SEED 형식 암호에서 S 박스와 G 함수 구성에 관한 연구

종신회원 송 홍 복*, 정회원 조 경 연**

A study on the constitution of S box and G function in SEED-type cipher

Hong-bok Song*, Gyeong-yeon Cho** *Regular Members*

요 약

본 논문에서는 우리나라 128 비트 블록 암호 알고리즘 표준인 SEED와 유사한 구조를 가지는 암호방식에서 최적화한 S 박스와 G 함수를 구성하는 방식을 제안한다. S 박스는 비선형함수와 아핀변환으로 구성한다. 비선형함수는 차분공격과 선형공격에 강한 특성을 가지며, '0'과 '1'을 제외하고 입력과 출력이 같은 고정점과 출력이 입력의 1의 보수가 되는 역고정점을 가지지 않는 GF(2^8) 상의 역수로 구성한다. 아핀변환은 입력과 출력간의 상관을 최저로 하면서 고정점과 역고정점이 없도록 구성한다. G 함수는 4개의 S 박스 출력을 GF(2^8) 상의 4×4 행렬식을 사용하여 확산선형변환한다. G 함수는 MDS(Maximum Distance Separable) 코드를 생성하고, SAC(Strict Avalanche Criterion)를 만족하고, 고정점과 역고정점 및 출력이 입력의 2의 보수가 되는 약한 입력이 없으며, 하드웨어 구현이 용이하도록 구성한다. 본 논문에서 제안한 S 박스와 G 함수는 차분공격과 선형공격에 강하고, 약한 입력이 없으며, 하드웨어 구현이 용이하며, 확산 특성이 우수하므로 안전성이 높은 암호 방식의 구성 요소로 활용할 수 있다.

ABSTRACT

In this paper, a way of constituting optimized S box and G function was suggested in the block cipher whose structure is similar to SEED, which is KOREA standard of 128-bit block cipher. S box can be formed with nonlinear function and an affine transform. Nonlinear function must be strong with differential attack and linear attack, and it consists of an inverse number over GF(2^8) which has neither a fixed point, whose input and output are the same except 0 and 1, nor an opposite fixed number, whose output is one's complement of the input. Affine transform can be constituted so that the input/output correlation can be the lowest and there can be no fixed point or opposite fixed point. G function undergoes diffusive linear transform with 4 S-box outputs using the matrix of 4×4 over GF(2^8). G function can be constituted so that MDS(Maximum Distance Separable) code can be formed, SAC(Strict Avalanche Criterion) can be met, there can be no weak input, where a fixed point, an opposite fixed point, and output can be two's complement of input, and the construction of hardware can be made easy. The S box and G function suggested in this paper can be used as a constituent of the block cipher with high security, in that they are strong with differential attack and linear attack with no weak input and they are excellent at diffusion.

Ⅰ. 서 론

1977년 IBM이 개발하고, 미국 정부에 의해 수정되어 미국 정부의 암호 표준으로 채택된 DES(Data Encryption Standard)[1]는 최근까지 널리 사용되고 있는 암호 알고리즘이었다. 그러나 최근 컴퓨터 계

* 동의대학교 전기.전자.정보통신.메카트로닉스 공학부(hbsong@dongeui.ac.kr)
** 부경대학교 전자 컴퓨터 정보통신 공학부(gycho@pknu.ac.kr)
논문번호 : 020013-0112, 접수일자 : 2002년 1월 16일

산 능력과 암호 해독 기술의 발달로 인해 DES가 해독되는 등 보안, 관리상의 취약점 및 문제점이 발견되었다.

이에 미국 국립 표준 기술연구소(NIST, National Institute Standards & Technology)에서는 새로운 표준 블록 암호 알고리즘을 선정하기 위해 1997년 초 새로운 표준 암호 알고리즘(AES, Advanced Encryption Standard) 선정 프로젝트를 발표했다. NIST는 차세대 표준 암호 알고리즘으로 128비트 블록 암호 알고리즘, 다양한 길이의 키(128, 192, 256비트)를 사용할 수 있고, 약한 키를 갖지 않으며, 소프트웨어와 하드웨어 상에서 효율적이며, 스마트 카드 상에서도 동작할 수 있는 알고리즘 등의 기준을 제시했다. AES 선정 프로젝트는 전 세계적으로 후보 알고리즘을 공모하였고, 여러 차례 평가에 의해 5개의 후보 알고리즘(Rijndael, Twofish, MARS, RC6, Serpent)을 선정하였다. 5개의 후보 중 NIST 자체평가와 전 세계적인 공개 검증을 통해 2000년 10월에 최종적으로 Rijndael이 선정되었다[2][3].

한편 암호 알고리즘에 대한 대부분의 연구는 미국, 유럽 국가 등 몇몇 암호 선진국에서 주도하고 있으며, 정보의 유출을 방지하기 위해서 암호 기술 및 제품에 대한 수출을 규제하고 있다. 이에 자국의 정보를 보호하기 위하여 각 나라들은 자체적인 암호 알고리즘을 연구하고 있으며, 우리나라에서는 한국정보보호진흥원을 주축으로 관련 전문가들과 공동으로 128 비트 블록 암호 알고리즘인 SEED[4,5,6,7,8]를 개발하여 공개하였다.

SEED는 16 회전을 수행하는 피스탈 네트워크(Feistel network) 구조를 가지며, 64 비트 평문을 2개의 32 비트 블록으로 나누고, 하나의 32 비트 블럭을 G 함수에 의하여 변환하고, 변환한 32 비트와 변환하지 않은 32 비트 블록을 연산하여 32 비트의 결과를 구한다. 이러한 과정을 3회 반복하여 하나의 F 함수를 구성한다. G 함수는 4개의 8 비트 S 박스 출력을 결합하여 구성한다.

본 논문에서는 SEED와 같은 형식을 가지는 암호 알고리즘에서 사용할 수 있는 안전성이 높은 S 박스와 G 함수를 생성하는 방법을 제안한다.

S 박스는 비선형함수와 아핀변환으로 구성한다. 비선형함수는 차분공격[9]과 선형공격[10]에 강한 nl(X) ==> X^{-1} over GF(2^8) 변환[11]을 채택하고, 원시다항식은 '0'과 '1' 이외의 약한 입력을 가지지 않으면서 입출력의 상관계수가 작은 것을 선정한다. 아핀변환은 차분공격과 선형공격의 특성에 영향을 주지 않지만[12], 수식의 복잡도를 증가시켜서 보간공격(interpolation attack)[13]에 강하도록 한다. 본 논문에서는 입력과 출력이 같거나 출력이 입력의 1의 보수가 되는 약한 입력을 가지지 않으면서 입출력의 상관계수가 가장 작은 아핀변환을 구성한다.

SEED G 함수는 4개의 S 박스 출력을 간단한 수식으로 결합하여 입력의 변화를 출력에 확산시키므로 충분한 확산이 이루어지지 않는다. 이러한 단점을 개선하기 위하여 본 논문에서는 F 함수의 특성을 분석하여 G 함수는 전단사함수가 되어야 하며, MDS(Maximum Distance Separable) 코드[14,15,16]를 생성하여야 하며, SAC(Strict Avalanche Criterion)[17]를 충족시켜야 함을 제시한다. 또한 G 함수의 약한 입력으로 고정점과 역고정점 및 출력이 입력의 2의 보수가 되는 입력이 약한 입력임을 보인다.

본 논문에서는 이러한 G 함수의 특성을 고려하여 MDS 코드를 생성하고, SAC를 만족하여 확산 특성이 우수하고, 약한 입력을 가지지 않는 전단사 함수이고, 하드웨어 구현이 용이한 G 함수를 생성하는 알고리즘을 제안하고, SEED에 활용할 수 있는 3개의 G 함수를 생성하여 특성을 분석한다.

본 논문의 구성은 2장에서 SEED 알고리즘을 소개하고, 3장에서 S 박스를 구성하고, 4장에서 G 함수의 특성을 분석하고, 5장에서는 하드웨어 구현이 용이한 G 함수를 생성하는 방식을 제안하고, 6장에서 제안한 G 함수를 구현하고 성능을 분석하고 SEED와 비교를 수행하고, 7장에서 결론을 맺는다.

본 논문에서는 제안하는 S 박스와 G 함수는 차분공격과 선형공격에 강하고, 약한 입력이 없으며, 확산 특성이 우수하므로 안전성이 높은 암호 방식의 구성 요소로 활용할 수 있다.

Ⅱ. SEED 알고리즘

SEED 암호 알고리즘은 피스탈(Feistel) 구조로 이루어져 있으며, 128 비트의 평문 블록단위당 128 비트 키로부터 생성된 16개의 64 비트 라운드 키를 입력으로 사용하여 총 16 라운드를 거쳐 128 비트 암호문 블록을 출력한다.

각 라운드에서는 128 비트 입력 평문 블록을 L과 R 2개의 64 비트 블록으로 나누고, R 64 비트 블록과 2개의 32 비트 라운드 키를 F 함수에서 처리

한다. F 함수 출력은 L 64 비트 블럭과 XOR를 취하여 중간 결과를 형성한다. 중간 결과의 L과 R을 교환하고 다음 라운드를 동일한 방식으로 수행한다.

F 함수는 입력된 64 비트를 32 비트 블록 2개(R0, R1)로 분할하고, 2개의 32 비트 라운드 키와 함께 그림 1과 같은 처리를 수행하여 32 비트 블록 2개(R0', R1')를 출력한다. 그림 1에서 F 함수는 키 처리 부분과 32 비트 모듈러 덧셈기를 포함한 3개의 G 함수로 구성되어 있다.

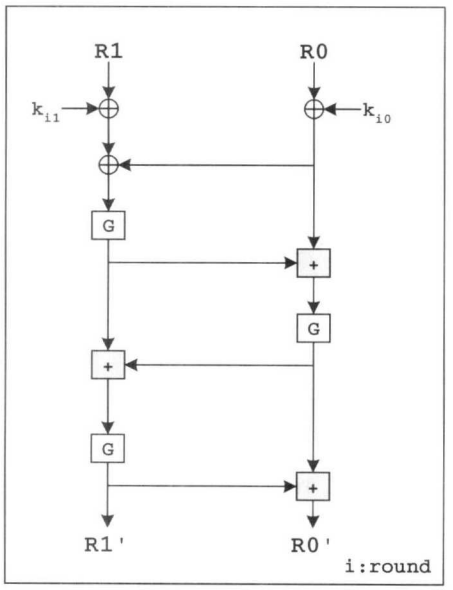


그림 1. SEED F 함수 처리도

G 함수는 다음과 같이 기술된다.

$$Y_3 = S_2(X_3), \quad Y_2 = S_1(X_2),$$
$$Y_1 = S_2(X_1), \quad Y_0 = S_1(X_0)$$

$$Z_3 = (Y_0\&m_3)\oplus(Y_1\&m_0)\oplus(Y_2\&m_1)\oplus(Y_3\&m_2)$$
$$Z_2 = (Y_0\&m_2)\oplus(Y_1\&m_3)\oplus(Y_2\&m_0)\oplus(Y_3\&m_1)$$
$$Z_3 = (Y_0\&m_1)\oplus(Y_1\&m_2)\oplus(Y_2\&m_3)\oplus(Y_3\&m_0)$$
$$Z_3 = (Y_0\&m_0)\oplus(Y_1\&m_1)\oplus(Y_2\&m_2)\oplus(Y_3\&m_3)$$

where m0=0xfc, m1=0xf3, m2=0xcf,
m3=0x3f (& : bit-wise AND)

G 함수에는 두 개의 S 박스 S1과 S2를 사용한다. S1과 S2는 각각 GF(2^8) 상의 원시다항식 0x163에서 X^{247}과 X^{251}을 비선형함수로 사용하고, 비선형함수를 아핀변환하여 구성한다.

또한 G 함수는 4개의 확장된 32 비트 SS 박스로 다음과 같이 구현할 수 있다.

$$Z = SS3(X3)\oplus SS2(X2)\oplus SS1(X1)\oplus SS0(X0)$$

SS3 = S2(X3)&m2 || S2(X3)&m1 ||
S2(X3)&m0 || S2(X3)&m3
SS2 = S1(X2)&m1 || S1(X2)&m0 ||
S1(X2)&m3 || S1(X2)&m2
SS3 = S2(X1)&m0 || S2(X1)&m3 ||
S2(X1)&m2 || S2(X1)&m1
SS3 = S1(X0)&m3 || S1(X0)&m2 ||
S1(X0)&m1 || S1(X0)&m0
(||는 concatenation)

Ⅲ. S 박스 생성

S 박스는 비선형함수와 아핀변환으로 구성한다. 비선형함수는 SEED와 같은 대칭 키 블록 암호를 공격하는 가장 효율적인 공격 수단인 차분공격과 선형공격에 강한 특성을 가져야 한다.

차분공격에 강한 S 박스를 구성하기 위해서는 차분 XOR 테이블에서 모든 첫 번째 컬럼이 '0'이 되어야 하며, '0'인 성분의 수가 작아야 하며, 최대 성분 값이 작아야 한다[18,19,20]. 첫 번째 조건을 만족시키기 위해서는 S 박스는 전단사함수가 되어야 한다. 두 번째 조건을 만족하기 위해서 '0' 또는 '2'인 성분이 많아야 하며, 세 번째 조건을 만족하기 위해서 최대 성분 값이 '4'인 전단사함수를 선정해야 한다. 한편 선형공격에 강한 S 박스를 구성하기 위해서는 입력과 출력의 상관계수가 작아야 한다.

또한 S 박스는 S(A)=A인 고정점과 S(A)=~A인 역고정점이 없어야 한다. 고정점과 역고정점은 S 박

표 1. 고정점과 역고정점을 가지지 않는 GF(28) 상의 역수 특성

원시 다항식	입출력 상관계수	원시 다항식	입출력 상관계수	원시 다항식	입출력 상관계수	원시 다항식	입출력 상관계수
0x1a3	0.010	0x171	0.037	0x1cf	0.080	0x17b	0.124
0x163	0.015	0x1dd	0.047	0x15f	0.080	0x12d	0.133
0x18b	0.019	0x19f	0.052	0x1a9	0.081	0x11d	0.182
0x12b	0.020	0x1c3	0.066	0x165	0.111	0x1b1	0.196

스가 치환 기능을 수행하지 않는 입력으로 약한 입력이다.

이들 조건을 만족하는 비선형 전단사함수를 찾기 위해서 nl(X) ==> X^{-1} over GF(2^8)을 계산하였다. 단, 입력 '0'과 '1'은 예외적인 경우로 고정점 판단에서 제외한다. 조건을 만족하는 역수의 계산 결과를 표 1에 보인다.

GF(2^8) 상의 역수는 다음과 같은 성질을 가진다.

1) 차분 XOR 테이블의 최대 값은 4이며, 각 행에서 오직 하나의 성분만이 4이며, 나머지는 0과 2로만 구성되어 차분공격에 강하다.

2) 아핀변환과 최소거리는 2^4이다.

3) 출력 비트들의 선형결합에 대한 비선형계수(nonlinear order)는 7이다.

표 1은 입출력 상관계수가 작은 순서대로 나열하였다.

S 박스는 비선형함수와 아핀변환을 결합하여 식(1)과 같이 표현된다.

$$S(X) = ((X^{-1} \bmod Q) * M) \bmod P + C \quad (1)$$

• M과 P는 GF(2^8)상에서 서로소(relatively prime)

아핀변환은 차분공격과 선형공격 특성에 변화를 주지 않지만 수식의 복잡도를 증가시켜서 보간공격에 강한 특성을 나타내며, 비선형함수의 고정점을 없애는 기능을 수행한다.

식(1)에서 고정점과 역고정점을 가지지 않으면서 상관계수를 최소로 하는 아핀함수를 구하여 정리한 것을 표 2에 보인다.

Ⅳ. G 함수의 특징

G 함수는 32 비트 입력 X를 32 비트 출력 Z로 변환하는 G : X → Z 함수로 정의할 수 있으며 다음과 같은 특징을 가진다.

4.1. 전단사함수

F 함수를 변수 T0와 T1에 아래바침자를 붙여서 변화하는 모습을 알기 쉽도록 표현하면 다음과 같이 된다.

```
T0_0 = R0 ^ roundkey[Ki][0] ;
T1_0 = R1 ^ roundkey[Ki][1] ;

T1_1 = T0_0 ^ T1_0 ;
T1_2 = G(T1_1) ;
T0_1 = T0_0 + T1_2 ;
T0_2 = G(T0_1) ;
T1_3 = T0_2 + T1_2 ;
T1_4 = G(T1_3) ;
T0_3 = T0_2 + T1_4 ;

R0' =  T0_3;
R1' =  T1_4;
```

G 함수의 출력 Z의 상태수 N_z가 '$N_z < 2^{32}$'이면, T1_2와 T0_2의 상태수가 N_z가 되며, 이어서 T1_3

표 2. 최적화된 S 박스 계수표

비선형함수 원시다항식 Q	아핀변환 곱항 M	법연산 생성다항식 P	아핀변환 상수 C	S 박스 입출력 상관계수	비고
0x1a3	0x38	0x1c5	0x94	0.000056	G1-S1 박스
0x163	0xc8	0x159	0x3f	0.000303	G1-S2 박스
0x18b	0xba	0x1cd	0x11	0.000112	G2-S1 박스
0x12b	0x71	0x182	0xd9	0.000296	G2-S2 박스
0x171	0xc1	0x1a4	0x40	0.000411	G3-S1 박스
0x1dd	0x4d	0x1fc	0x2f	0.000149	G3-S2 박스
0x19f	0x84	0x17b	0x09	0.000095	
0x1c3	0xf4	0x1e1	0x08	0.000130	
0x1cf	0x76	0x127	0x63	0.000400	
0x15f	0x44	0x16f	0x2c	0.000202	
0x1a9	0x43	0x1bf	0x2e	0.000518	
0x165	0x06	0x11b	0x18	0.000450	
0x17b	0xc8	0x1bf	0x67	0.000503	
0x12d	0x45	0x167	0x28	0.000318	
0x11d	0x35	0x1e1	0xdd	0.000442	
0x1b1	0x08	0x187	0x56	0.000020	

와 T1_4 및 T0_3의 상태수가 2^{32}보다 작아진다. 이러한 현상을 방지하기 위해서는 G 함수의 Z 출력의 상태수는 반드시 2^{32}가 되어야 한다.

한편 G 함수는 X 입력을 Z 출력으로 변환하는 기능을 수행하는 함수로 차분공격에 강하기 위해서 단사함수가 되어야 한다. 따라서 G 함수는 전단사 함수가 되어야 한다.

4.2. MDS 코드

G 함수는 그림 2와 같이 S 박스와 GF(2^8) 상의 4×4 행렬식으로 표현되는 확산선형변환 부분의 결합으로 표현할 수 있다. 그림 2의 확산선형변환 부분은 식(2)의 행렬식으로 표현된다. 32 비트 입력 X는 4개의 8 비트 $X_i \in \{X_3, X_2, X_1, X_0\}$로 분할되어 $S_i \in \{S_2, S_1\}$ 박스에서 치환되고, 확산선형변환 행렬식에서 선형변환되어 4개의 8 비트 $Z_i \in \{Z_3, Z_2, Z_1, Z_0\}$로 출력된다. X_3와 Z_3가 최상위 바이트이다.

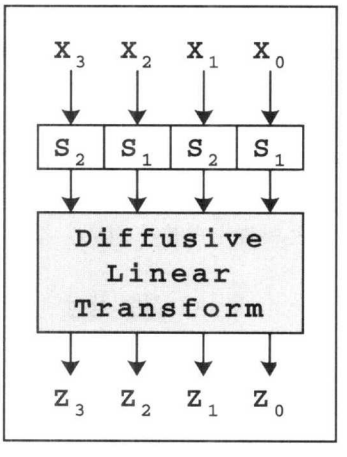


그림 2. G 함수 구조도

$$\begin{vmatrix} Z_0 \\ Z_1 \\ Z_2 \\ Z_3 \end{vmatrix} = \begin{vmatrix} A_{00} & A_{01} & A_{02} & A_{03} \\ A_{10} & A_{11} & A_{12} & A_{13} \\ A_{20} & A_{21} & A_{22} & A_{23} \\ A_{30} & A_{31} & A_{32} & A_{33} \end{vmatrix} \begin{vmatrix} S1(X_0) \\ S2(X_1) \\ S1(X_2) \\ S2(X_3) \end{vmatrix} \quad (2)$$

식 (2)의 확산선형변환 행렬식은 차분공격과 선형공격에 강하도록 설계되어야 한다. 이를 위해서 차분확산계수와 선형확산계수가 모두 최고 값을 가지도록 설계해야 한다.

식 (2)의 G 함수에서 S 박스의 8 비트 단위 출력을 $P_i \in \{P_0, P_1, P_2, P_3\}$라 하면, 확산선형변환 행렬식은 Pi를 확산시켜서 출력 $Z_i \in \{Z_0, Z_1, Z_2, Z_3\}$를 생성한다. Wh(a)를 GF($2^8$) 상의 a의 해밍가중치라고 하면,

$$\begin{aligned} Wh(a) &= 0 \text{ if } a = 0 \\ &= 1 \text{ if } a \neq 0 \end{aligned}$$

이 된다.

확산선형변형 행렬식의 차분확산계수(differential branch number) Bd는 식(3)과 같이 정의할 수 있다

$$Bd = \min\left(\sum_{i=0}^{3} Wh(Pi) + \sum_{i=0}^{3} Wh(Zi)\right) \quad \text{while } P \neq 0 \quad (3)$$

식-3에서 ∑Wh(Pi)와 ∑Wh(Zi)의 최대 값은 4이고 최소 값은 1이 된다. 따라서 차분확산계수는 Bd≤5가 된다. 차분확산계수는 입력 Pi가 변화하였을 때 출력 Zi가 변화하는 GF(2^8) 상의 코드 수의 최소 값을 나타낸다. 차분공격에 있어서 비활성 S 박스 입력의 차분은 '0'이며, 따라서 출력의 차분도 '0'이다. SEED의 F 함수는 그림 1과 같이 3개의 G 함수를 직렬로 연결한 구조이다. 따라서 차분공격에 강하기 위해서는 G 함수의 차분확산계수는 최고 값인 5가 되어야 한다.

한편 확산선형변형 행렬식의 선형확산계수(linear branch number) Bl는 식(4)와 같이 정의할 수 있다

$$Bl = \min\left(\sum_{i=0}^{3} Wh(Ai) + \sum_{i=0}^{3} Wh(Bi)\right) \quad \text{while } B \neq 0 \quad (4)$$

식 4에서 Ai와 Bi는 각각 입력 마스크 값과 출력 마스크 값을 나타내며, 식-2의 확산선형변환 행렬식을 M이라고 하면 $A = M^t \times B$가 된다[21]. 식-4에서 ∑Wh(Ai)와 ∑Wh(Bi)의 최대 값은 4이고 최소 값은 1이 된다. 따라서 선형확산계수는 Bl≤5가 된다. 선형확산계수는 출력 Zi에 대하여 선형 결합된 입력 Pi의 GF(2^8) 상의 코드 수의 최소 값을 나타낸다. SEED의 F 함수는 그림 1과 같이 3개의 G 함수를 직렬로 연결한 구조이다. 따라서 선형공격에 강하기 위해서는 G 함수의 선형확산계수는 최고 값인 5가 되어야 한다.

확산계수가 최고 값을 가지는 코드를 MDS (Maximum Distance Separable) 코드라고 한다. 따라서 G 함수는 MDS 코드를 생성해야 한다.

식(3)과 식(4)로부터 확산선형변환 행렬식 M이 대칭 행렬이면 'Bd = Bl'이 된다[22]. 본 논문에서는 대칭 행렬을 구성하기 위하여 한 행의 성분 4개를 선정하고, 아래 행은 윗 행의 성분을 왼쪽으로 회전시켜서 구성한다. 이렇게 구성한 G 함수를 식(5)에

보인다.

$$\begin{vmatrix} Z_0 \\ Z_1 \\ Z_2 \\ Z_3 \end{vmatrix} = \begin{vmatrix} A_0 & A_1 & A_2 & A_3 \\ A_1 & A_2 & A_3 & A_0 \\ A_2 & A_3 & A_0 & A_1 \\ A_3 & A_0 & A_1 & A_2 \end{vmatrix} \begin{vmatrix} S1(X_0) \\ S2(X_1) \\ S1(X_2) \\ S2(X_3) \end{vmatrix} \quad (5)$$

4.3. SAC

식(2)의 G 함수에서 두 개의 입력 X와 Xi에 대하여, Xi는 i($0 \le i \le 31$) 비트만이 X와 다르다고 할 때, Vij는 Z의 j($0 \le j \le 31$) 비트가 변경될 확률이라고 정의하면, 입력 $X \in \{0,1,2,,,2^{32}-1\}$에서 Vij가 평균값 0.5를 가지는 정규분포를 이루면 SAC를 만족시킨다고 한다.

선형공격 및 차분공격에 강하기 위해서는 입력의 작은 비트의 변화가 출력의 많은 비트에 나타나야 하며, 또한 입력의 많은 비트의 변화가 출력의 적은 비트의 변화로 나타나야 한다.

SEED의 F 함수는 그림-1과 같이 G 함수의 출력이 다음 G 함수의 입력이 되는 구조이다. 또한 입력은 3개의 G 함수를 거쳐서 출력이 된다. 이와 같은 구조에서 G 함수가 SAC를 만족하지 않으면 입력의 변화가 일부 출력에만 나타나고, 이러한 과정이 3번 반복되면 출력의 특정 부분에만 변화가 집중될 수 있다.

SEED에서는 G 함수 출력에 대하여 덧셈 연산을 하여서, 캐리 전파에 의하여 확산을 시키고 있다. 그런데 캐리 전파는 더하는 두 개의 비트가 '0'과 '1'인 경우에만 발생하므로, 캐리 전파 확률은 50%이다. 즉, 덧셈 연산의 캐리 전파만으로는 충분한 확산을 기대할 수 없다.

따라서 입력의 변화를 출력에 충분히 확산시키기 위해서 G 함수는 SAC를 충족시켜야 한다.

4.4. 약한 입력

G 함수는 X 입력을 Z 출력으로 변환하는 32 비트 치환 블럭이다. 따라서 'G(X) = X'인 고정점과 'G(X)=~X'인 역고정점은 약한 입력이 된다. 또한 F 함수는 G 함수의 출력에 대하여 덧셈 연산을 수행한다. 'G(X) = -X'가 되는 입력 X에 대하여서 F 함수는

```
T0_0 = R0 ^ roundkey[Ki][0] ;
T1_0 = R1 ^ roundkey[Ki][1] ;

T1_1 = T0_0 ^ T1_0 ;
T1_2 = G(T1_1) ;
T0_1 = T0_0 + T1_2 ;
T0_2 = G(T0_1) = -T0_1 = -T0_0 - T1_2 ;
T1_3 = T0_2 + T1_2 = -T0_0 ;
T1_4 = G(T1_3) = T0_0 ;
T0_3 = T0_2 + T1_4
     = -T0_0 - T1_2 +T0_0 = -T1_2 ;

R0' = T0_3 = -(T0_0 ^ T1_0) ;
R1' = T1_4 = T0_0 ;
```

가 된다. 따라서 'G(X) = -X'가 되는 입력은 약한 입력이다.

G 함수가 약한 입력을 가지면, 약한 입력에 대해서는 치환 기능을 수행하지 않는다. 따라서 G 함수는 약한 입력을 가지지 않아야 한다.

4.5. 간단한 하드웨어

그림-2의 하드웨어 구현은 S 박스는 8×8 ROM으로 구성하고, 확산선형변환은 2 입력 XOR 게이트 어레이로 구성할 수 있다. 확산선형변환 부분은 비규칙적인 회로가 되므로 반도체상에서 넓은 면적을 차지하게 된다.

따라서 G 함수의 회로를 간단하게 구성하기 위해서는 확산선형변환 행렬식을 구성하는 2 입력 XOR 게이트 어레이 부분의 회로가 간단해야 된다.

V. G 함수 생성

SEED의 G 함수는 두 종류의 S 박스를 각각 두 개씩 사용하여 그림-2와 같이 구성되어 있다. F 함수는 그림-1과 같이 G 함수를 3번 반복 사용하여 연산한다. 본 논문에서는 F 함수에서 서로 다른 3개의 G 함수를 사용하여 수식의 복잡도를 높이는 경우를 고려하여 3개의 G 함수, G1, G2와 G3 함수를 생성한다. 각 G 함수에 사용되는 2 종류의 S 박스는 S 박스의 비선형함수의 입출력상관계수가 낮은 순서대로 선정하였다. 각 G 함수에 사용한 S 박스를 표-2의 비고란에 표시하였다.

식-2의 확산선형변환 행렬식은 Zi에 대한 함수로 Z_0는 다음처럼 표현된다.

$$Z_0 = \sum(Ai * Si(Xi)) \bmod Ap$$

Ai는 $GF(2^8)$상에서 곱항으로 확산선형변환 행렬식의 한 성분이고, Ap는 $GF(2^8)$ 상의 원시다항식이다. $GF(2^n)$ 상의 곱셈은 곱항과 원시다항식이 정해

지면 GF(2)상의 행렬식으로 표현할 수 있다.

$GF(2^n)$ 상의 n 비트 수 A와 B 및 원시다항식 P는 다음과 같이 다항식으로 표현된다.

$$A=\sum_{i=0}^{n-1}a_ix^i,\quad B=\sum_{i=0}^{n-1}b_ix^i,\quad P=\sum_{i=0}^{n}p_ix^i$$

단위 함수 u()를 다음과 같이 정의하면,

$$u(i,j,m) = 1 \text{ if } m = i + j$$
$$= 0 \text{ if } m \neq i + j$$

A와 B의 곱 C는 다음과 같이 된다.

$$C = A\times B = \sum_{i=0}^{2n-2}c_ix^i$$

$$c_m=\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}u(i,j,m)\cdot a_i\cdot bj$$

단, $m\in\{0,1,,,2n-2\}$

곱 C를 $GF(2^n)$ 상의 원시다항식 P로 나눈 나머지는 다음과 같은 방식으로 계산할 수 있다.

```
for (i=2n-2 ; i > n-1 ; i--)
    for (j=0 ; j < n+1 ; j++)
        c_{i-j} = c_i · p_{n-j} + c_{i-j}
```

이렇게 계산된 $c_i\in\{c_{n-1},c_{n-2},,,c_0\}$는 식(6)과 같이 표현할 수 있다.

$$c_i = \sum_{j=0}^{n-1}f_j(b_{n-1},b_{n-2},,,b_0,p_{n-1},,,p_0)\cdot a_j \qquad (6)$$

식(6)에서 B와 P가 $b_i\in\{0,1\}$, $p_i\in\{0,1\}$ 로 주어지면 $f_j(...)\in\{0,1\}$가 된다. 따라서 $GF(2^n)$ 상의 A와 B의 곱은 $f_j(...)$를 성분으로 하는 GF(2)상의 행렬식으로 표현할 수 있다. 그림 3에 예를 보인다.

그림 3a는 간단한 선형변환 행렬식의 예이고, 그림 3b는 복잡한 선형변환 행렬식의 예를 보이고 있다. 그림 3의 행렬식에서 각 행의 '1' 성분은 Ci 생성식의 항을 나타내며, '1' 성분의 수가 N 개이면, Ci를 생성하기 위해서는 'N-1'개의 2 입력 XOR 게이트가 필요하다. 또한 전달지연 시간은 $\lfloor\log_2 N\rfloor$가 된다.

$$\begin{vmatrix}C_7\\C_6\\C_5\\C_4\\C_3\\C_2\\C_1\\C_0\end{vmatrix} = \begin{vmatrix}0&0&0&0&0&0&0&1\\1&0&0&0&0&0&0&1\\0&1&0&0&0&0&0&1\\0&0&1&0&0&0&0&0\\0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0\\0&0&0&0&0&1&0&0\\0&0&0&0&0&0&1&1\end{vmatrix} \begin{vmatrix}A_7\\A_6\\A_5\\A_4\\A_3\\A_2\\A_1\\A_0\end{vmatrix}$$

그림 3a. GF(28) 상 곱셈 C = (A * 0xe1) mod 0x1c3 의 GF(2)상 행렬식 표

$$\begin{vmatrix}C_7\\C_6\\C_5\\C_4\\C_3\\C_2\\C_1\\C_0\end{vmatrix} = \begin{vmatrix}0&1&1&1&0&1&0&0\\0&1&0&0&1&1&1&0\\0&1&0&1&0&0&1&1\\1&0&1&0&1&0&0&1\\1&1&0&1&0&1&0&0\\1&0&0&1&1&1&1&0\\0&0&1&1&1&0&1&1\\1&1&1&0&1&0&0&1\end{vmatrix} \begin{vmatrix}A_7\\A_6\\A_5\\A_4\\A_3\\A_2\\A_1\\A_0\end{vmatrix}$$

그림 3b. GF(28) 상 곱셈 C = (A * 0x33) mod 0x1cf 의 GF(2)상 행렬식 표현

하드웨어를 최소로 하는 G 함수의 확산선형변환 행렬식을 표-3의 알고리즘에 의하여 생성한다.

표 3의 알고리즘을 적용하여 생성한 G 함수를 표 4에 보인다.

표 4에서 Sn={Q, M, P, C}는 식(1)의 Q, M, P 및 C를 각각 나타낸다.

표 3. G 함수의 확산선형변환 행렬식 생성 알고리즘

step-1) 2 입력 XOR 하드웨어를 최소로 필요로 하는 $GF(2^8)$ 상의 행렬식 성분 4개를 무작위로 발췌하여 확산선형변환 형렬식을 구성한다.

step-2) 행렬식의 역행렬식을 구한다. 역행렬식이 구해지지 않으면 전단사함수가 되지 않으므로 step-1로 되돌아간다.

step-3) 역행렬식의 모든 성분이 '0'이 아닌 것을 확인한다. '0'인 성분이 있으면 확산이 제대로 이루어지지 않으므로 step-1로 되돌아간다.

step-4) 확산선형변환 행렬식이 입력 $P\in\{1,2,3,,,,2^{32}-1\}$에 대하여 식-3의 차분확산계수가 5인가를 판단한다. 5 미만이면 MDS 코드를 생성하지 않으므로 step-1으로 되돌아간다.

step-5) $X\in\{0,1,2,,,,2^{32}-1\}$에 대하여 고정점 'G(X)=X'와 역고정점 'G(X)=~X' 및 'G(X)=-X'가 되는 약한 입력점을 가지지 않는 것을 확인한다. 약한 입력점을 가지면 step-1로 되돌아간다.

표 4. S 박스와 확산선형변환 행렬식의 G 함수

	S 박스	확산선형변환 행렬식
G1	S1 = {0x1a3, 0x38, 0x1c5, 0x94} S2 = {0x163, 0xc8, 0x159, 0x3f}	Ai = {0x04, 0x01, 0x01, 0xc3} Ap = 0x187
G2	S1 = {0x18b, 0xba, 0x1cd, 0x11} S2 = {0x12b, 0x71, 0x182, 0xd9}	Ai = {0x02, 0x01, 0xa2, 0x02} Ap = 0x187
G3	S1 = {0x171, 0xc1, 0x1a4, 0x40} S2 = {0x1dd, 0x4d, 0x1fc, 0x2f}	Ai = {0x02, 0x02, 0x01, 0x91} Ap = 0x187

Ai={A_0, A_1, A_2, A_3}는 식-5 GF(2^8) 상의 4×4 행렬식의 각 성분을 나타내며, Ap는 행렬식의 원시 다항식이다.

Ⅵ. 구현 및 평가

G 함수를 소프트웨어로 구현하는 경우에는 32 비트 SS 박스를 만드는 것이 효율적이다. SS 박스는 2개의 8×32 메모리를 필요로 한다.

하드웨어로 구현하는 경우에는 4개의 8×8 S 박스 ROM을 사용하는 경우[5]와 4개의 8×32 SS 박스 ROM을 사용하는 경우[8]가 있다. 8×8 S 박스 ROM으로 구현하는 경우에 확산선형변환 행렬식을 구현하는 데 소요되는 2 입력 XOR 게이트의 수 및 최대 전달지연시간을 표-5에 보인다.

확산 특성은 MDS 코드이면서 SAC를 만족하는가로 판단할 수 있다. 본 논문의 G 함수는 MDS 코드를 생성하므로 SAC를 만족하는 가만을 판단하면 된다. Vij의 평균 SACavr, Vij의 표준편차 SACdev를 산출하여 표-5에 보인다.

표 5에서 Vij(SACavr±n SACdev) 항목은 SACavr의 위, 아래 n배의 SACdev 구간에 분포하는 Vij를 나타낸다. 모든 Vij가 평균값 주위에 밀집한 정규분포를 이루므로 본 논문의 G 함수는 SAC를 만족시킨다.

한편 SEED G 함수의 차분확산계수 Bd는 4로 MDC 코드를 생성하지 않으며, SACavr과 SACdev는 각각 0.376과 0.219로 SAC를 만족시키지 못한다. 또한 SEED의 S1 박스, S2 박스 및 G 함수는 다음과 같은 약한 입력을 가진다. SEED G 함수는 2 입력 XOR 게이트 64개가 필요하며 최대 전달지연시간은 2 gate이다.

SEED S1 박스의 약한 입력 :
고정점 : S1(0x17) = 0x17, S1(0xe6) = 0xe6
역고정점 : S1(0x48) = 0xb7, S1(0xfa) = 0x05

SEED S2 박스의 약한 입력 :
고정점 : S2(0x1c) = 0x1c

SEED G 함수의 약한 입력 :
고정점 : G(0x32f732f7) = 0x32f732f7
G(0xa867a867) = 0xa867a867
역고정점 : G(0x5ae96fb0) = 0xa516904f
G(0x6fb05ae9) = 0x904fa516
2의 보수 : G(0xd60b3903) = 0x29f4c6fd

Ⅶ. 결 론

컴퓨터와 정보통신이 발달할수록 더욱 많은 정보

표 5. G 함수 특성표

	G1	G2	G3
No of 2 input XOR gates	128 gates	140 gates	140 gates
Maximum Tpd	3 gates	3 gates	3 gates
SACavr	0.501	0.505	0.504
SACdev	0.032	0.031	0.031
Vij(SACavr±1 SACdev)	67 %	62 %	65 %
Vij(SACavr±2 SACdev)	97 %	98 %	99 %
Vij(SACavr±3 SACdev)	100 %	100 %	100 %

를 처리하게 된다. 이에 따라서 정보에 대한 보안이 중요한 문제로 대두되면서 정보를 보호하고 불법적인 유출을 방지하기 위해서 암호의 필요성이 증대되고 있다. 이러한 필요성에 따라서 우리나라에서도 학국정보보호센터를 주축으로 관련 전문가들과 공동으로 128 비트 블록 암호 알고리즘인 SEED를 1998년에 개발하여 공개하였다.

SEED는 피스탈 네트워크(Feistel network) 구조를 가지며, 64 비트 평문을 2개의 32 비트 블록으로 나누고, 하나의 32 비트 블럭을 G 함수에 의하여 변환하고, 변환한 32 비트와 변환하지 않은 32 비트 블록을 연산하여 32 비트를 결과를 구한다. 이러한 과정을 3회 반복하여 하나의 F 함수를 구성한다. G 함수는 4개의 8 비트 S 박스를 선형결합하여 구성한다. 이러한 SEED 구조에서는 안전성은 S 박스와 G 함수의 안전성에 의존적이다.

본 논문에서는 SEED와 같은 형식을 가지는 암호 알고리즘에서 안전성이 높은 S 박스와 G 함수를 생성하는 방법을 제안하였다.

S 박스는 $GF(2^8)$ 상에서 비선형함수와 아핀변환으로 구성하였다. 비선형함수는 차분공격과 선형공격에 강한 특성을 가지는

$nl(X) ==> X^{-1}$ over $GF(2^8)$ 변환을 채택하고, 원시다항식은 '0'과 '1' 이외의 약한 입력을 가지지 않으면서 입출력의 상관계수가 작은 것을 선정하였다. 아핀변환은 입력과 출력이 같은 고정점과 출력이 입력의 1의 보수가 되는 역고정점인 약한 입력을 가지지 않으면서 입출력의 상관계수가 가장 작게 되도록 구성하였다.

G 함수는 4개의 S 박스 출력을 $GF(2^8)$ 상에서 4×4 행렬식으로 확산선형변환하여 구성하였다. G 함수는 MDS((Maximum Distance Separable) 코드를 생성하고, SAC(Strict Avalanche Criterion)를 만족하여 확산 기능이 좋도록 하였다. 또한 고정점과 역고정점 및 출력이 입력의 2의 보수가 되는 약한 입력을 가지지 않으며, 하드웨어로 구현시 회로가 간단해지도록 구성하였다.

본 논문에서는 제안한 S 박스와 G 함수는 차분공격과 선형공격에 강하고, 약한 입력이 없으며, 하드웨어 구현이 용이하고, 확산 특성이 우수하므로 안전성이 높은 암호 방식의 구성 요소로 활용할 수 있다.

참 고 문 헌

[1] ANSI X3.92, "American National Standard for Data Encryption Algorithm(DEA)," NIST, 1983

[2] NIST, "Advanced Encryption Standard Development Effort." http://csrc.nist.gov/encryption/ aes.

[3] Joan Daemen, Vincent Rijmen, "AES Proposal: Rijndael", 1999

[4] 한국정보보호센터, 128 비트 블록 암호알고리즘(SEED) 개발 및 분석 보고서, Dec. 1998

[5] Young-Ho Seo, Jong-Hyeon Kim and Dong-Wook Kim, "Hardware Implementation of 128-bit Symmetric Cipher SEED," The Second IEEE Asia Pacific Conference on ASICs, pp. 183-186, Aug. 2000

[6] 이 명동, "SEED 암호 알고리즘의 FPGA 구현을 위한 RTL 수준 VHDL 설계," 한남대학교 대학원 컴퓨터공학과 석사학위논문, 2001

[7] 정 찬호, "SEED에 대한 효과적인 Brute-Force 공격 알고리즘," 한국항공대학교 컴퓨터공학과 석사학위논문, 2001

[8] 전 신우, 정 용진, "128 비트 SEED 암호 알고리즘의 고속처리를 위한 하드웨어 구현," 통신정보보호학회지, Vol. 11, No. 1, pp. 13-23, Feb. 2001

[9] E. Biham and A. Shamir, "Differential cryptoanalysis of DES-like cryptosystems," Journal of Cryptology, Vol. 4, No. 1, pp. 3-72, 1991

[10] M. Matsui, "The first experimental cryptanalysis of the Data Encryption Standard," Advances in Cryptology, Proceedings of CRYPTO '94, Springer-Verlag, Berlin, pp. 1-11, 1994

[11] K. Nyberg, "Differentially uniform mappings for cryptography," Advances in Cryptology, Proceedings of Eurocrypt '93, LNCS 765, T. Helleseth, ED., Springer-Verlag, pp. 55-64, 1994

[12] Serge Mister and Carlisle Adams, "Practical S-box Design," Workshop record of the workshop on selected area in Cryptography(SAC '96), Queen's University, pp. 61-76, Aug. 1996

[13] T. Jakobsen and L.R. Knudsen, "The interpolation attack on block cipher," Fast

Software Encryption, LNCS 1267, E. Biham, Ed., Springer-Verlag, pp. 28-40, 1997

[14] S. Vaudenay, "On the need for multi-permutations: Cryptanalysis of MD4 and SAFER," Proceedings of Fast Software Encryption (2), LNCS 1008, Springer-Verlag, pp. 286-297, 1995

[15] V. Rijmen, J. Daemen, B. Preneel, A. Bosselaers and E. De Win, "The cipher SHARK," Fast Software Encryption, LNCS 1039, D. Gollmann, Ed., Springer-Verlag, pp. 99-112, 1996

[16] J. Daemen, L. Knudsen and V. Rijmen, "The block cipher SQUARE," Proceedings of Fast Software Encryption (4), LNCS, Springer-Verlag, 1997

[17] Webster, A. and S. Tavares, "On the Design of S-Boxes," Advances on Cryptology, CRYPTO '85, pp. 523-534, 1985

[18] A.M. Youssef, Z.G. Chen and S.E. Tavares, "Construction of Highly Nonlinear Injective S-boxes With Application to CAST-like Encryption Algorithms," Proceedings of the Canadian Conference on Electrical and Computer Engineering(CCECE'97), 1997

[19] Jennifer Seberry, Xian-Mo Zhang and Yuliang Zheng, "Systematic Generation of Cryptographically Robust S-boxes," The preceedings of the First ACM Conference on Computer and Communications Security, pp. 172-182, Nov. 1993

[20] Nyberg, K., "Perfect nonlinear S-boxes," In Advances in Cryptology, EUROCRYPT'91, Vol. 547, Lecture Notes in Computer Science, Springer-Verlag, pp. 378-386, 1991

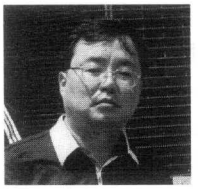
[21] J. Daemen, R. Govaerts and J. Vandewalle, "Correlation Matrixes," Fast Software Encryption, LNCS 1008, Spring-Verlag, pp. 275-285, 1994

[22] J. S. Kang, C. S. Park, S. J. Lee and J. L. Lim, "On the optimal diffusion layer with practical security against Differential and Linear Cryptanalysis," Proceedings of ICISC'99, LNCS 1787, Spring-Verlag, pp. 33-52, 1999

송 홍 복 종신회원

1983년 : 광운대학교 전자통신공학과 졸업(공학사)
1985년 : 인하대학교 대학원 전자공학과 졸업(공학석사)
1985-1990년 : 동의공업대 전자통신과 조교수
1989-1990년 : 일본 구주공대 정보공학부 객원 연구원
1990년 : 동아대학교 대학원 전자공학과 졸업 (공학박사)
1994 ~ 1995년 : 일본 미야자키대학교 전기·전자공학부(POST-DOC)
1991년 ~ 현재 : 동의대학교 전기·전자·정보통신·메카트로닉스공학부 부교수
<주관심 분야> 다치 논리 이론 및 다치 논리 시스템 설계, VLSI 설계, 마이크로 프로세스 응용 등임

조 경 연 정회원



1990 : 인하대학교 공과대학 전자공학과 정보공학 전공 (공학박사)
1983-1991 : 삼보컴퓨터 기술연구소 책임연구원
1991-현재 : 부경대학교 공과대학 전자컴퓨터정보통신공학부 부교수
1991 ~ 2001 : 삼보컴퓨터 기술연구소 비상임 기술고문
1998 ~ 현재 : 에이디칩스 사외이사 겸 비상임 기술고문
<주관심 분야> 전산기구조, 반도체회로설계, 암호 알고리즘