$
$$C_2 = P_1 k_{12} + P_2 k_{22} + \cdots + P_m k_{m2}$$
$$\cdots$$
$$C_m = P_1 k_{1m} + P_2 k_{2m} + \cdots + P_m k_{mm}$$

Note

The key matrix in the Hill cipher needs to have a multiplicative inverse.

3.2.2 Continued

Example 3.20

For example, the plaintext “code is ready” can make a 3 × 4 matrix when adding extra bogus character “z” to the last block and removing the spaces. The ciphertext is “OHKNIHGKLISS”.

Figure 3.16 Example 3.20

$$\begin{bmatrix} 14 & 07 & 10 & 13 \\ 08 & 07 & 06 & 11 \\ 11 & 08 & 18 & 18 \end{bmatrix} = \begin{bmatrix} 02 & 14 & 03 & 04 \\ 08 & 18 & 17 & 04 \\ 00 & 03 & 24 & 25 \end{bmatrix} \begin{bmatrix} 09 & 07 & 11 & 13 \\ 04 & 07 & 05 & 06 \\ 02 & 21 & 14 & 09 \\ 03 & 23 & 21 & 08 \end{bmatrix}$$

C = P × K

a. Encryption

$$\begin{bmatrix} 02 & 14 & 03 & 04 \\ 08 & 18 & 17 & 04 \\ 00 & 03 & 24 & 25 \end{bmatrix} = \begin{bmatrix} 14 & 07 & 10 & 13 \\ 08 & 07 & 06 & 11 \\ 11 & 08 & 18 & 18 \end{bmatrix} \begin{bmatrix} 02 & 15 & 22 & 03 \\ 15 & 00 & 19 & 03 \\ 09 & 09 & 03 & 11 \\ 17 & 00 & 04 & 07 \end{bmatrix}$$

P = C × K^{-1}

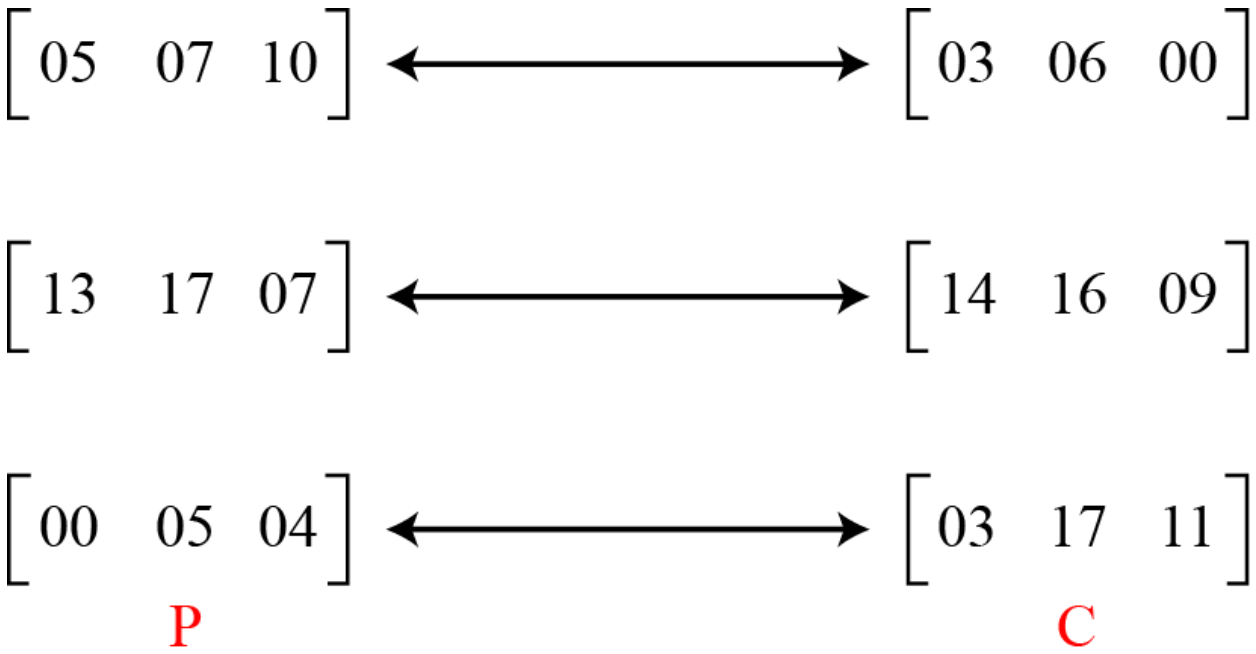
b. Decryption

3.2.2 Continued

Example 3.21

Assume that Eve knows that m = 3. She has intercepted three plaintext/ciphertext pair blocks (not necessarily from the same message) as shown in Figure 3.17.

Figure 3.17 Example 3.21



3.2.2 Continued

Example 3.21 (Continued)

She makes matrices P and C from these pairs. Because P is invertible, she inverts the P matrix and multiplies it by C to get the K matrix as shown in Figure 3.18.

Figure 3.18 Example 3.21

$$\begin{bmatrix} 02 & 03 & 07 \\ 05 & 07 & 09 \\ 01 & 02 & 11 \end{bmatrix} = \begin{bmatrix} 21 & 14 & 01 \\ 00 & 08 & 25 \\ 13 & 03 & 08 \end{bmatrix} \begin{bmatrix} 03 & 06 & 00 \\ 14 & 16 & 09 \\ 03 & 17 & 11 \end{bmatrix}$$

$$K \qquad P^{-1} \qquad C$$

Now she has the key and can break any ciphertext encrypted with that key.

3.2.2 Continued

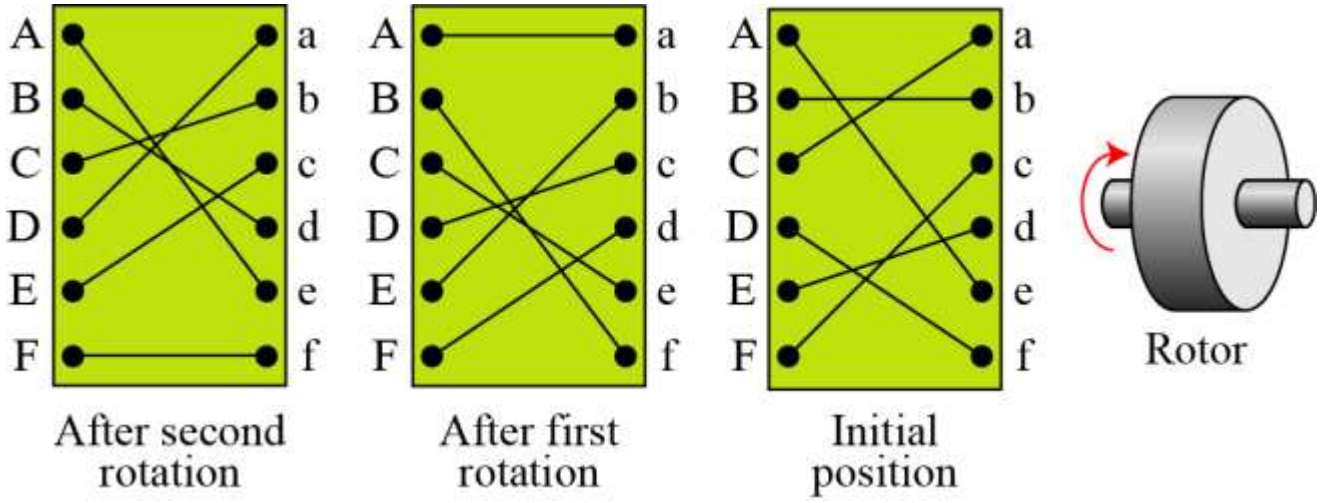
One-Time Pad

One of the goals of cryptography is perfect secrecy. A study by Shannon has shown that perfect secrecy can be achieved if each plaintext symbol is encrypted with a key randomly chosen from a key domain. This idea is used in a cipher called one-time pad, invented by Vernam.

3.2.2 Continued

Rotor Cipher

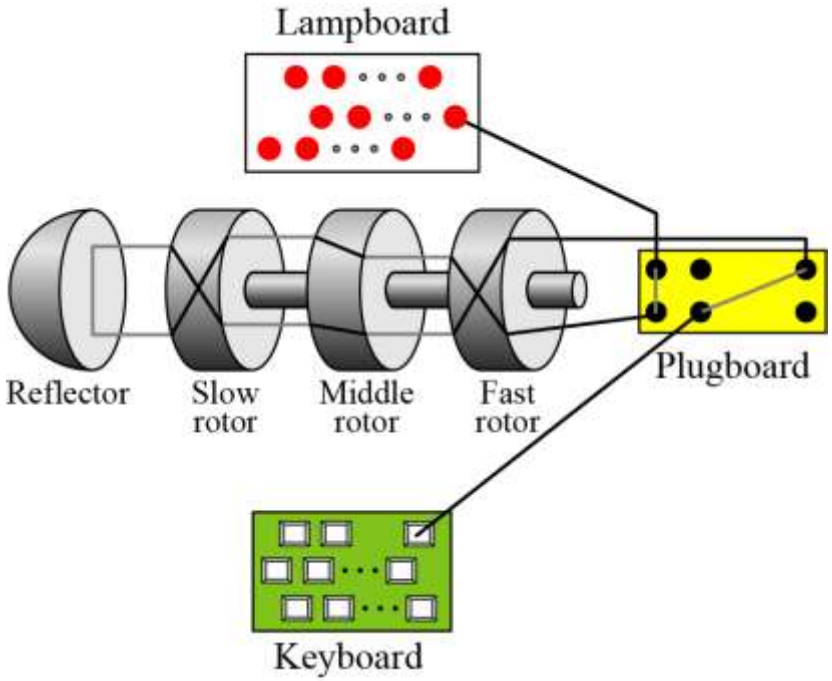
Figure 3.19 A rotor cipher



3.2.2 Continued

Enigma Machine

Figure 3.20 A schematic of the Enigma machine



3-3 TRANSPOSITION CIPHERS

A transposition cipher does not substitute one symbol for another, instead it changes the location of the symbols.

Note

A transposition cipher reorders symbols.

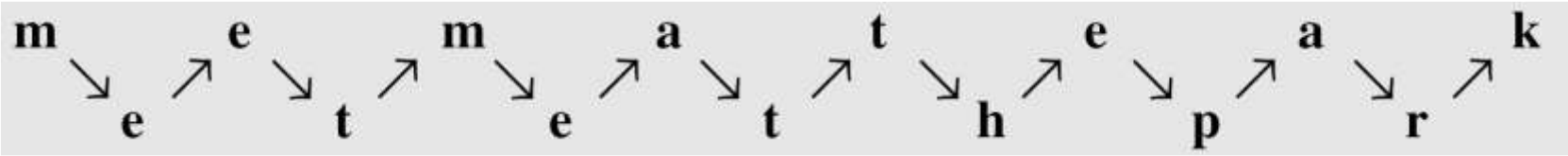
Topics discussed in this section:

3.3.1 Keyless Transposition Ciphers

Simple transposition ciphers, which were used in the past, are keyless.

Example 3.22

A good example of a keyless cipher using the first method is the rail fence cipher. The ciphertext is created reading the pattern row by row. For example, to send the message "Meet me at the park" to Bob, Alice writes



She then creates the ciphertext "MEMATEAKETETHPR".

3.3.1 Continued

Example 3.23

Alice and Bob can agree on the number of columns and use the second method. Alice writes the same plaintext, row by row, in a table of four columns.

m	e	e	t
m	e	a	t
t	h	e	p
a	r	k	

She then creates the ciphertext "MMTAEEHREAEKTTP".

3.3.1 Continued

Example 3.24

The cipher in Example 3.23 is actually a transposition cipher. The following shows the permutation of each character in the plaintext into the ciphertext based on the positions.

01	02	03	04	05	06	07	08	09	10	11	12	13	14	15
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
01	05	09	13	02	06	10	13	03	07	11	15	04	08	12

The second character in the plaintext has moved to the fifth position in the ciphertext; the third character has moved to the ninth position; and so on. Although the characters are permuted,
there is a pattern in the permutation: (01, 05, 09, 13), (02, 06, 10, 13), (03, 07, 11, 15), and (08, 12). In each section, the difference between the two adjacent numbers is 4.

3.3.2 Keyed Transposition Ciphers

The keyless ciphers permute the characters by using writing plaintext in one way and reading it in another way
The permutation is done on the whole plaintext to create the whole ciphertext.

Another method is to divide the plaintext into groups of predetermined size, called blocks, and then use a key to permute the characters in each block separately.

3.3.2 Continued

Example 3.25

Alice needs to send the message “Enemy attacks tonight” to Bob..

e n e m y a t t a c k s t o n i g h t z

The key used for encryption and decryption is a permutation key, which shows how the character are permuted.

Encryption ↓

3	1	4	5	2
1	2	3	4	5

↑ Decryption

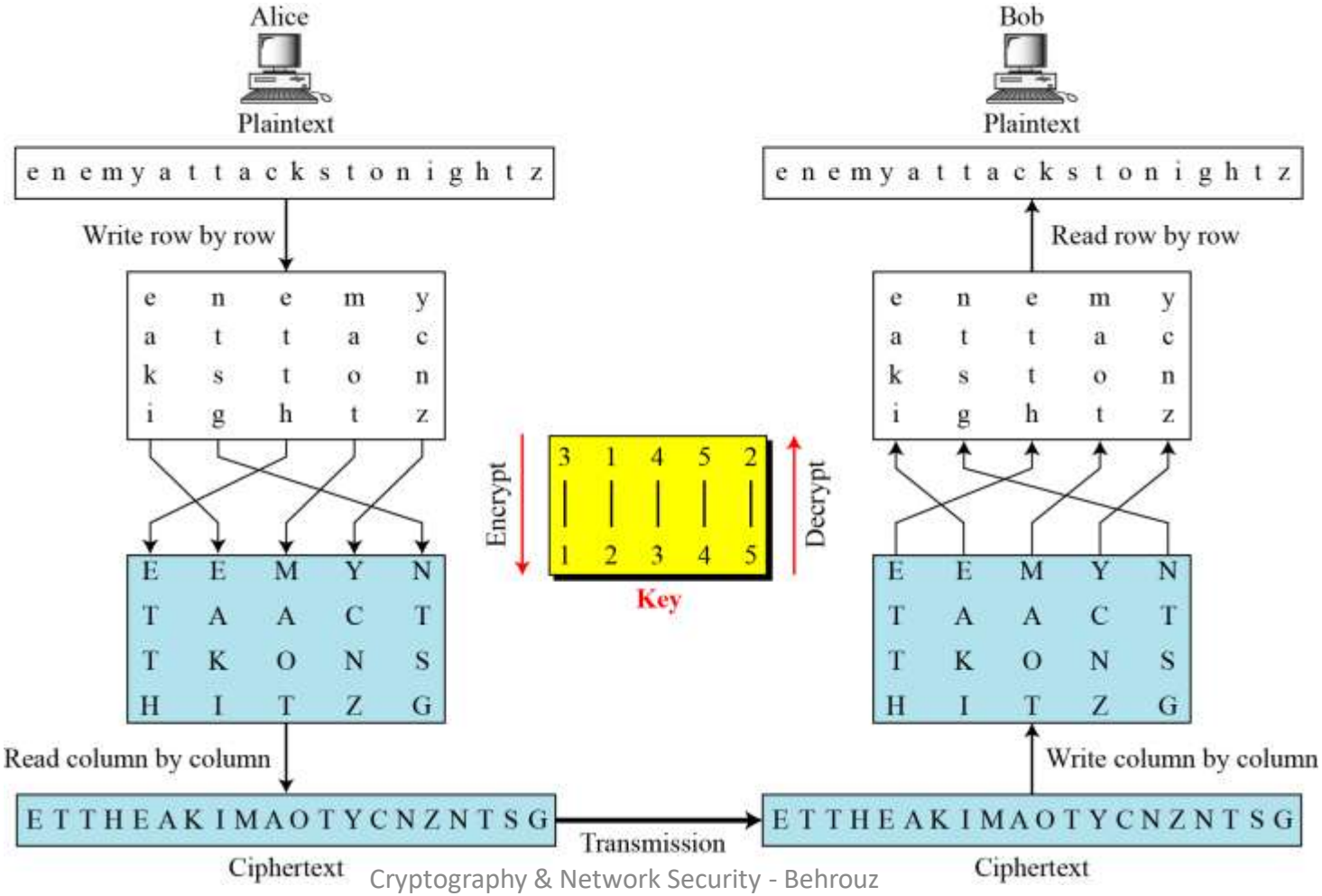
The permutation yields

E E M Y N T A A C T T K O N S H I T Z G

3.3.3 Combining Two Approaches

Example 3.26

Figure 3.21

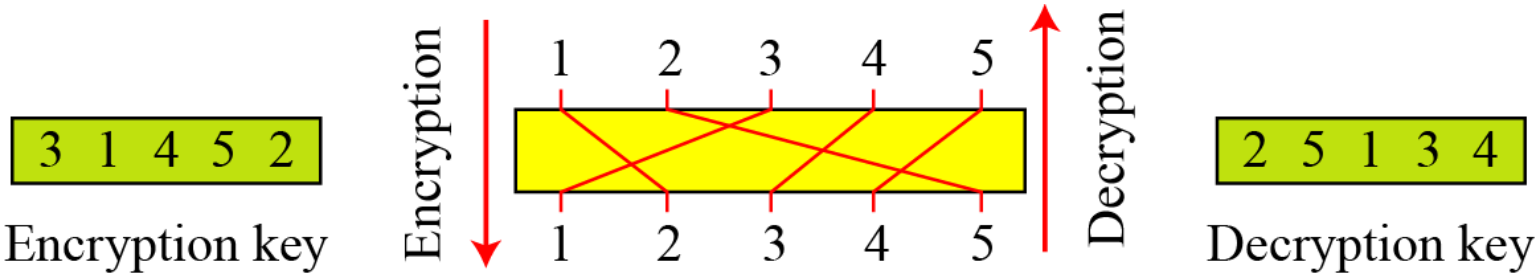


3.3.3 Continued

Keys

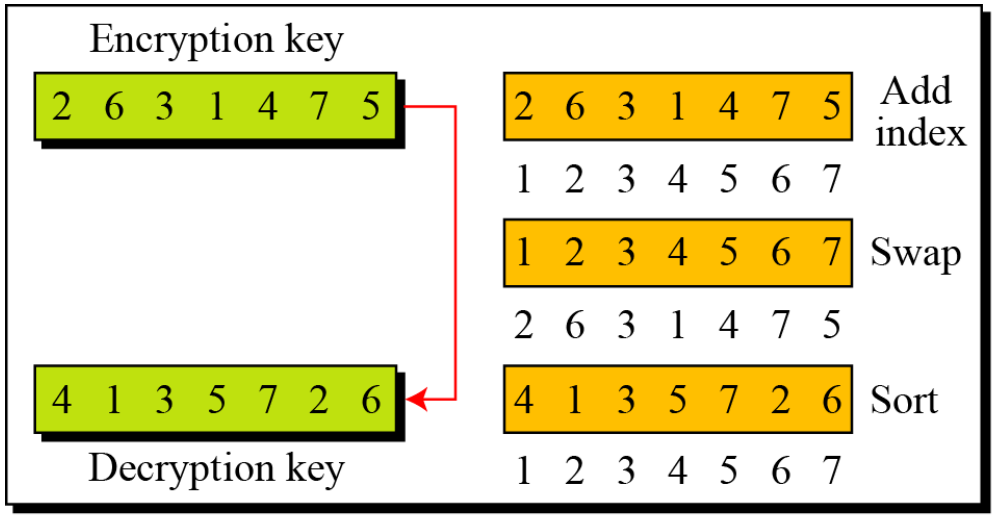
In Example 3.27, a single key was used in two directions for the column exchange: downward for encryption, upward for decryption. It is customary to create two keys.

Figure 3.22 Encryption/decryption keys in transpositional ciphers



3.3.3 Continued

Figure 3.23 Key inversion in a transposition cipher



a. Manual process

```
Given: EncKey [index]
index ← 1
while (index ≤ Column)
  {
   DecKey[EncKey[index]] ← index
   index ← index + 1
  }
Return : DecKey [index]
```

b. Algorithm

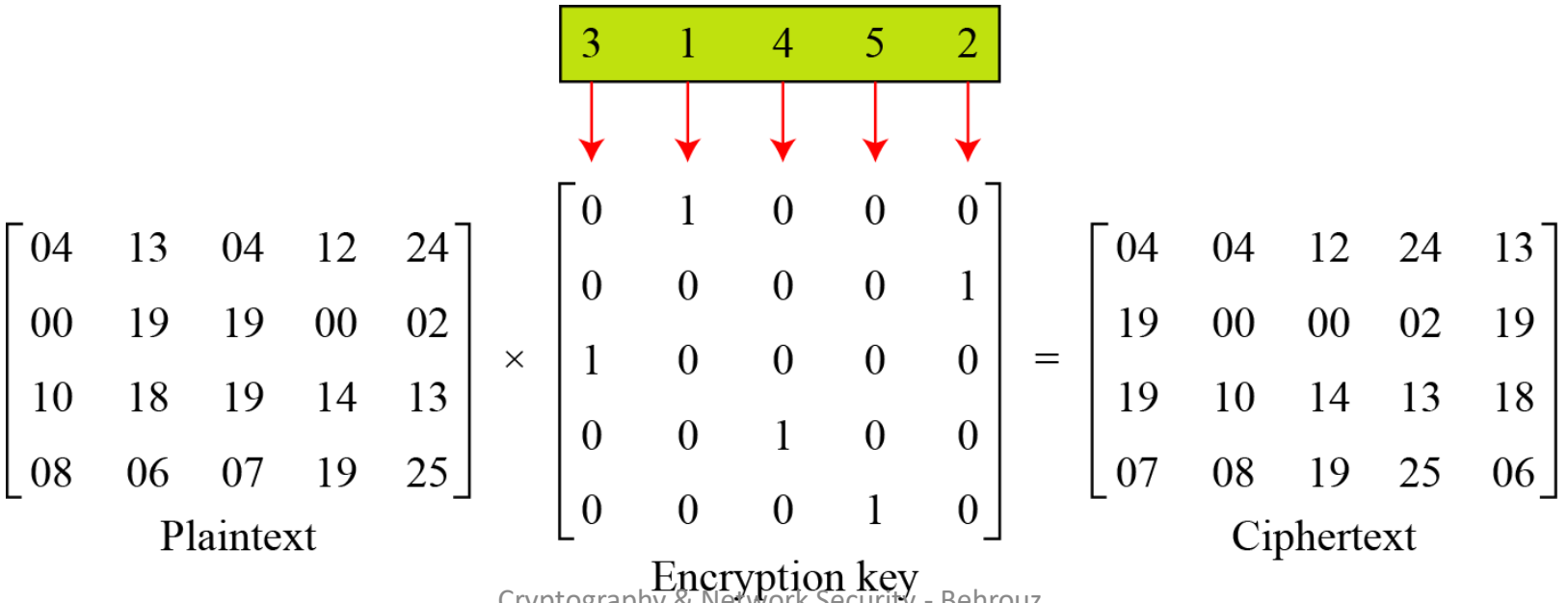
3.3.3 Continued

Using Matrices

We can use matrices to show the encryption/decryption process for a transposition cipher.

Example 3.27

Figure 3.24 Representation of the key as a matrix in the transposition cipher

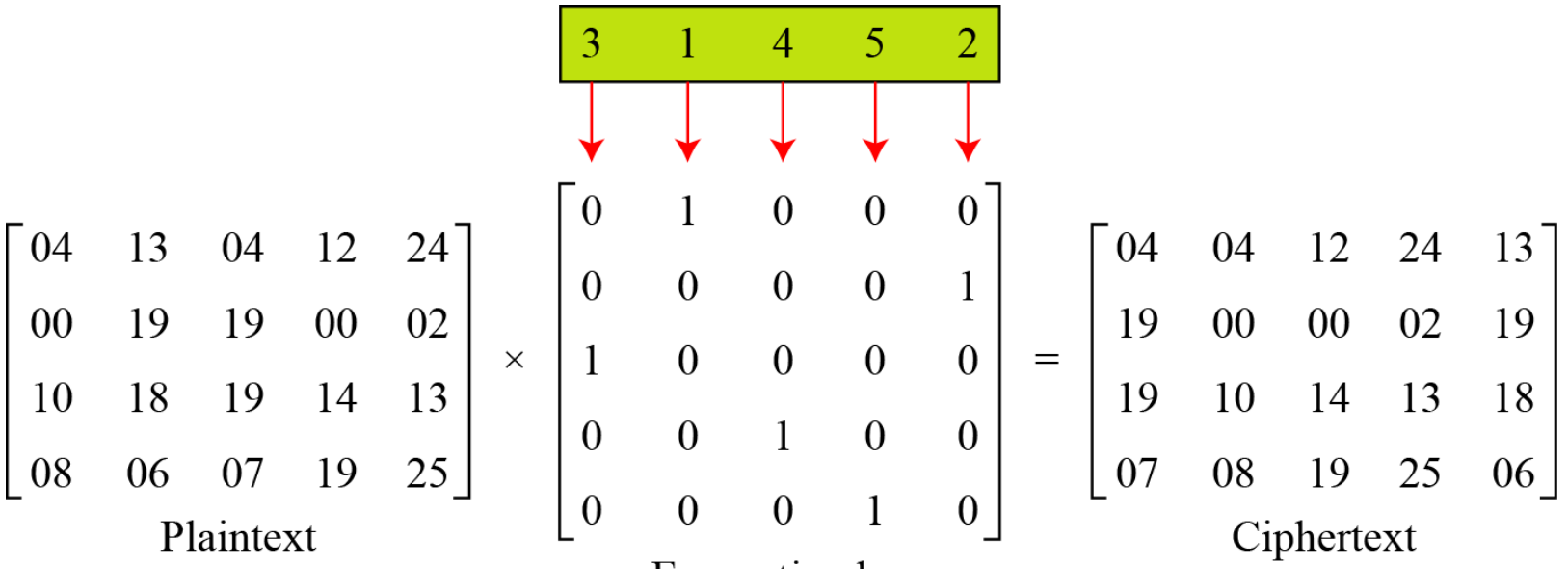


3.3.3 Continued

Example 3.27

Figure 3.24 shows the encryption process. Multiplying the 4 × 5 plaintext matrix by the 5 × 5 encryption key gives the 4 × 5 ciphertext matrix.

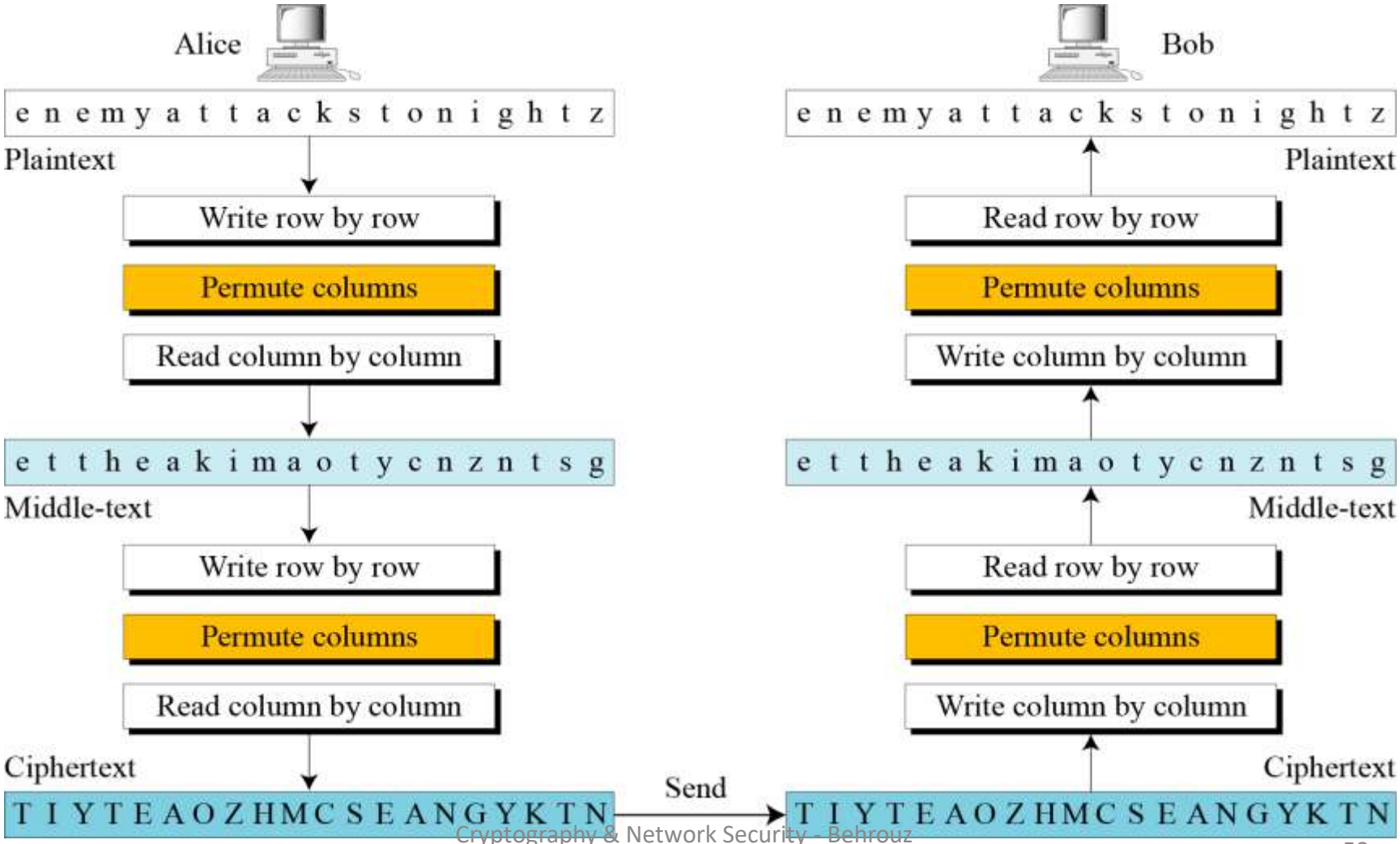
Figure 3.24 Representation of the key as a matrix in the transposition cipher



3.3.3 Continued

Double Transposition Ciphers

Figure 3.25 Double transposition cipher



3-4 STREAM AND BLOCK CIPHERS

The literature divides the symmetric ciphers into two broad categories: stream ciphers and block ciphers. Although the definitions are normally applied to modern ciphers, this categorization also applies to traditional ciphers.

Topics discussed in this section:

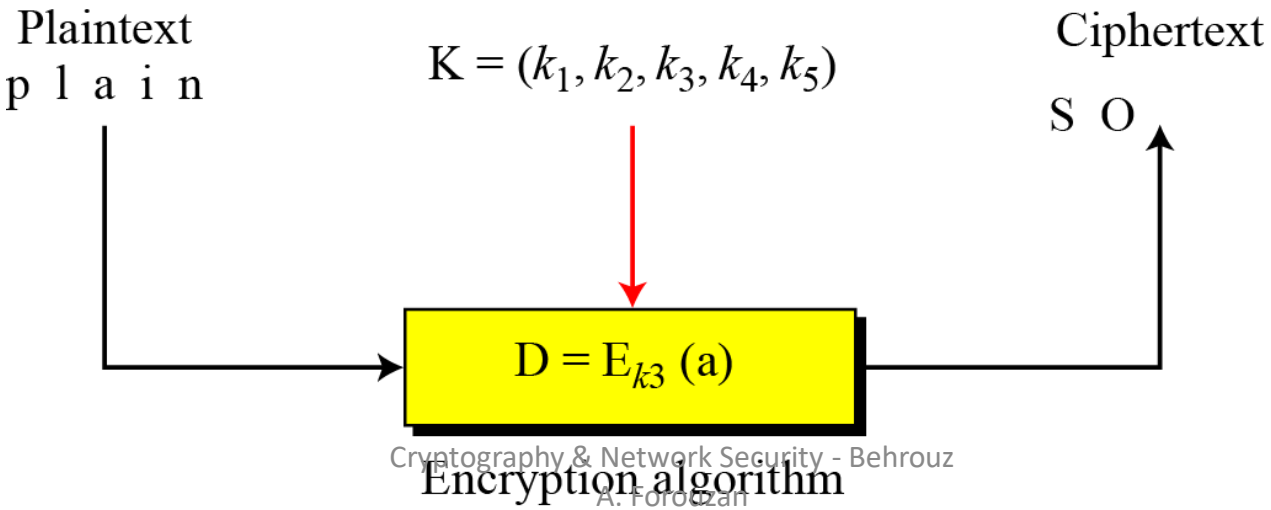
3.4.1 Stream Ciphers

Call the plaintext stream P, the ciphertext stream C, and the key stream K.

$$P = P_1 P_2 P_3, \ldots \qquad C = C_1 C_2 C_3, \ldots \qquad K = (k_1, k_2, k_3, \ldots)$$

$$C_1 = E_{k1}(P_1) \qquad C_2 = E_{k2}(P_2) \qquad C_3 = E_{k3}(P_3) \ldots$$

Figure 3.26 Stream cipher



3.4.1 Continued

Example 3.30

Additive ciphers can be categorized as stream ciphers in which the key stream is the repeated value of the key. In other words, the key stream is considered as a predetermined stream of keys or K = (k, k, …, k). In this cipher, however, each character in the ciphertext depends only on the corresponding character in the plaintext, because the key stream is generated independently.

Example 3.31

The monoalphabetic substitution ciphers discussed in this chapter are also stream ciphers. However, each value of the key stream in this case is the mapping of the current plaintext character to the corresponding ciphertext character in the mapping table.

3.4.1 Continued

Example 3.32

Vigenere ciphers are also stream ciphers according to the definition. In this case, the key stream is a repetition of m values, where m is the size of the keyword. In other words,

$$K = (k_1, k_2, \ldots k_m, k_1, k_2, \ldots k_m, \ldots)$$

Example 3.33

We can establish a criterion to divide stream ciphers based on their key streams. We can say that a stream cipher is a monoalphabetic cipher if the value of k_i does not depend on the position of the plaintext character in the plaintext stream; otherwise, the cipher is polyalphabetic.

3.4.1 Continued

Example 3.33 (Continued)

- Additive ciphers are definitely monoalphabetic because k_i in the key stream is fixed; it does not depend on the position of the character in the plaintext.

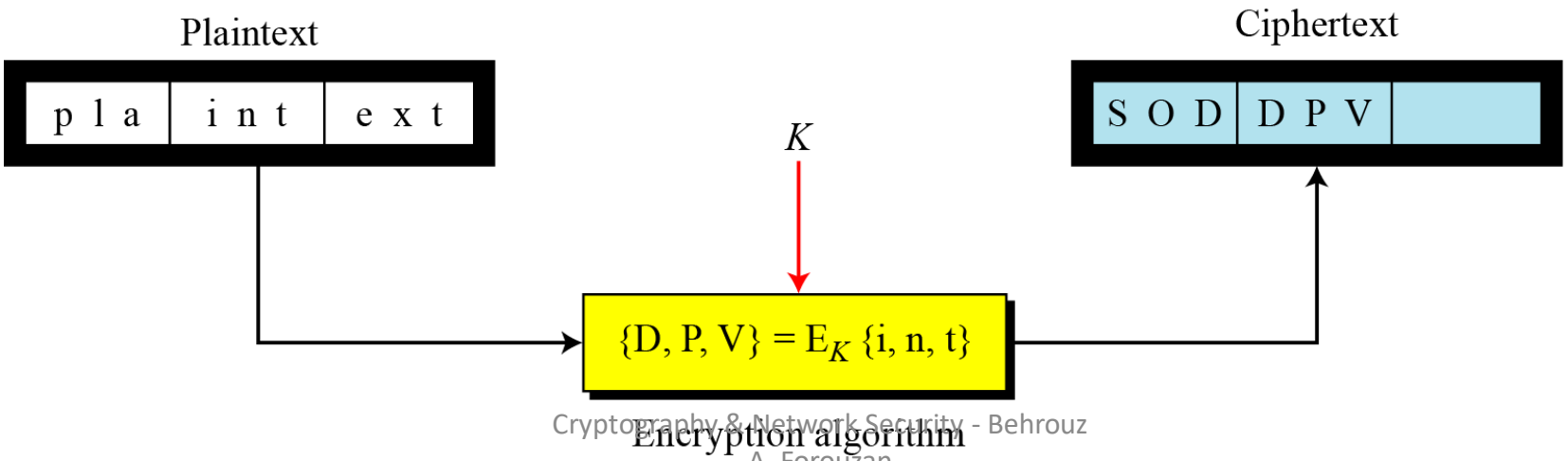
- Monoalphabetic substitution ciphers are monoalphabetic because k_i does not depend on the position of the corresponding character in the plaintext stream; it depends only on the value of the plaintext character.

- Vigenere ciphers are polyalphabetic ciphers because k_i definitely depends on the position of the plaintext character. However, the dependency is cyclic. The key is the same for two characters m positions apart.

3.4.2 Block Ciphers

In a block cipher, a group of plaintext symbols of size m (m > 1) are encrypted together creating a group of ciphertext of the same size. A single key is used to encrypt the whole block even if the key is made of multiple values. Figure 3.27 shows the concept of a block cipher.

Figure 3.27 Block cipher



3.4.2 Continued

Example 3.34

Playfair ciphers are block ciphers. The size of the block is m = 2. Two characters are encrypted together.

Example 3.35

Hill ciphers are block ciphers. A block of plaintext, of size 2 or more is encrypted together using a single key (a matrix). In these ciphers, the value of each character in the ciphertext depends on all the values of the characters in the plaintext. Although the key is made of m × m values, it is considered as a single key.

Example 3.36

From the definition of the block cipher, it is clear that every block cipher is a polyalphabetic cipher because each character in a ciphertext block depends on all characters in the plaintext block.

3.4.3 Combination

In practice, blocks of plaintext are encrypted individually, but they use a stream of keys to encrypt the whole message block by block. In other words, the cipher is a block cipher when looking at the individual blocks, but it is a stream cipher when looking at the whole message considering each block as a single unit.

Data Encryption Standard (DES)

Objectives

- To review a short history of DES
- To define the basic structure of DES
- To describe the details of building elements of DES
- To describe the round keys generation process
- To analyze DES

The Data Encryption Standard (DES) is a symmetric-key block cipher published by the National Institute of Standards and Technology (NIST).

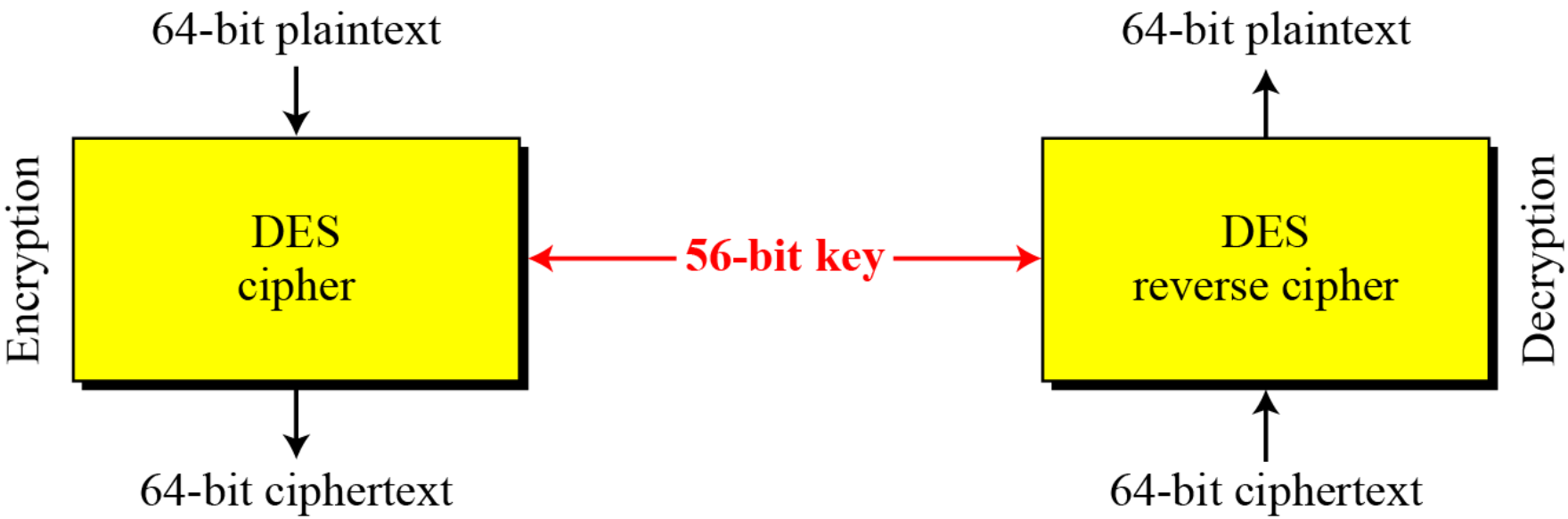
In 1973, NIST published a request for proposals for a national symmetric-key cryptosystem.

A proposal from IBM, a modification of a project called Lucifer, was accepted as DES.

DES was published in the Federal Register in March 1975 as a draft of the Federal Information Processing Standard (FIPS).

DES is a block cipher, as shown in Figure 6.1.

Figure 6.1 *Encryption and decryption with DES*



The encryption process is made of two permutations (P-boxes), which we call initial and final permutations, and sixteen Feistel rounds.

Topics discussed in this section:

Figure 6.2 *General structure of DES*

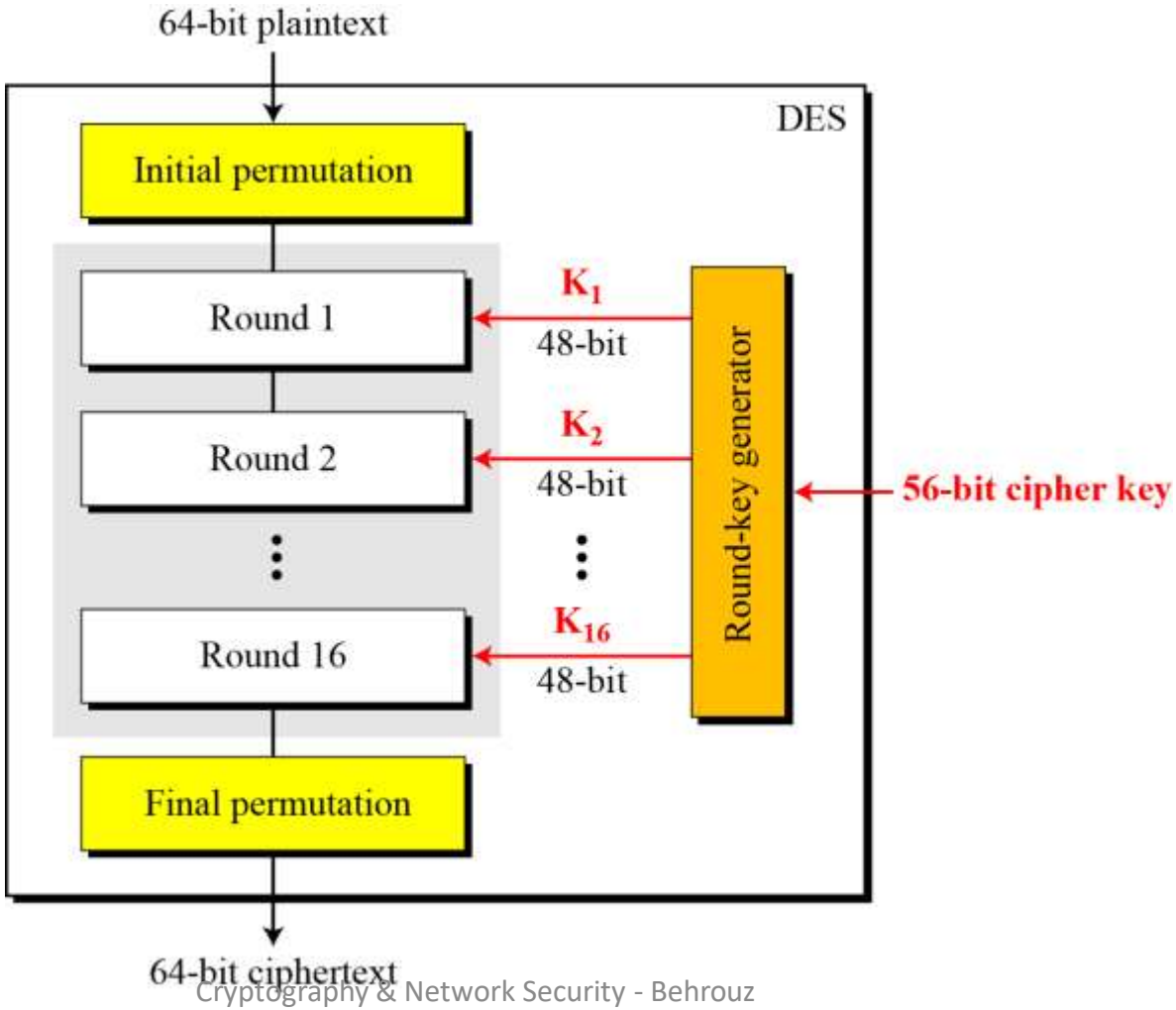


Figure 6.3 *Initial and final permutation steps in DES*

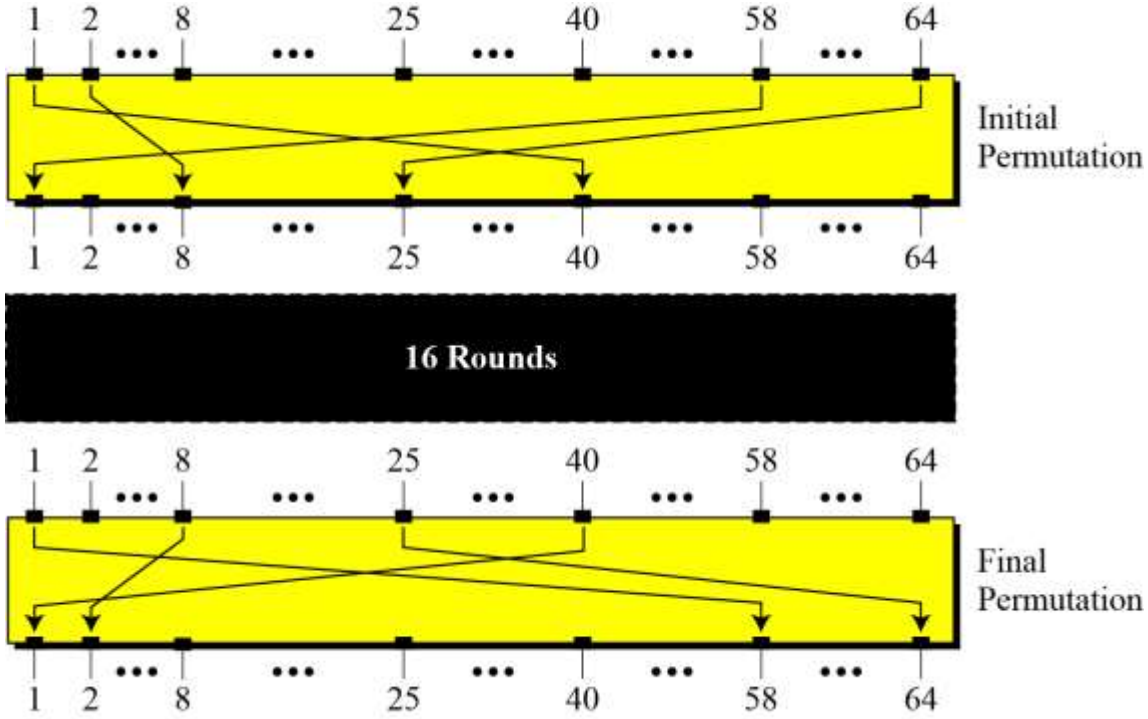


Table 6.1 *Initial and final permutation tables*

Initial Permutation								*Final Permutation*							
58	50	42	34	26	18	10	02	40	08	48	16	56	24	64	32
60	52	44	36	28	20	12	04	39	07	47	15	55	23	63	31
62	54	46	38	30	22	14	06	38	06	46	14	54	22	62	30
64	56	48	40	32	24	16	08	37	05	45	13	53	21	61	29
57	49	41	33	25	17	09	01	36	04	44	12	52	20	60	28
59	51	43	35	27	19	11	03	35	03	43	11	51	19	59	27
61	53	45	37	29	21	13	05	34	02	42	10	50	18	58	26
63	55	47	39	31	23	15	07	33	01	41	09	49	17	57	25

Note

The initial and final permutations are straight P-boxes that are inverses of each other.
They have no cryptography significance in DES.

DES uses 16 rounds. Each round of DES is a Feistel cipher.(the structure consist of identical rounds of processing)

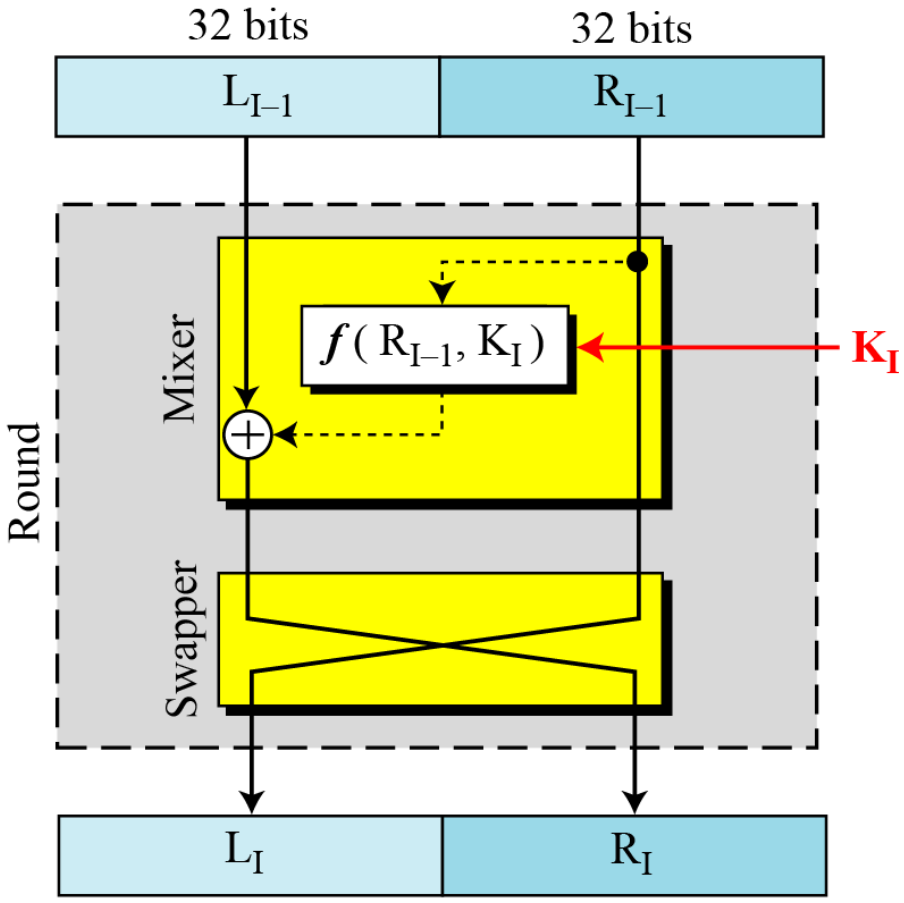


Figure 6.4
A round in DES (encryption site)

DES Function

The heart of DES is the DES function. The DES function applies a 48-bit key to the rightmost 32 bits to produce a 32-bit output.

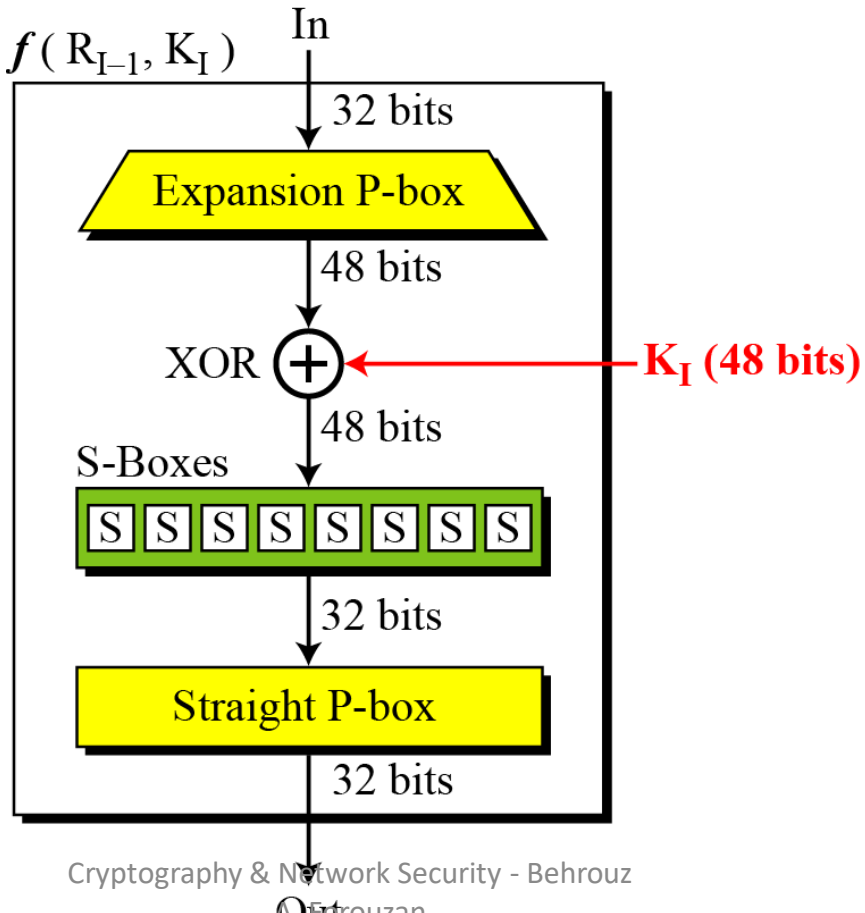
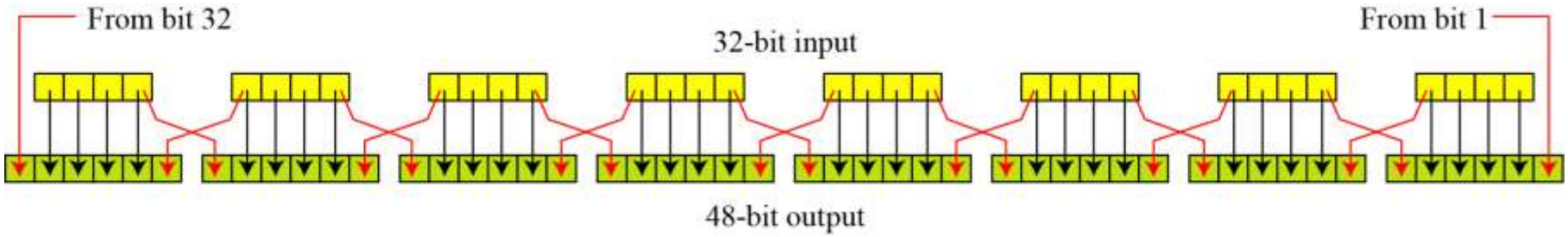


Figure 6.5
DES function

Expansion P-box

Since R_{I-1} is a 32-bit input and K_I is a 48-bit key, we first need to expand R_{I-1} to 48 bits.

Figure 6.6 *Expansion permutation*



Although the relationship between the input and output can be defined mathematically, DES uses Table 6.2 to define this P-box.

Table 6.6 *Expansion P-box table*

32	01	02	03	04	05
04	05	06	07	08	09
08	09	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	31	31	32	01

Whitener (XOR)

After the expansion permutation, DES uses the XOR operation on the expanded right section and the round key. Note that both the right section and the key are 48-bits in length. Also note that the round key is used only in this operation.

S-Boxes

The S-boxes do the real mixing (confusion). DES uses 8 S-boxes, each with a 6-bit input and a 4-bit output. See Figure 6.7.

Figure 6.7 *S-boxes*

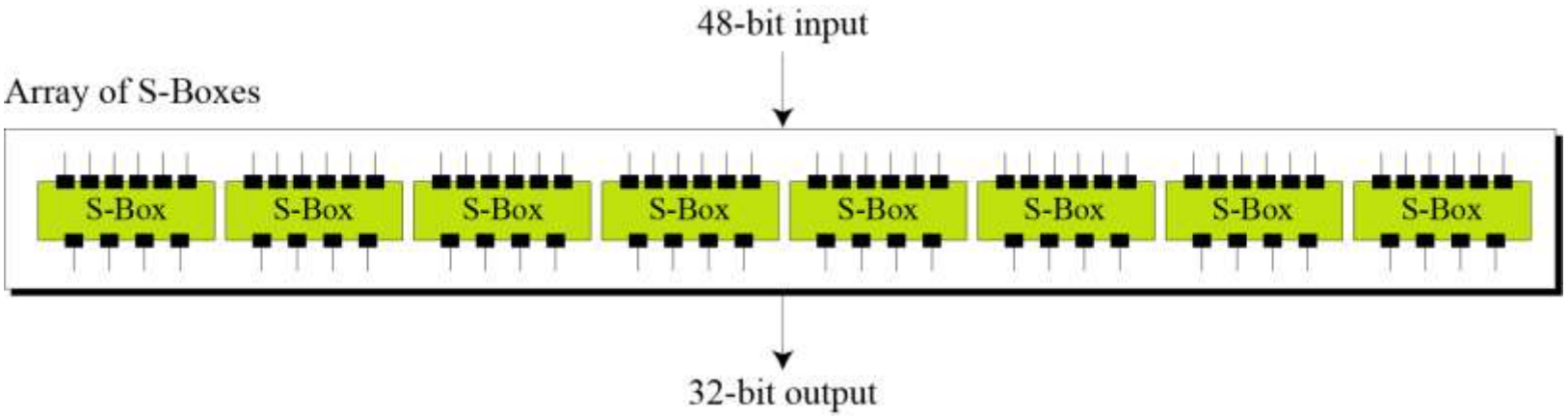
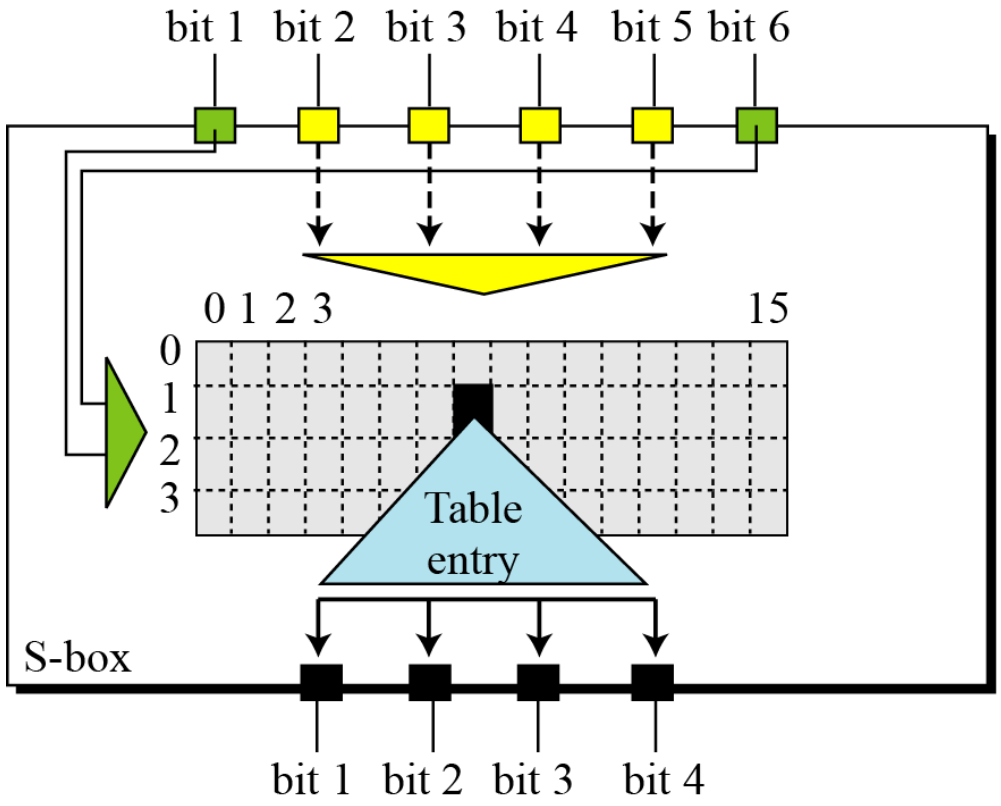


Figure 6.8 *S-box rule*



Straight Permutation

Table 6.11 *Straight permutation table*

16	07	20	21	29	12	28	17
01	15	23	26	05	18	31	10
02	08	24	14	32	27	03	09
19	13	30	06	22	11	04	25

Using mixers and swappers, we can create the cipher and reverse cipher, each having 16 rounds.

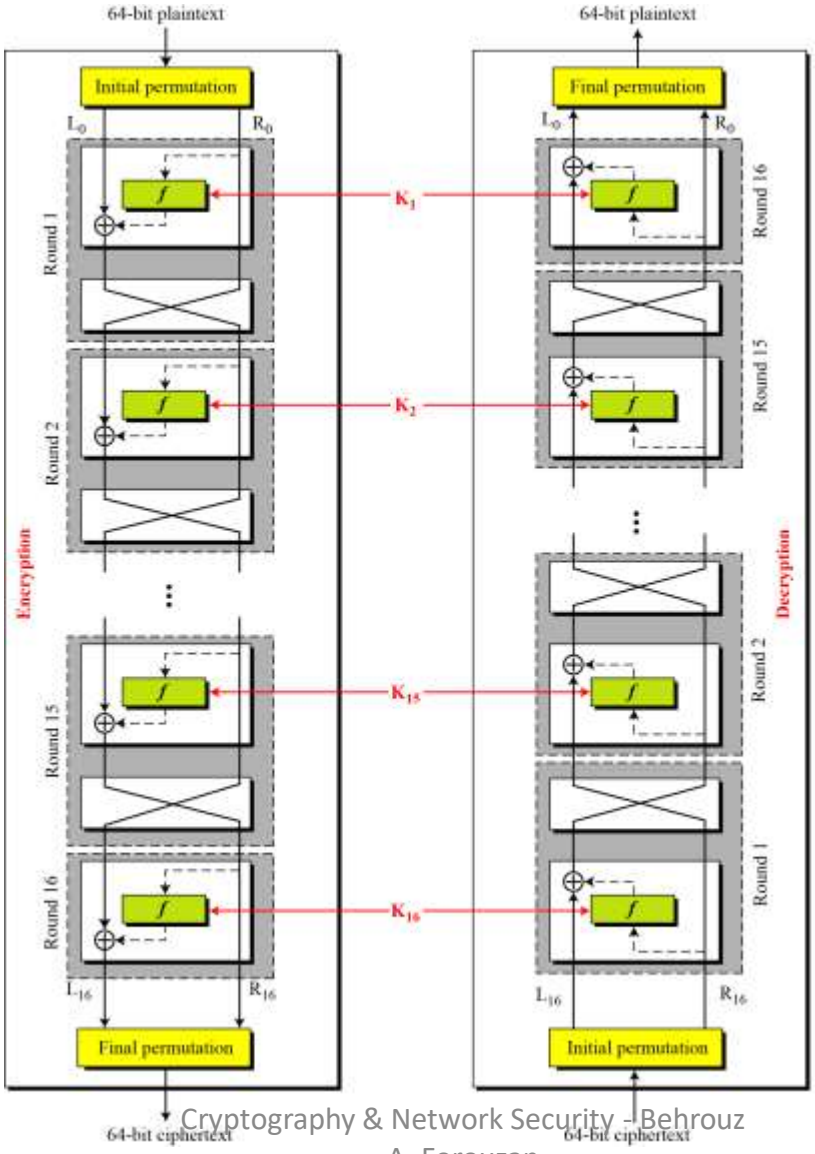
First Approach

To achieve this goal, one approach is to make the last round (round 16) different from the others; it has only a mixer and no swapper.

Note

In the first approach, there is no swapper in the last round.

Figure 6.9 *DES cipher and reverse cipher for the first approach*



Algorithm 6.1 *Pseudocode for DES cipher*

```
Cipher (plainBlock[64], RoundKeys[16, 48], cipherBlock[64])
{
    permute (64, 64, plainBlock, inBlock, InitialPermutationTable)
    split (64, 32, inBlock, leftBlock, rightBlock)
    for (round = 1 to 16)
    {
        mixer (leftBlock, rightBlock, RoundKeys[round])
        if (round!=16) swapper (leftBlock, rightBlock)
    }
    combine (32, 64, leftBlock, rightBlock, outBlock)
    permute (64, 64, outBlock, cipherBlock, FinalPermutationTable)
}
```

Algorithm 6.1 *Pseudocode for DES cipher* (Continued)

```
mixer (leftBlock[48], rightBlock[48], RoundKey[48])
{
    copy (32, rightBlock, T1)
    function (T1, RoundKey, T2)
    exclusiveOr (32, leftBlock, T2, T3)
    copy (32, T3, rightBlock)
}
swapper (leftBlock[32], rigthBlock[32])
{
    copy (32, leftBlock, T)
    copy (32, rightBlock, leftBlock)
    copy (32, T, rightBlock)
}
```

Algorithm 6.1 *Pseudocode for DES cipher* (Continued)

```
function (inBlock[32], RoundKey[48], outBlock[32])
{
    permute (32, 48, inBlock, T1, ExpansionPermutationTable)
    exclusiveOr (48, T1, RoundKey, T2)
    substitute (T2, T3, SubstituteTables)
    permute (32, 32, T3, outBlock, StraightPermutationTable)
}
```

Alternative Approach

We can make all 16 rounds the same by including one swapper to the 16th round and add an extra swapper after that (two swappers cancel the effect of each other).

Key Generation

The round-key generator creates sixteen 48-bit keys out of a 56-bit cipher key.

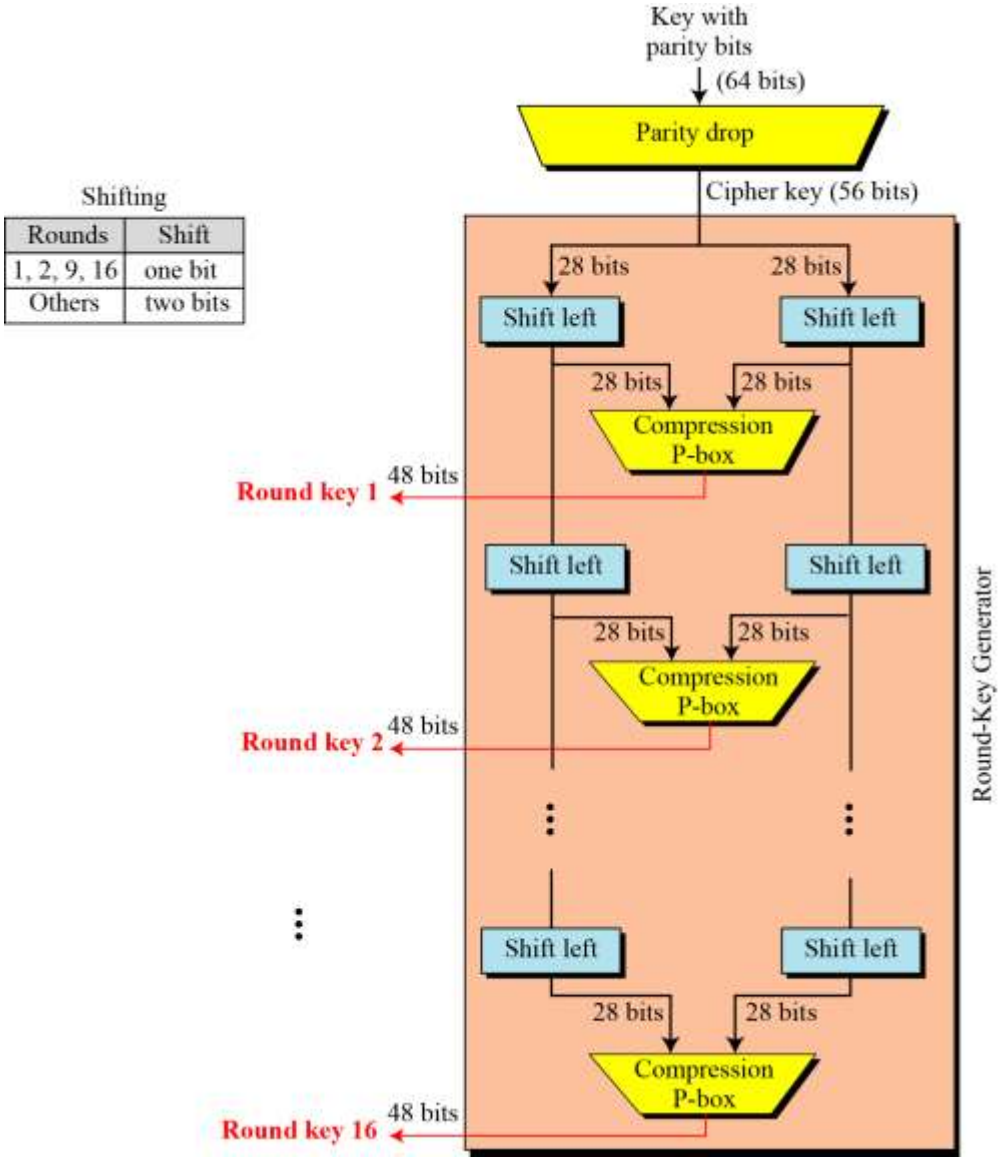


Figure 6.10
Key generation

Table 6.12 *Parity-bit drop table*

57	49	41	33	25	17	09	01
58	50	42	34	26	18	10	02
59	51	43	35	27	19	11	03
60	52	44	36	63	55	47	39
31	23	15	07	62	54	46	38
30	22	14	06	61	53	45	37
29	21	13	05	28	20	12	04

Table 6.13 *Number of bits shifts*

Round	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Bit shifts	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Table 6.14 *Key-compression table*

14	17	11	24	01	05	03	28
15	06	21	10	23	19	12	04
26	08	16	07	27	20	13	02
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	53	46	42	50	36	29	32

Algorithm 6.2 *Algorithm for round-key generation*

```
Key_Generator (keyWithParities[64], RoundKeys[16, 48], ShiftTable[16])
{
    permute (64, 56, keyWithParities, cipherKey, ParityDropTable)
    split (56, 28, cipherKey, leftKey, rightKey)
    for (round = 1 to 16)
    {
        shiftLeft (leftKey, ShiftTable[round])
        shiftLeft (rightKey, ShiftTable[round])
        combine (28, 56, leftKey, rightKey, preRoundKey)
        permute (56, 48, preRoundKey, RoundKeys[round], KeyCompressionTable)
    }
}
```

Critics have used a strong magnifier to analyze DES. Tests have been done to measure the strength of some desired properties in a block cipher.

Topics discussed in this section:

Two desired properties of a block cipher are the avalanche effect and the completeness.

Example 6.7

To check the avalanche effect in DES, let us encrypt two plaintext blocks (with the same key) that differ only in one bit and observe the differences in the number of bits in each round.

Plaintext: 0000000000000000 Key: 22234512987ABB23
Ciphertext: 4789FD476E82A5F1

Plaintext: 0000000000000001 Key: 22234512987ABB23
Ciphertext: 0A4ED5C15A63FEA3

Completeness effect

Completeness effect means that each bit of the ciphertext needs to depend on many bits on the plaintext.

S-Boxs

The design provides confusion and diffusion of bits from each round to the next.

P-Boxes

They provide diffusion of bits.

Number of Rounds

DES uses sixteen rounds of Feistel ciphers. the ciphertext is thoroughly a random function of plaintext and ciphertext.

During the last few years critics have found some weaknesses in DES.

Weaknesses in Cipher Design

1. *Weaknesses in S-boxes*
2. *Weaknesses in P-boxes*
3. *Weaknesses in Key*

Table 6.18 *Weak keys*

Keys before parities drop (64 bits)	*Actual key (56 bits)*
0101 0101 0101 0101	0000000 0000000
1F1F 1F1F 0E0E 0E0E	0000000 FFFFFFF
E0E0 E0E0 F1F1 F1F1	FFFFFFF 0000000
FEFE FEFE FEFE FEFE	FFFFFFF FFFFFFF

Figure 6.11 *Double encryption and decryption with a weak key*

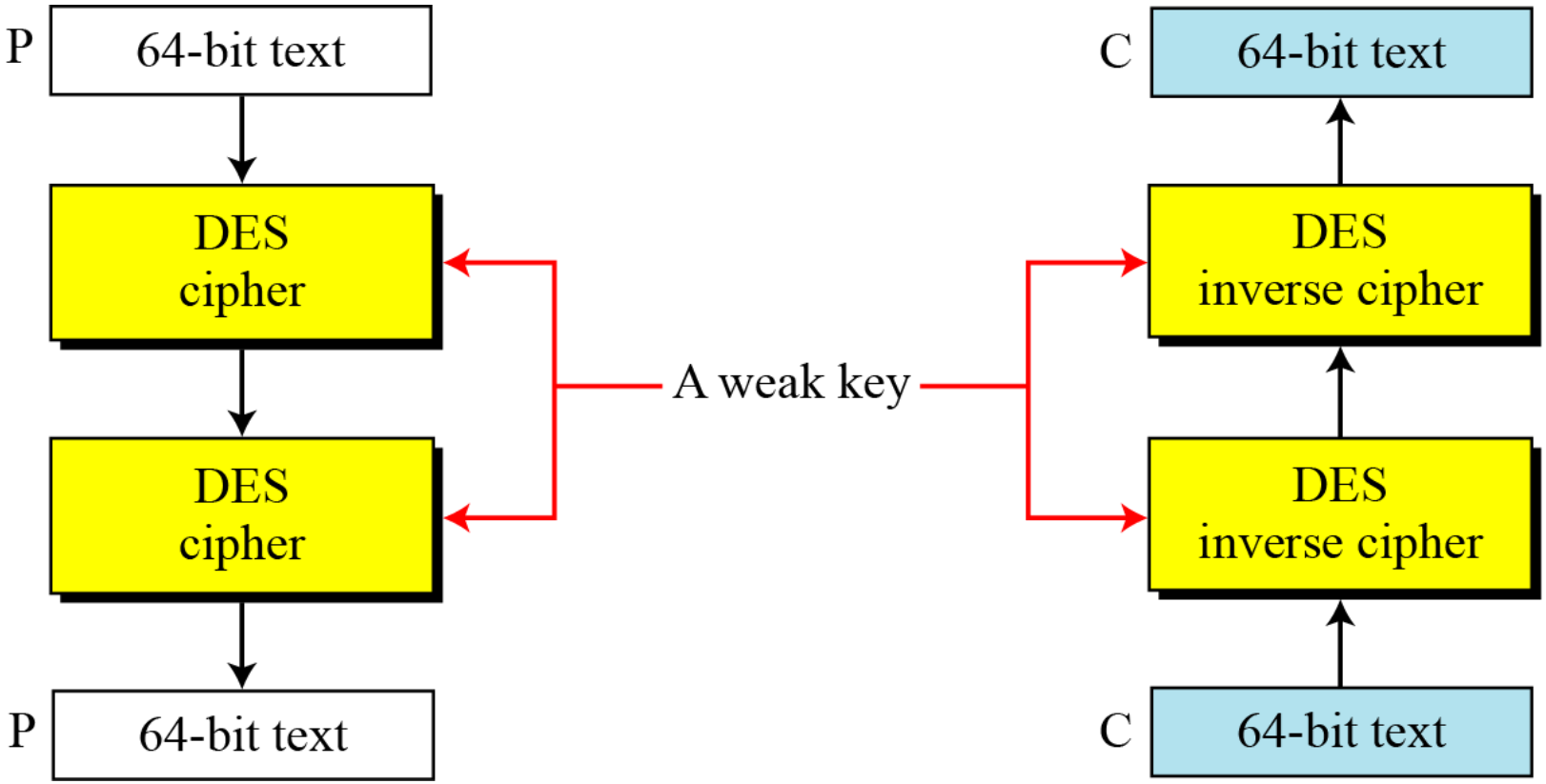
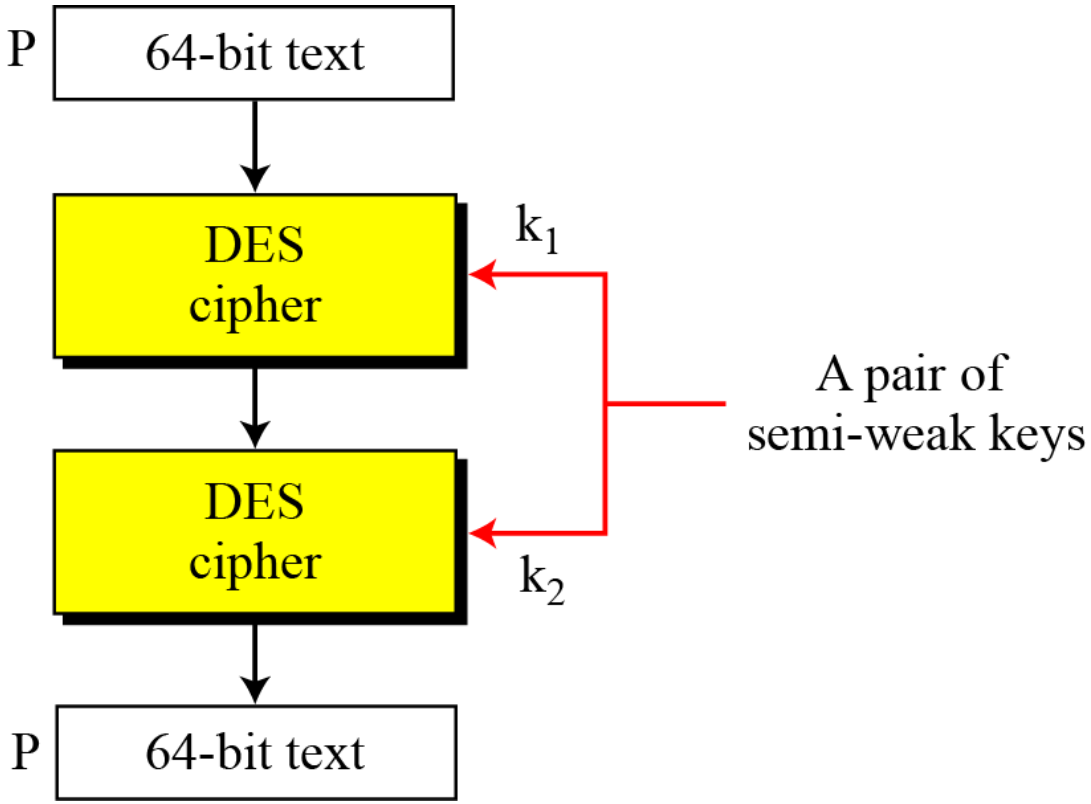


Figure 6.12 *A pair of semi-weak keys in encryption and decryption*



The major criticism of DES regards its key length. Fortunately DES is not a group. This means that we can use double or triple DES to increase the key size.

Topics discussed in this section:

The first approach is to use double DES (2DES).

Meet-in-the-Middle Attack

However, using a known-plaintext attack called meet-in-the-middle attack proves that double DES improves this vulnerability slightly (to 2^{57} tests), but not tremendously (to 2^{112}).

Figure 6.14 *Meet-in-the-middle attack for double DES*

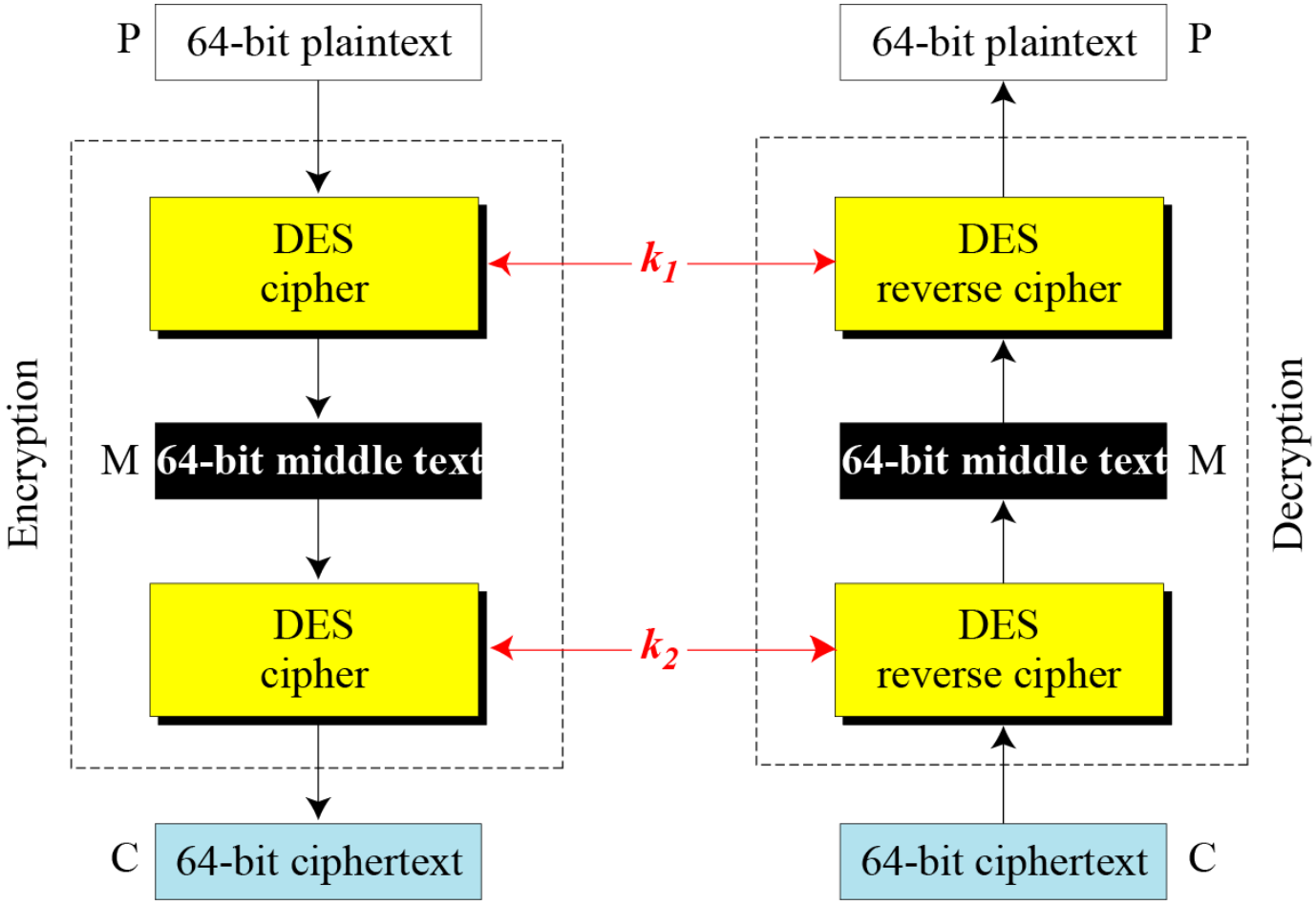


Figure 6.15 *Tables for meet-in-the-middle attack*

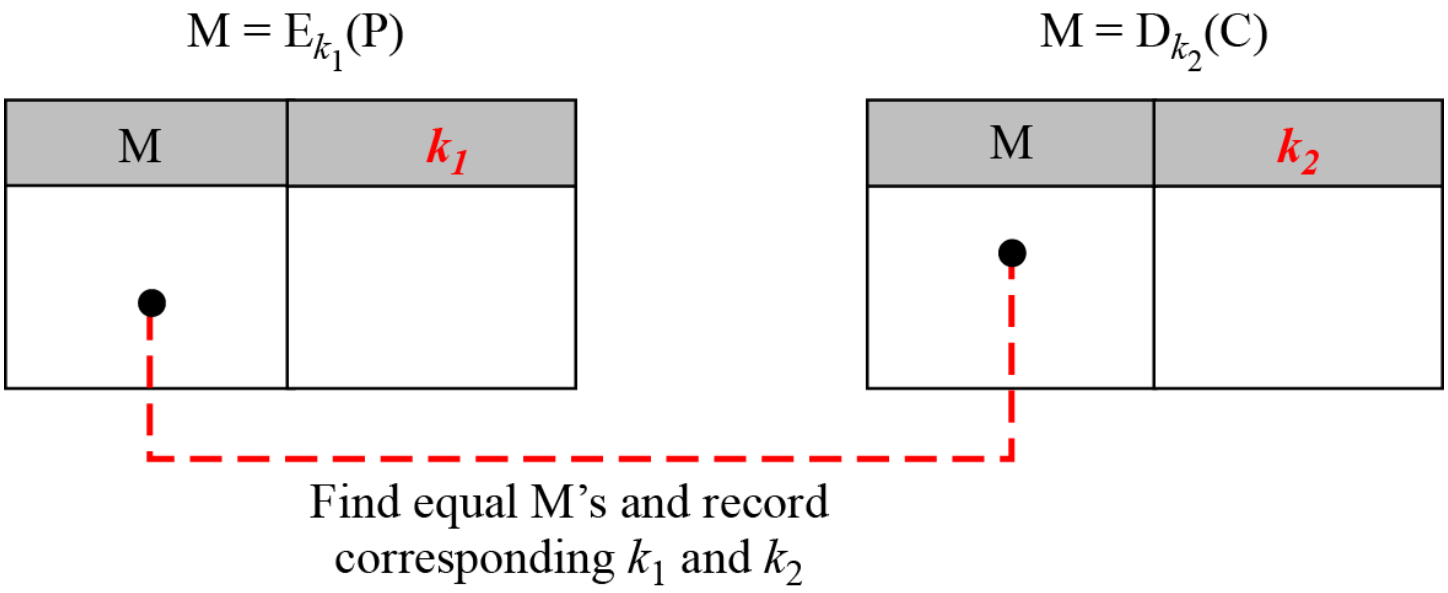
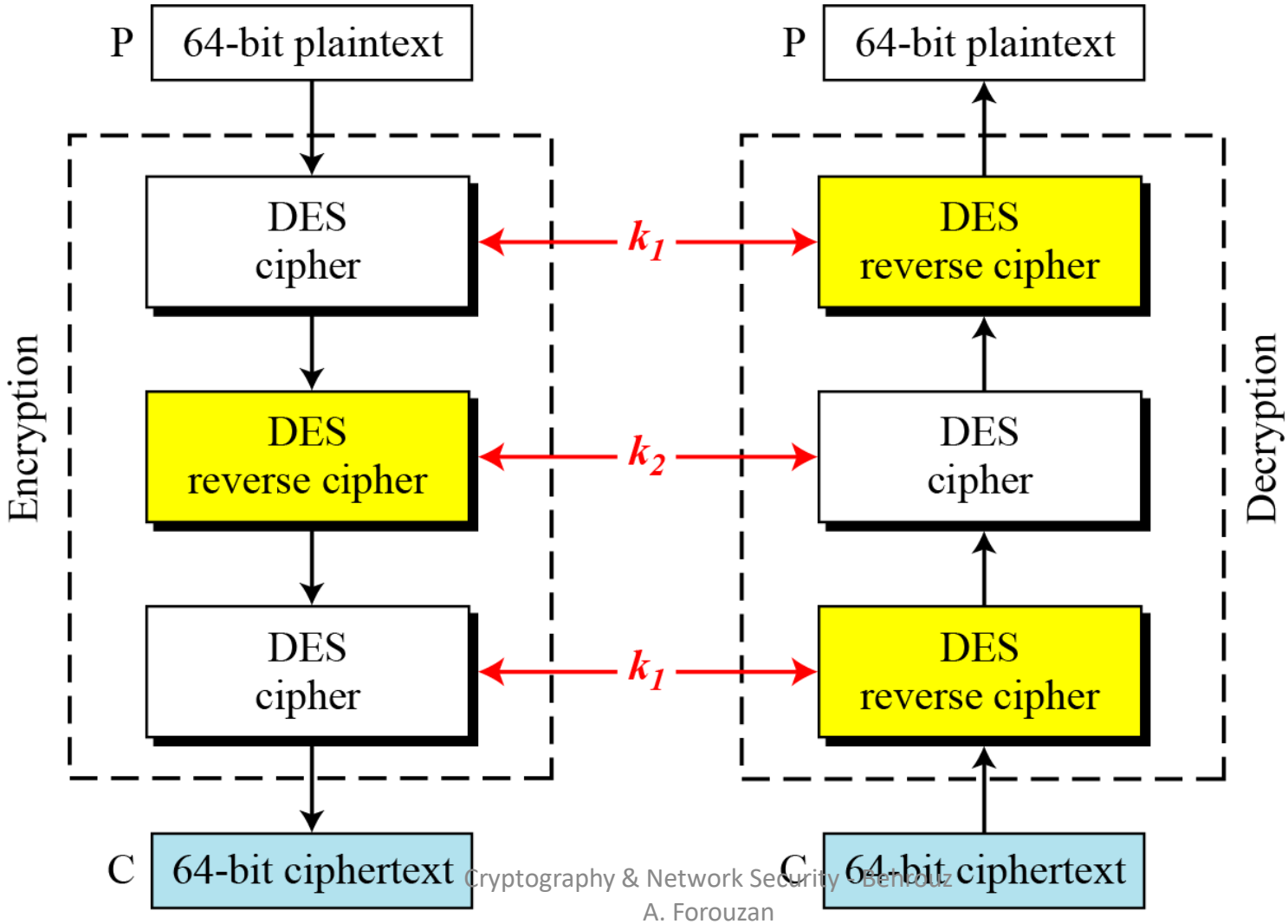


Figure 6.16 *Triple DES with two keys*



Triple DES with Three Keys

The possibility of known-plaintext attacks on triple DES with two keys has enticed some applications to use triple DES with three keys. Triple DES with three keys is used by many applications such as PGP .

DES, as the first important block cipher, has gone through much scrutiny. Among the attempted attacks, three are of interest: brute-force, differential cryptanalysis, and linear cryptanalysis.

Topics discussed in this section:

We have discussed the weakness of short cipher key in DES. Combining this weakness with the key complement weakness, it is clear that DES can be broken using 2^{55} encryptions.

It has been revealed that the designers of DES already knew about this type of attack and designed S-boxes and chose 16 as the number of rounds to make DES specifically resistant to this type of attack.

Note

We show an example of DES differential cryptanalysis in Appendix N.

Linear cryptanalysis is newer than differential cryptanalysis. DES is more vulnerable to linear cryptanalysis than to differential cryptanalysis. S-boxes are not very resistant to linear cryptanalysis. It has been shown that DES can be broken using 2^{43} pairs of known plaintexts. However, from the practical point of view, finding so many pairs is very unlikely.

Note

We show an example of DES linear cryptanalysis in Appendix N.

Advanced Encryption Standard (AES)

Objectives

- To review a short history of AES
- To define the basic structure of AES
- To define the transformations used by AES
- To define the key expansion process
- To discuss different implementations

7-1 INTRODUCTION

The Advanced Encryption Standard (AES) is a symmetric-key block cipher published by the National Institute of Standards and Technology (NIST) in December 2001.

Topics discussed in this section:

The criteria defined by NIST for selecting AES fall into three areas:

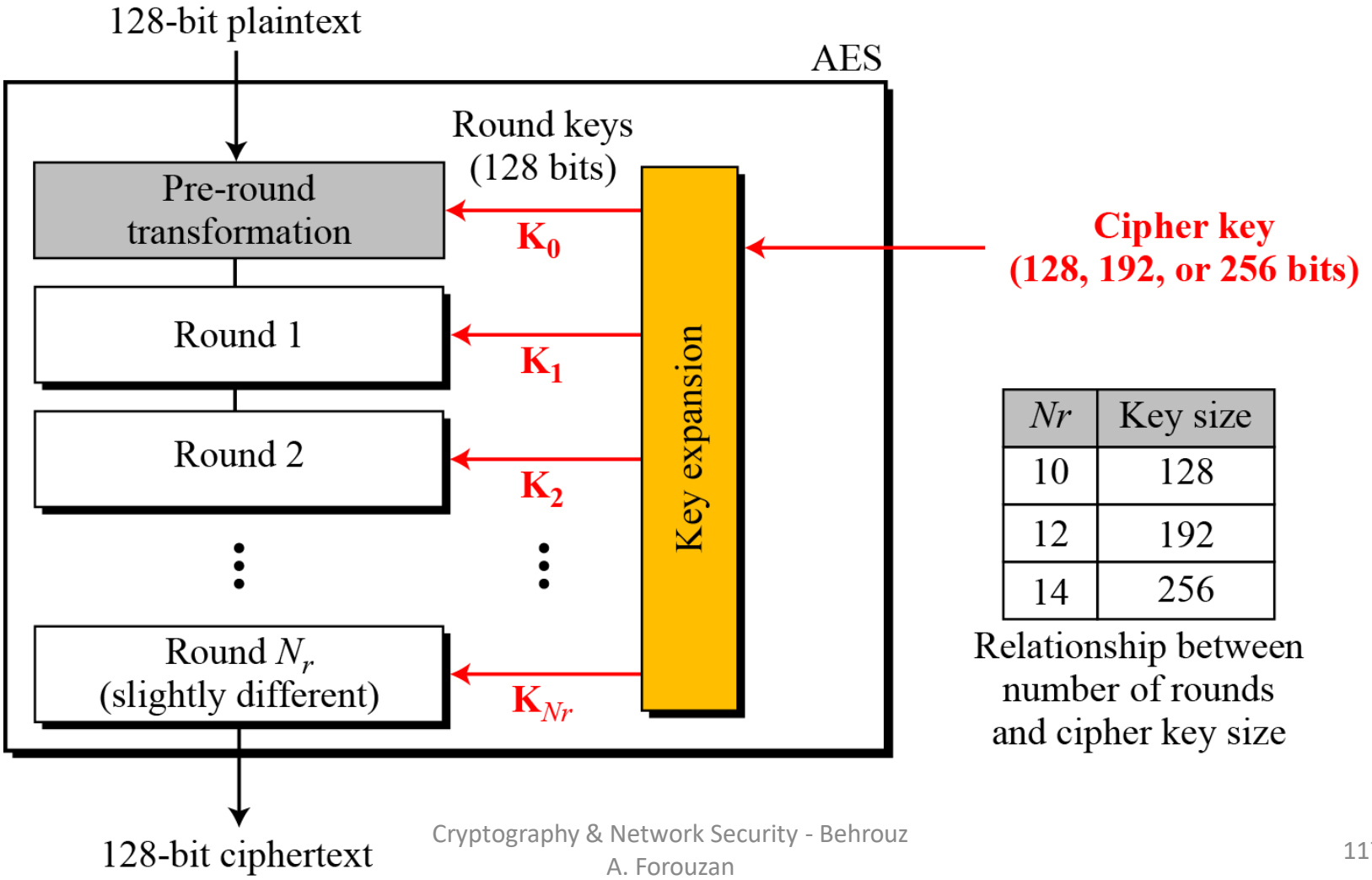
1. *Security*
2. *Cost*
3. *Implementation.*

AES is a non-Feistel cipher that encrypts and decrypts a data block of 128 bits. It uses 10, 12, or 14 rounds. The key size, which can be 128, 192, or 256 bits, depends on the number of rounds.

Note

AES has defined three versions, with 10, 12, and 14 rounds.
Each version uses a different cipher key size (128, 192, or 256), but the round keys are always 128 bits.

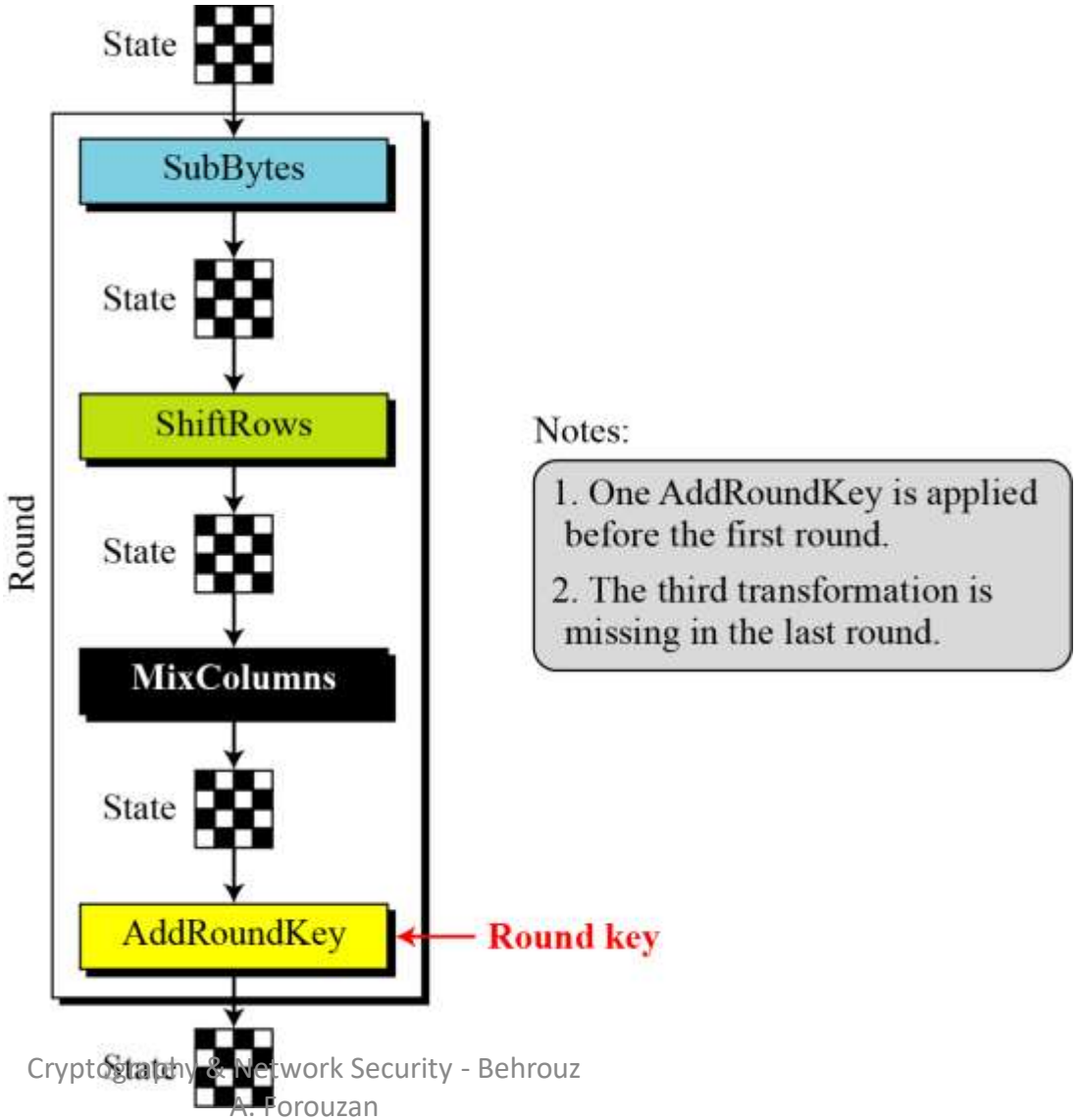
Figure 7.1 *General design of AES encryption cipher*



Nr	Key size
10	128
12	192
14	256

Relationship between number of rounds and cipher key size

Figure 7.5 *Structure of each round at the encryption site*



7-2 TRANSFORMATIONS

To provide security, AES uses four types of transformations: substitution, permutation, mixing, and key-adding.

Topics discussed in this section:

AES, like DES, uses substitution. AES uses two invertible transformations.

SubBytes

The first transformation, SubBytes, is used at the encryption site. To substitute a byte, we interpret the byte as two hexadecimal digits.

Note

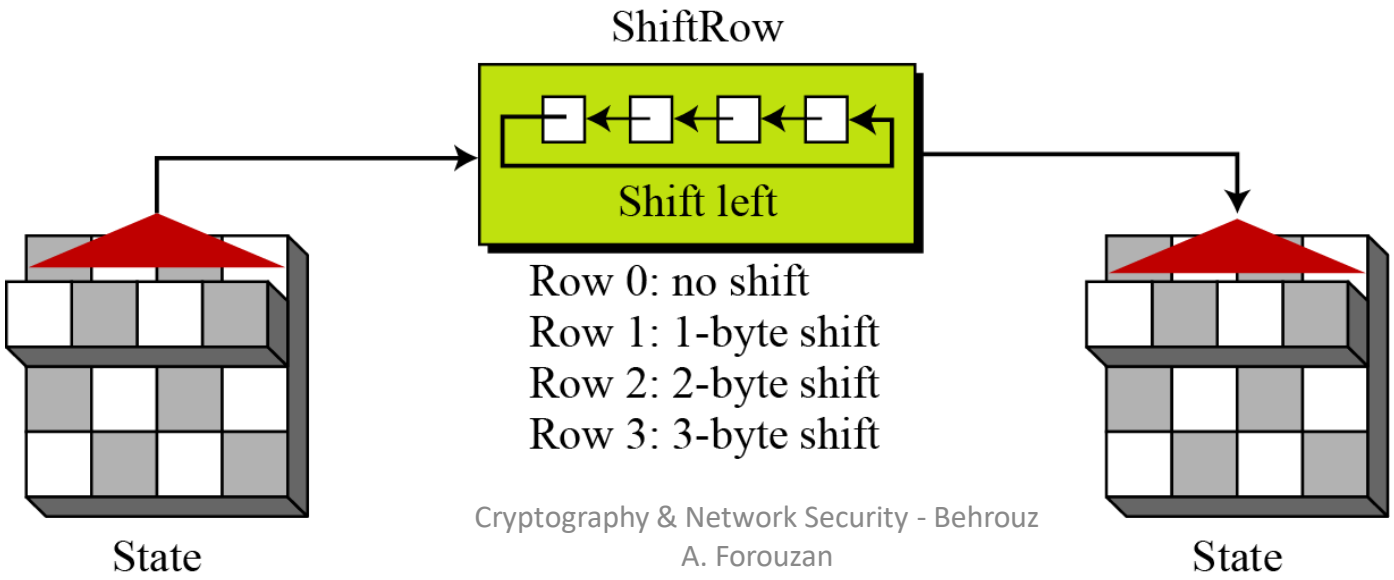
The SubBytes operation involves 16 independent byte-to-byte transformations.

Another transformation found in a round is shifting, which permutes the bytes.

ShiftRows

In the encryption, the transformation is called ShiftRows.

Figure 7.9 *ShiftRows transformation*

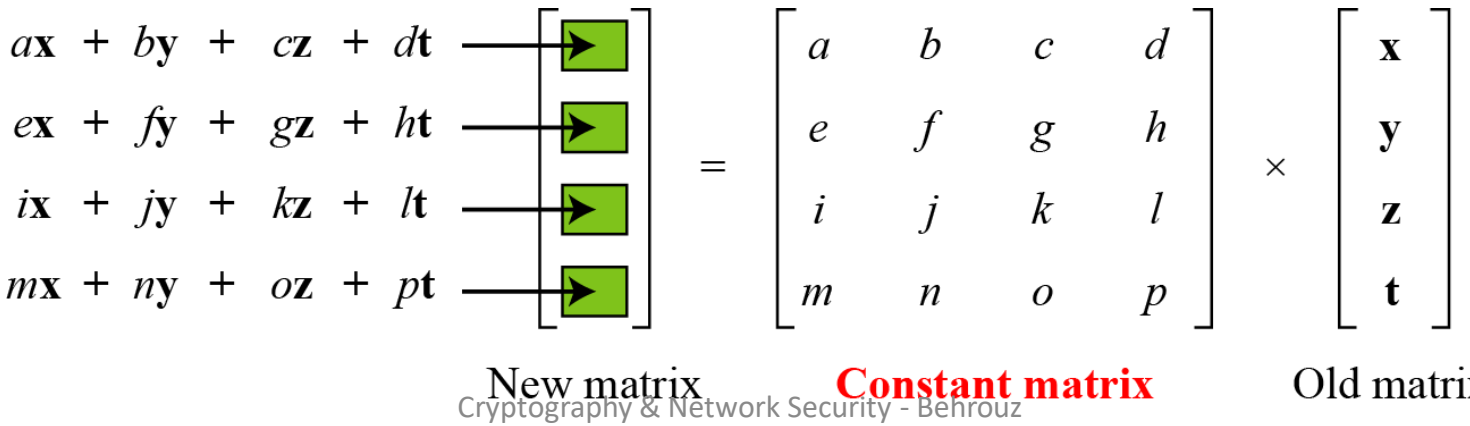


InvShiftRows

In the decryption, the transformation is called InvShiftRows and the shifting is to the right.

We need an interbyte transformation that changes the bits inside a byte, based on the bits inside the neighboring bytes. We need to mix bytes to provide diffusion at the bit level.

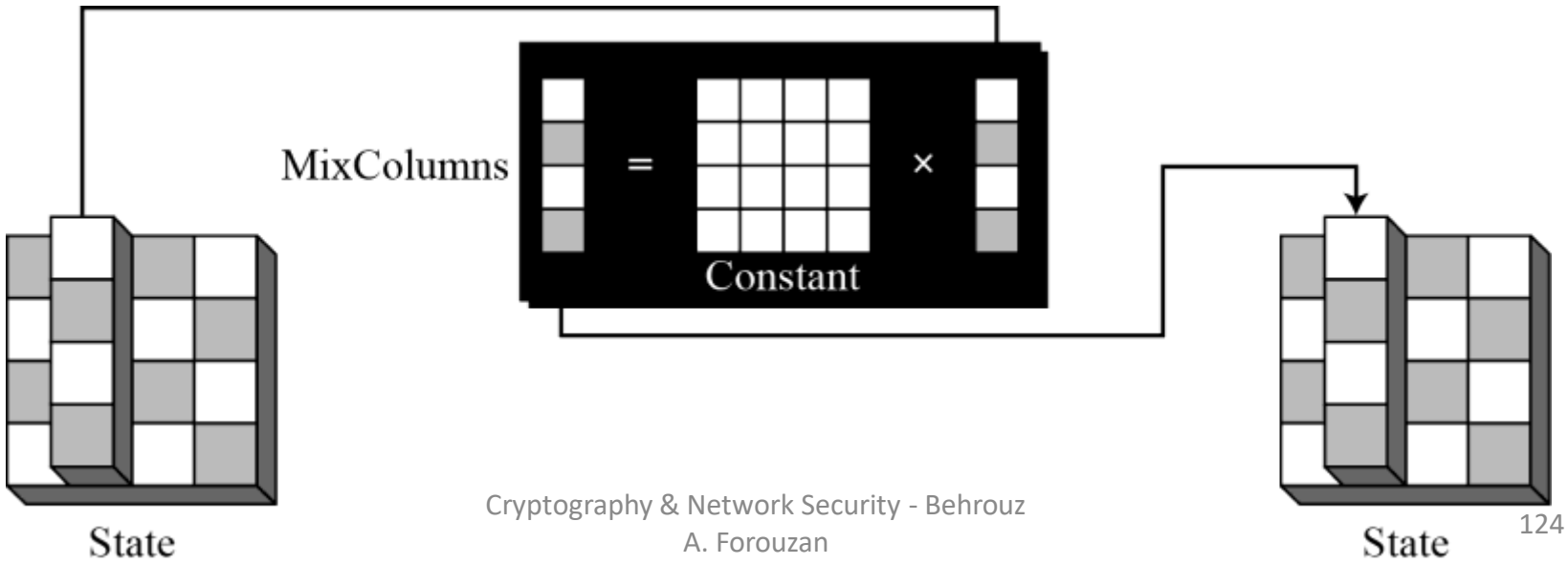
Figure 7.11 *Mixing bytes using matrix multiplication*



MixColumns

The MixColumns transformation operates at the column level; it transforms each column of the state to a new column.

Figure 7.13 *MixColumns transformation*



InvMixColumns

The InvMixColumns transformation is basically the same as the MixColumns transformation.

Note

The MixColumns and InvMixColumns transformations are inverses of each other.

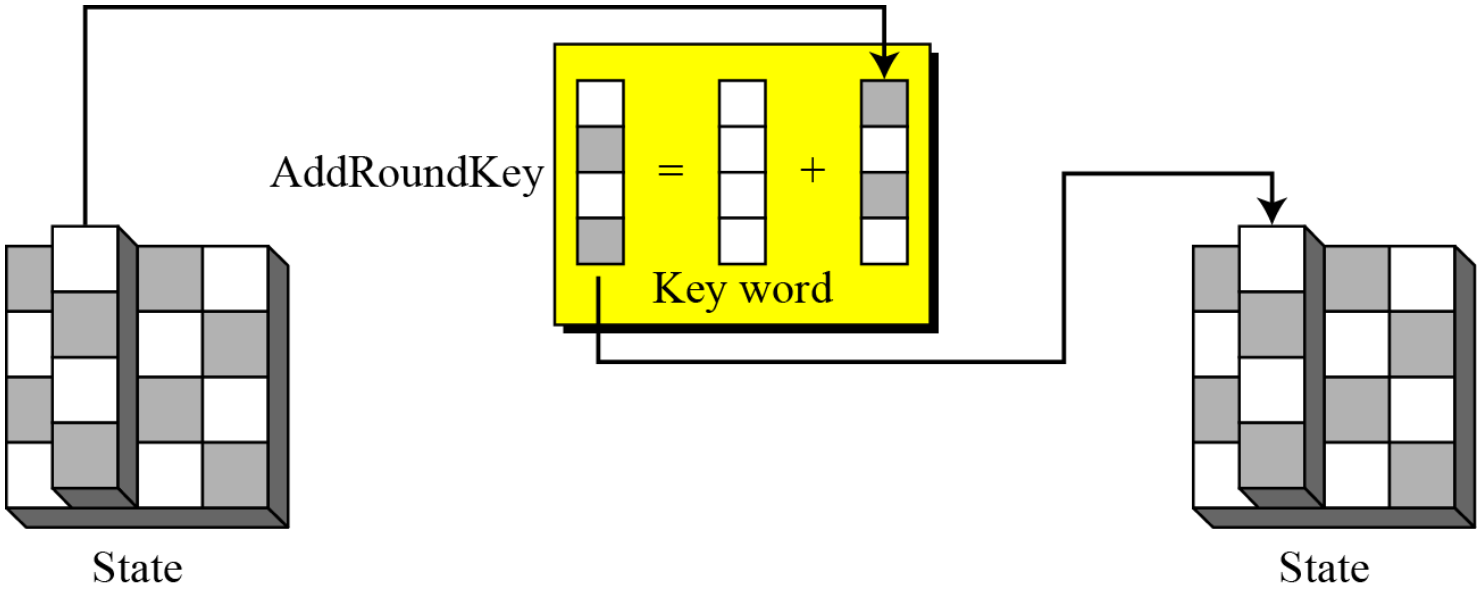
AddRoundKey

AddRoundKey proceeds one column at a time. AddRoundKey adds a round key word with each state column matrix; the operation in AddRoundKey is matrix addition.

Note

The AddRoundKey transformation is the inverse of itself.

Figure 7.15 *AddRoundKey transformation*



Algorithm 7.4 *Pseudocode for AddRoundKey transformation*

```
AddRoundKey (S)
{
    for (c = 0 to 3)
        s_c ← s_c ⊕ w_round + 4c
}
```

7-6 ANALYSIS OF AES

This section is a brief review of the three characteristics of AES.

Topics discussed in this section:

AES was designed after DES. Most of the known attacks on DES were already tested on AES.

Brute-Force Attack

AES is definitely more secure than DES due to the larger-size key.

Statistical Attacks

Numerous tests have failed to do statistical analysis of the ciphertext.

Differential and Linear Attacks

There are no differential and linear attacks on AES as yet.

AES can be implemented in software, hardware, and firmware. The implementation can use table lookup process or routines that use a well-defined algebraic structure.

The algorithms used in AES are so simple that they can be easily implemented using cheap processors and a minimum amount of memory.