LUCIFER 형태의 암호화 알고리즘에 관한 연구

강 해 동 *, 이 창 순 **, 문 상 재 *

* 경북대학교 전자공학과

** 대구 공업 전문대학 전산학과

A Study on Lucifer-type Encryption Algorithm

H. D. Gahng *, C. S. Ri **, S. J. Moon *

* Dept. of Electronics, Kyungpook National Univ.

** Dept. of Computer Science, Taegu technical Jr. Coll.

Abstract

Lucifer - type algorithms with variable length of block size are presented. They are investigated and compared in the viewpoint of the intersymbol dependence. The results show that the S-box and the number of rounds are very important factors in achieving the complete dependence.

1. 머 리 말

주된 암호법에는 대체(substitution)와 치환(permutation)을 병용하여 암호화하는 것과 정수나 다항식 개념을 데이타에 도입하고 이들에게 지수승을 취하여 적절히 암호화하는 것으로 대별할 수 있다. 대표적인 것으로 Lucifer [1]와 DES[2]등은 전자에, 그리고 RSA방식 [3]과 SEEK방식 [4]등은 후자에 속한다.

암호화방법은 처리하는 형태에 따라 스트림암호화방법과 블록암호화방법으로 나눌 수 있다. 스트림암호화방법은 문자 그대로 입력되는 정보비트들을 개별적으로 그리고 연속적으로 암호화시키는 것이며, 블록암호화방법은 일정한 크기의 블록으로 정보비트들을 나누어 한번에 한 블록씩 암호화하는 것으로 Lucifer와 DES가 대표적인 예이다.

본 논문에서는 DES의 전신이라 할 수 있는 Lucifer의 대체상자의 성질을 분석한다. 또 블록의 크기를 32, 64, 그리고 256비트등으로 변화시켜 Lucifer형태의 암호화 알고리즘을 구성하고, 암호의 불법적 분석을 방지하는데 중요한 척도가 되는 심볼간 상호의존도 [5,6]를 구하여 이들을 블록크기가 128비트인 Lucifer의 심볼간 상호의존도와 비교 분석한다.

2. Lucifer알 고 리 즘

Lucifer는 128비트 키이, 128비트의 블록크기를 가진 블록암호화 알고리즘이며 키이의 제어 아래 비선형과 선형의 치환을 교대로 사용하는 product암호법이다. Lucifer는 같은 형태의 과정을 열여섯번 반복 수행한다. 한 라운드의 암호화과정을 그림1 에 나타내었다.

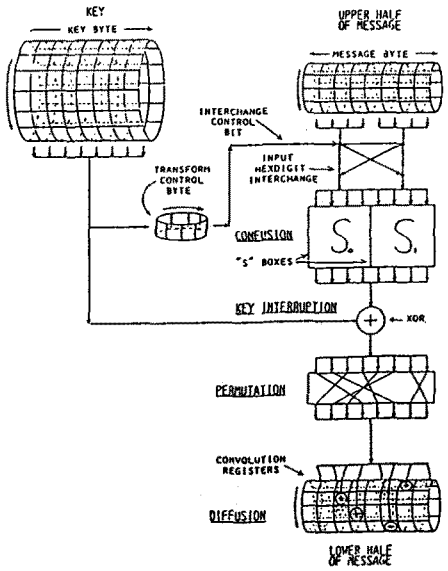

그림1.CID 블록선도

평어 128비트는 64비트씩 하반부,상반부로 나누어진다. 매라운드마다 상반부와 키이의 영향을 받아서 새로운 64비트를 구성한다.이 변형된 64비트는 다음라운드에서 상반부가 되고 다음라운드의 하반부는 이전라운드의 상반부를 그대로 사용한다. 128비트 키이는 16바이트로 구분되고 64비트 평어 상반부는 8바이트로 구분되어 각기 그림1 의 원통과 같이 구성된다.라운드마다 사용되는 키이비트수는 평어

의 상반부비트수와 같이 64비트이다.즉8바이트가 사용된다. 매라운드에서 마지막 :i이 바이트가 수행되고나면 그 바이트는 회전하지 않고 정지하여 다음라운드의 첫 키이바이트가 된다. 표1은 매라운드의 64비트 키이 발생을 위한 스케쥴이다.매라운드마다 사용되는 키이의 첫바이트(표1의0열)는 변환조정바이트(transform-control-byte)로 사용된다. 이 바이트의 8개 비트들은 상호교환조정비트(interchange-control-bit)가 되어,매비트는 정보어 상반부가 어떤 방법으로 2개의 대체상자에 입력되는 지를 결정한다. 상호교환조정비트가 "0"이면 정보어 상반부 바이트의 좌우 4비트의 두십진값이 각각 So와 S1의 입력이 되고,"1"이면 두십진값이 서로 바뀌어 대체상자에 입력된다.

표1. 키이바이트사용스케쥴

		message byte							
		0	1	2	3	4	5	6	7
	1	0	1	2	3	4	5	6	7
	2	7	8	9	10	11	12	13	14
C	3	14	15	0	1	2	3	4	5
I	4	5	6	7	8	9	10	11	12
D	5	12	13	14	15	0	1	2	3
	6	3	4	5	6	7	8	9	10
r	7	10	11	12	13	14	15	0	1
o	8	1	2	3	4	5	6	7	8
u	9	8	9	10	11	12	13	14	15
n	10	15	0	1	2	3	4	5	6
d	11	6	7	8	9	10	11	12	13
	12	13	14	15	0	1	2	3	4
	13	4	5	6	7	8	9	10	11
	14	11	12	13	14	15	0	1	2
	15	2	3	4	5	6	7	8	9
	16	9	10	11	12	13	14	15	0

라운드마다 두개의 대체상자와 한번의 치환상자를 거치는 과정을 8번 반복 수행한다. 한번 수행시마다 그 과정은 표2에 의하여 이루어지는데 그림1 에서와 같이 키이 바이트와 대체상자 출력 바이트 간에 비트별 mod-2가산이 행하여지고 그 결과가 치환상자에 입력된다.

표2.대체상자와 치환상자

S-BOX INTERNAL PERMUTATION

So	Si
0 — 12	0 — 7
1 — 15	1 — 2
2 — 7	2 — 14
3 — 10	3 — 9
4 — 14	4 — 3
5 — 13	5 — 11
6 — 11	6 — 0
7 — 0	7 — 4
8 — 2	8 — 12
9 — 6	9 — 13
10 — 3	10 — 1
11 — 1	11 — 10
12 — 9	12 — 6
13 — 4	13 — 15
14 — 5	14 — 8
15 — 8	15 — 5

FIXED PERMUTAION

0 — 3
1 — 5
2 — 0
3 — 4
4 — 2
5 — 1
6 — 7
7 — 6

치환과정을 거쳐 얻어진 한 바이트의 8비트들은 평어하반부의 특정 비트들과 비트별 mod-2가산을 하는데 그 연산이 행해지는 방법은 그림2 에 도시되어 있다.

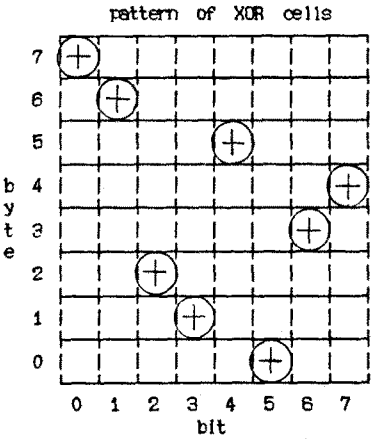

그림2.펼쳐진 폰볼루션 레지스터

에로서 치환상자의 출력인 첫라운드 첫바이트의 각 비트들이 순서대로 평어하반부의 7,6,2,1,5,0,3,과4번 바이트의 해당비트들과 각각 연산이 수행되고 나면 상반부와 하반부의 원통들은 한바이트씩 회전을 한다. 치환과정에서 출력되는 그 다음 바이트의 각 비트들도 이전 바이트처럼 회전된 하반부와 똑같이 mod-2연산을 수행하게 된다.

위 설명과 같이 매라운드 암호화 과정(CID:confusion, key interruption,and diffusion cycle)이 16번 반복되어 최종출력 128비트가 얻어진다. 수신측에서의 암호해독을 위한 복호는 암호화의 역순으로 그림1 의 CID과정을 수행하되 키이바이트사용스케쥴(표1)의 아래에서 위로 왼쪽에서 오른쪽의 차례로 키이바이트를 사용하게된다.

3. 대 체 상 자 의 성 질

Lucifer의 대체상자는 Lucifer알고리즘의 중요 구성요소이다.우선 NSA가 DES설계자들에게 제시한,DES의 대체상자들이 만족해야 할 성질들중[7],대체상자의 입력 출력 비트수와 관계없는 것들만 Lucifer에 적용하여 Lucifer의 대체상자의 성질을 조사한다.

Lucifer에도 적용될 수 있는 성질들을 나열하면,

P0. 대체상자의 각 열은 0에서 15까지의 십진수를 치환한 것
P1. 비선형이고 unaffine 할 것.

등이다.

성질 P0는 Lucifer의 대체상자에도 만족됨을 표1 로 쉽게 알 수 있다.

대체상자의 4 출력비트들은 각각 입력 4 비트의 함수로 볼 수 있다. 입력 비트들을 w,x,y,와 z로 두고, 이 4 함수를 부울린 함수 $f_i(w,x,y,z)$, i=0,1,2,3로 나타내면 So와 S1의 부울린 함수들은

$$\left.\begin{aligned} f_0 &= \bar{w}\bar{y} + \bar{w}\bar{x}z + \bar{w}x\bar{z} + x\bar{y}\bar{z} + wxyz \\ f_1 &= \bar{w}\bar{y} + \bar{y}z + \bar{w}\bar{x}\bar{z} + \bar{w}\bar{x}\bar{y}z \\ f_2 &= \bar{w}\bar{x}z + \bar{w}x\bar{z} + w\bar{x}\bar{y} + \bar{x}\bar{y}\bar{z} \\ f_3 &= yz + w\bar{x}y + wx\bar{z} + \bar{w}\bar{y} \end{aligned}\right\} \quad (3.1)$$

$$\left.\begin{aligned} f_0 &= \bar{w}\bar{x}y + w\bar{x}\bar{y} + w\bar{x}z + x\bar{y}z + wxy\bar{z} \\ f_1 &= w\bar{y} + \bar{w}\bar{x}\bar{z} + xyz \\ f_2 &= \bar{w}y + wx\bar{y} + wxz + w\bar{x}yz \\ f_3 &= wxz + w\bar{y}z + \bar{w}\bar{y}\bar{z} + \bar{x}\bar{y}z + \bar{w}\bar{x}yz + w\bar{x}\bar{y}\bar{z} \end{aligned}\right\} \quad (3.2)$$

이다.

식 3.1 과 3.2 에서 선형이거나 affine 한 함수, 즉 $f(w,x,y,z)=w+x+y+z$ 이거나 $f(w,x,y,z)=w+y+z$ 인 함수는 없으므로 Lucifer의 대체상자는 성질P1도 만족함을 알 수 있다.

다음으로 각 대체상자의 출력비트들이 대체상자에 입력되는 모든 비트의 함수라는 것이 입력비트가 모든 출력비트에 균등한 영향을 준다는 것은 아니다. 각 출력4비트에 대하여, 16개의 입력에 영향을 받은 횟수를 더한 값을 16개 입력에 전 출력비트가 영향을 받은 횟수에 대한 백분율로 나타내면 표3과 같다. 표3 으로부터 So가 S1 보다 균등한 분포를 가짐을 알 수 있다.

표 3.대체상자 입력에 각 출력비트가 받는 영향

출력비트	So	S1
0	31.53 %	18.75 %
1	21.05	25.00
2	21.05	37.50
3	26.32	18.75

4. 32,64,256비트블록크기 Lucifer형태의 암호화알고리즘

(1) 구 성

Lucifer가 완성된 1970년의 LSI 기술수준은 Lucifer 알고리즘구조에 많은 제약을 주었다[1].그 결과 Lucifer 하드웨어는 시프트레지스트를 이용하여 한 바이트 키이 및 평어 상하반부를 억세스한다. 이러한 바이트를 기본단위로 하는Lucifer의 성질을 그대로 가지도록 하면서 한 블록의 길이를 128비트보다 길거나 짧게한 Lucifer 형태 알고리즘은 표1의 키이바이트사용스케줄과 그림2의 펼쳐진 론블루션 레지스터를 각 블록길이에 맞도록 적절히 수정하므로써 구성할 수 있다.표4와 그림3에 64 비트블록에 대한 키이바이트사용스케줄과 펼쳐진 론블루션레지스트를 예로 나타내었다.그림3은 하반부바이트수의 8멱승개의 가능한 XOR 패턴 중의 한 예이다. 표4에서 알 수 있듯이 블록당 수행되는 전 라운드수는 전체 키이바이트수와 동일하다. 대체상자와 치환상자는 바이트 단위로 연산이 수행되므로 128비트 Lucifer의 그것들을 그대로 이용한다.

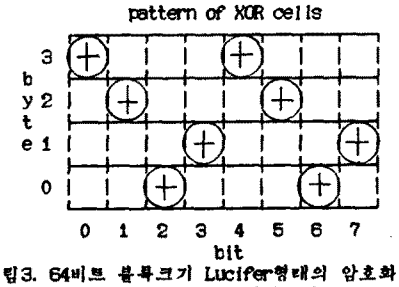

그림3. 64비트 블록크기 Lucifer형태의 암호화 알고리즘의 펼쳐진 론블루션 레지스터

표 4. 64비트 블록크기 Lucifer형태의 암호화 알고리즘의 키이바이트사용스케줄

		message byte 0	1	2	3
C	0	0	1	2	3
I	1	3	4	5	6
D	2	6	7	0	1
	3	1	2	3	4
r	4	4	5	6	7
o	5	7	0	1	2
u	6	2	3	4	5
n	7	5	6	7	0
d					

(2) 심 볼 간 상 호 의 존 도

라운드마다 출력되는 비트는 그 라운드에 입력되는 비트와 키이비트들의 함수로 볼 수 있다.출력비트를 입력비트의 함수로 볼 수 있는 성질을 백분율(%)로 표현한 의존도를 조사하므로써 암어방법의 복잡성을 비교할 수있다.

각 블록크기에 맞는 표4와 그림3을 작성하고 각 블록크기의 Lucifer형태의 암호화 알고리즘을 구성하여, 이들의 심볼간상호의존도[5]를 키이의존도와 정보의존도로 나누어 구한 결과를 표5에 나타내었다.Lucifer의 심볼간 상호의존도는 DES와는 달리 키이에 따라 각 의존도가 달라지게 된다. 본 논문에서는 상호교환조정비트가 1이 나올 확률과 0 이나올 확률이 똑같이 1/2이라 가정하고 키이비트를 랜덤하게 1과 0이 나올 확률이 각각 1/2이 되도록 발생하였다.

표5.각 블록크기 Lucifer의 상호의존도

	정 보				키 이			
라운드	32	64	128	256	32	64	128	256
1	9.38	4.69	2.34	1.17	6.05	3.08	1.55	0.78
2	36.13	19.92	10.05	5.06	14.55	8.65	4.53	2.31
3	75.38	53.91	32.98	18.64	43.66	27.20	16.87	9.54
4	97.07	87.89	68.25	48.60	78.62	54.37	46.47	30.25
5		100.0	93.36	81.63		77.64	77.89	66.66
6			100.0	97.79		89.80	93.53	93.73
7				100.0		95.61	98.75	99.70
8						99.17	99.76	100.0
9							100.0	

키이에 따라 각 의존도가 달라지므로 전체 라운드 수가 비교적 적은 32 비트와 64 비트 Lucifer들은 입력되는 키이에 따라서는 100% 의존도를 얻지 못할 경우도 있다.

암호화에 사용되는 치환은 랜덤하게 하여야 complete S-P 네트워크를 이룰 수 있다[8]. 바이트를 연산의 기본 단위로 하는 Lucifer의 확산치환은 완전히 랜덤하지는 못하므로 심볼간 상호의존도가 DES의 그것보다 떨어지는 요인중에 하나가 된다. 예로서 64비트 Lucifer의 경우 치환과정에서 한라운드에 출력되는 4바이트를 모두 모아 한꺼번에 표6과 같이 랜덤하게 확산치환하면 키이와 정보의존도를 약간(실험결과 0.2%) 높일 수 있다.

표6. 64비트Lucifer의 랜덤확산치환

	bit 0	1	2	3	4	5	6	7
b 0	4	25	14	23	20	9	30	7
y 1	8	29	18	27	24	13	2	11
t 2	12	1	22	31	28	17	6	15
e 3	16	5	26	3	0	21	10	19

5. 결 론

본 논문에서는 키이바이트 사용스케줄과 확산치환의 패턴을 적절히 바꾸어 32,64,그리고 256 비트의 블록크기를 가진 Lucifer 형태의 암호화 알고리즘을 구성하고, 이들의 심볼간상호의존성에 대해 비교 조사하였다.

라운드수가 비교적 적은 32비트와 64비트용 Lucifer 형태의 알고리즘에서는 키이에 따라서는 100%의 심볼간 상호의존도를 얻지 못할 수도 있었다.

Lucifer 형태의 알고리즘에서는 비선형이고 unaffine한 대체상자를 가지나 바이트를 연산의 기본 단위로 하므로 확산치환이 국소적으로 이루어져 전체 라운드수가 비교적 적을 때는 심볼간 상호의존성이 약하였다.그러나 치환과정으로부터 한 라운드에 출력되는 바이트들을 모두 모아 한꺼번에 랜덤하게 확산치환을 하면 키이와 정보의 심볼간 상호의존도를 약간 높일 수 있었다.

참 고 문 헌

[1] A.Sorkin, "Lucifer, a cryptographic algorithm," Cryptologia,vol.8,no.1,pp.22-35,Jan.1984

[2] National Bureau of Standards, Data Encryption Standard, U.S. FIPP PUB 46, pp.1-18,1977

[3] R.Rivest, A.Shamir, and L.Adleman, "A method for obtaining digital signatures and pubulic - key cryptosystems,"Comm.ACM,vol.21,no.2,pp.120-6,1986

[4] L.Neuwith,"A comparison of four key distribution methods,"Telecommunication,July1986

[5] C.Meyer and S.Matyas, Crypography:A New Dimension in Computer Data Security,NewYork,John Wiley Sons 1982

[6] 이훈재,문상재,"Lucifer 와 DES 에서의 심볼간 상호의존성에 관한 연구,"경북대학교 전자기술 연구소,제8권 pp.89-92,Aug.1987

[7] E.F.Brickell and et.al.,"Structure in the S-Boxes of the DES,"extended abstract performed while the author was visiting Bell Communication Research

[8] Kam,J.B., and Davida,G.I. "Structured Design of Substition-Permutation Encryption Network," IEEE Transactions on Computer,vol.28,no.10,747,1979