함수 풀에 기반한 개선된 SPEED 암호시스템
(An Improved SPEED Cryptosystem based on a Function Pool)

이 문 규 † 박 근 수 †† 조 유 근 ††
(Mun-kyu Lee) (Kunsoo Park) (Yookun Cho)

요 약 본 논문에서는 Zheng이 제시한 비밀키 암호시스템인 SPEED를 개선한 암호시스템을 제안한다. 제안된 암호시스템에서는 암호화에 사용되는 부울 함수를 키에 따라 가변적으로 함수 풀(function pool)로부터 선택함으로써, 함수 선택에 대한 약간의 오버헤드만으로 선형 공격(linear cryptanalysis) 및 차분 공격(differential cryptanalysis) 등 알려진 공격에 대해 향상된 저항성을 가지도록 하였다.

Abstract In this paper we propose a cryptosystem which improves the secret-key cryptosystem SPEED introduced by Zheng. The new cryptosystem has enhanced security against known attacks such as linear cryptanalysis and differential cryptanalysis by selecting encryption functions from a function pool according to the value of the encryption key.

1. 서 론

SPEED(Secure Package for Encrypting Electronic Data)[1]는 Financial Cryptography '97에서 Zheng에 의해 발표된 비밀키 암호시스템으로, 이전의 많은 블록 암호시스템들이 가졌던 형태인 Feistel 구조[2, 3]에 기반하고 있으나 각 라운드 변환에 부울 함수를 이용하고 있다는 점에서 이전의 암호시스템들과 구분된다. SPEED를 구성하는 핵심 요소인 부울 함수들은, 비선형성(nonlinearity), 전달 특성(propagation criterion), 균형성(balance) 등 안전한 함수가 되기 위한 특성들을 만족하고 있다.

부울 함수의 특성에 대한 연구로서, 먼저 Meier와 Staffelbach[4]는 암호시스템에 사용되는 부울 함수의 안전성 평가의 기준으로서 비선형성을 연구하였으며, Preneel 등[5]은 SAC(Strict Avalanche Criterion) 및 완전 비선형 특성(perfect nonlinearity criterion)의 일반화된 기준으로서 전달 특성에 관한 연구를 수행하였다. Seberry 등[6]과 Zhang 및 Zheng[7]은 높은 비선형성 및 전달 특성, 균형성을 만족하는 부울 함수의 구성에 관해 연구하였으며, Millan 등[8, 9]은 비교적 높은 비선형성을 갖는 함수들을 효율적으로 생성해 내는 유전 알고리즘(genetic algorithm)을 제시한 바 있다.

SPEED에서는 LC(linear cryptanalysis)[10]를 막기 위해 [6] 및 [7]의 방법으로 구성된 비선형 함수들(nonlinear functions)을 각 라운드 변환에 이용하고 있으며, 또한 DC(differential cryptanalysis)[11]를 방지하기 위해 데이타에 의존한 연산을 사용하고 있다. 그러나 Hall 등은 SAC '98에서 SPEED의 안전성에 대해 몇 가지 문제점을 지적하고 제한적인 공격 방법을 제시하였다[12].

본 논문에서는 SPEED의 각 패스(pass)를 구성하는 부울 함수가 고정되어 있는 점을 개선하여 여러 개의 부울 함수들로 구성된 함수 풀(function pool)로부터 암호화 키에 따라 가변적으로 함수를 선택할 수 있도록 함으로써, [12]에서 지적된 SPEED의 문제점들 중 일부를 개선하고 LC 및 DC에 대한 저항성을 향상시켰다. 함수 풀을 구성하는 부울 함수들은 [7]의 방법을 이용하여 생성하였으며, 이들은 균형성, 높은 전달 특성 및 비선형성 등 안전성의 기준을 만족한다.

이 논문의 구성은 다음과 같다. 먼저 2장에서는 부울 함수에 관한 용어들을 정의하며, 3장에서는 SPEED 암호시스템에 관해 설명한다. 4장에서는 [12]에서 지적한 SPEED의 안전상의 문제점을 설명하고, 5장에서는 부울

· 이 논문은 2000년도 두뇌한국21 사업에 의하여 지원되었음.
† 정 회 원 : 서울대학교 컴퓨터공학부
mklee@theory.snu.ac.kr
†† 종신회원 : 서울대학교 컴퓨터공학부 교수
kpark@theory.snu.ac.kr
cho@csc.snu.ac.kr
논문접수 : 2000년 2월 1일
심사완료 : 2000년 8월 2일

함수들을 생성하고 이들을 이용하여 함수 풀을 구성하는 방법을 제시한다. 6장에서는 개선된 SPEED 암호시스템의 안전성 분석 결과를 제시하며, 7장에서는 개선된 SPEED와 원래의 SPEED의 수행 속도 비교 결과를 제시한다. 8장에서는 결론을 맺는다.

2. 용어의 정의

이 절에서는 [6] 및 [7]에 따라 부울 함수들에 관한 용어들을 정의한다. 먼저, 부울 함수 $f:\{0,1\}^n \rightarrow \{0,1\}$ 의 진리표(truth table)는 $(f(0,0,\ldots,0), f(0,0,\ldots,1), \ldots, f(1,1,\ldots,1))$ 인 길이 2^n의 비트열로 정의되며, 두 함수 f와 g의 해밍 거리(Hamming distance)는 f와 g의 진리표에서 서로 다른 비트의 개수로 정의된다. 비트열 $\alpha \in \{0,1\}^n$에 대한 해밍 웨이트(Hamming weight) $W(\alpha)$는 α에 들어있는 1의 개수를 나타낸다.

안전성을 평가하기 위한 부울 함수의 특성들은 다음과 같이 정의된다.

- **균형성** : 부울 함수의 진리표가 같은 수의 0과 1로 구성될 때, 즉 그 함수의 출력에서 0 또는 1이 편중되지 않고 균등하게 나타날 때 그 함수는 **균형성을 만족한다**고 말한다.
- **비선형성** : 유사 선형 함수(affine function)는 $f(x_1,\ldots,x_n)=a_1x_1 \oplus \cdots \oplus a_nx_n \oplus c$ (단, $a_j, c \in \{0,1\}$) 의 형태로 구성된 함수를 말하며, 특별히 $c=0$인 경우 선형 함수(linear function)라 한다. n-입력 함수 f의 **비선형성** N_f는 모든 가능한 n-입력 유사 선형 함수와 f와의 해밍 거리 중 최소값으로 정의된다. 즉, 비선형성이란 어떤 함수가 유사 선형 함수들과 얼마나 다른 구조를 가지고 있는가를 평가하는 기준으로서, LC에 대한 저항성의 정도를 나타낸다고 할 수 있다. Walsh-Hadamard 변환을 이용하면 함수의 비선형성을 빠르게 계산할 수 있다[4].
- **전달 특성** : 어떤 함수 $f(x)$ $(x \in \{0,1\}^n)$ 및 비트열 $\alpha \in \{0,1\}^n$에 대해 $f(x) \oplus f(x \oplus \alpha)$가 균형성을 만족하는 함수이면 f는 **α에 대해 전달 특성을 만족한다**고 정의한다. 이것은 f의 입력에 α를 XOR하여 적용해도 f의 출력이 정확히 50%만 바뀌게 되는 것을 의미하며, 따라서 이런 α를 많이 가질수록 함수는 안전하다고 말할 수 있다. $1 \le W(\alpha) \le k$를 만족하는 모든 α에 대해 f가 전달 특성을 만족할 경우, f는 **차수 (degree) k의 전달 특성을 만족한다**고 정의한다. 이것은 f의 입력 비트들 중 k개 이하의 임의의 비트들이 바뀌어도 출력 비트들은 정확히 반이 바뀌게 되는 것을 의미한다. 전달 특성은 부울 함수의 안전성 평가의 중요한 기준으로서 SAC(strict avalanche criterion) 및 완전 비선형 특성(perfect nonlinearity criterion)의 일반화된 기준이며, SAC는 차수 1의 전달 특성과, 완전 비선형 특성은 차수 n의 전달 특성과 같다[5, 6]. 완전 비선형 특성을 만족하는 함수를 벤트 함수(bent function)라 하는데, 벤트 함수는 n이 짝수인 경우에만 존재한다[5].

3. SPEED 암호시스템

SPEED는 길이 w인 평문 M을 입력으로 받아 길이 l인 키 K를 이용하여 다시 길이 w인 암호문 C를 생성하는데, 여기서 w, l 및 암호화 라운드 수 r은 가변적인 값을 갖는다. 구체적으로 w는 64, 128, 256의 값을 취할 수 있으며, l은 $48 \le l \le 256$을 만족하는 16의 배수로, r은 32 이상 임의의 4의 배수로 정할 수 있다. Kaliski와 Yin[13]은 RC5[14]가 12 이상의 라운드로 구성될 경우 LC 및 DC에 대해 안전하다는 것을 보인 바 있는데, [1]에서는 RC5와 SPEED의 각 라운드의 구조상의 유사성을 들어 SPEED의 경우 32 이상의 라운드가 사용되면 안전할 것으로 분석하였다.

SPEED의 암호화 과정을 간략하게 나타내면 그림 1과 같다. 암호화 키인 K는 키 스케줄 과정을 통해 4개의 서브키 $K_1,\ldots,K_4$들로 만들어지는데, 이 서브키들은 각각 $wr/32$ 비트로 구성된다. 평문 M은 네 개의 패스(pass) $P_1,\ldots,P_4$를 지나면서 각각 $K_1,\ldots,K_4$에 의해 차례로 암호화되어 암호문 C를 생성하게 된다.

그림 2는 i $(1 \le i \le 4)$번째 패스 P_i를 나타내는데, 각 패스는 각각 $r/4$개의 라운드로 구성되어 있다. P_i에 대한 서브키 K_i는 $r/4$개로 구분되어 각각 $w/8$ 비트인 K_{ij}들을 구성하며, 이들은 각각 그 패스 내의 j번째 라운드 R_{ij}를 위한 키로 사용된다.

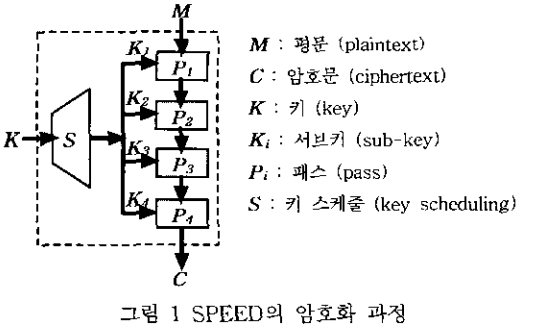

그림 1 SPEED의 암호화 과정

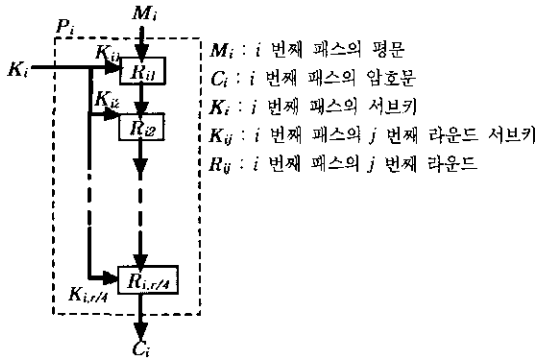

그림 2 SPEED의 i 번째 패스

SPEED의 각 라운드는 Feistel 변환(Feistel transform)[2, 3]에 기반하고 있다. Feistel 변환은 L, R을 같은 길이의 비트열, f를 비트열의 길이를 보존하는 함수, $\oplus$를 XOR라 할 때 $s(L,R)=(R, L\oplus f(R))$의 형태로 표현되는 변환이며, SPEED의 각 라운드는 그림 3과 같이 Feistel 변환의 보다 일반화된 형태로 구성되어 있다. 각 라운드에서 입력 m_{ij}는 각각 $w/8$ 비트인 m_{ijk} $(1\le k\le 8)$들로 구분되는데, 이들을 각각 하나의 **워드(word)**라 정의한다. 이들 평문 블럭들은 각 패스별로 정해져 있는 부울 함수 F_i 및 원형 쉬프트 d와 v, 그리고 키 K_{ij}와의 모듈라 덧셈 등의 연산을 거쳐 출력 블럭들인 c_{ijk} $(1\le k\le 8)$를 생성하게 된다.

F_i $(1\le i\le 4)$는 각 패스마다 정해져 있는 7워드 입력-1워드 출력의 부울 함수로서, 비선형 부울 함수(nonlinear boolean function)들에 관한 최근의 연구 결과들[6, 7]에 기반하여 구성된 함수들이다. 함수의 비선형성은 LC에 대항하기 위한 것이다. 구체적으로 이들 함수들은 아래와 같다. 각 F_i들은 7비트 입력-1비트 출

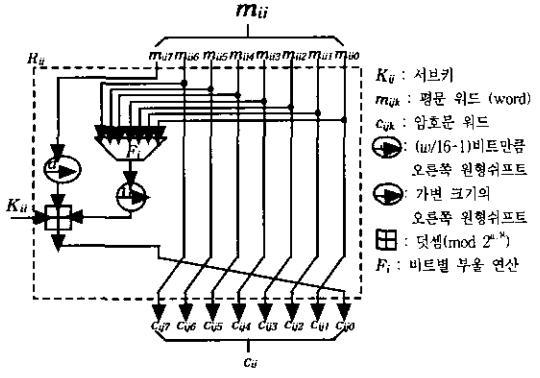

그림 3 i 번째 패스의 j 번째 라운드

력의 부울 함수들이지만, 이것을 7워드 입력-1워드 출력의 함수로 확장하여 해당 비트별로 연산을 적용하면 각 단계의 암호화 함수로 사용할 수 있다.

$$F_1(x_6, x_5, \ldots, x_0) = x_6x_3 \oplus x_5x_1 \oplus x_4x_2 \oplus x_1x_0 \oplus x_0$$

$$F_2(x_6, x_5, \ldots, x_0) = x_6x_4x_0 \oplus x_4x_3x_0 \oplus x_5x_2 \oplus x_4x_3 \oplus x_4x_1 \oplus x_3x_0 \oplus x_1$$

$$F_3(x_6, x_5, \ldots, x_0) = x_5x_4x_0 \oplus x_6x_4 \oplus x_5x_2 \oplus x_3x_0 \oplus x_1x_0 \oplus x_3$$

$$F_4(x_6, x_5, \ldots, x_0) = x_6x_4x_2x_0 \oplus x_6x_5 \oplus x_4x_3 \oplus x_3x_2 \oplus x_1x_0 \oplus x_2$$

그림 3에서 F_i를 거친 결과에는 원형 쉬프트(cyclic shift) v가 적용되는데, 몇 비트가 쉬프트될 것인지는 F_i 결과의 상위 하프워드(half-word)와 하위 하프워드에서 각각 위쪽 $\log_2(w/8)$ 비트씩을 뽑아서 더한 값으로 결정된다. 즉 데이타에 따라 쉬프트의 양이 결정되는데, 이것은 RC5[14]에서도 사용된 방법으로 DC를 막기 위한 효과적인 방법이다.

복호화 과정은 같은 서브키 K_i들을 적용하면서 각 패스를 반대 순서로 수행하는 과정으로, 비밀키 암호시스템인 SPEED의 특성상 암호화 과정과 거의 같은 구조를 가진다.

4. SPEED의 문제점

Hall 등은 SPEED의 안전성에 대해 몇 가지 문제점들을 지적하고, 이를 바탕으로 제한적인 공격 방법을 제시하였다[12]. 이들이 지적한 문제점들은 다음과 같다.

- **비전사성(non-surjectivity)** : F_i를 거친 결과에서는 해당 길이를 갖는 모든 비트열이 나타날 수 있지만, 여기에 데이타 의존 쉬프트 v를 적용하고 난 결과에는 한정된 비트열만이 나타난다. 즉, F_i와 v를 합성한 함수 $v\circ F_i$는 전사함수가 아니다. Rijmen 등[15]은 라운드 함수의 치역을 알고 있다는 가정 하에, 전사가 아닌 라운드 함수를 갖는 암호시스템들에 대해 공격이 가능함을 보인 바 있다.
- **함수의 상호 연관된 출력(correlated outputs)** : SPEED의 부울 함수들의 연속된 출력 간에는 상호 연관성이 존재한다. 예를 들면 1/2+1/32의 확률로 $F_1(x_6, \ldots, x_0) = F_1(x_7, \ldots, x_1)$가 성립하는데, 이것은 F_1 함수의 연속된 출력 사이에 무시할 수 없는 연관성이 존재한다는 것을 의미한다. 이런 연관성은 F_3과 F_4에서도 비슷하게 나타난다.
- **DC의 가능성** : 각 F_i 함수는 각 워드 내의 해당 비트 위치들마다 독립적으로 적용되므로 한 워드 내의 서로 다른 위치의 비트들간에는 영향을 주지 못하는, 즉 비트 위치간 확산(diffusion) 효과가 작은 성질이 있다.

이 사실을 이용하면 각 위치별로 독립적인 차분 특성(differential characteristic)을 구해 DC를 수행하는 것이 가능하다. 물론 각 라운드가 부울 함수들 이외에 모듈라 덧셈이나 원형 쉬프트 등 확산 효과를 가지는 연산들을 가지고 있으므로, 실제로 DC가 쉽게 적용되지는 않는다. [12]에서는 적은 라운드 수를 가지는 SPEED에 대해 DC를 수행한 실험 결과를 제시하였는데, r 라운드의 SPEED에 대해 80% 이상의 공격 성공률을 얻기 위해서는 $2^{2(r-1)}$개 이상의 평문-암호문 순서쌍이 필요한 것으로 나타났다.

5. 함수 풀(Function Pool)

SPEED에서는 4개의 패스에서 서로 다른 부울 함수 F_i $(1 \le i \le 4)$들을 고정시켜 놓고 암호화에 사용한다. 본 논문에서는 부울 함수들로 풀(pool)을 구성한 후 각 패스에서 사용될 함수를 암호화 키 K에 따라 이 풀로부터 선택하도록 SPEED를 개선하였는데, 이런 개선으로 암호화 과정이 원래의 SPEED보다 더욱 가변적이 되어 공격에 대한 저항성이 증대된다.

5.1에서는 주어진 부울 함수들을 이용하여 함수 풀을 구성하는 방법에 대해 설명하고, 5.2에서는 함수 풀을 구성하기 위해 각각의 함수들을 생성하는 방법에 관해 설명한다.

5.1 함수 풀의 구성

암호화에 사용할 부울 함수들이 주어지면 이들을 풀(pool)로 구성하여 암호화 함수로 사용할 수 있는데, k $(4 \le k \ll 255$, k는 소수)개의 함수 $f_1, \ldots, f_k$로 구성된 함수 풀로부터 우리는 i $(1 \le i \le 4)$번째 패스에 사용할 함수 F_i를 다음과 같이 선택하기로 한다.

$$F_i = f_{\{a+b(i-1) \bmod k\}+1} \quad (1 \le i \le 4)$$

여기서 a, b는 암호화 키 K의 처음 8비트 및 두 번째 8비트를 각각 $K_{(1)}$ 및 $K_{(2)}$라 정의할 때 $a = K_{(1)} \bmod k$ 및 $b = \{K_{(2)} \bmod (k-1)\}+1$로 계산되는 값이며, 따라서 $0 \le a \le k-1$ 및 $1 \le b \le k-1$의 범위를 갖는다.

아래의 정리 1은 이러한 방법으로 선택된 각 패스별 함수들이 모두 다르다는 사실을 보여주고 있다.

【정리 1】 위와 같은 방법으로 선택된 F_i들에 대해, $i_1 \ne i_2$ $(1 \le i_1 \le 4,\ 1 \le i_2 \le 4)$이면 $F_{i_1} \ne F_{i_2}$이다.

【증명】 $F_{i_1} = F_{i_2}$ $(i_1 \ne i_2)$라 가정하자. 그러면

$$\{a+b(i_1-1) \bmod k\}+1 = \{a+b(i_2-1) \bmod k\}+1$$

이므로, $b \cdot i_1 \equiv b \cdot i_2 \bmod k$이 성립해야 한다. 그런데 $i_1 \ne i_2$ 및 $1 \le b \le k-1$이고 k가 소수이므로 $b \cdot (i_1 - i_2)$가 k의 배수가 될 수 없어서 모순이 된다. 따라서 $F_{i_1} \ne F_{i_2}$이다. □

다음에는 임의로(random) 선택된 키 K에 대해 위의 방법을 적용했을 때 함수 풀 내의 각 함수 f_j $(1 \le j \le k)$들이 선택되는 빈도에 대해 알아본다. 먼저 아래의 보조정리는 $4 \le k \ll 255$일 경우 a를 해당 범위 내의 임의수(random number)로 간주할 수 있음을 보이고 있다.

【보조정리 2】 $4 \le k \ll 255$라 가정할 때, a가 $0 \le a \le k-1$ 범위 내의 각 값을 취할 수 있는 확률 $P(a=0)$, $P(a=1)$, ... $P(a=k-1)$에 대해

$$P(a=0) \approx P(a=1) \approx \cdots \approx P(a=k-1) \approx \frac{1}{k}$$

이 성립한다.

【증명】 $K_{(1)}$은 $0 \le K_{(1)} \le 255$ 범위에서 임의로 선택되므로, k를 고정시킨 상태라면 a가 취할 수 있는 값들의 확률은 각각 아래와 같다.

$$\begin{aligned}
P(a=0) &= (\lfloor 255/k \rfloor + 1) / 255 \\
&\vdots \\
P(a = 255 \bmod k) &= (\lfloor 255/k \rfloor + 1) / 255 \\
P(a = 255 \bmod k + 1) &= \lfloor 255/k \rfloor / 255 \\
&\vdots \\
P(a = k-1) &= \lfloor 255/k \rfloor / 255
\end{aligned}$$

$k \ll 255$일 경우 $\lfloor 255/k \rfloor \approx \lfloor 255/k \rfloor + 1$이므로

$$P(a=0) \approx P(a=1) \approx \cdots \approx P(a=k-1) \approx \frac{1}{k}$$

이다. (그러나 $k \ll 255$가 만족되지 않으면 이 보조정리는 성립하지 않는다.) □

【정리 3】 a가 $0 \le a \le k-1$ 범위의 임의수라고 가정하자. F_i로 함수 f_j $(1 \le j \le k)$가 선택될 확률을 p_{ij}라 하면, 각 i $(1 \le i \le 4)$에 대해

$$p_{i1} = p_{i2} = \cdots = p_{ik} = \frac{1}{k}$$

이다.

【증명】 F_i의 선택 과정에서 계산되는 $\{a+b(i-1) \bmod k\}+1$ 값을 e라고 정의하자. 일단 e의 계산 과정 중 $b(i-1)$ 부분을 먼저 계산한다고 가정해도 일반성을 잃지 않으므로 먼저 $K_{(2)}$로부터 계산된 $b(i-1)$ 값을 고정시키자. a가 $0 \le a \le k-1$ 범위의 임의수이므로 $b(i-1)$값에 a를 더해 $\bmod k$ 연산한 결과인

$a+b(i-1) \bmod k$는 역시 a와 마찬가지로

$$0 \le a+b(i-1) \bmod k \le k-1$$

범위의 임의수가 되며, 따라서 $e = \{a+b(i-1) \bmod k\}+1$도

$$1 \le e \le k$$

범위의 임의수가 된다. 이것은 $K_{(2)}$값이 무엇이든지 그대로 성립된다.

따라서 $p_{i1} = P(e = 1)$, $p_{i2} = P(e = 2)$, ..., $p_{ik} = P(e = k)$는 모두 같으며, $p_{i1}+p_{i2}+\cdots+p_{ik}=1$이므로 $p_{i1} = p_{i2} = \cdots = p_{ik} = 1/k$이다. □

보조정리 2와 정리 3은 $4 \le k \ll 255$라고 가정할 때 키에 따라 함수 풀 내의 각 함수 $f_j\ (1 \le j \le k)$들이 각 패스별 함수로 선택되는 빈도가 거의 균일함을 보여주고 있다.

본 논문에서는 $4 \le k \ll 255$를 만족하는 소수 중 하나인 $k=7$로 시스템을 구성하였는데, 이 경우 예를 들어 키가 $K=$ 00110100 00011111 ... 이라면 a와 b는 각각 $a=52 \bmod 7=3$ 및 $b=31 \bmod 6+1=2$로 결정되고, 따라서 $F_1 \leftarrow f_4$, $F_2 \leftarrow f_6$, $F_3 \leftarrow f_1$, $F_4 \leftarrow f_3$으로 각 패스별 부울 함수가 결정된다.

5.2 함수의 생성

함수 풀을 구성하기 위해서는 $k\ (4 \le k \ll 255)$개의 함수 $f_1, \ldots, f_k$들을 생성해야 하는데, 이 논문에서는 $k=7$인 시스템을 구성하였다. 이 절에서는 함수 풀을 구성하는 함수들을 생성하는 방법을 설명한다.

먼저 $f_1, \ldots, f_4$는 원래의 SPEED에서 사용된 $F_1, \ldots, F_4$를 그대로 사용한 것으로, 아래와 같다.

$$f_1(x_6, x_5, \ldots, x_0) = x_6x_3 \oplus x_5x_1 \oplus x_4x_2 \oplus x_1x_0 \oplus x_0$$

$$f_2(x_6, x_5, \ldots, x_0) = x_6x_4x_0 \oplus x_4x_3x_0 \oplus x_5x_2 \oplus x_4x_3 \oplus x_4x_1 \oplus x_3x_0 \oplus x_1$$

$$f_3(x_6, x_5, \ldots, x_0) = x_5x_4x_0 \oplus x_6x_4 \oplus x_5x_2 \oplus x_3x_0 \oplus x_1x_0 \oplus x_3$$

$$f_4(x_6, x_5, \ldots, x_0) = x_6x_4x_2x_0 \oplus x_6x_5 \oplus x_4x_3 \oplus x_3x_2 \oplus x_1x_0 \oplus x_2$$

이 함수들은 아래와 같은 특성들을 만족한다[1].

- 모두 균형성을 만족한다.
- 이들의 비선형성은 7-입력 부울 함수에서 얻을 수 있는 최대값[6]인 56이다.
- 전달 특성을 만족하지 않는 비트열을 최대 5개만 가진다.

한편 f_5, f_6, f_7은 [7]의 8.3에서 제시된 방법을 이용하여 새로 생성한 함수들인데, 그 방법은 다음과 같다. 먼저 홀수 $t \ge 5$와 짝수 s에 대해, $g_{(t)}$를 t-입력 함수, $h_{(s)}$를 s-입력 벤트 함수라고 하자. 단 $g_{(t)}$는 전달 특성을 만족하지 않는 비트열을 5개만 가져야 하며, 균형성을 만족해야 한다[7]. 이들 두 함수를 이용하여 $(t+s)$-입력 함수 $g_{(t+s)}$를 다음과 같이 생성한다.

$$g_{(t+s)}(x_1, \ldots, x_{t+s}) = g_{(t)}(x_1, \ldots, x_t) \oplus h_{(s)}(x_{t+1}, \ldots, x_{t+s})$$

이렇게 생성된 함수는 아래와 같은 안전성 요건들을 만족하게 된다[7].

- 균형성을 만족한다.
- $(t+s)$-입력 부울 함수에서 얻을 수 있는 최대 비선형성[6]인 $2^{t+s-1}-2^{(t+s-1)/2}$을 가진다.
- 전달 특성을 만족하지 않는 비트열을 5개만 가진다.

기본 함수 $g_{(t)}$로부터 시작하여 위의 생성 방법을 반복 적용하면, 더 긴 입력을 가지는 함수를 차례로 생성할 수 있으며, [7]에서는 기본 함수로서 다음과 같은 5-입력 함수 $g_{(5)}$를 사용하고 있다.

$$g_{(5)}(x_1, x_2, x_3, x_4, x_5) = (1 \oplus x_1)(1 \oplus x_2)x_3 \oplus (1 \oplus x_1)x_2x_4 \oplus x_1(1 \oplus x_2)(x_3 \oplus x_4) \oplus x_1x_2(x_4 \oplus x_5)$$

이 기본 함수와 2-입력 벤트 함수들 $(h_{(2)})$을 이용하여 본 논문에서 생성한 7-입력 함수들 $(g_{(7)})$인 f_5, f_6, f_7들은 다음과 같다.

$$f_5(x_6, x_5, \ldots, x_0) = x_6x_5x_3 \oplus x_6x_5x_2 \oplus x_6x_3 \oplus x_5x_3 \oplus x_5x_2 \oplus x_3x_2 \oplus x_1x_0 \oplus x_4 \oplus x_0$$

$$f_6(x_6, x_5, \ldots, x_0) = x_3x_2 \oplus x_3x_4 \oplus x_3x_6 \oplus x_3x_5 \oplus x_2x_1 \oplus x_1x_4x_0 \oplus x_6x_5 \oplus x_1x_0 \oplus x_5$$

$$f_7(x_6, x_5, \ldots, x_0) = x_6x_2x_1 \oplus x_6x_2x_0 \oplus x_6x_1x_0 \oplus x_2x_1x_0 \oplus x_6x_5 \oplus x_6x_4 \oplus x_6x_3 \oplus x_5x_0 \oplus x_4x_2 \oplus x_3x_1 \oplus x_6$$

이 함수들은 위에서 말한 안전성 요건들을 만족하는데, 예를 들어 f_5의 경우 균형성을 만족하며, 7-입력 부울 함수에서 얻을 수 있는 최대의 비선형성인 56을 가짐을 확인할 수 있다. 또한 (0, 0, 0, 0, 0, 0, 0), (0, 0, 0, 0, 1, 0, 0), (0, 0, 0, 1, 0, 0, 0), (0, 0, 1, 0, 1, 0, 0), (0, 0, 1, 1, 0, 0, 0)를 제외한 길이 7인 모든 비트열에 대해서 전달 특성을 만족한다.

6. 안전성

SPEED의 구조가 함수 풀에 의해 가변화됨으로써, 우리는 [12]에서 지적된 문제점들 중 비전사성 문제에 대한 개선을 기대할 수 있으며, 또한 DC 및 LC에 대한 안전성의 향상도 기대할 수 있다. 그러나 SPEED의 문제점 중 하나인 함수의 상호 연관된 출력 부분은 제안된 방식으로 해결되지 않는다.

- **비전사성 문제** : 비전사 라운드 함수를 갖는 암호 시스템들에 대한 Rijmen 등[15]의 공격은 라운드 함수의 치역에 대한 예측을 기반으로 하고 있다. 그러나 [12]에서는 SPEED의 각 라운드에서 $v \cdot F_i$가 전사함수가 아니라는 점을 지적하면서도, $v \cdot F_i$에 다시 서브키

가 결합되므로 치역을 예측하기가 간단하지 않다는 점을 언급하고 있다. 본 논문에서 제안한 개선된 시스템은 가변적인 함수를 사용하여 치역을 예측하기가 더욱 어려워짐으로써, 치역의 예측에 기반한 [15]의 공격 방법은 적용이 불가능하다.

- **DC** : [12]에서는 비트간 확산(diffusion) 효과가 결여되어 있다는 점을 이용하여 DC를 적용한 결과를 제시하였는데, r 라운드의 SPEED에 대해 80% 이상의 높은 성공률을 얻기 위해서는 실험적으로 $2^{2(r-11)}$개 이상의 평문-암호문 순서쌍이 필요할 것으로 추정하였다. 여기에 k개의 함수를 이용한 함수 풀을 적용할 경우, 키에 관한 정보가 없는 공격자 입장에서는 키와 함수 선택을 독립적이라고 가정하여야 하므로, a와 b의 값을 임의로 뽑는다고 보면 $k(k-1)$배의 순서쌍이 필요할 것이다. 물론 공격 과정에서 함수와 키의 의존 관계에 관한 추가적인 정보를 얻게 될 것이므로 이 수치가 그대로 적용되지는 않겠지만, 이런 추가적인 정보를 배제할 경우 함수 풀에 의한 암호시스템 구조의 가변화는 DC에 대한 저항성을 최대 $k(k-1)$배 향상시킨다고 할 수 있다. 예를 들어 [1]에서 안전성을 보장받기 위한 최소의 라운드 수로 제시한 $r=32$를 그대로 적용하고 이 논문에서처럼 7개의 함수로 풀을 구성할 경우, DC에 필요한 순서쌍은 $2^{42}\cdot 42$개 이상이 된다. 이 수치는 함수 풀의 크기 k를 크게 함으로써 임의로 크게 할 수 있다.
- **LC** : LC는 암호시스템의 비선형 부분들(nonlinear components)을 확률적인 선형 근사식으로 대체하여 키의 특정 비트들을 찾는 방식에 기반하고 있다. 따라서 키에 의존한 연산이나 높은 비선형성을 갖는 연산에 대해서는 확률이 높은 근사식을 만드는 것이 어려운 것으로 알려져 있다[1, 16]. 본 논문의 개선된 SPEED에서 암호화 함수들은 키에 의존하여 선택되고 각 함수들은 모두 최대의 비선형성을 만족하고 있으므로, 개선된 SPEED는 LC에 대한 향상된 저항성을 가진다. 또한, 이 함수들은 비선형성 이외에 균형성, 전달 특성 등의 추가적인 안전성 요건들을 만족하고 있다.

7. 수행 속도

이 절에서는 원래의 SPEED와 개선된 SPEED의 수행 속도를 비교한다. 블록 크기 w, 키의 크기 l 및 암호화 라운드 수 r은 동일하게 $w=256$, $l=128$, $r=32$로 하여 실험하였다. 표 1은 이들의 키 스케줄 시간 및 암호화 속도를 측정하여 성능을 비교한 결과를 보여주고 있는데, 키 스케줄 시간은 10,000,000개의 키를 임의로 생성하여 평균을 측정하였으며, 암호화 속도도 역시 임의의 256비트 블록을 10,000,000개 생성하여 평균을 측정하였다. 실험은 Celeron, Pentium III 및 UltraSPARC 등 여러 가지 환경에서 수행하였다.

표 1 개선된 SPEED 암호시스템의 성능

암호 시스템	수행 환경	키 스케줄 시간(μsec)	암호화 속도 (Mbits/sec)
SPEED	①	2.44	114.35
	②	2.00	138.38
	③	5.18	57.04
개선된 SPEED	①	2.44	94.73 (94.73 ÷ 114.35 ≈ 82.8%)
	②	2.00	123.01 (123.01 ÷ 138.38 ≈ 88.9%)
	③	5.20	36.65 (36.65 ÷ 57.04 ≈ 64.3%)

①: Celeron 433MHz / 128MB RAM / Windows NT 4.00.1381 / Visual C++ 6.0
②: Pentium III 667MHz / 128MB RAM / linux 2.3.99-pre9 / gcc 2.91.66
③: UltraSPARC 167MHz / 128MB RAM / Solaris 2.6 / gcc 2.95.2

성능 비교 결과에서는 개선된 SPEED의 키 스케줄 시간은 원래의 SPEED와 거의 같고 암호화 속도는 원래의 SPEED에 비해 11.1% ~ 35.7% 가량 늦어지는 것으로 나타났다. 암호화 속도가 늦어지는 것은 부울 함수가 많아지고 가변적이 되면서 함수를 선택하기 위한 오버헤드가 추가되기 때문인 것으로 생각된다. 함수 선택 시에는 mod, 즉 나눗셈 연산이 필요한데, Pentium이나 Celeron CPU는 UltraSPARC보다 곱셈, 나눗셈을 더 효율적으로 수행할 수 있는 구조로 되어 있으므로 UltraSPARC에서 속도의 감소가 더 두드러지는 것을 관찰할 수 있다.

일반적으로 SPEED를 비롯한 Feistel 변환 기반의 암호시스템들에서는 라운드 수를 늘리면 안전성이 높아지는 것으로 알려져 있다[1]. 그러나 라운드 수의 증가는 암호화 속도의 감소를 의미한다. 본 논문에서 제안한 함수 풀을 이용한 방식은 라운드 수의 증가가 필요하지 않으므로, 안전성을 향상시키면서도 표 1에서 보여주는 것처럼 원래의 암호시스템의 속도를 크게 떨어뜨리지 않는 것을 알 수 있다.

8. 결 론

본 논문에서는 암호화에 사용되는 부울 함수들을 키

에 따라 가변적으로 선택함으로써 LC 및 DC 등의 공격 방법들에 대한 저항성을 높인 개선된 SPEED 암호 시스템을 제안하였다. 앞으로의 연구에서는 함수 호출 단계의 간소화 등 구현시의 최적화를 통해, 함수 풀의 속도 감소의 요인 중 한가지인 함수 선택의 오버헤드 문제를 개선할 수 있을 것으로 예상된다.

참 고 문 헌

[1] Y. Zheng, "The SPEED cipher," *Proc. of Financial Cryptography '97, LNCS*, Vol.1318, pp. 24-28, Springer-Verlag, 1997.

[2] H. Feistel, "Cryptography and computer privacy," *Scientific American*, Vol.228, pp. 15-23, 1973.

[3] H. Feistel, W. A. Notz and J. L. Smith, "Some cryptographic techniques for machine-to-machine data communications," *Proceedings of IEEE*, Vol.63, No.11, pp. 1545-1554, 1975.

[4] W. Meier and O. Staffelbach, "Nonlinearity criteria for cryptographic functions," *Proc. of EUROCRYPT '89, LNCS*, Vol.434, pp. 549-562, Springer-Verlag, 1990.

[5] B. Preneel, W. V. Leekwijck, L. V. Linden, R. Govaerts and J. Vandewalle, "Propagation characteristics of Boolean functions," *Proc. of EUROCRYPT '90, LNCS*, Vol.473, pp. 161-173, Springer-Verlag, 1991.

[6] J. Seberry, X. M. Zhang and Y. Zheng, "Nonlinearity and propagation characteristics of balanced Boolean functions," *Information and Computation*, Vol.119, No.1, pp. 1-13, 1995.

[7] X. M. Zhang and Y. Zheng, "Characterizing the structures of cryptographic functions satisfying the propagation criterion for almost all vectors," *Design, Codes and Cryptography*, Vol.7, No.2, pp. 111-134, 1996.

[8] W. Millan, A. Clark and E. Dawson, "An effective genetic algorithm for finding highly nonlinear Boolean functions," *International Conference on Information and Communications Security '97*, pp. 149-158, 1997.

[9] W. Millan, A. Clark and E. Dawson, "Heuristic design of cryptographically strong balanced Boolean functions," *Proc. of Eurocrypt '98, LNCS*, Vol.1403, pp. 489-499, Springer-Verlag, 1998.

[10] M. Matsui, "Linear cryptanalysis method for DES cipher," *Proc. of EUROCRYPT '93, LNCS*, Vol.765, pp. 386-397, Springer-Verlag, 1994.

[11] E. Biham and A. Shamir, "Differential cryptanalysis of the full 16-round DES," *Proc. of CRYPTO '92, LNCS*, Vol.740, pp. 487-496, Springer-Verlag, 1993.

[12] C. Hall, J. Kelsey, V. Rijmen, B. Schneier and D. Wagner, "Cryptanalysis of SPEED," *Fifth Annual Workshop on Selected Areas in Cryptography (SAC '98), LNCS*, Vol.1556 pp. 319-338, Springer-Verlag, 1999. (Also in the rump session of *Financial Cryptography '98*)

[13] B. S. Kaliski Jr. and Y. L. Yin, "On differential and linear cryptanalysis of the RC5 encryption algorithm," *Proc. of CRYPTO '95, LNCS*, Vol.963, pp. 171-184, Springer-Verlag, 1995.

[14] R. Rivest, "The RC5 encryption algorithm," *Proc. of Fast Software Encryption, LNCS*, Vol.1008, pp. 86-96, Springer-Verlag, 1995.

[15] V. Rijmen, B. Preneel and E. De Win, "On weaknesses of non-surjective round functions," *Design, Codes and Cryptography*, Vol.12, No.3, pp. 253-266, November 1997.

[16] P. Hawkes and L. O'Connor, "On applying linear cryptanalysis to IDEA," *Proc. of ASIACRYPT '96, LNCS*, Vol.1163, pp. 105-115, Springer-Verlag, 1996.

이 문 규

1996년 서울대학교 컴퓨터공학과 학사. 1998년 서울대학교 컴퓨터공학과 석사. 1998년 ~ 현재 서울대학교 컴퓨터공학부 박사과정. 관심분야는 컴퓨터이론, 암호학

박 근 수

1983년 서울대학교 컴퓨터공학과 학사. 1985년 서울대학교 컴퓨터공학과 석사. 1991년 미국 Columbia University 전산학 박사. 1991년 ~ 1993년 영국 University of London, King's College 조교수. 1993년 ~ 현재 서울대학교 컴퓨터공학부 부교수. 관심분야는 컴퓨터이론, 암호학, 병렬계산.

조 유 근

1971년 서울대학교 건축공학과 학사. 1978년 미국 University of Minnesota 전산학 박사. 1979년 ~ 현재 서울대학교 컴퓨터공학부 교수. 1984년 ~ 1985년 미국 University of Minnesota 교환교수. 1993년 ~ 1995년 서울대학교 중앙교육연구전산원장. 1995년 한국정보과학회 부회장. 1999년 ~ 현재 서울대학교 공과대학 부학장. 관심분야는 운영체제, 알고리즘 설계 및 분석, 암호학.