通信情報保護學會誌
第4卷 第1號, 1994. 3

ISO/IEC JTC1/SC27의 국제표준소개(4) : ISO/IEC IS 8372

정보처리-64비트 블럭 암호 알고리즘의 운영 모드 〔Information processing-Modes of operation for a 64-bit block cipher algorithm〕

이 필 중*

요 약

지난호에는 IS 10116 : n비트 블럭 암호 알고리즘의 운영모드(Modes of operation for an n-bit block cipher algorithm)를 소개하였다. 이번 호에는 보다 n=64의 경우에 한정되어 제한적이기는 하지만 훨씬 먼저인 1987년에 만들어져 사용 되어오고 있으며 1992년 정보보안 국제총회에서 이미 많은 제품이 IS 8372를 근거로 만들어져 있기 때문이라는 이유로 다시 5년간 국제표준으로서 수명의 연장을 받은 IS 8372를 소개한다.

1. 범위와 응용분야 (Scope and field of application)

이 국제표준은 비밀키를 사용한 어떠한 64비트 블럭 암호 알고리즘(그것이 어떤 것이든)의 4가지 운영모드를 기술하고 있다. 〔This International Standard describes four modes of operation for any 64-bit block cipher algorithm using a secret key.〕

주 부록은 이 국제표준에 포함되지는 않지만 각 모드의 특성에 관한 주석을 포함하고 있다. 〔NOTE : The Annex, which does not form part of this International Standard, contains comments on the properties of each mode.〕

이 국제표준은 4개의 정의된 운영모드를 지정함으로써, 어떠한 64비트 블럭 암호 알고리즘의 응용(예를 들면 데이터 전송, 데이터 저장, 인증)들에 있어서, 이 국제표준은 운영모드, 시작 변수의 구성, 그리고 파라미터 값들을 (적절하도록)결정하는 데에 유용한 참고자료가 될 수 있다. 〔This Internaional Standard establishes four defined modes of operation so that in application of a 64-bit block cipher(for example data transmission, data storage authentication) this International Standard will provide a useful reference for the specification of the mode of operation, the

* 종신회원, 포항공과대학 전자전기공학과

formation of the starting variable, and the values of parameters(as appropriate).]

주 암호 피드백(CFB)운영모드(7절 참조)에는 j와 k 두 개의 파라미터가 정의되어 있다. 출력 피드백(OFB)운영모드(8절 참조)에는 한 개의 파라미터 j가 정의되어 있다. 이들 운영 모드중의 하나가 사용될 때 해당 파라미터 값(들)이 선택되어지고, 모든 통신 참여자들에 의해 사용되어질 필요가 있다. [NOTE : For the Output Feedback(OFB) mode of operation (see clause 8), one parameter, j, is defined. When one of these modes of operation is used the parameter value (s) need(s) to be chosen and used by all communicating parties.]

2. 참조[Reference]

ANSI X3.92-1981, Data Encryption Algorithm.

3. 정의[Definitions]

3.1 평문[plaintext]

암호화되지 않은 정보[Unenciphered information.]

3.2 암호문[ciphertext]

암호화된 정보[Enciphered information.]

3.3 블럭 체이닝[block chaining]

암호문의 각 블럭이 선행 암호문 블럭에 암호적으로 연관성이 있게 한 정보의 암호화. [The encipherment of information such that each block of ciphertext is cryptographically dependent upon the preceding ciphertext block.]

3.4 초기값[initializing value(IV)]

암호화 과정의 시작점 설정에 사용되는 값. [Value used in defining the starting point of an encipherment process.]

3.5 시작변수[starting variable(SV)]

초기값에서 유도되어 운영모드의 시작점 설정에 사용되는 변수. [Variable derived from the initializing value and used in defining the starting point of the modes of operation.]

주 이 국제표준에서는 초기값에서 시작 변수를 유도하는 방법이 정의되어 있지 않다. 운영모드의 어떠한 응용에서든 기술될 필요가 있다. [NOTE : The method of deriving the starting variable from the initializing value is not defined in this International Standard. It needs to be described in any application of the modes of operation.]

3.6 암호동기[cryptographic synchronization]

암호화와 복호화 과정의 시작점 조정. [The coordination of the encipherment and decipherment process.]

4. 표기법[Notation]

이 국제표준에서는 블럭암호 알고리즘에 의해 정의된 함수관계를 다음과 같이 표시한다.

$C=eK(P)$

여기서 P는 평문 블럭, C는 암호문 블럭, K는 열쇠이다. 이 eK라는 표현은 열쇠 K를 사용한 암호화 연산이다. [For the purpose of this International Standard the functional relation defined by the block encipherment algorithm is written $C=eK(P)$ where P is the plaintext block; C is the ciphertext block ; K is the key. The expression eK is the operation of encipherment using the key K.]

대응하는 복호화 함수는 다음과 같다.

$P=dK(C)$

대문자로 표기된 변수는, 가령 위에서의 P와 C 같이, 비트들의 일차원 배열을 나타낸다. 예를 들면,

$A=\{a_1, a_2, \ldots, a_m\}$ 그리고
$B=\{b_1, b_2, \ldots, b_m\}$

는 1에서 m까지 번호가 붙여진 m비트의 배열들이다. [The corresponding decipherment function is written $P=dK(C)$. A variable, such as P and C above, denoted by a capital letter represents a one-dimensional array of bits. For example, $A=\{a_1, a_2, \ldots, a_m\}$ and $B=\{b_1, b_2, \ldots, b_m\}$ are arrays of m bits, numbered form 1 to m.]

"배타적 논리합"으로서 알려진 이진 덧셈 연산은 기호 ⊕로 표시된다. A와 B의 배열에 대한 그 연산은 다음과 같이 정의된다.

$A\oplus B=\{a_1\oplus b_1, a_1\oplus b_2, \ldots, a_m\oplus b_m\}$

[The operation of addition, modulo 2, also known as the "exclusive or" function, is shown by the symbol ⊕. The operation applied to arrays such as A and B is defined as $A\oplus B=\{a_1\oplus b_1, a_1\oplus b_2, \ldots, a_m\oplus b_m\}$.]

j비트 배열을 생성하기 위해 A의 왼쪽에서 j비트를 선택하는 연산은 다음과 같이 표시한다.

$A\sim j=\{a_1, a_2, \ldots, a_j\}$

이 연산은 A의 비트수 m이 $j\leq m$일 경우에만 정의된다. [The operation of selecting the leftmost j bits of A to generate a j-bit array is written $A\sim j=\{a_1,a_2,\ldots,a_j\}$. This operation is defined only when $j\leq m$, where m is the number of bits in A.]

"쉬프트 함수" S_k는 다음과 같이 정의된다. m비트의 변수 X와 k비트의 변수 F(단, $k\leq m$)가 주어질때, 쉬프트 함수 $S_k(X|F)$의 결과는 (다음과 같은) m비트의 변수가 된다.

$S_k(X|F)=\{x_{k+1},x_{k+2},\ldots,x_m,f_1,f_2,\ldots,f_k\}$

결과는 배열 X의 비트를 k만큼 왼쪽으로 쉬프트하여, $x_1,\ldots,x_k$를 없애고 X의 오른쪽에서 k비트만큼을 배열 F와 바꾸어 놓은 것이다. [A "shift function" S_k is defined as follows. Given an m-bit variable X and a k-bit variable F where $k\leq m$, the effect of a shift function $S_k(X|F)$ is to produce the m-bit variable $S_k(X|F)=\{x_{k+1},x_{k+2},\ldots,x_m,f_1,f_2,\ldots,f_k\}$. The effect is to shift the bits of array X left by k places, discarding $x_1\ldots x_k$ and to place the array F in the rightmost k places of X.]

연속하는 "1"비트들로 이루어진 k비트 변수 $I(k)$을 시작으로, j비트의 변수 C를 그 안으로 쉬프트한 (단, $j\leq k$), 이 쉬프트 함수의 특별한 경우가 사용된다. 그 결과는 다음과 같다.

$S_j(I(k)|C)=(1,1,\ldots,1,c_1,c_2,\ldots,c_j)$

여기서 왼쪽으로부터 $k-j$개의 비트들은 "1"이다. [A special case of this function is used which begins with the k-bit variable $I(k)$ of successive "1" bits and shifts the variable C of j bits into it, where $j\leq k$. The result is $S_j(I(k)|C)=(1,1,\ldots,1,c_1,c_2,\ldots,c_j)$where there

are $k-j$"ones"on the left of the resultant array.)

5. 전자 코드북(ECB) 모드 (Electronic Codebook(ECB) Mode)

64비트의 평문 블럭 P가 주어진다면 암호화 알고리즘은 64비트의 암호화 블럭 C는 다음과 같이 기술된다.

$$C_i=eK(P)$$

〔Given a plaintext block P of 64 bits, the encipherment algorithm produces a ciphertext block C of 64 bits, *i.e.* : $C=eK(P)$.〕

복호 알고리즘은 다음과 같이 기술된다.

$$P=dK(C)$$

〔The decipherment algorithm produces $P=dK(C)$〕

암호화 알고리즘을 사용하는 이 모드는 "전자 코드북"으로 알려져 있다. 〔This mode of using the encipherment algorithm is known as "electronic codebook".〕

6. 암호 블럭 체이닝(CBC) 운영모드 (Cipher Block Chaining(CBC) Mode)

CBC 운영 모드에 사용되는 변수들은 아래와 같다.

a) 각기 64비트인 n개의 평문 블럭열 P_1, P_2, ..., P_n
b) 열쇠 K
c) 64비트의 시작 변수 SV
d) 각기 64비트인 n개의 암호문 블럭열 C_1, C_2, ..., C_n

〔The variables employed for the CBC mode of encipherment are : a) a equence of n plaintext blocks P_1, P_2, ..., P_n, each of 64bits. b) a key K. c) a starting variable SV of 64 bits. d) the resultant sequence of n ciphertext blocks C_1, C_2, ..., C_n, each of 64 bits.〕

주 SV 형성 방법은 이 국제표준에서는 기술되어 있지 않다. 〔NOTE : The method of forming SV is not described in this International Standard.〕

CBC 암호화 모드는 다음과 같이 기술된다. 최초의 평문 블럭의 암호화 :

$$C_1=eK(P_1\oplus SV) \quad (1)$$

계속하여:

$$C_i=eK(P_i\oplus C_{i-1}) \; i=2, 3, .., n \quad (2)$$

〔The CBC mode of encipherment is described as follows : Encipherment of the first plaintext variable, $C_1=eK(P_1\oplus SV)$ ···(1) subsequently, $C_i=eK(P_i\oplus C_{i-1})$ ···(2) for $i=2, 3, ..., n$)〕

이 절차는 그림1의 상부에 나타내었다. 시작변수 SV는 최초의 암호문 출력을 만들 때 사용된다. 그 다음부터는 암호문은 암호화되기 전의 다음 평문에 이진 덧셈 연산으로 더해진다. 〔This procedure is illustrated in the upper part of figure 1. The starting variable SV is used in the generation of the first ciphertext output. Subsequently, the ciphertext is added, modulo 2, to the next plaintext before encipherment.〕

CBC 복호화 모드는 다음과 같이 기술된다.
최초의 암호문 블럭의 복호화:

$$P_1=dK(C_1)\oplus SV \quad (3)$$

계속하여:

$$P_i=dK(C_i)\oplus C_{i-1} \; i=2, 3, .., n \quad (4)$$

이 절차는 그림1의 하부에 나타내었다. [The CBC mode of decipherment is described as follows : Decipherment of the first ciphertext block, $P_1=dK(C_1)\oplus SV$ ···(3) subsequently, $P_i=dK(C_i)\oplus C_{i-1}$ for $i=2,3,\ldots,n$ ···(4). This procedure is illustrated in the lower part of figure 1.]

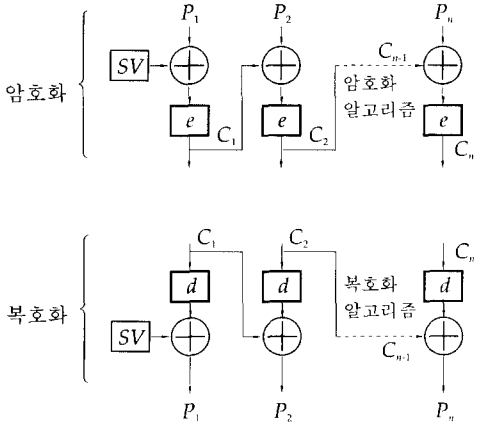


그림 1 암호 블럭 체이닝(CBC) 운영 모드

7. 암호 피드백(CFB) 운영 모드 [Cipher Feedback(CFB) Mode]

7.1 두 개의 파라미터가 CFB 운영 모드를 정의한다.

a) 피드백 변수의 크기 $k(1\leq k\leq 64)$

b) 평문 변수의 크기 j $(1\leq j\leq k)$

[Two parameters define a CFB mode of operation : a) the size of feedback variable, k, where $1\leq k\leq 64$ b) the size of plaintext variable, j, where $1\leq j\leq k$.)

CFB 운영 모드에 이용되는 변수들은 아래와 같다.

a) 입력변수 :

1) 각기 j비트인 n개의 평문 변수들 P_1, P_2, ..., P_n

2) 열쇠 K

3) 64비트의 시작 변수 SV

[The variables employed for the CFB mode of operation are a) The input variables : 1) a sequence of n plaintext variables P_1, P_2, ..., P_n, each of j bits. 2) a key K, 3) a starting variable SV of 64 bits.]

b) 중간결과 :

1) 각기 64비트인 n개의 알고리즘 입력 변수열 X_1, X_2, ..., X_n

2) 각기 64비트인 n개의 알고리즘 출력 변수열 Y_1, Y_2, ..., Y_n

3) 각기 j비트인 n개의 변수열 E_1, E_2, ..., E_n

4) 각기 k비트인 n-1개의 피드백 변수열 F_1, F_2, ..., F_{n-1}

[b) The intermediate results : 1) a sequence of n algorithm input variables X_1, X_2, ..., X_n, each of 64 bits. 2) A sequence of n algorithm output variables Y_1, Y_2, ..., Y_n each of 64 bits. 3) A sequence of n variables E_1, E_2, ..., E_n, each of j bits. 4) A sequence of n-1 feedback variables F_1, F_2, ..., F_{n-1}, each of k bits.]

c) 출력 변수 즉, 각기 j비트인 n개의 암호문 변수열 C_1, C_1, ..., C_n.[The output variables, *i.e.* a sequence of n ciphertext variables C_1, C_1, ..., C_n each of j bits.]

주 SV를 형성하는 방법은 이 국제표준에는 기술되어 있지 않다. [NOTE : The method of forming SV is not described in this International Standard.]

변수 X의 초기값은 다음과 같이 지정한다.

$$X_1=SV \qquad (5)$$

〔The variable X is set to its initial value $X_1=SV$ …(5).〕

7.2 각 평문 블럭을 암호화하는 운영은 다음 5개의 단계를 따른다.

a) 암호 알고리즘의 이용, $Y_i=eK(X_i)$ (6)

b) 왼쪽 j비트의 선택, $E_i=Y_i \sim j$ (7)

c) 암호문 블럭의 생성, $C_i=P_i \oplus E_i$ (8)

d) 피드백 변수의 생성, $F_i=S_j(I(K)|C_i)$ (9)

e) 쉬프트 함수, $X_{i+1}=S_k(X_i|F_i)$ (10)

〔The operation of enciphering each plaintext variable employs the following five steps : a) use of encipherment algorithm, $Y_i=eK(X_i)$ …(6), b) selection of leftmost j bits, $E_i=Y_i \sim j$ …(7), c) generation of ciphertext block, $C_i=P_i \oplus E_i$ …(8), d) generation of feedback block, $F_i=S_j(I(K)|C_i)$ …(9), e) shift function on X, $X_{i+1}=S_k(X_i|F_i)$ …(10).〕

이 과정들은 $i=1,2,\ldots,n$,까지 반복되어, 마지막 사이클인 식 (8)에서 끝난다. 이 절차를 그림 2의 좌측에 나타내었다. 암호화 알고리즘의 출력 블럭 Y의 왼쪽 j비트는 j비트의 평문 블럭을 이진수 덧셈 연산으로 암호화하는데 사용된다. Y의 나머지 비트는 생략된다. 평문과 암호문 블럭의 비트들은 1에서 j까지 번호가 붙여진다. 〔These steps are repeated for $i=1,2,\ldots,n$, ending with equation (8) on the last cycle. The procedure is shown in the left side of figure 2. The leftmost j bits of the output block Y of the encipherment algorithm are used to enciper the j-bit plaintext block by modulo 2 addition. The remaining bits of Y are discard. The bits of the plaintext and ciphertext blocks are numbered from 1 to j.〕

암호문 블럭은 왼쪽 비트 위치에 k-j개의 "1"을 대치하여 k비트 배열 F가 된다. 그때 그 다음에 배열 X의 비트는 k만큼 왼쪽으로 쉬프트되고, 배열 F는 오른쪽에서 k만큼 삽입되어 새로운 X값을 생성한다. 이 쉬프트 연산에서는 X의 왼쪽 k비트가 생략된다. 배열 X의 초기치는 시작변수(SV)이다. 〔The ciphertext block is augmented by placing k-j "ones" in its leftmost bit positions to become F a k-bit array, then the bits of array X are shifted left by k places and array F is inserted in the rightmost k places to produce the new value of X. In this shift operation, the leftmost k bits of X are discarded. The initial value of array X is the starting variable (SV).〕

7.3 복호화하는데 사용된 변수들은 암호화할 때 사용된 변수들과 같다. 입력 변수 X는 초기값으로 $X_1=SV$가 지정된다. 각 암호문 블럭을 복호화하는 운영은 다음 다섯 가지 절차를 따른다.

a) 암호 알고리즘의 이용, $Y_i=eK(X_i)$ (11)

b) 왼쪽 j비트의 선택, $E_i=Y_i \sim j$ (12)

c) 복호문 블럭의 생성, $P_i=C_i \oplus E_i$ (13)

d) 피드백 변수의 생성, $F_i=S_j(I(K)|C_i)$ (14)

e) 쉬프트 함수, $X_{i+1}=S_k(X_i|F_i)$ (15)

〔The variables employed for decipherment are the same as those employed for encipherment. The variable X is set to its initial value $X_1=SV$. The operation of deciphering each ciphertext block employs the following five steps : a) use of encipherment algorithm, $Y_i=ek(X_i)$ …(11), b) selection of leftmost j bits, $E_i=Y_i \sim j$ … (12), c) generation of plaintext block, $P_i=C_i \oplus E_i$ …(13), d) generation of feedback block, $F_i=S_j(I(K)|C_i)$ …(14), e) shift function on X, $X_{i+1}=S_k(X_i|F_i)$ …(15).〕

이 단계들은 i=1, 2, …, n번까지 반복되어, 마지막 사이클인 식(13)에서 끝난다. 이 절차를 그림 2의 오른쪽에 나타내었다. 암호화 알고리즘

에서 출력 블럭 Y의 왼쪽 j비트들은 j비트의 암호문 블럭을 이진수의 가산으로 복호화하는데 사용된다. Y의 나머지 비트는 생략된다. 평문과 암호문 블럭은 1에서 j까지 번호가 붙여진 비트를 갖는다. [These steps are repeated for i=1, 2, ..., n, ending with equation (13) on the last cycle. The procedure is illustrated on the right side of figure 2. The leftmost j bits of the output Y of the encipherment algorithm are used to decipher the j-bit ciphertext block by modulo 2 addition. The remaining bits of Y are discarded. The plaintext and ciphertext blocks have bits numbered from 1 to j.]

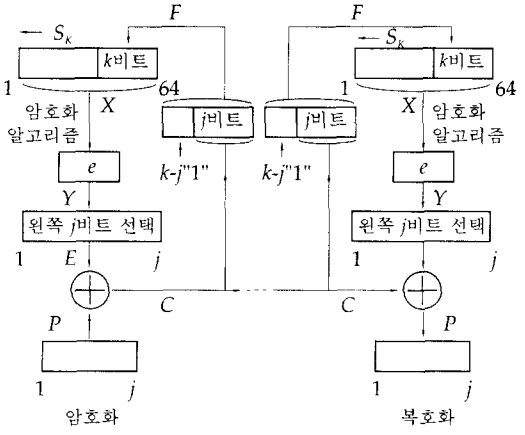


그림 2 암호 피드백(CFB) 운영 모드

암호문 블럭은 왼쪽 비트 위치에 k-j개의 "1"을 대치하여 k비트 배열 F가 된다. 그때 배열 X의 비트는 k만큼 왼쪽으로 쉬프트되고, 배열 F는 오른쪽에서 k만큼 삽입되어 새로운 X값을 생성한다. 배열 X의 초기값은 초기 변수(SV)가 된다.[The ciphertext block is augmented by placing k-j "ones" in its leftmost bit positions to become F, a k-bit array, then the bits of the array X are shifted left by k places and the array F is inserted in the rightmost k places to produce the new value of X. In this shift operation, the leftmost k bits of X are discarded. The initial value of the array X is the starting variable(SV).]

7.4 CFB에서는 같은 j와 k값을 사용하도록 권고한다. 이 권고안(j=k)에서 식(9)와 (14)는 다음과 같이 나타낸다. F_i=C_i(j=k 경우) [It is recommended that CFB should be used with equal values of j and k. In this recommended form (j=k)the equations (9) and (14) can be written, F_i=C_i(case j=k)]

8. 출력 피드백(OFB) 모드 [Output Feedback(OFB)Mode]

8.1 한개의 파라미터가, 즉 평문 변수의 크기 j($1 \le j \le 64$), OFB 운영모드를 정의한다. OFB운영 모드에서 사용되는 변수들은 다음과 같다.

a) 입력변수 :

1) 각기 j비트인 n개의 평문 변수들 P_1, P_1, ..., P_n

2) 열쇠 K

3) 64비트의 시작 변수 SV

[One parameter defines an OFB mode of operation, *i.e.* the size of plaintext variable j where $1 \le j \le 64$. The variables employed for the OFB mode of operation are : a) The input variables : 1) a sequence of n plaintext blocks P_1, P_2, ..., P_n, each of j bits; 2) a key K; 3) a starting variable SV of 64 bits.]

b) 중간결과 :

1) 각기 64비트인 n개의 입력 변수열 X_1, X_2, ..., X_n

2) 각기 64비트인 n개의 출력 변수열 Y_1, Y_2, ..., Y_n

3) 각기 j비트인 n개의 변수열 E_1, E_2, ..., E_n

[b) The intermediate results : 1) a sequence of n algorithm input variables X_1, X_2, ..., X_n, each of 64 bits 2) a sequence of n algorithm output variables Y_1, Y_2, ..., Y_n, each of 64 bits. 3) a sequence of n variables E_1, E_2, ..., E_n, each of j bits.]

c) 출력 변수들은 각기 j비트인 n개의 암호문 변수열 C_1, C_2, ..., C_n. [The output variables, *i.e.* a sequence of n ciphertext variables C_1, C_2, ..., C_n, each of j bits.]

주 *SV* 형성 방법은 이 국제 표준에는 기술되어 있지 않다. [NOTE : The method of forming *SV* is not described in this International Standard.]

입력 블럭 X의 초기값은 다음과 같이 지정한다.

$$X_1 = SV \qquad (16)$$

[The variable X is set to its initial value $X_1=SV$ ···(16)]

8.2 각 평문 블럭들을 암호화하는 운영은 다음 네 가지 절차를 따른다.

a) 암호 알고리즘의 이용, $Y_i = eK(X_i)$ (17)

b) 왼쪽 j비트의 선택, $E_i = Y_i \sim j$ (18)

c) 암호문 블럭의 생성, $C_i = P_i \oplus E_i$ (19)

d) 피드백 연산, $X_{i+1} = Y_i$ (20)

[The operation of enciphering each plaintext employs the following four steps : a) use of encipherment algorithm, $Y_i = eK(X_i)$ ···(17), b) selection of leftmost j bits, $E_i = Y_i \sim j$ ···(18), c) generation of ciphertext block, $C_i = P_i \oplus E_i$ ···(19), d) feedback operation, $X_{i+1} = Y_i$ ···(20).]

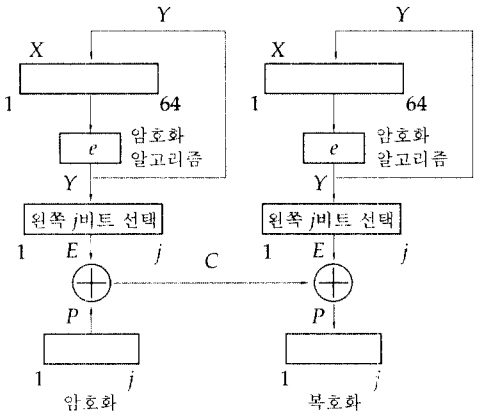


그림 3 출력 피드백(OFB)운영 코드

이 과정들은 $i=1, 2, \ldots, n$번까지 반복되며, 마지막 사이클인 식(19)에서 끝난다. 이 절차는 그림3의 왼쪽부분에 그려져 있다. 암호 알고리즘을 사용한 매 번의 결과 Y_i 는 피드백되고, X_i 의 다음 값, 즉 X_{i+1}이 된다. Y_i 의 좌측에서 j비트는 입력변수를 암호화하는데 사용된다. [These steps are repeated for $i=1, 2, \ldots, n$, ending with equation (19) on the last cycle. The procedure is illustrated on the left side of figure 3. The result of each use of the encipherment algorithm, which is Y_i is used to feedback and become the next value of X_i namely X_{i+1}. The leftmost j bits of Y_i are used to encipher the input block.]

8.3 복호화에 사용된 변수들은 암호화에 사용된 것과 동일하다. 변수 X는 초기값으로 $X_1 = SV$가 지정된다. 각 암호문 블럭을 복호화하는 운영은 다음 네 가지 절차를 따른다.

a) 암호 알고리즘의 이용, $Y_i = eK(X_i)$ (21)

b) 왼쪽 j비트의 선택, $E_i = Y_i \sim j$ (22)

c) 복호문 블럭의 생성, $P_i = C_i \oplus E_i$ (23)

d) 피드백 연산, $X_{i+1} = Y_i$ (24)

[The variables employed for decipher-

ment are the same as those employed for encipherment. The variable X is set to its initial value $X_1=SV$. The operation of deciphering each cipertext block employs the following four steps : a) use of encipherment algorithm, $Y_i=eK(X_i)$ ··· (21), b) selection of leftmost j bits, $E_i=Y_i \sim j$ ··· (22), c) generation of plaintext variable, $P_i=C_i \oplus E_i$ ··· (23), d) feedback operation, $X_{i+1}=Y_i$ ··· (24).]

이 과정들은 $i=1, 2, \ldots, n$까지 반복되어, 마지막 사이클인 식(23)에서 끝난다. 이 절차는 그림3의 오른쪽에 나타내었다. 변수들은 X_i와 Y_i는 암호화에 사용된 것들과 같고, 단지 식(23)이 다르다. [These steps are repeated for $i=1, 2, \ldots, n$ ending with equation (23) on the last cycle. The procedure is illustrated in the right side of figure 3. The values of variables, X_i and Y_i are the same as those used for encipherment ; only equation (23) is different.]

부 록 : 운영모드의 특성 [Annex : Properties of the modes of operation]

이 부록은 표준안에 기술되어 있는 네 가지 운영 모드의 특성에 대한 주석을 포함하고 있으나, 표준안의 꼭 필요한 부분은 아니다. [This annex contains comments on the properties of the four modes of operation described in this standard and is not an integral part of the body of the standard.]

A.1 전자 코드북(ECB) 운영모드의 특성 [Properties of the Electronic Codebook(ECB)Mode of Operation]

컴퓨터 또는 인간 상호간의 정보를 운반하는 메세지는 반복성을 갖고 있거나 자주 사용되는 열들로 되어있다. ECB 모드에서 동일한 평문은(같은 열쇠일 경우) 동일한 암호문을 생성한다. 이 특성이 ECB를 일반적인 사용을 부적당하게 만든다. ECB의 사용은 반복적 특성이 받아들여지는 목적을 위해 미래의 표준으로 규정될 것이다. [Messages that carry information between computers, or people, may have repetitons or commonly used sequences. In ECB mode, identical plaintext produces(for the same key) identical ciphertext blocks. This characteristic makes ECB unsuitable for general use. The use of ECB may be specified in future standards for those purposes where the repetition characteristic is acceptable.]

암호화와 복호화 사이에서 블럭 경계를 읽어버리면(예를 들면 비트 슬립 때문에) 올바른 블럭 경계가 재성립될 때까지 암호화와 복호화 작업 사이에서의 동기화가 되지 않을 것이다. 모든 복호화 작업의 결과는 올바르게 되지 않을 것이다. [If block boundaries are lost between encipherment and decipherment(for example a bit slip), synchronization between the encryption and decryption operations will be lost until correct block boundaries are re-established. The results of all decipherment operations will be incorrect.]

A.2 암호 블럭 체이닝(CBC) 운영모드의 특성 [Properties of the Cipher Block Chaining(CBC)Mode of Operation]

CBC 모드에서는 같은 열쇠와 같은 시작 변수로 동일한 평문이 암호화되면 언제나 동일한 암호

문이 생긴다. 이 특성에 대해서 걱정이 되는 사용자는 평문의 시작, 열쇠 또는 시작 변수를 바꾸기 위한 어떤 방법을 고안해야 한다. 그러한 방법중 한가지는 각 CBC 평문의 시작 부분에 유일한 식별자, (예를 들면 증가계수)를 포함시키는 것이다. 크기가 증가되어서는 안되는 레코드를 암호화할 때에 이용되는, 또 다른 방법으로는 그 내용(예를 들면 랜덤 액세스 저장소에 들어있는 블럭 주소)을 알지 않고도 레코드에서 계산될 수 있는 시작 변수와 같은 어떤 값을 사용하는 방법이 있다. [The CBC mode produces the same ciphertext whenever the same plaintext is enciphered using the same key and initialising value. Users who are concerned about this characteristic should devise some way of changing the start of the plaintext, the key or the starting variable. One possibility is to incorporate a unique identifier(for example an incremented counter) at the beginning of each CBC message. Another, which may be used when encrypting records whose size should not be increased, is to use some value as the initialising values which can be computed from the record without knowing its contents(for example the nubmer of the block which contains it in random access storage).]

CBC 방식은 블럭 암호화이기 때문에 정확히 64비트가 필요하다. 64비트보다 작은 블럭들은 특별한 조작(예를 들면, padding)이 필요하다. [Since the CBC mode is a block method of encipherment, it needs to operate on complete data blocks of 64 bits. Blocks of less than 64 bits require special handling.]

CBC 방식에서는 하나의 암호문 블럭내에 한 개 또는 그 이상의 비트의 오류는 두 블럭(오류가 발생한 블럭과 그 다음의 블럭)의 복호화에 영향을 준다. 만약 i번째 암호문 블럭에 오류가 발생하면, i번째의 복호된 평문 블럭은 평균 50%의 비트 오류율을 가질 것이다. $(i+1)$번째 복호된 평문 블럭은 오류가 있는 암호문 비트에 직접적으로 일치하는 비트에만 오류가 발생하게 된다. [In the CBC mode, one or more bit errors within a single ciphertext block will affect the decipherment of two blocks(the block in which the error occurs and the succeeding block). If the errors occur in the ith ciphertext block, each bit of the ith plaintext block will have an average error rate of 50%. The $(i+1)$th plaintext block will only those bits in error that correspond directly to the ciphertext bits in error.]

암호화와 복호화 사이에서 블럭 경계를 읽어버리면(예를 들면 비트 슬립 때문에) 올바른 블럭 경계가 재성립될 때까지 암호화와 복호화 운영 사이에서의 동기화가 되지 않을 것이다. 모든 복호화 운영 결과는 정확하지 않을 것이다. [If block boundaries are lost between encipherment and decipherment(for example a bit slip), synchronization between the encryption and decryption operations will be lost until correct block boundaries are re-established. The result of all decipherment operations will be incorrect.]

A.3 암호 피드백(CFB) 운영모드의 특성 [Properties of the Cipher Feedback (CFB)Mode of Operation]

CFB 모드의 경우에는 j-비트 암호문에 어떤 오류가 있을 때 CFB 입력 블럭에서 오류 비트들이 쉬프트되어 모두 빠져나갈 때까지 오류가 발생한 암호문과 이어지는 암호문의 복호화에 영향을 미친다. 최초로 영향을 받은 j-비트 평문의 오류

비트들의 위치는 암호문에서의 오류 비트들의 위치와 같다. 이어지는 복호화된 평문은 모든 에러가 입력 블럭 밖으로 쉬프트되어질 때까지 평균 50%의 오류율을 가질 것이다. 이 시간 동안에 추가의 오류가 없다는 가정하에서는 그 후에는 정확한 평문이 얻어질 것이다. 이 특성은 "제한된 오류 확장"이라고 한다. [In the CFB mode, errors in any *j*-bit unit of ciphertext will affect the decipherment of the garbled ciphertext and also the decipherment of succeeding ciphertext until the bits in error have been shifted out of the CFB input block. The first affected *j*-bit unit of plaintext will be garbled in exactly those places where the ciphertext is in error. Succeeding deciphered plaintext will have an average error rate of 50% until all errors have been shifted out of the input block. Assuming no additional errors are encountered during this time, the correct plaintext will then be obtained. This characteristic is referred to as "limited error extension".]

만약 *j*-비트 경계가 복호화 과정에서 상실된다면, 암호적 동기성은 *j*-비트 경계가 성립될 때까지 또는 *j*-비트 경제 다음의 64-비트가 재성립될 때까지 상실될 것이다. [If *j*-bit boundaries are lost during decryption, cryptographic synchronization will be lost until cryptographic initialisation is performed or until 64 bits after the *j*-bit boundaries have been re-established.]

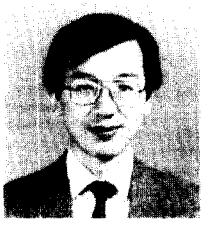
CFB 모드에서 암호화와 복호화 모두 암호화 알고리즘을 이용한다. [The encipherment and decipherment processes in the CFB mode both use the encipherment form of the algorithm.]

A.4 출력 피드백(OFB)운영모드의 특성 [Properties of the Output Feedback (OFB)Mode of Operation]

OFB 모드의 암호문의 오류를 그 결과로 나오는 평문의 출력에 전파되지 않도록 한다. 암호문상에서 한 개 비트의 오류는 복호화된 평문상에 오직 한 개의 비트 오류만을 발생시킨다. OFB 모드는 자기동기화가 되지 않는다. 만약 암호화와 복호화의 운영이 비동기화가 된다면 시스템을 재초기화 할 필요가 있다. 이와 같은 동기성의 상실은 j 비트 블럭($j > 1$일때)의 경계 상실(비트 슬립 때문에)에 의한 것이거나 한쪽에서 또는 다른 한쪽에서 변수 X의 오류에 의한 것인데, 이 오류는 재초기화가 이루어질 때까지 양쪽의 X값을 다르게 한다. [The OFB mode does not extend ciphertext errors in the resultant plaintext output. One bit in error in the ciphertext causes only one bit to be in error in the deciphered plaintext. It is not self-synchronizing. If the two operations of encipherment and decipherment desynchronize, the system needs to be re-initialized. Such a loss of synchronization might be due either to the loss of correct boundaries of the *j*-bit blocks(because of a bit slip) or an error in the value of variable X at one end or the other, causing the X values to differ at the two ends until re-initialisation takes place.]

각각의 재초기화는 같은 열쇠를 가지고 기존에 사용했던 SV값과는 다른 SV값이 사용되어야 한다. [Each re-initialisation should use a value of SV different from the SV values used before with the same key.]

□ 著者紹介



이 필 중(李 弼 中) 종신회원, 국제이사

1951년 12월 30일생
1974년 2월 서울대학교 전자공학과 학사
1977년 2월 서울대학교 전자공학과 석사
1982년 6월 U.C.L.A. System Science, Engineer
1985년 6월 U.C.L.A. Electrical Engineering, Ph.D.
1980년 6월 ~ 1985년 8월 Jet Propulsion Laboratory, Senior Engineer
1985년 8월 ~ 1990년 2월 Bell Communications Research, M.T.S.
1990년 2월 ~ 현재 포항공과대학 전자전기공학과, 부교수