基礎密碼學(二)

祕密金鑰密碼系統 (SECRET KEY CRYPTOSYSTEMS)

近代密碼系統

- 古代保密技術並不能保證系統的安全
- 應用上也受到相當大之限制
  - Ex: 中文系統很難用換位法以達到保密效果，如將「*今天我不回家*」之明文換位置成「*回不天家我今*」之密文，該密文一看就可重組成明文而不需藉其他工具來破解
- 需要使用一些以“*位元*”爲處理單位的*加密系統*

近代密碼系統 (Cont.)

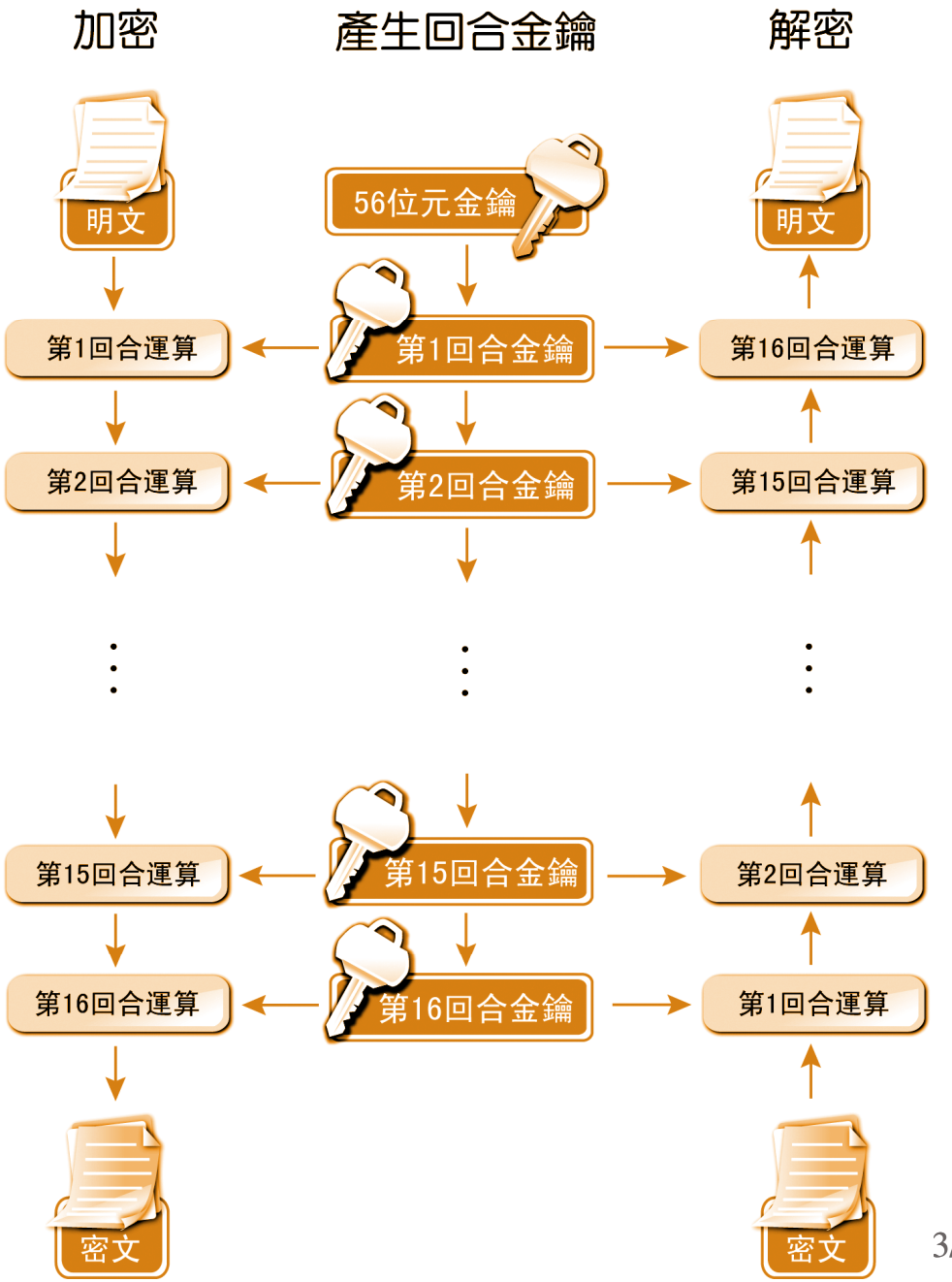
DES (Data Encryption Standard)

- 對稱式加密系統之代表
- 1970年代中期由IBM公司所發展
- 美國國家標準局公佈爲資料加密標準的一種**區塊加密法**(Block Cipher)
- DES 屬於區塊加密法，而區塊加密法就是對一定大小的明文或密文來做加密或解密動作
- 每次加密解密的區塊大小均爲 ***64 位元***(bits)

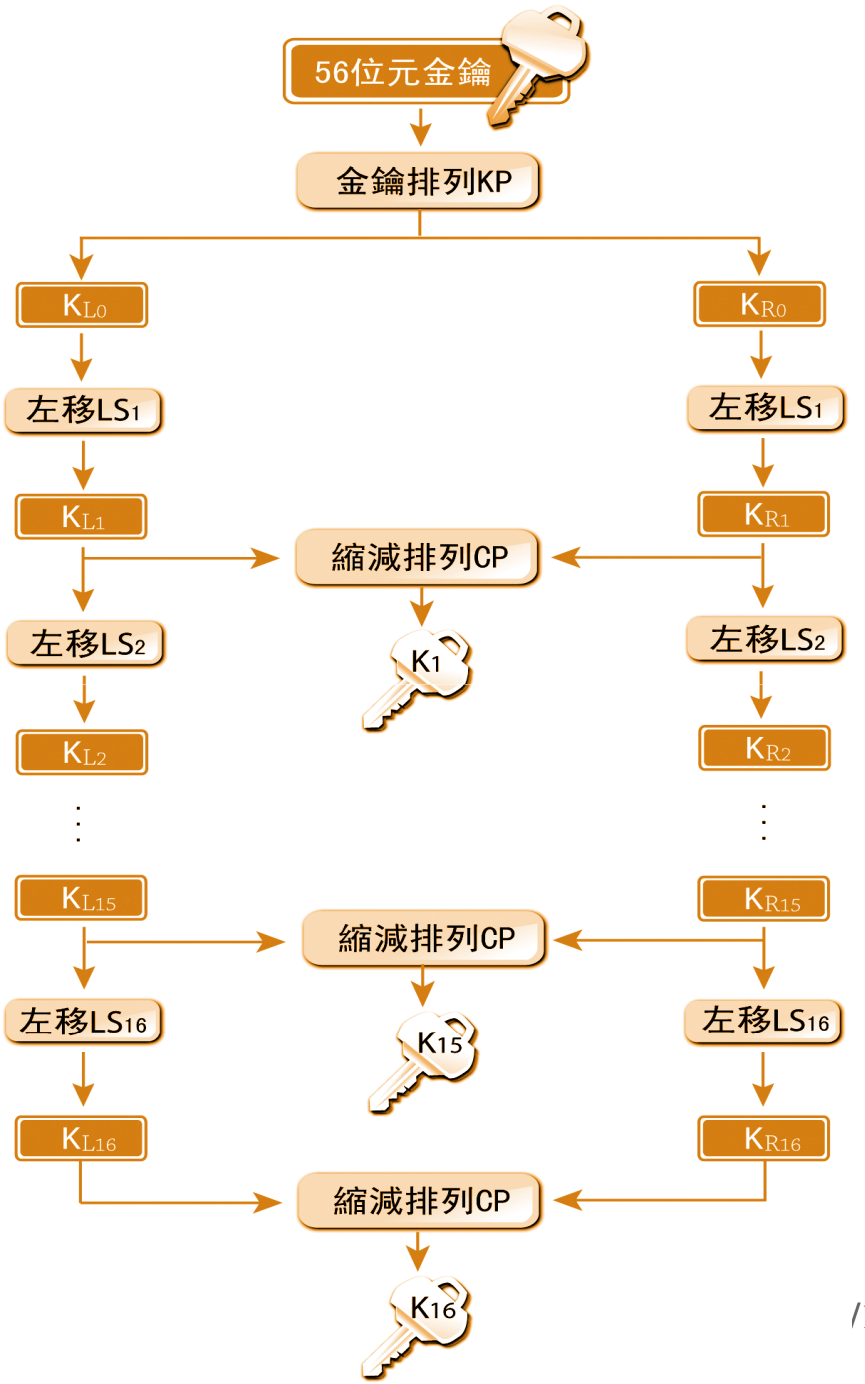
近代密碼系統 (Cont.)

- 就一般資料而言，資料通常大於64位元。只要將明文或密文中每64位元當作一個區塊加以切割，再將每個區塊做加密或解密即可
- 最後一個區塊大小可能小於64位元，此時就要將此區塊附加“0”位元，直到區塊大小成爲64位元爲止
- DES 所用**加密**或**解密金鑰**也是***64位元***大小。但其中有**8個位元是用來做錯誤更正**，**真正的金鑰有效長度只有 56 位元**

DES 加解密架構


加密
產生回合金鑰
解密
明文
56位元金鑰
明文
第1回合運算
第1回合金鑰
第16回合運算
第2回合運算
第2回合金鑰
第15回合運算
第15回合運算
第15回合金鑰
第2回合運算
第16回合運算
第16回合金鑰
第1回合運算
密文
密文

DES之回合金鑰產生運算過程



DES回合金鑰的產生

- 假設選定[science]作爲祕密金鑰 ([science]之ASCII編碼爲：[73 63 69 65 6e 63 65]$_{16}$)
- 一個字母需要8個位元來作編碼，所以[science]其二進位表示法爲：
  - [(01110011) (01100011) (01101001) (01100101) (01101110) (01100011) (01100101)]$_2$共56個位元
- 加入奇同位檢查碼後之結果如下:
  - [(01110011) (10110000) (11011010) (00101100) (01010111) (01110011) (10001100) (11001011)]共64個位元

ASCII 內碼表

字母	十進位內碼	十六進位內碼
a	97	61
b	98	62
c	99	63
d	100	64
e	101	65
f	102	66
g	103	67
h	104	68
i	105	69
j	106	6A
k	107	6B
l	108	6C
m	109	6D

字母	十進位內碼	十六進位內碼
n	110	6E
o	111	6F
p	112	70
q	113	71
r	114	72
s	115	73
t	116	74
u	117	75
v	118	76
w	119	77
x	120	78
y	121	79
z	122	7A

DES回合金鑰的產生 (Cont.)

DES的金鑰排列表-KP

57	49	41	33	25	17	9	1	58	50	42	34	26	18
10	2	59	51	43	35	27	19	11	3	60	52	44	36
63	55	47	39	31	23	15	7	62	54	46	38	30	22
14	6	61	53	45	37	29	21	13	5	28	20	12	4

再作初始金鑰排列(KP)，結果如下：

[(11000110) (10110101) (00101011) (00111011) (01010101) (10001100) (11000111)]

DES的左移位數表

回合數	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
左移位數	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

接著，將排列後的結果分成兩半，分別爲K_{L0}及K_{R0}，它們的內容分別爲：

K_{L0}=[11000110 10110101 00101011 0011]

K_{R0}=[10110101 01011000 11001100 0111]

向左循環一位元之後可分別得到：

K_{L1}=[10001101 01101010 01010110 0111]

K_{R1}=[01101010 10110001 10011000 1111]

DES回合金鑰的產生 (Cont.)

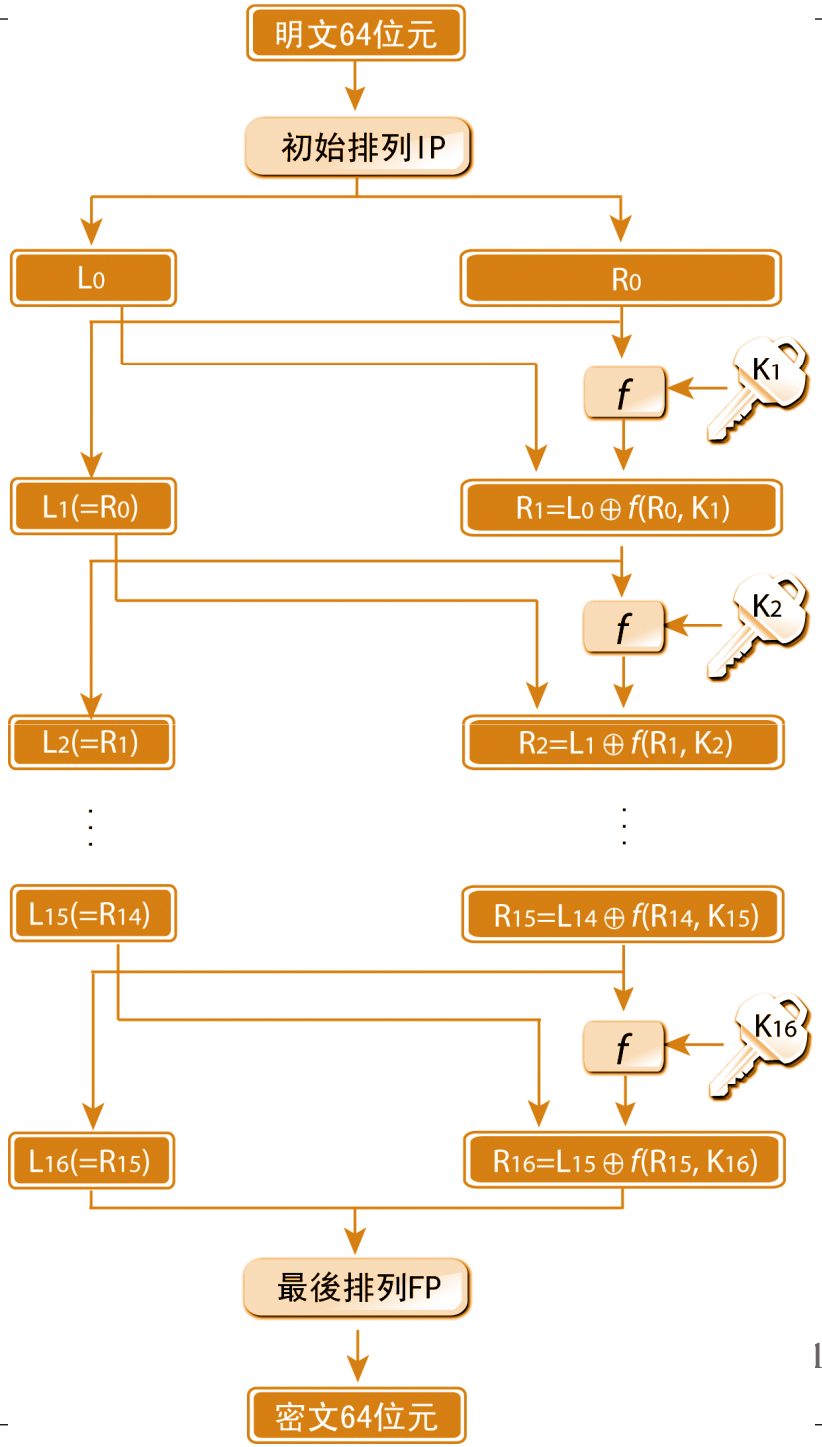
再經過**壓縮排列**後(CP)後，就可得到第一回合的回合金鑰K_1：
K_1=[00101101 11011000 11001110 00110111 01111111 01000000]

14	17	11	24	1	5	3	28	15	6	21	10
23	19	12	4	26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40	51	45	33	48
44	49	39	56	34	53	46	42	50	36	29	32

其餘各回合的回合金鑰可依序用類似的方法求得

位元	1-8	9-16	17-24	32-25	33-40	41-48	49-56	57-64
原始金鑰	73	63	69	65	6e	63	65	
奇同位元碼	72	b1	db	2d	56	72	8d	ca
KP	c6	b5	2b	3b	55	8c	c7	
K_{L_1}	8d	6a	56	7				
K_{R_1}	6a	b1	98	f				
回合金鑰 1	2d	d8	ce	37	7f	40		
回合金鑰 2	c5	37	98	4e	1a	c7		
回合金鑰 3	56	9e	d5	d6	c1	fd		
回合金鑰 4	5f	f8	42	03	9f	c9		
回合金鑰 5	2a	e5	ee	da	b5	31		
回合金鑰 6	f8	45	0f	6b	4f	2c		
回合金鑰 7	61	8b	39	58	79	9a		
回合金鑰 8	85	b8	b7	e5	50	3d		
回合金鑰 9	98	c7	4e	ab	ce	d0		
回合金鑰 10	60	5b	1f	59	e7	13		
回合金鑰 11	25	b9	65	bf	44	0c		
回合金鑰 12	83	6c	f3	c8	73	c6		
回合金鑰 13	fd	67	b0	b4	e2	ad		
回合金鑰 14	d6	97	89	f2	1e	c3		
回合金鑰 15	5b	92	57	9e	a3	3b		
回合金鑰 16	b9	6e	39	a5	45	af		

DES的加密流程


明文64位元
初始排列IP
L0
R0
f
K1
L1(=R0)
R1=L0 ⊕ f(R0, K1)
f
K2
L2(=R1)
R2=L1 ⊕ f(R1, K2)
L15(=R14)
R15=L14 ⊕ f(R14, K15)
f
K16
L16(=R15)
R16=L15 ⊕ f(R15, K16)
最後排列FP
密文64位元

DES 加密法範例

- 假設要對64位元的明文[security]作加密

- 明文[security]的ASCII編碼爲：[73 65 63 75 72 69 74 79]，轉換爲二進位表示爲：
- [01110011 01100101 01100011 01110101 01110010 01101001 01110100 01111001]

DES 加密法範例 (Cont.)

初始排列表（IP）

58	50	42	34	26	18	10	2	60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6	64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1	59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5	63	55	47	39	31	23	15	7

- 明文：
  - [01110011 01100101 01100011 01110101 01110010 01101001 01110100 01111001]
- 經過起始排列(IP)後的結果爲：
  - [11111111 11011001 01001010 10101111 00000000 11111111 10100000 00010101]

DES 加密法範例 (Cont.)

- 將重組後的結果分爲L及R兩半
- L_0：00000000 11111111 10100000 00010101
- R_0：10011010 00100110 00100111 00100100
- 作下列運算

$$L_j = R_{j-1}$$

$$R_j = L_{j-1} \oplus f(R_{j-1}, K_j)$$

$f(R_{j-1}, K_j)$ 函式運算

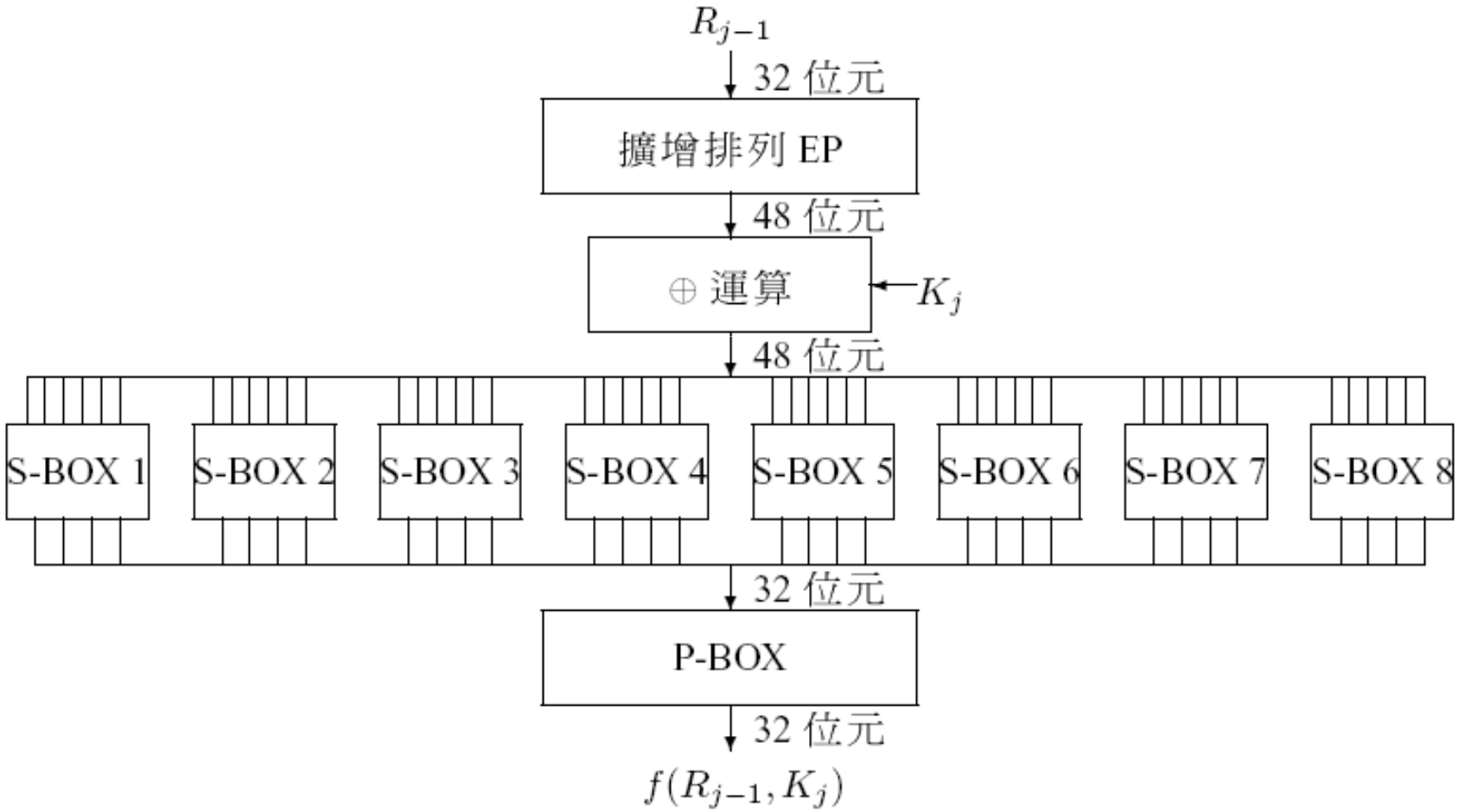
- 將32位元的R_{j-1}做位元擴充成48位元
  - DES 擴增排列表

32	1	2	3	4	5	4	5	6	7	8	9
8	9	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	30	31	32	1

R_0: 10011010 00100110 00100111 00100100

R'_0: 01001111 01000001 00001100 00010000 11101001 00001001

DES 之f()函數運算過程



列／行	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S-BOX 1																
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S-BOX 2																
0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
S-BOX 3																
0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
S-BOX 4																
0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
S-BOX 5																
0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

DES 加密法-S-BOX

將做完互斥或運算的48位元資料分成八個區塊(S-BOX)，每個(S-BOX)內有6個位元

EX. S-BOX5 (000100)

列索引(00)=0

行索引(0010)=2

查表對應值：

4→(0100)

DES 加密法-S-BOX(續)

列／行	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S-BOX 6																
0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
S-BOX 7																
0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
S-BOX 8																
0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

R'_0:

010011	(1, 9) ➔6➔0110
110100	(2, 10)➔12➔1100
000100	(0, 2)➔9➔1001
001100	(0, 6)➔9➔1001
000100	(0, 2)➔4➔0100
001110	(0, 7)➔8➔1000
100100	(2, 2)➔11➔1011
001001	(1, 4)➔10➔1010

DES 加密法 P-BOX

- 將8個S-BOX的輸出結合成32位元
- 經P-BOX之重排後完成一次f()函數之運算

P-BOX

16	7	20	21	29	12	28	17	1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9	19	13	30	6	22	11	4	25

01101100 10011001 01001000 10111010

10011110 00001111 10000111 01010001

最後排列表

- 重複執行上述步驟16回合 ，將R16及L16合併成64位元的密文輸出

最後排列表

40	8	48	16	56	24	64	32	39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30	37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28	35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26	33	1	41	9	49	17	57	25

DES 加密法範例

經過第一回合加密運算所產生的密文爲：
[00000000 11111111 10100000 00010101 01110110 01101000 00101011 01010000]

接下來各回合所產生的密文如右邊之表格

位元	1-8	9-16	17-24	32-25	33-40	41-48	49-56	57-64
明文	73	65	63	75	72	69	74	79
IP	ff	d9	4a	af	00	ff	a0	15
第 1 回合密文	00	ff	a0	15	76	68	2b	50
第 2 回合密文	76	68	2b	50	23	41	49	be
第 3 回合密文	23	41	49	be	d1	39	35	55
第 4 回合密文	d1	39	35	55	7b	23	ff	91
第 5 回合密文	7b	23	ff	91	ff	73	21	3c
第 6 回合密文	ff	73	21	3c	99	33	bc	d5
第 7 回合密文	99	33	bc	d5	c1	65	99	d1
第 8 回合密文	c1	65	99	d1	85	8a	70	4c
第 9 回合密文	85	8a	70	4c	7e	75	44	22
第 10 回合密文	7e	75	44	22	bc	a3	98	4e
第 11 回合密文	bc	a3	98	4e	11	c6	ea	37
第 12 回合密文	11	c6	ea	37	16	64	fb	de
第 13 回合密文	16	64	fb	de	8b	98	2b	a0
第 14 回合密文	8b	98	2b	a0	d6	f4	05	f8
第 15 回合密文	d6	f4	05	f8	ae	b7	cf	be
第 16 回合密文	ae	b7	cf	be	8e	a2	0a	f2
FP（密文）	14	ff	d5	cd	13	73	06	f7

DES 解密流程

- DES解密過程與加密過程類似，僅將：

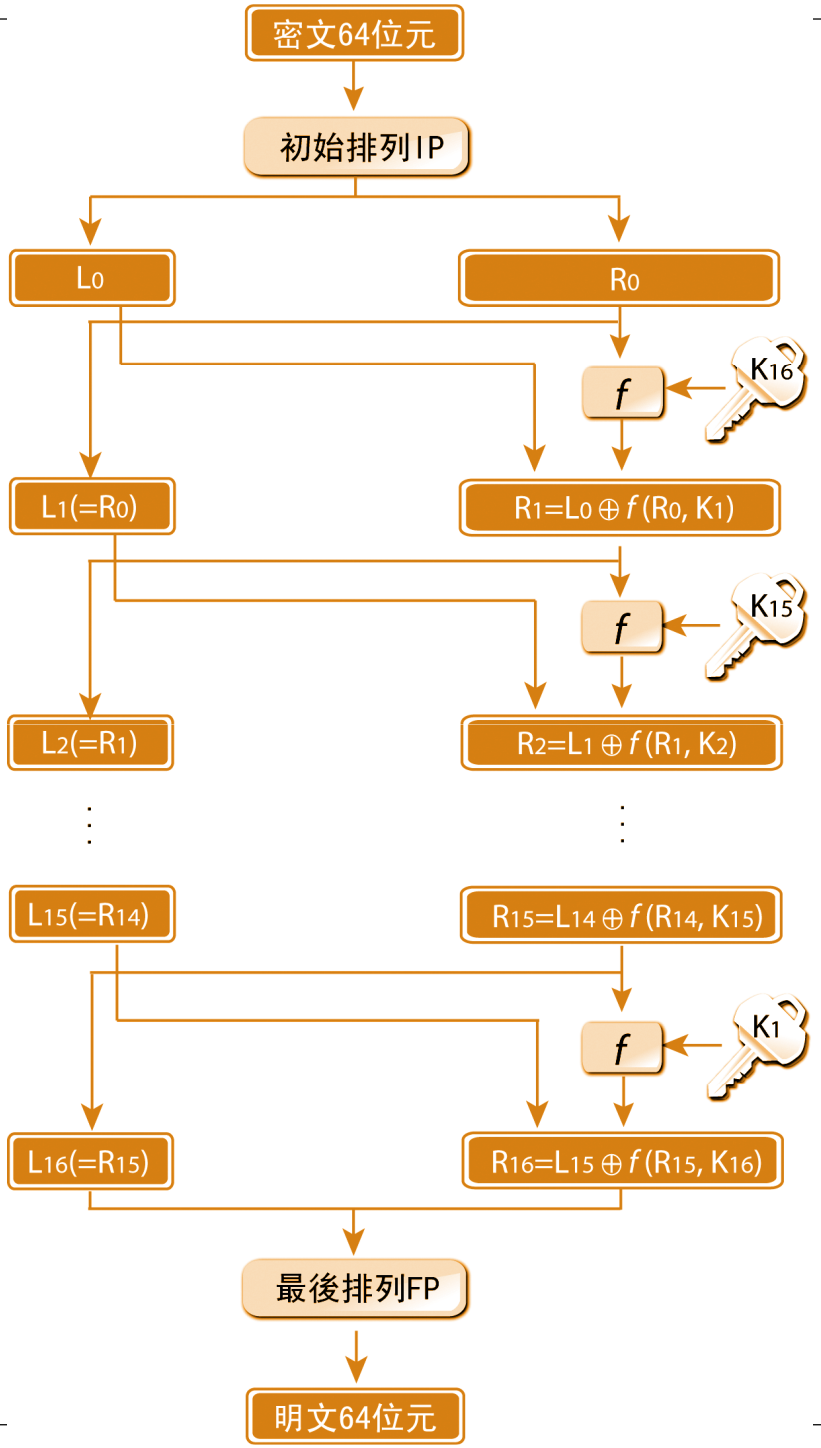
$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$$

$$R_i = L_{i-1} \oplus f(R_{i-1}, K_{17-i})$$

- 改成

$$\begin{aligned} R_1 &= L_0 \oplus f(R_0, K_{16}) \\ R_2 &= L_1 \oplus f(R_1, K_{15}) \\ &\vdots \\ R_{16} &= L_{15} \oplus f(R_{15}, K_1) \end{aligned}$$

DES 解密過程


密文64位元
初始排列IP
L0
R0
f
K16
L1(=R0)
R1=L0 ⊕ f (R0, K1)
f
K15
L2(=R1)
R2=L1 ⊕ f (R1, K2)
L15(=R14)
R15=L14 ⊕ f (R14, K15)
f
K1
L16(=R15)
R16=L15 ⊕ f (R15, K16)
最後排列FP
明文64位元

DES的強度

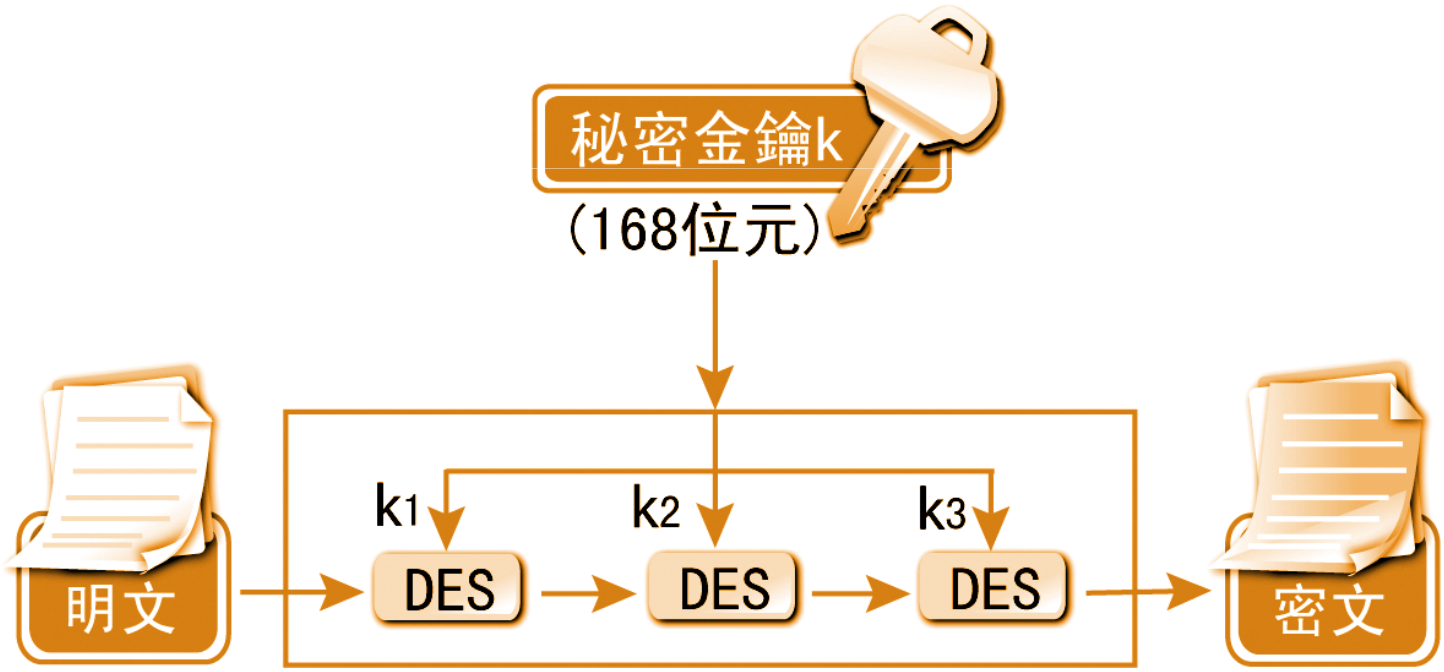
- 密碼學系統兩個最重要的部份
  - **密碼的演算法**
  - **金鑰**
- DES的演算法是**公開**的
- DES的安全性在**金鑰**的部份
  - 使用56位元的金鑰
  - 2^{56}種可能(約有$7.2*10^{16}$把金鑰)
  - 個人電腦運算需要超過1000年

DES的變動

- 隨著電腦硬體的大幅改進
  - 處理器速度的提升
  - 平行運算能力
  - 叢集運算、格網運算、雲端運算…
- DES是可能被破解的
- 因爲DES的演算法被證明是很難破解的，故只需要加強安全性即可
- 改進DES
  - **雙重DES** (Double DES)
  - **三重DES** (Triple DES)

三重DES密碼系統

- 使用三次DES對明文加密
- 依金鑰數量有不同作法
  - 三把金鑰
  - 兩把金鑰

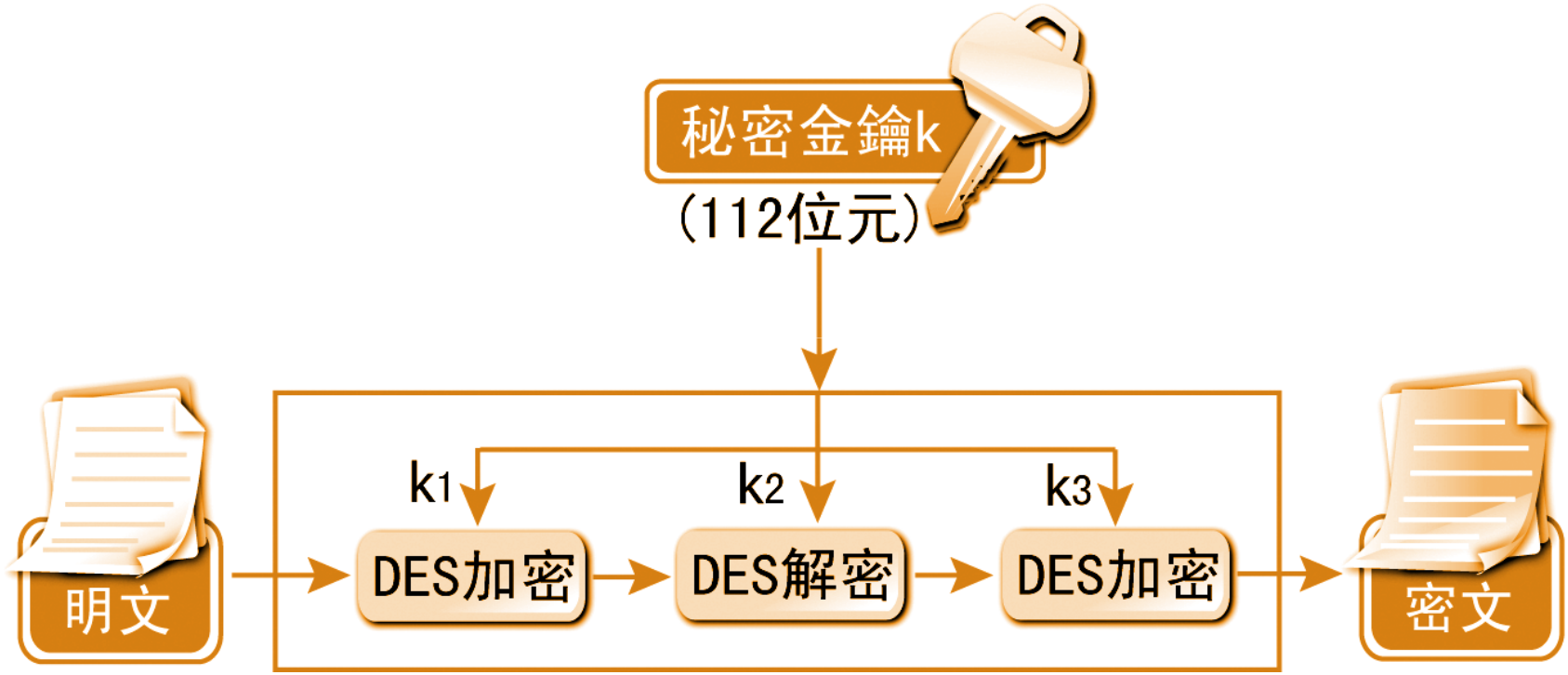


3DES 加密架構

3DES使用方式

- ***EEE3***：用三把不同祕密金鑰（即金鑰長度爲168位元）以加密-加密-加密依序處理產生密文。
- ***EDE3***：用三把不同祕密金鑰，以加密-解密-加密依序處理產生密文。
- ***EEE2***：用兩把不同祕密金鑰（即金鑰長度爲112位元）任選兩個DES金鑰設爲相同（例如，第一個及第二個DES金鑰相同，但與第三個DES金鑰不同），並以加密-加密-加密依序處理產生密文。
- ***EDE2***：用兩把不同祕密金鑰（即金鑰長度爲112位元）第一個DES金鑰與第三個DES金鑰相同，但與第二個DES金鑰不同，並以加密-解密-加密依序處理產生密文。

3DES-EDE2



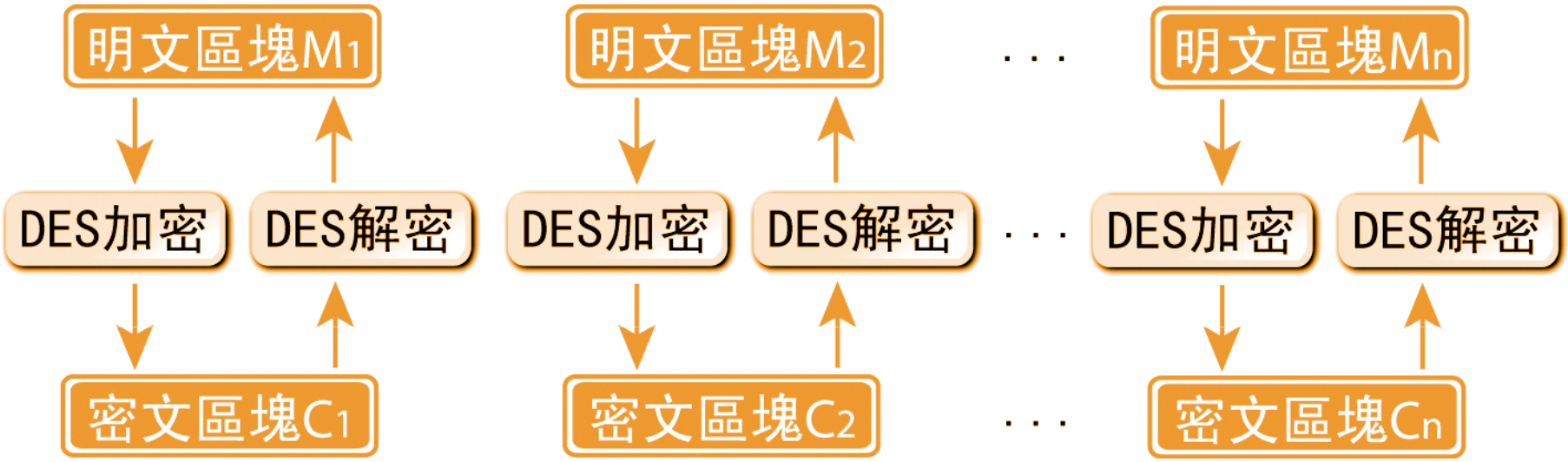
祕密金鑰密碼系統的加密模式

- ***四種加密模式***
  - **ECB** 加密模式 (**E**lectronic **C**ode **B**ook Mode) 電子碼冊
  - **CBC**加密模式 (**C**ipher **B**lock **C**haining Mode) 密碼區塊鏈結
  - **CFB**加密模式 (**C**ipher **F**eed**b**ack Mode) 密碼回饋
  - **OFB**加密模式 (**O**utput **F**eed**b**ack Mode) 輸出回饋

ECB 加密模式

- 最簡單的操作模式
- 輸入的明文訊息被分割成每區塊64位元
- 每區塊被獨立的加密
- 適合加密小訊息

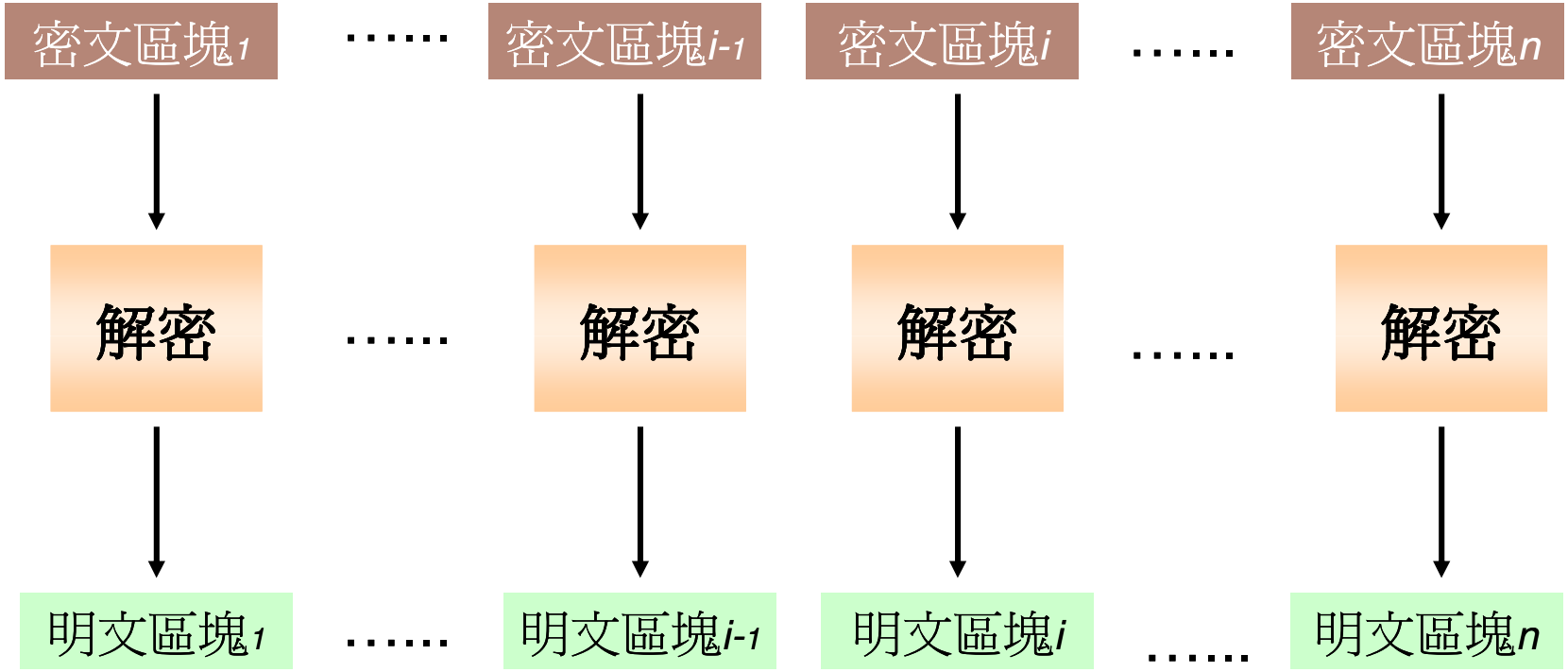
ECB 加密模式



密文區塊*i* = 加密(明文區塊*i*)

優點: 可以平行處理
缺點: 相同明文產生相同密文，內容可被剪貼替換

ECB 加密模式 (Cont.)

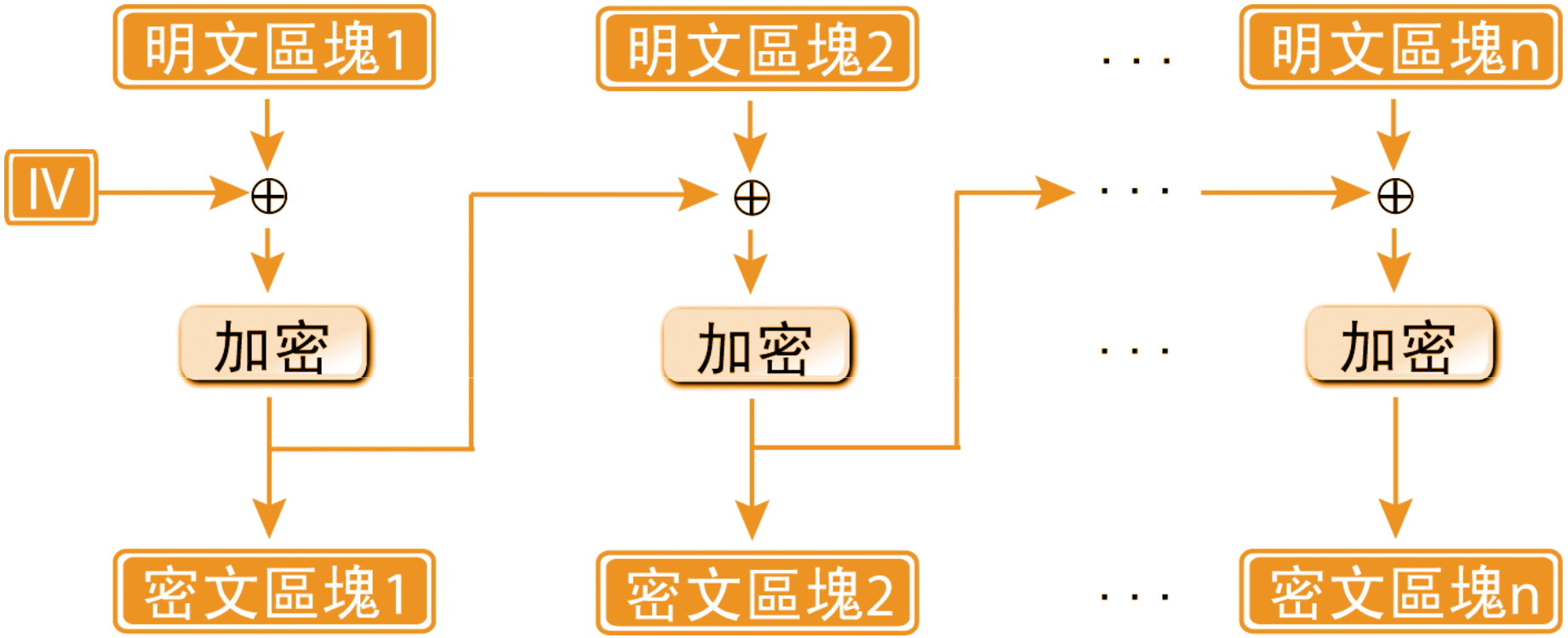


明文區塊i = 解密(密文區塊i)

密碼區塊鏈結模式(CBC)

- 確保一個明文區塊在輸入端重複出現，這相同的明文區塊在輸出端會產生位元數全部相同的密文區塊
- 使用回饋機制來達成

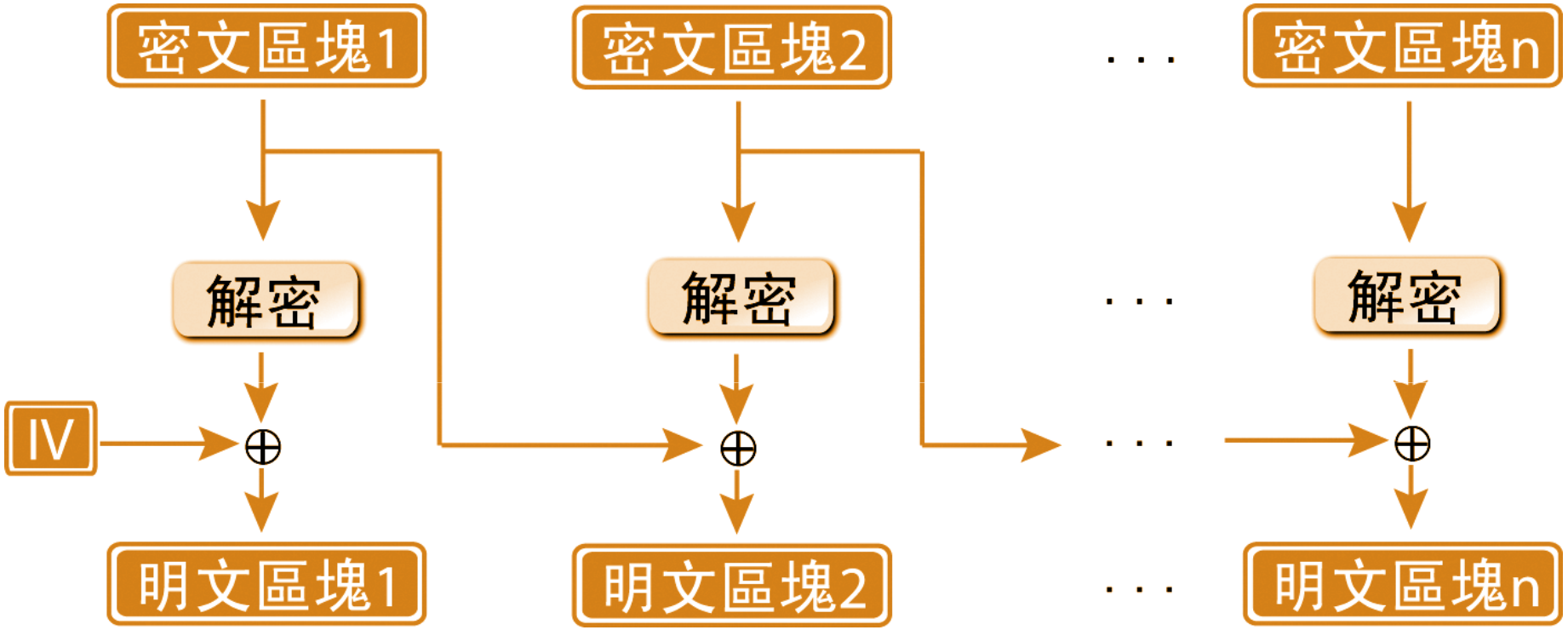
CBC 模式-加密處理



密文區塊$_1$ =加密(明文區塊$_1$ ⊕ IV)

密文區塊$_i$ =加密(明文區塊$_i$ ⊕ 密文區塊$_{i-1}$)

CBC 模式-加密處理 (Cont.)

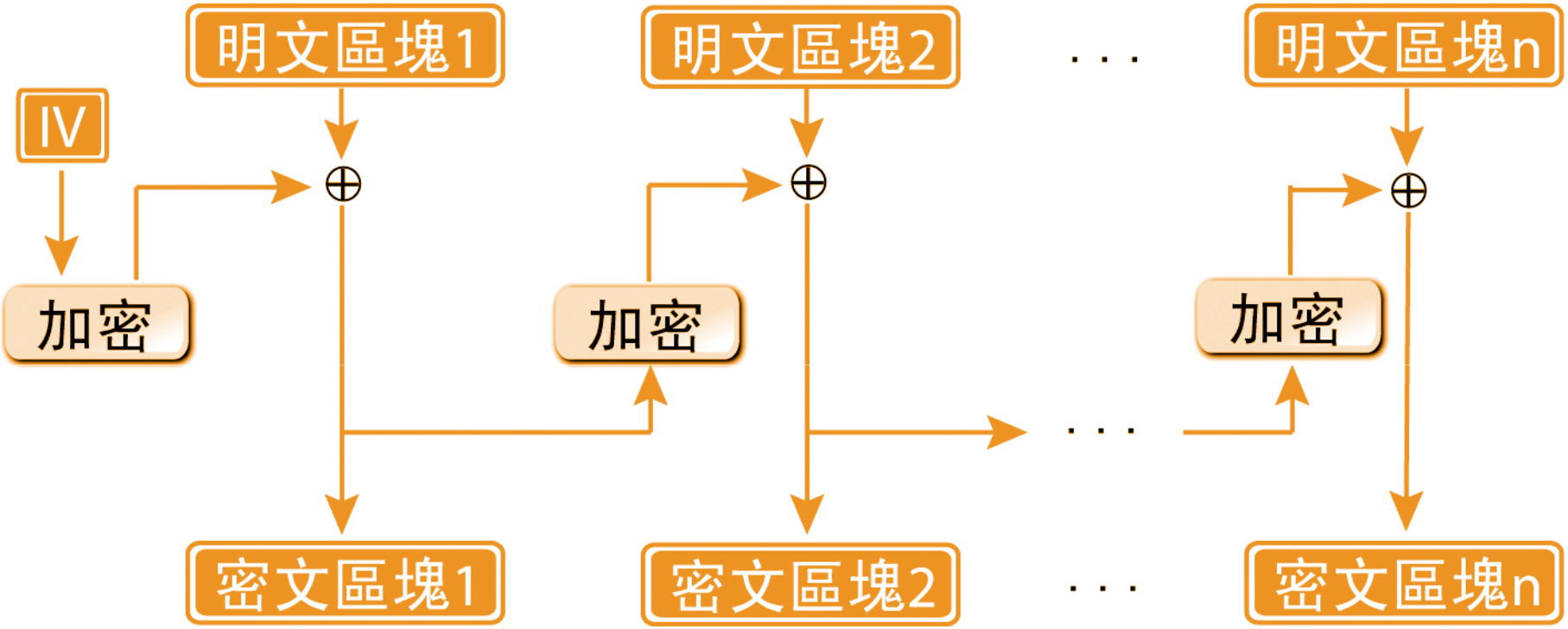


優點: 相同明文可以產生不相同密文，不會有***剪貼替換***問題
缺點: 不能***平行處理***；一個位元錯，之後全錯

密碼回饋(CFB)

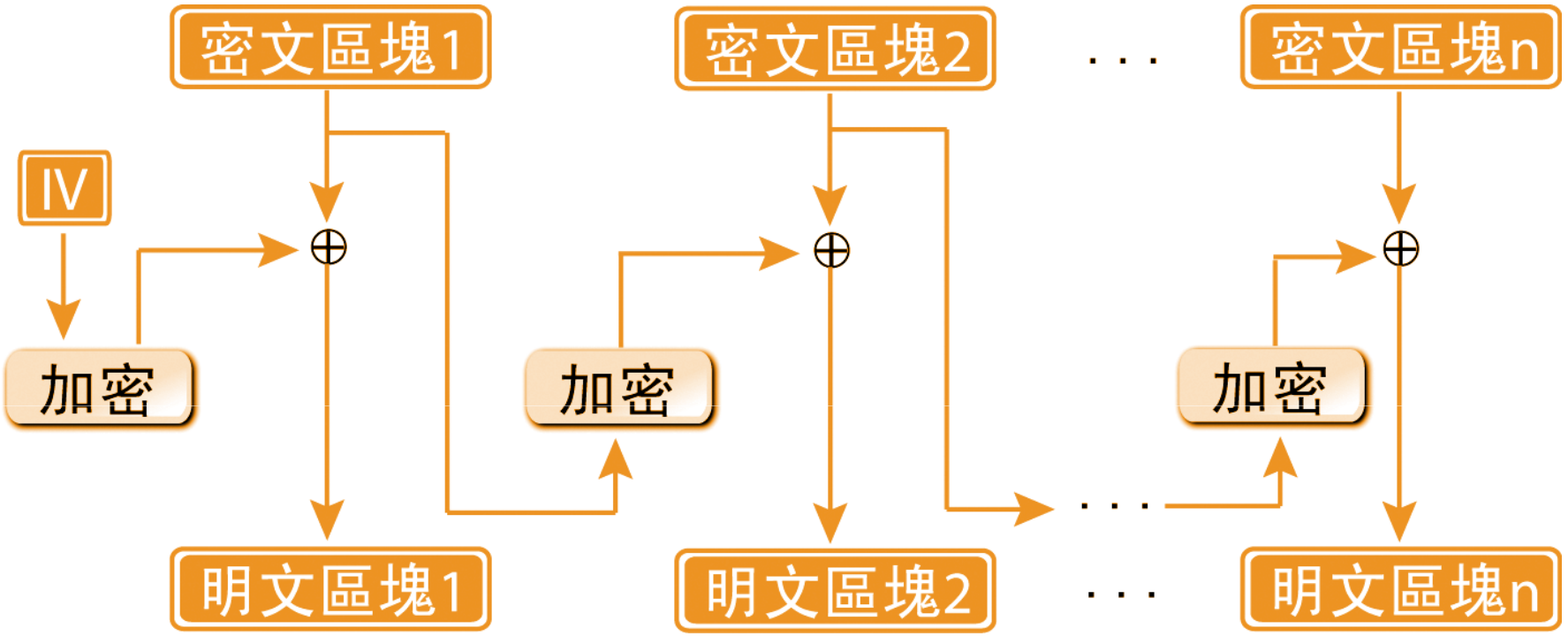
- 不是所有的應用都可以使用資料區塊方式處理
- 終端機的連線過程需要加密
- 資料加密單位(通常爲8位元)會小於一個區塊大小(通常爲64位元)

CFB 模式-加密處理過程



密文區塊*1* =明文區塊*1* ⊕ 加密(IV)
密文區塊*i* =加密(密文區塊*i-1*) ⊕ 明文區塊*i*

CFB 模式-加密處理過程



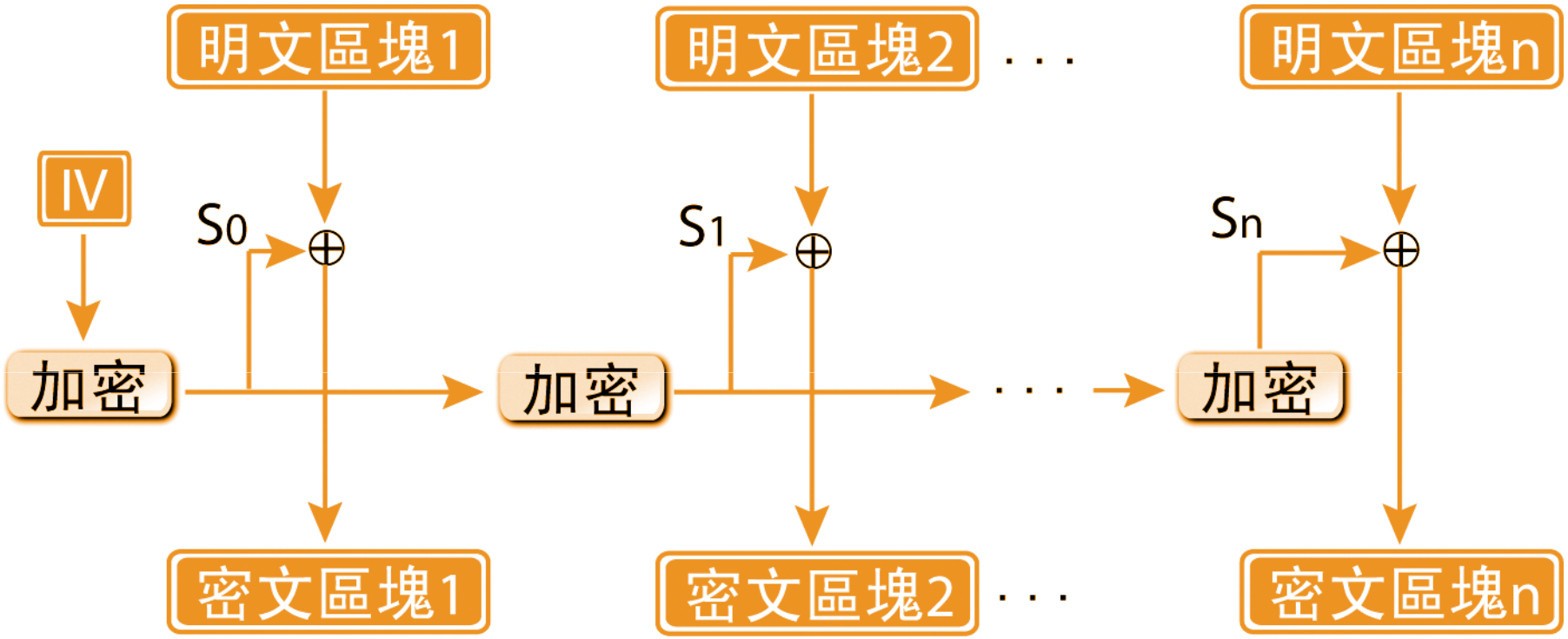
優點: 一個密文區塊錯誤僅影響目前區塊及下一個區塊；加密區塊長度*可變動*

缺點: 不能平行處理

輸出回饋(OFB)

- 與密碼回饋(CFB)模式類似
- 唯一差異
  - CFB的下一回合加密過程採用密文來加密
  - OFB的下一回合加密過程採用加密後IV的輸出值

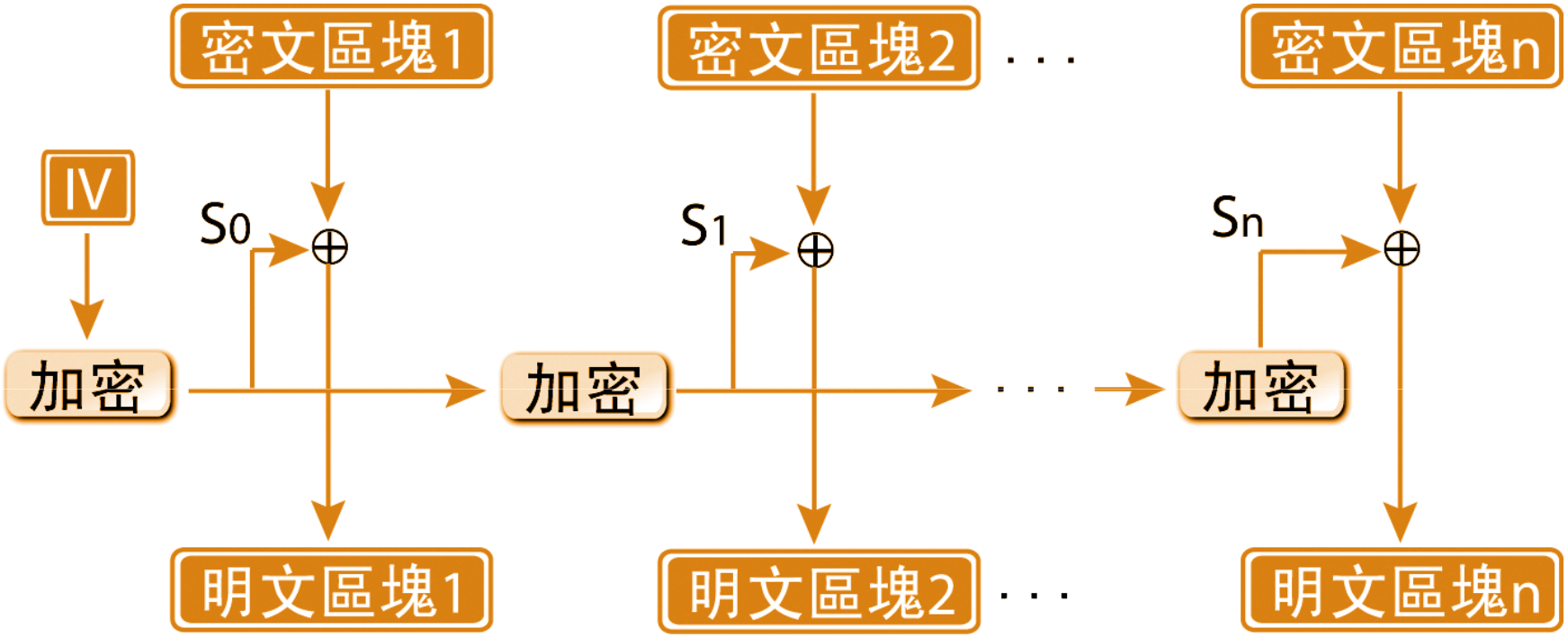
OFB 模式-加密處理過程



S_0 =加密(IV)　　S_i =加密(S_{i-1})

密文區塊$_i$ = S_i ⊕ 明文區塊$_i$

OFB 模式-加密處理過程 (Cont.)



S_0 =解密(IV)　　S_i =解密(S_{i-1})

明文區塊$_i$ = $S_i \oplus$ 密文區塊$_i$