通信情報保護學會論文誌
第 8 卷 第 3 號, 1998. 9

DES 알고리즘에 사용된 S-box의 비선형 구조에 관한 연구

김 지홍*, 윤 석창*

A Study on Nonlinear Analysis of S-box in DES Algorithm

Ji-Hong Kim*, Suck-Chang Yun*

요 약

본 논문에서는 DES(Data Encryption Standard) 시스템에서의 핵심부분에 해당되는 f 함수에 대한 비선형 해석을 다룬다. 먼저 S-box의 입출력 형태를 비선형 결합함수 형태로 구성하여 f 함수에 대한 또 다른 형태의 분석방식을 제시한다. 이러한 분석방법은 DES 뿐만 아니라 블록 암호시스템의 안전성 문제를 분석할 수 있으며, 또한 보다 안전한 블록암호 시스템을 제안하기 위한 기초자료로 사용될 수 있을 것이다.

Abstract

In this paper, we deal with f function, which is a core part in DES system. First of all we analyse S-box structrure with nonlinear combining function and present another method to analyse f function in DES system. This analysis method will be used in proposing more secure block cipher system as well as analysing the security of block cipher system

Ⅰ. 서 론

1977년 미국 NIST 에서 표준안으로 승인된 이래 지금까지 세계 각국에서 다양한 데이타 보호용도로 응용되고 있는 DES 알고리즘은 ATM(Automatic Teller Machines, 현금자동입출력기)과 POS(Point of Sale) 단말장치에서 PIN(Personal Identification Number, 개인식별번호)의 암호화 및 데이타의 내용변조등을 막기 위해 응용되어 왔다[6,8,9]. DES 알고리즘은 평문을 64비트 단위로 블록암호화하며, 64비트의 키 블록 중 56비트가 암호화 및 복호화에

* 세명 대학교 전자공학과
이 논문은 1996년도 세명대학교 교내 학술연구비 지원에 의해 수행된 연구임.

사용된다. DES 알고리즘은 기본적으로 치환(P-box), 대치(S-box), 및 키 스케줄을 이용하여 16라운드 동안 동일한 동작과정의 반복으로 이루어 진다[4].

현재까지 블록암호 알고리즘에 대한 공격방법으로 제시된 방법은 1992년 Biham과 Shamir에 의해 제안된 Differential Cryptanalysis[1]와 일본의 Matsui에 의해 제안된 Linear Cryptanalysis[5]이 있으며, 현재에도 이러한 연구가 계속 진행되고 있다. 위의 두 방식들은 평문 입력쌍과 암호문 출력쌍과의 관계를 이용하여 확률론적으로 키를 찾는 방법을 제안한 방식이다. 그러나 이러한 암호시스템에 대한 공격방법에 대한 연구와는 달리 블록암호 시스템의 핵심부분에 해당되는 S-box에 대한 설계에 대한 연구는 거의 없었다. 이에 본 논문에서는 Rueppel이 제안한 비선형 등가회로 방식[7]을 적용하여 S-box의 입출력관계를 C언어 프로그램을 이용하여 일종의 비선형 함수 형태로 구하여, f함수에 대한 등가 시스템 구조를 알아낸다.

본 논문의 결과는 현재 진행되고 있는 블록암호시스템의 표준화와 함께 보다 안전한 블록 암호시스템을 시스템으로 구현하기 위한 기초자료로 사용될 수 있을 것으로 기대된다.

Ⅱ. 비선형 결합구조와 DES의 f함수

2.1. 비선형 결합구조

Rueppel은 키스트림 생성기[7]를 최대장 계열을 생성할 수 있는 L단 LFSR 시스템에 대하여 선형복잡도와 출력계열의 비예측성을 높히기 위하여 비선형결합함수(nonlinear function)를 적용하여 구현하였다. L차 생성다항식에 의해 구성되는 LFSR 시스템의 출력계열의 주기 p는 2^L-1이다. 일정한 초기조건하에서 발생되는 출력계열을 살펴보면 비선형 결합함수의 차수가 1차일때, 즉 선형시스템의 경우에 대한 출력계열의 갯수는 각 단에서 생성되는 각각의 출력계열을 생성할 수 있기 때문에 전체 갯수는 ${}_LC_1$ 이다. 비선형 결합함수의 차수가 2차일때 출력계열의 갯수는 L단 LFSR 시스템의 두개의 단에서 생성되는 출력들의 곱으로 생성되며, 이러한 계열들의 갯수는 ${}_LC_2$ 이며, .. 비선형 차수가 k차인 경우, 출력계열의 갯수는 ${}_LC_k$ 이다. 따라서 L단 LFSR 시스템에 대한 1차, 2차, .. L차의 비선형 결합함수의 형태의 전체 갯수는 식(1)과 같다.

$$ {}_LC_1+{}_LC_2+{}_LC_3+..{}_LC_L=2^L-1=p \qquad (1)$$

이러한 p개의 비선형 결합함수에 의해 생성되는 계열들에 대하여 선형결합을 적용하면 주기 p이내의 2^p-1 개(영계열 제외)의 모든계열을 생성할 수 있다. 따라서 비선형결합기를 사용한 구조의 선형복잡도는 최대 (2^L-1)까지 가능하며, 주어진 비선형 차수에 따라서 다양한 선형복잡도를 얻을 수 있다.

만약 최대차수 k차의 비선형 결합함수를 적용한 경우의 k차 비선형 결합함수의 형태는 식(2)와 같이 표시할 수 있다.

$$\begin{aligned} f(x) = {} & a_0+a_1x_1+a_2x_2+a_3x_3+a_Nx_N \\ & +a_{12}x_1x_2+a_{13}x_1x_3+a_{14}x_1x_4+\cdots \\ & +a_{123}x_1x_2x_3+a_{124}x_1x_2x_4+a_{125}x_1x_2x_5+\cdots \\ & +a_{1234}x_1x_2x_3x_4+a_{12345}x_1x_2x_3x_5+\cdots \\ & +\cdots\cdots \\ & +a_{123..k}x_1x_2x_3x_4\cdots x_k\ldots+a_{N-k+1..N}x_{N-k+1N}x_{N-k}\ldots x_N \end{aligned} \qquad (2)$$

비선형 결합기를 사용한 출력계열 Z는 p개의 비선형 함수에 의해 생성된 계열들의 선형결합 형태로 구성되므로, 이는 $p\times p$ 형태의 곱행렬 P와 이들에 대한 계수행렬 A의 곱의 형태로 표시될 수 있다[7].

$$Z = P^T \cdot A$$
$$A = (P^T)^{-1} \cdot Z \qquad (3)$$

식(3)은 임의의 출력계열 행렬과 주어진 초기상태를 이용하여 생성된 곱행렬의 역행렬을 곱하면, 주어진 초기상태하에서 키스트림 생성기에서 사용된 비선형 결합함수를 찾을 수 있음을 나타낸다.

2.2. DES의 f함수 구조

DES 알고리즘[4]은 64비트 블록단위로 암호화되며 64비트의 키 블록중 56비트가 암호화 및 복호화에 사용된다. DES 알고리즘은 기본적으로 치환(P-box), 대치(S-box), 및 키 스케줄을 이용하여 16라운드 동안 동일한 동작과정의 반복으로 이루어 진다. 암호화 과정은 64비트의 평문을 초기치환(IP)시킨후, 좌우 32비트씩 양분되며, 오른쪽 32비트는 다음 라운드의 왼쪽부분을 차지하게 되고 전체 16 라운드의 오른쪽 부분과 암호함수 f를 적용한 결과가 전 라운드의 왼쪽부분과 XOR되어 새로운 라운드의 오른쪽 부분을 차지하게 된다. 암호함수 f는 그림 1과 같이 비트 확장표 E와 치환표 P 및 비선형구조를 갖는 lookup table 형식의 S-box로 구성되어 있다.

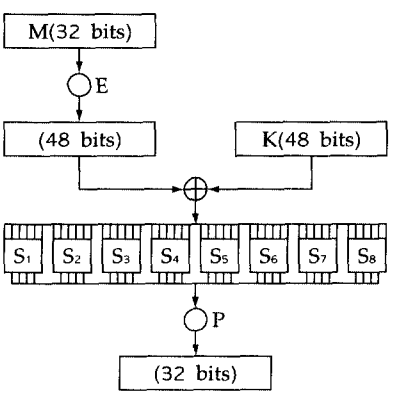


그림 1. f 함수 구조
Fig. 1. The structure of f function

Ⅲ. f함수의 비선형 구조분석

3.1. S-box에 대한 비선형 결합방정식

본 절에서는 DES 알고리즘[4]의 핵심부분에 해당되는 비선형구조의 S-box 구조를 분석하기 위하여 전 장에서 설명된 비선형 결합기를 이용한다. 원래 S-box구조는 table look-up 방식을 사용하지만, 이를 비선형 결합함수를 이용한 동가시스템을 구성한다. S-box_1 의 선택테이블은 표 1과 같으며 입력으로 주어지는 6비트중에서 처음과 마지막 비트는 행(row) 번호를 표시하며, 가운데 4비트는 열(column) 번호를 의미한다. 예로서 입력이 "101100"이라면, 행 번호는 "10", 즉 2행을 의미하며, 열 번호는 "0110", 즉 6열을 의미한다. 따라서 표 1에 의하면 2행 6열의 데이타인 "2"가 출력된다.

표 1. S-box_1 선택 테이블
Table 1. S-box_1 Selection Table

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
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-box의 6비트 입력에 대한 4비트 출력과의 관계를 분석하기 위하여, table look-up 방식으로 표시된 S-box를 비선형 결합함수로 파악하기 위하여 웨이트 분포에 의한 입력과 출력관계로 표시할 수 있다.

웨이트가 "1"인 입력(32,16,8,4,2,1)을 우선으로 하고, 다음으로 웨이트가 "2"인 입력(48,40,36,34,33,24,20,18,17,12,10,9,6,5,3), 웨이트 "3"인 입력(56,52,50,49,44,42,41,38,37,35,28,26,25,22,21,19,14,13,11,7), 웨이트 "4"인 입력(60,58,57,

54,53,51,46,45,43,39,30,29,27,23,15), 웨이트가 "5"인 입력 (62,61,59,55,47,31), 그리고 마지막으로 웨이트 가 "6"인 입력(63)으로 순서를 정하였고 이를 곱행렬 형태로 구성함으로서, S-box_1의 4개의 출력비트에 대한 6개의 입력함수와의 비선형결합함수를 구하면 다음과 같다.

$$y_{11} = 1+x_1+x_2+x_3+x_5+x_6+x_1x_4+x_1x_5+x_2x_3+x_3x_4 +x_1x_2x_3+x_1x_2x_4+x_1x_3x_4+x_1x_3x_5+x_1x_4x_6+x_2x_3x_4 +x_3x_4x_5+x_3x_4x_6+x_4x_5x_6+x_1x_2x_3x_4+x_1x_2x_4x_5+ x_1x_2x_4x_6+x_1x_2x_5x_6+x_1x_3x_4x_5+x_1x_3x_4x_5+x_1x_2x_3x_4x_6+x_1x_2x_3x_5x_6 \quad (4)$$

$$y_{12} = 1+x_2+x_3+x_6+x_1x_2+x_1x_3+x_1x_5+x_1x_6+x_2x_4+ x_2x_6+x_3x_5+x_4x_5+x_4x_6+x_5x_6+x_1x_2x_3+x_1x_2x_5+ x_1x_2x_6+x_1x_3x_4+x_1x_4x_5+x_2x_3x_6+x_2x_4x_5+x_2x_4x_6 +x_3x_4x_6+x_3x_5x_6+x_1x_2x_3x_4+x_1x_2x_3x_5+x_1x_2x_3x_6 +x_1x_3x_5x_6+x_3x_4x_5x_6+x_1x_2x_3x_4x_6+x_1x_2x_3x_5x_6+ x_1x_2x_4x_5x_6+x_1x_3x_4x_5x_6$$

$$y_{13} = 1+x_1+x_4+x_5+x_6+x_1x_2+x_1x_5+x_2x_3+x_2x_4+x_2x_5 +x_2x_6+x_3x_4+x_3x_5+x_3x_6+x_4x_5+x_1x_2x_3+x_1x_2x_4 +x_1x_2x_6+x_1x_3x_4+x_1x_5x_6+x_2x_3x_4+x_2x_3x_5+x_2x_3x_6+x_2x_4x_6+x_3x_4x_6+x_4x_5x_6+x_1x_2x_3x_4+x_1x_2x_3x_5 +x_1x_2x_3x_6+x_1x_2x_4x_5+x_1x_2x_4x_6+x_1x_2x_5x_6+x_2x_3x_4x_6+x_2x_4x_5x_6+x_1x_2x_3x_4x_6+x_1x_2x_4x_5x_6+x_1x_3x_4x_5x_6$$

$$y_{14} = x_2+x_4+x_1x_3+x_1x_4+x_1x_5+x_1x_6+x_2x_5+x_2x_6+x_3x_5 +x_5x_6+x_1x_2x_3+x_1x_2x_5+x_1x_3x_4+x_1x_3x_5+x_1x_4x_5+ x_1x_4x_6+x_1x_5x_6+x_2x_3x_6+x_2x_4x_5+x_2x_4x_6+x_1x_2x_3x_4 +x_1x_2x_4x_5+x_1x_2x_5x_6+x_1x_3x_4x_5+x_1x_3x_4x_6+x_2x_3x_5x_6+x_1x_2x_3x_4x_6+x_1x_2x_3x_5x_6+x_1x_3x_4x_5x_6$$

마찬가지로 8개의 S-box에 대한 각각의 출력 y_{n1}, y_{n2}, y_{n3}, y_{n4}에 관한 비선형결합식은 부록에 첨부하였다.

위에서 보인 예제에 대하여, 본 논문에서 제시한 비선형함수를 이용하여 S-box의 출력을 구한다. S-box_1의 입력이 "101100" 이라면, 비선형 함수의 입력은 $x_1 = x_3 = x_4 = 1$, $x_2 = x_5 = x_6 = 0$이다. S-box_1의 출력 4비트는 y_{11}, y_{12}, y_{13}, y_{14}이며 이들의 값을 식(4)에 대입하면, $y_{11} = 0$, $y_{12} = 0$, $y_{13} = 1$, $y_{14} = 0$를 찾을 수 있다. 따라서 표 1에 의한 2행 6열의 데이타인 "2"와 동일한 출력이 나타난다.

3.2. f함수 분석

DES 암호화 과정은 64비트 평문을 초기치환(IP)시킨후, 좌우 32비트씩 양분되며, 오른쪽 32비트는 다음 라운드의 왼쪽부분을 차지하게 되고, 전체 16 라운드의 오른쪽 부분과 암호함수 f를 적용한 결과가 전 라운드의 왼쪽부분과 XOR되어 새로운 라운드의 오른쪽 부분을 차지하게 된다.

매 라운드에서 사용되는 암호함수 f는 그림 1과 같이 구성되며, 비트 확장표 E의 출력과 키스트림 함수 K의 출력과 이진합되어, S-box의 lookup table 구조를 통과한 후, 마지막으로 치환표 P에 의해 선형치환되는 구조로 되어있다.

먼저 f함수에 입력되는 32 비트열을 M, E함수의 출력을 E, 48 비트의 키스트림을 K, P함수의 출력, 즉 최종 f함수의 출력을 P, 각각의 S-box 함수의 출력을 y_{n1}, y_{n2}, y_{n3}, y_{n4}이라 정의한다.

$$M = [m_1, m_2, m_3, m_4, \ldots, m_{29}, m_{30}, m_{31}, m_{32}] \quad (5)$$
$$E = [e_1, e_2, e_3, e_4, \ldots, e_{45}, e_{46}, e_{47}, e_{48}]$$
$$K = [k_1, k_2, k_3, k_4, \ldots, k_{45}, k_{46}, k_{47}, k_{48}]$$
$$P = [p_1, p_2, p_3, p_4, \ldots, p_{29}, p_{30}, p_{31}, p_{32}]$$

f함수의 최종출력의 제1비트 p_1은 S-box_4의 출력, 즉, y_{44}에 해당되므로, 이는 S-box_4의 6개의 입력비트와 연관된다. 따라서 E함수의 6

비트의 출력값(e_{19}, e_{20}, e_{21}, e_{22}, e_{23}, e_{24})과 K함수의 6비트의 출력값(k_{19}, k_{20}, k_{21}, k_{22}, k_{23}, k_{24})과 연관된다. 이는 초기의 6개의 입력비트 (m_{12}, m_{13}, m_{14}, m_{15}, m_{16}, m_{17})와 관계된다. 결과적으로 f함수의 최종출력중 제1비트는 입력 32비트중 6개의 입력비트와 키스트림 48비트중 6개의 입력비트와 관련이 있으며, 다른 비트들의 영향은 전혀 없는 것으로 나타난다. 표 2는 f함수의 최종 출력계열을 입력 M과 키스트림 K, S-box의 비선형 함수와의 관계를 나타낸다.

표 2. f함수의 출력

Table 2. The output of f function

f함수의 출력	입력(M)과의 연관된 6비트	키스트림(K)과의 연관된 6비트	S-box 출력함수
p_1	m_{12} - m_{17}	k_{19} - k_{24}	y_{44}
p_2	m_4 - m_9	k_7 - k_{12}	y_{23}
p_3	m_{16} - m_{21}	k_{25} - k_{30}	y_{54}
p_4	m_{20} - m_{25}	k_{31} - k_{36}	y_{61}
p_5	m_{28} - m_1	k_{43} - k_{48}	y_{81}
p_6	m_8 - m_{13}	k_{13} - k_{18}	y_{34}
p_7	m_{24} - m_{29}	k_{37} - k_{42}	y_{74}
p_8	m_{16} - m_{21}	k_{25} - k_{30}	y_{51}
p_9	m_{32} - m_5	k_1 - k_6	y_{11}
p_{10}	m_{12} - m_{17}	k_{19} - k_{24}	y_{43}
p_{11}	m_{20} - m_{25}	k_{31} - k_{36}	y_{53}
p_{12}	m_{24} - m_{29}	k_{37} - k_{42}	y_{72}
p_{13}	m_4 - m_9	k_7 - k_{12}	y_{21}
p_{14}	m_{16} - m_{21}	k_{25} - k_{30}	y_{52}
p_{15}	m_{28} - m_1	k_{43} - k_{48}	y_{83}
p_{16}	m_8 - m_{13}	k_{13} - k_{18}	y_{32}
p_{17}	m_{32} - m_5	k_1 - k_6	y_{12}
p_{18}	m_4 - m_9	k_7 - k_{12}	y_{24}
p_{19}	m_{20} - m_{25}	k_{31} - k_{36}	y_{64}
p_{20}	m_{12} - m_{17}	k_{19} - k_{24}	y_{42}
p_{21}	m_{28} - m_1	k_{43} - k_{48}	y_{84}
p_{22}	m_{24} - m_{29}	k_{37} - k_{42}	y_{73}
p_{23}	m_{32} - m_5	k_1 - k_6	y_{13}
p_{24}	m_8 - m_{13}	k_{13} - k_{18}	y_{31}
p_{25}	m_{16} - m_{21}	k_{25} - k_{30}	y_{33}
p_{26}	m_{12} - m_{17}	k_{19} - k_{24}	y_{41}
p_{27}	m_{28} - m_1	k_{43} - k_{48}	y_{82}
p_{28}	m_4 - m_9	k_7 - k_{12}	y_{22}
p_{29}	m_{20} - m_{25}	k_{31} - k_{36}	y_{62}
p_{30}	m_8 - m_{13}	k_{13} - k_{18}	y_{33}
p_{31}	m_{32} - m_5	k_1 - k_6	y_{14}
p_{32}	m_{24} - m_{29}	k_{37} - k_{42}	y_{71}

$m_{12}\oplus k_{19}=x_1$, $m_{13}\oplus k_{20}=x_2$, $m_{14}\oplus k_{21}=x_3$, $m_{15}\oplus k_{22}=x_4$, $m_{16}\oplus k_{23}=x_5$, $m_{17}\oplus k_{24}=x_6$이라면, f함수의 출력비트중 제1비트 p_1의 경우에는 식(6)으로 표시될 수 있다.

$$p_1=f_{y44}(x_1,\ x_2,\ x_3,\ x_4,\ x_5,\ x_6) \qquad (6)$$

또한 부록에 표시된 비선형 함수y_{44}의 계수행렬을 A[y_{44}]라고 하면

A[y_{44}]:1 101100 110011011000111 11100110110 011010000 101100000101011 001010 0

위의 A[y_{44}]에 대한 비선형 결합함수를 분석하면, 상수항이 "1"이며, 1차항은" $x_1+x_3+x_4$", 2차항은 "$x_1x_2+x_1x_3+x_1x_6+x_2x_3+x_2x_5+x_2x_6+x_4x_5+x_4x_6+x_5x_6$ "3차항의 경우에는 "$x_1x_2x_3+x_1x_2x_4+x_1x_2x_5+x_1x_3x_5+x_1x_3x_6+x_1x_4x_6+x_1x_5x_6+x_2x_3x_6+x_2x_4x_5+x_2x_5x_6$ "4차항의 경우에는 "$x_1x_2x_3x_4+x_1x_2x_3x_6+x_1x_2x_4x_5+x_1x_4x_5x_6+x_2x_3x_4x_6+x_2x_4x_5x_6+x_3x_4x_5x_6$" 5차항의 경우에는 "$x_1x_2x_3x_5x_6+x_1x_3x_4x_5x_6$", 6차항의 경우에는 없다. 따라서 이를 함수식으로 표시하면 다음과 같다.

$$\begin{aligned}y_{44}=1&+x_1+x_3+x_4+x_1x_2+x_1x_3+x_1x_6+x_2x_3+x_2x_5\\&+x_2x_6+x_4x_5+x_4x_6+x_5x_6+x_1x_2x_3+x_1x_2x_4+x_1x_2x_5\\&+x_1x_3x_5+x_1x_3x_6+x_1x_4x_6+x_1x_5x_6+x_2x_3x_6+x_2x_4x_5\\&+x_2x_5x_6+x_1x_2x_3x_4+x_1x_2x_3x_6+x_1x_2x_4x_5+x_1x_4x_5x_6\\&+x_2x_3x_4x_6+x_2x_4x_5x_6+x_3x_4x_5x_6+x_1x_2x_3x_5x_6+x_1x_3\\&x_4x_5x_6.\end{aligned}$$

키 계열중 초기 6비트의 키입력 K(k_1,k_2,k_3,k_4,

k_5,k_6)는 표 2와 같이 f함수의 입력중 6개의 입력비트 M($m_{32},m_1,m_2,m_3,m_4,m_5$)과 관련이 있으며, 이는 곧 f함수의 출력 p의 p_9, p_{17}, p_{23}, p_{31}에 영향을 준다.

$$p_9 = f_{y_{11}}(m_{32}\oplus k_1,\ m_1\oplus k_2,\ m_2\oplus k_3,\ m_3\oplus k_4,\ m_4\oplus k_5,\ m_5\oplus k_6) \quad (7)$$

$$p_{17} = f_{y_{12}}(m_{32}\oplus k_1,\ m_1\oplus k_2,\ m_2\oplus k_3,\ m_3\oplus k_4,\ m_4\oplus k_5,\ m_5\oplus k_6)$$

$$p_{23} = f_{y_{13}}(m_{32}\oplus k_1,\ m_1\oplus k_2,\ m_2\oplus k_3,\ m_3\oplus k_4,\ m_4\oplus k_5,\ m_5\oplus k_6)$$

$$p_{31} = fy_{14}(m_{32}\oplus k_1,\ m_1\oplus k_2,\ m_2\oplus k_3,\ m_3\oplus k_4,\ m_4\oplus k_5,\ m_5\oplus k_6)$$

따라서 1개의 f함수의 입력계열과 최종 출력계열쌍을 알고 있다고 가정하면, 비선형 함수방법을 이용하여 식(7)을 만족하는 메시지와 키쌍을 찾을 수 있다.

3.3. 제안된 방법을 이용한 f 함수 해석

3.2절의 내용을 설명하기 위하여 다음과 같은 예를 보인다. 32비트의 f함수 입력 M과 48비트의 키 입력 K이 다음과 같을 때, DES 암호화 방식에 의하여 계산된 f함수 출력 P는 다음과 같다.

M = (0001 0001 0001 0001 0001 0001 0001 0001)

K = (0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100)

P = (1100 0110 1101 0011 0110 1101 0010 1101)

만일 입력 M과 출력 P만을 알고 있다고 가정하고, 전 절에서 제시한 방법을 이용하여 분석하기로 한다. 키 입력 비트중 초기의 k_1, k_2, k_3, k_4, k_5, k_6을 알고자 할 경우에는 입력 M과 K중에서 식(7)을 이용한다. 키 계열중 초기 6비트의 키입력 K(k_1,k_2,k_3,k_4,k_5,k_6)는 표 2에서와 같이 f함수의 입력 M중에서 6개의 입력비트 M($m_{32},m_1,m_2,m_3,m_4,m_5$)와 f함수의 출력 P중 4개의 출력비트 P(p_9,p_{17},p_{23},p_{31}) 만이 관련이 있으며, 다른 입 출력값과는 무관하다.

따라서 입력 ($m_{32}=1$, $m_1=0$, $m_2=0$, $m_3=0$, $m_4=1$, $m_5=0$)과 $p_9=1$, $p_{17}=0$, $p_{23}=0$, $p_{31}=0$에 대하여 각각 f_{y11}, f_{y12}, f_{y13}, f_{y14} 함수를 적용할 수 있다. $m_{32}\oplus k_1$을 x_1, $m_1\oplus k_2$를 x_2, $m_2\oplus k_3$를 x_3, $m_3\oplus k_4$를 x_4, $m_4\oplus k_5$를 x_5, $m_5\oplus k_6$을 x_6이라면, 식(7)은 식(8)과 같이 표시될 수 있다.

$$p_9 = f_{y_{11}}(x_1,\ x_2,\ x_3,\ x_4,\ x_5,\ x_6) = 1 \quad (8)$$

$$p_{17} = f_{y_{12}}(x_1,\ x_2,\ x_3,\ x_4,\ x_5,\ x_6) = 0$$

$$p_{23} = f_{y_{13}}(x_1,\ x_2,\ x_3,\ x_4,\ x_5,\ x_6) = 0$$

$$p_{31} = f_{y_{14}}(x_1,\ x_2,\ x_3,\ x_4,\ x_5,\ x_6) = 0$$

식(7)을 이용하여 6개의 입력에 대한 4개의 비선형 함수를 적용한 결과를 알고 있으므로, 이를 이용하여 6개의 입력의 총 가짓수 $2^6=64$ 개중 위의 4가지 조건을 만족시키는 경우를 찾으면 실제로 다음과 같은 4가지 경우가 존재한다.

001110, $p_9=1$, $p_{17}=0$, $p_{23}=0$, $p_{31}=0$ ① (9)

011111, $p_9=1$, $p_{17}=0$, $p_{23}=0$, $p_{31}=0$ ②

100101, $p_9=1$, $p_{17}=0$, $p_{23}=0$, $p_{31}=0$ ③

100110, $p_9=1$, $p_{17}=0$, $p_{23}=0$, $p_{31}=0$ ④

식(9)에서 ①항을 만족하는 키계열을 찾기 위해서는, 입력계열 $x_1=m_{32}\oplus k_1=1$, $x_2=m_1\oplus k_2=0$, $x_3=m_2\oplus k_3=0$, $x_4=m_3\oplus k_4=1$, $x_5=m_4\oplus k_5=0$, $x_6=m_5\ \oplus k_6=1$을 만족시키는 키계열을 찾기 위하여, $m_{32}=1$, $m1=0$, $m_2=0$, $m_3=0$, $m_4=1$, $m_5=0$를 대입하면, $k_1=0$, $k_2=0$, $k_3=0$, $k_4=1$, $k_5=1$, $k_6=1$을 찾을 수 있다. 따라서 키 비트계열 K의 초기 6 비트의 값은 (000111)이 된다.

마찬가지 방법으로 추측가능한 키 비트계열을 모두 구하면 다음과 같다.

$$001110 \oplus 100010 = 101100 \quad ① \qquad (10)$$
$$011111 \oplus 100010 = 111101 \quad ②$$
$$100101 \oplus 100010 = 000111 \quad ③$$
$$100110 \oplus 100010 = 000100 \quad ④$$

위의 4 가지 경우중에 본 논문에서 가정한 6비트 키계열은 ④항에 해당된다. 만일 다음과 같이, f함수의 입력과 출력쌍을 한 개를 더 알고 있다고 가정한다.

M = (0001 0010 0011 0100 0101 0110 0111 1000)

K = (0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100)

P = (0110 1010 0011 0011 0101 1001 1001 0111)

위의 방법을 이용하여, $m_{32}=0$, $m_1=0$, $m_2=0$, $m_3=0$, $m_4=1$, $m_5=0$와 $p_9=0$, $p_{17}=0$, $p_{23}=0$, $p_{31}=1$을 만족하는 입력계열은 다음과 같다.

$$000110,\ p_9=0,\ p_{17}=0,\ p_{23}=0,\ p_{31}=1 \quad ① \qquad (11)$$
$$001111,\ p_9=0,\ p_{17}=0,\ p_{23}=0,\ p_{31}=1 \quad ②$$
$$100010,\ p_9=0,\ p_{17}=0,\ p_{23}=0,\ p_{31}=1 \quad ③$$
$$101101,\ p_9=0,\ p_{17}=0,\ p_{23}=0,\ p_{31}=1 \quad ④$$

식(11)을 만족하는 키계열을 찾기 위해서는, $x1 = m_{32} \oplus k_1$, $x_2 = m_1 \oplus k_2$, $x_3 = m_2 \oplus k_3$, $x_4 = m_3 \oplus k_4$, $x_5 = m_4 \oplus k_5$, $x_6 = m_5 \oplus k_6$을 적용하여 키 비트계열을 구하면 다음과 같다.

$$000110 \oplus 000010 = 000100 \quad ① \qquad (12)$$
$$001111 \oplus 000010 = 001101 \quad ②$$
$$100010 \oplus 000010 = 100000 \quad ③$$
$$101101 \oplus 000010 = 101111 \quad ④$$

따라서 식(10)과 식(12)를 동시에 만족하는 키비트 계열은 "000100"이다.

위의 예는 초기 6비트의 키를 알기 위하여 진행된 과정이며, 마찬가지 방법을 적용하면, 48 비트의 키비트를 추정할 수 있을 것이다. 또한 만일 다수의 f함수 입력과 출력쌍을 알고 있다면 이와 같은 방법으로 모든 키비트를 찾을 수 있다.

Ⅳ. 결 론

전 장에서 f함수의 입출력관계를 설명한 바와 같이, f함수의 각각의 출력비트들은 입력계열 M 32 비트중 6개의 비트와 키스트림 계열 K 48 비트중 6개의 비트들과 관련되며, 나머지 비트들과는 전혀 관련없이 나타난다. 이와같은 결론을 이용하여 Biham 과 Shamir의 "Differential Cryptanalysis"를 적용하면 출력비트와 상관관계가 있는 입력비트들 만을 이용함으로서 평문쌍에 대한 암호문쌍과의 관계를 분석할 수 있을 것으로 기대된다. 또한 본 논문에서 나타난 1단 f함수의 결과를 16단으로 확장하여 해석하면, DES 뿐만 아니라, 블록 암호시스템의 안전성 문제를 분석할 수 있을 뿐 아니라, 현재 진행되고 있는 블록암호시스템의 표준화 및 블록암호시스템을 이용한 스마트 카드 개발과 더불어 더욱 더 안전한 블록 암호시스템을 제안하기 위한 기초자료로서 활용될 수 있을 것으로 기대된다.

Ⅴ. 참고 문헌

[1] E.Biham and A.Shamir, "Differential Cryptanalysis of the Full 16-round DES", Proc. of Crypto'92, pp12.1-12.5, 1992.

[2] E.F.Brickell, J.H.Moore and M.R.Purtill, "Structure in the S-boxes of the DES",

Proc. of Crypto' 86, pp.3-7, 1986.

[3] C.H.Meyer and S.M. Matyas, Cryptography : A New Dimension in Computer Data Security, John Wiley & Sons, New York, 1982.

[4] NBS, Data Encryption Standard, FIPS Pub-46, 1977.

[5] M.Matsui, "Linear Cryptanalysis of DES Cipher", SCIS93, pp.83-98, 1993.

[6] M.Y.Rhee, Cryptography and Secure Communications, McGraw-Hill, 1993.

[7] R.A.Rueppel, Analysis and Design of Stream Ciphers, Springer- Verlag, Berlin, Germany, 1986

[8] B.Schneier, Applied Cryptography : Protocols, Algorithms, and Source Code in C, Wiley, 1994.

[9] J.Seberry and J. Pieprzyk, Cryptography : An Introduction to Computer Security, Prentice Hall, Sydney, 1989.

(부록 : S-box의 출력 y_{n1}, y_{n2}, y_{n3}, y_{n4}에 관한 비선형 결합식의 계수)

S-box_1 :

A[y_{11}] 1 111011 001101000100000 1100110010 1000001101 100111110000000 011000 0

A[y_{12}] 1 011001 110110101010111 1011100100 0011100110 111000001000001 011110 0

A[y_{13}] 1 100111 100101111111100 1101100001 1110100101 111111000001010 011110 0

A[y_{14}] 0 010100 011110011010001 1010110111 0011100000 100101110000100 011010 0

S-box_2 :

A[y_{21}] 1 101011 000001101000100 1001000101 0011000000 001101001100000 000100 0

A[y_{22}] 1 110111 000001100001000 0000000000 0000100001 010100000000001 001100 0

A[y_{23}] 1 110110 110000000110000 1111110101 0010111100 110010110000010 001100 0

A[y_{24}] 1 101100 110010001011000 0010011001 0101000001 001011001100110 001000 0

S-box_3 :

A[y_{31}] 1 011010 111010100110110 1100100110 1101000001 100100001100111 000110 0

A[y_{32}] 0 101001 100001111010110 1111000000 1110110001 111001000100100 001010 0

A[y_{33}] 1 110111 001011110111010 0111010110 1011011111 100011011101001 010110 0

A[y_{34}] 0 110101 110110000010100 1011010110 0000000000 011001000000000 010000 0

S-box_4 :

A[y_{41}] 0 100111 001001011011011 0110100011 0101010001 110101111001110 001010 0

A[y_{42}] 1 111000 000101001011111 0100010011 1000010001 100101011001110 001010 0

A[y_{43}] 1 011011 101110001000111 0110011111
1010011000 111100100101011 001010 0
A[y_{44}] 1 101100 110011011000111 1110011011
0011010000 101100000101011 001010 0

S-box_5 :
A[y_{51}] 0 010011 010100100101111 0100001011
0011101100 101111101000101 000100 0
A[y_{52}] 0 101111 000000100001000 1001000101
0010000110 101010100101001 010000 0
A[y_{53}] 1 110110 011010010111111 1111111101
1101111100 100000110001101 011110 0
A[y_{54}] 0 001000 111010111011101 1111001100
0100111110 111101110000011 001010 0

S-box_6 :
A[y_{61}] 1 010010 010111000101111 0000011011
0000001111 001010001101001 011100 0
A[y_{62}] 1 111111 010000100010000 1000001100
0000001000 011100101100010 011000 0
A[y_{63}] 0 000101 110111000010000 1000011000
0101010001 000100001100000 001100 0

A[y_{64}] 0 101010 000011100100000 0001110100
1000001101 001010110101011 000110 0

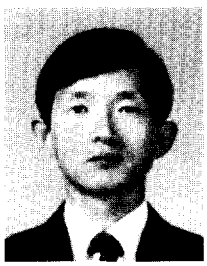
S-box_7 :
A[y_{71}] 0 001011 101111100000000 1100011001
1010001000 111100100101001 010010 0
A[y_{72}] 1 110110 111011101000000 0000000000
0000000000 101010100000011 000100 0
A[y_{73}] 0 011110 010110000001101 1100010001
0001000101 011100011001011 011010 0
A[y_{74}] 0 111011 000001000100100 0000000010
0000101000 001011010100010 000010 0

S-box_8 :
A[y_{81}] 1 101011 000010111100010 0000111101
1001110101 101010010101000 011100 0
A[y_{82}] 1 010111 011001110010000 1110110011
0001000000 100100011000000 010000 0
A[y_{83}] 0 111010 001100001010100 0010010111
0010110000 010000000101010 001100 0
A[y_{84}] 1 011110 010110000000111 1100010000
0011010110 010111011100001 011000 0

□ 著者紹介

김 지 홍

1982년 2월 한양대학교 공과대학 전자공학과(공학사)
1984년 2월 한양대학교 대학원 전자통신공학과(공학석사)
1996년 2월 한양대학교 대학원 전자통신공학과(공학박사)
1984년 - 1991년 금성전선 연구소 근무
1991년 3월 - 현재 세명대학교 전자공학과 부교수



윤 석 창

1975년 2월 한양대학교 공과대학 전자공학과(공학사)
1977년 2월 성균관대학교 대학원 전기공학과(공학석사)
1988년 2월 성균관대학교 대학원 전자공학과(공학박사)
1981년 3월 - 1991년 2월 안양 전문대학 교수
1991년 3월 - 현재 세명대학교 전자공학과 부교수