情報保護學會論文誌
第17卷 第6號, 2007. 12

블록암호에 대한 새로운 다중선형공격법*

홍 득 조[1†], 성 재 철[2], 홍 석 희[1‡], 이 상 진[1]
[1]고려대학교 정보보호기술연구센터(CIST), [2]서울시립대학교 수학과

New Multiple Linear Cryptanalysis of Block Ciphers

Deukjo Hong[1†], Jaechul Sung[2], Seokhie Hong[1‡], Sangjin Lee[1]
[1]Korea University, CIST, [2]University of Seoul, Dep. of Mathematics

요 약

다중선형공격은 선형공격을 강화하기 위한 방법으로 연구되어왔다. 본 논문에서는 다중선형공격에 관한 최신 이론인 Biryukov의 공격 알고리즘이 비선형 키스케줄을 가진 블록암호에 적용이 어려움을 지적하고, 새로운 다중선형공격법을 제안한다. 작은 블록암호에 대한 실험을 통하여 새로운 다중선형공격법에 관한 이론이 실제로도 매우 잘 적용될 수 있음이 보여진다.

ABSTRACT

Multiple linear cryptanalysis has been researched as a method building up the linear attack strength. We indicate that the lastest linear attack algorithm using multiple approximations, which was proposed by Biryukov et al. is hardly applicable to block ciphers with highly nonlinear key schedule, and propose a new multiple linear attack algorithm. Simulation of the new attack algorithm with a small block cipher shows that theory fot the new multiple linear cryptanalysis works well in practice.

Keywords : Multiple Linear Cryptanalysis, Linear Attack, Multiple Linear Approximations

Ⅰ. 서 론

1.1. 블록암호에 대한 선형공격의 역사

블록암호에 대한 선형공격은 1993년 M. Matsui에 의해 처음으로 소개되었다[7, 8]. 선형공격은 차분공격[1]과 더불어 블록암호에 가장 널리 적용되는 공격법이다. 차분공격이 고계차분공격, 부정차분공격, 불능차분공격, 부메랑공격, 렉탱글공격 등으로 다양하게 발전해왔듯이, 선형공격을 발전시키기 위한 다양한 연구와 시도가 있었다. 그 중 대표적인 것이 다중선형공격이다.

다중선형공격의 아이디어는 간단하다. Matsui가 최초에 제안한 두 개의 공격 알고리즘은 가장 좋은 바이어스를 가질 것으로 기대되는 하나의 선형근사식만을 이용했다[7]. 다중선형공격은 이 알고리즘들을 여러 개의 선형근사식을 이용하는 공격법으로 확장시키는 것이다. 그럼으로써 공격의 데이터 복잡도 혹은 시간 복잡도를 낮추는 효과를 얻으려는 것이다. 이 아이디어는

접수일: 2007년 7월 2일; 채택일: 2007년 9월 10일
* 본 연구는 정보통신부 및 정보통신연구진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었습니다.
† 주저자, hongdj@cist.korea.ac.kr
‡ 교신저자, hsh@cist.korea.ac.kr

1994년 Kaliski Jr.과 Robshaw에 의하여 제안되었다[5]. 그러나 그들의 알고리즘은 키비트 마스크가 동일한 선형근사식들만을 이용해야한다는 제약을 갖고 있었다.

다중선형공격의 다른 흐름을 제시한 것은 2003년의 Junod 논문이다[3,4]. 그는 Matsui가 12대의 병렬연결된 워크스테이션으로 DES[9]에 대한 선형공격을 구현한 결과를 발표했음에 주목했다[8]. 그 공격에서 Matsui는 각각 13 비트의 키 정보를 갖는 두 개의 선형근사식을 이용했다. Junod는 블록암호에 대한 공격을 키정보에 대한 직접적인 분할-정복 (Divide- and-conquer) 공격으로 보지 않고, 모든 가능한 키 후보에 대한 목록을 작성하는 것으로 보았다. Junod의 관점에서 Matsui의 공격은 최적화되지 못한 방법이었다. 왜냐하면, 두 개의 선형근사식에서 작성되는 두 개의 부분키의 리스트를 하나로 합치는 과정이 최적이 아니었기 때문이다. 그는 Neyman-Pearson의 binary hypothesis testing problem을 적용하여 이것을 해결했으며, 두 개 이상의 키 후보 목록에 대해서도 일반적으로 적용할 수 있음을 주장했다. Junod의 연구는 다중선형공격 그 자체에 관한 것은 아니었으며, 그렇다고 그가 데이터 복잡도나 시간 복잡도와 성공확률간의 관계를 설명한 것도 아니었다. 그러나 Junod의 관점은 다중선형공격에 관한 최신 이론인 Biryukov 등의 연구에 큰 영향을 준 것은 사실이다.

Biryukov 등은 새로운 다중선형공격법을 제시하면서 Kaliski Jr.과 Robshaw가 제안한 공격법이 갖는 제약을 해결했다고 주장했으며, 이를 DES에 대한 실험을 통해 입증해보였다[2]. 그들의 연구에는 블록암호에 대한 공격을 모든 가능한 키 후보를 관측된 통계량에 따라 재정렬하는 것으로 보는 Junod의 관점이 바탕이 되었다. 그러나, Biryukov 등이 제시한 다중선형공격법은 DES와 같이 선형적인 키스케줄을 가진 블록암호에는 효과적일 수 있으나, 비선형 키스케줄을 갖는 블록암호에는 적용이 어렵다. 선형 키스케줄을 갖는 경우, Biryukov 등의 알고리즘은 마스터키의 부분적인 키정보를 대표값으로 갖는 키후보의 coset들을 정렬하게 된다. 그러나 비선형 키스케줄의 경우에는 선형근사식의 우변이 근사식의 개수보다 훨씬 더 많은 키 정보를 포함하게 되어 오히려 키공간은 분할되지 않게 된다. 비선형 키스케줄의 설계가 그렇게 어렵지 않다는 점에서 그들이 제시한 공격 알고리즘의 유용성에 의문이 남는다.

1.2. 본 논문의 연구 결과

본 논문에서는 비선형 키스케줄을 갖는 블록암호에 Birukov의 다중선형공격법이 적용될 수 없는 이유를 구체적으로 밝히고, 비선형 키스케줄을 갖는 블록암호에 효과적으로 적용될 수 있는 새로운 다중선형공격법을 제시한다. 본 논문의 실험결과는 새로운 다중선형공격에 관한 이론은 실제에서도 잘 적용된다는 것을 보여준다.

1.3. 기호 정리

우선 블록암호 E의 입출력들을 정의하자. P, C, K를 각각 블록암호 E의 평문, 암호문, 키라고 하자. E의 블록길이는 n이며, 키 길이는 k라고 하자. 그러므로 P, C를 n차 벡터공간 $GF(2)^n$의 원소로, K를 k차 벡터공간 $GF(2)^k$의 원소로 여길 수 있다. 블록암호 E는 총 r라운드로 구성되어 있으며, 키스케줄이 q개의 l비트의 서브키 $W_1, \cdots, W_q$를 생성한다고 하자. $W_1, \cdots, W_q$는 l차 벡터공간 $GF(2)^l$의 원소로 볼 수 있다. i번째 $(1 \le i \le r)$ 라운드함수는 $X_i = F_i(X_{i-1})$로 표시한다. X_{i-1}은 i번째 라운드함수 F_i의 입력이며, X_i는 F_i의 출력인 동시에 $i+1$번째 라운드함수 F_{i+1}의 입력이기도 하다. 따라서 $X_0 = P$, $X_r = C$ 이다.

Ⅱ. Biryukov의 다중선형공격

2.1. Biryukov의 다중선형공격 개요

키스케줄이 선형이라 가정하고, Biryukov 등이 제시한 공격알고리즘 중 Algorithm MK1을 살펴보자[2]. 이것은 Matsui의 Algorithm 1을 확장시킨 것이다. 키스케줄이 선형이기 때문에, 공격에 이용되는 m개의 선형근사식을 다음과 같이 표현할 수 있다.

$$\begin{aligned} &\alpha_1 \cdot P \oplus \beta_1 \cdot C \oplus b_1 = \gamma_1 \cdot K = z_1 \\ &\alpha_2 \cdot P \oplus \beta_2 \cdot C \oplus b_2 = \gamma_2 \cdot K = z_2 \\ &\qquad \vdots \\ &\alpha_m \cdot P \oplus \beta_m \cdot C \oplus b_m = \gamma_m \cdot K = z_m \end{aligned} \tag{1}$$

$\cdot$는 두 이진벡터간의 내적을 의미한다. 상수 비트인 b_i는 각 선형근사식이 성립할 확률 p_i가 1/2보다 크도록 하는 값으로 고정된다. 각 p_i는 다음과 같이 표현된다.

$$p_i = \frac{1}{2} + \varepsilon_i = \frac{1+c_i}{2} \quad (2)$$

ε_i를 i번째 근사식의 바이어스라 하며, c_i를 i번째 근사식의 임밸런스(imbalance)라 한다. $\boldsymbol{z} = (z_1, z_2, \cdots, z_m)$를 선형근사식의 우변값들의 벡터라고 하자. 거의 모든 선형공격법과 마찬가지로, Biryukov의 다중선형공격법은 주어진 데이터로부터 필요한 통계량을 추출하는 **통계량 추출과정**, 통계량을 분석하고 키후보를 정렬하는 **키 후보 정렬과정**, 옳은 키를 검색하는 **키 검색과정**으로 구성되어 있다.

통계량 추출과정에서는 N개의 평문-암호문쌍 (P_i, C_i)이 주어지면, 각 식의 좌변값을 계산하고, i번째 식의 좌변이 0이 될 때마다 카운터 t_i의 값을 1씩 증가시킨다.

키 후보 정렬과정에서는 이전 단계에서 추출된 통계량인 $t_1, \cdots, t_m$을 $\hat{c}_i = 2t_i/N - 1$로 표현하여 각 근사식의 불균형성에 대한 관측값으로 활용한다. $z_i = 0$이라면 N이 커질수록 $\hat{c}_i$는 c_i에 근접하게 되지만, 반대로 $z_i = 1$이라면 $\hat{c}_i$는 $-c_i$에 근접하게 된다. $\hat{\boldsymbol{c}}$와 $\boldsymbol{c}_{\boldsymbol{z}}$를 각각 다음과 같이 정의하자.

$$\hat{\boldsymbol{c}} = (\hat{c}_1, \hat{c}_2, \cdots, \hat{c}_z), \quad \boldsymbol{c}_{\boldsymbol{z}} = ((-1)^{z_1}c_1, (-1)^{z_2}c_2, \cdots, (-1)^{z_m}c_m) \quad (3)$$

또, 설명의 편의를 위해 모든 z_i가 독립이라고 가정하자. 그러면 $\boldsymbol{z}$에 대한 2^m개의 모든 가능한 후보는 다음 값에 따라 오름차순으로 정렬된다.

$$|\hat{\boldsymbol{c}} - \boldsymbol{c}_{\boldsymbol{z}}|^2 = \sum_{i=1}^{m} (\hat{c}_i - (-1)^{z_i}c_i)^2 \quad (4)$$

이것은 각 키 후보 $\boldsymbol{z}$에 대응되는 불균형성의 벡터 $\boldsymbol{c}_{\boldsymbol{z}}$와 m차 공간의 한 점 $\hat{\boldsymbol{c}}$간의 거리가 가까운 순으로 키후보를 정렬한 것이라 볼 수 있다.

마지막으로 키 검색과정에서는 이전 단계에서 만들어진 키 후보의 정렬된 목록에 따라 옳은 키를 검색한다. 키 후보 정렬과 키 검색 과정은 혼합되어 적용될 수도 있다.

[2]에는 Matsui의 Algorithm 2를 확장시킨 Algorithm MK2도 제시되어 있는데, 다음과 같은 형태의 선형근사식들을 이용한다.

$$\begin{aligned} \alpha_1 \cdot X_1 \oplus \beta_1 \cdot X_{r-1} \oplus b_1 &= \gamma_1 \cdot K, \\ &\vdots \\ \alpha_m \cdot X_1 \oplus \beta_m \cdot X_{r-1} \oplus b_m &= \gamma_m \cdot K. \end{aligned} \quad (5)$$

[2]에 의하면, 좌변의 $\alpha_i \cdot X_1$과 $\beta_j \cdot X_{r-1}$을 계산하는데 필요한 서브키 비트들의 벡터를 $\boldsymbol{z}_I$로, 우변의 $z_1, \cdots, z_m$ 비트들의 벡터를 $\boldsymbol{z}_O$로 정의한다. Algorithm MK2는 각 가능한 $\boldsymbol{z}_I$ 값에 대하여, 그것에 대응되는 $\boldsymbol{z}_O$의 목록을 만드는 것이다. 각 $\boldsymbol{z}_I$에 대하여 $\hat{\boldsymbol{c}}$가 다르게 계산되므로, $\boldsymbol{z}_I$와 연관된 $\hat{\boldsymbol{c}}$를 $\hat{\boldsymbol{c}}_{\boldsymbol{z}_I} = (\hat{c}_{\boldsymbol{z}_I 1}, \cdots, \hat{c}_{\boldsymbol{z}_I m})$로 표기하자. 각 목록들은 다음 값에 따라 $(\boldsymbol{z}_I, \boldsymbol{z}_O)$에 대한 하나의 통합된 목록으로 만들어질 수 있다.

$$|\hat{\boldsymbol{c}}_{\boldsymbol{z}_I} - \boldsymbol{c}_{\boldsymbol{z}_O}|^2 = \sum_{j=1}^{m} (c_{\boldsymbol{z}_I, j} - (-1)^{z_j}c_j)^2 \quad (6)$$

대부분의 경우 Algorithm MK2가 Algorithm MK1보다 효율적이다.

2.2. Biryukov의 다중선형공격이 비선형 키스케줄을 갖는 블록암호에 적용되기 어려운 이유

[2]는 DES의 8라운드에 대한 Algorithm MK1과 MK2의 실험결과를 보여줌으로써, 두 공격 알고리즘이 Matsui의 알고리즘보다 효과적임을 역설하고 있다. 그러나 이것은 모든 블록암호에 적용될 수 있는 결과가 아니다. 블록암호의 키스케줄이 비선형적이라 가정하자. 그것도 비선형성이 매우 높다고 하자. 효율성을 고려하지 않는다면 비선형 키스케줄을 만드는 방법은 그렇게 어렵지 않다. 암호화 알고리즘의 평문에 키를, 각 서브키에 고정된 상수를 입력하여 각 라운드의 출력값을 서브키로 얻는 것도 하나의 방법이다. 여기서 우리가 지적하고자 하는 것은 블록암호의 키스케줄이 높은 비선형성을 가질 때, Algorithm MK1과 MK2가 제공하는 정보가 모호해진다는 것이다.

$\mathbb{W} = (W_1, \cdots, W_q)$를 서브키의 벡터라 하자. 그러

면, Algorithm MK1에 이용되는 m개의 선형근사식들은 (1)이 아닌 다음과 같은 형태를 갖게 된다.

$$\begin{aligned}&\alpha_1 \cdot P \oplus \beta_1 \cdot C \oplus b_1 = \delta_1 \cdot \mathbb{W} = z_1,\\ &\alpha_2 \cdot P \oplus \beta_2 \cdot C \oplus b_2 = \delta_2 \cdot \mathbb{W} = z_2,\\ &\quad\vdots\\ &\alpha_m \cdot P \oplus \beta_m \cdot C \oplus b_m = \delta_m \cdot \mathbb{W} = z_m.\end{aligned} \tag{7}$$

모두 다른 (α_j,β_j)에 대해서, Algorithm MK1을 적용하면 모든 가능한 $\boldsymbol{z}=(z_1,\cdots,z_m)$에 대하여 정렬된 목록을 얻을 수 있다. 그러나 그 목록으로 마스터키 K 또는 서브키 $W_1,\cdots,W_q$의 부분 정보를 얻기는 쉽지 않을 것이다. 마스터키 K를 복구하는 관점에서 일반적으로 $\boldsymbol{z}$의 목록에서 얻을 수 있는 것은 K의 비트들에 대한 수많은 고차단항식들의 선형결합식들에 대한 부분정보이다. 이러한 비선형 결합식에 대한 해를 얻기 위해서는 엄청나게 많은 수의 관계식이 필요하다. 그러한 해법이 가능할 정도의 선형근사식을 얻는 것도 불가능에 가까울뿐더러, 설령 얻을 수 있다고 해도 다중선형공격 알고리즘의 시간복잡도를 크게 증가시킬 것이다. 또, 라운드키를 복구하는 관점에서도, $\boldsymbol{z}$의 목록에서 얻을 수 있는 것은 서브키 비트들의 선형결합식들에 대한 부분정보이다. 그러나 여기에는 평문을 암호화하는데 이용된 모든 서브키의 비트들이 포함된다. 만약, $\boldsymbol{z}$로부터 첫 번째와 마지막 라운드에 사용된 서브키들만의 관계식을 얻어낼 수 있다면 그 서브키들의 부분정보를 얻을 수 있겠지만, 일반적으로 블록 암호에 대하여 그런 일이 발생할 확률은 매우 작다. 블록암호에 의하여 출력되는 암호문의 각 비트도 평문과 키의 비트들에 대한 고차 방정식으로 표현될 수 있지만 그것으로부터 평문이나 키에 대한 정보를 얻어내기가 무척 어려운 것과 비슷한 이치이다.

그렇다면 Algorithm MK2의 경우는 어떤지 살펴보자. 비선형 키스케줄을 갖는 블록암호로부터 얻어진 아래와 같은 식이 Algorithm MK2가 적용된다고 하자.

$$\begin{aligned}&\alpha_1 \cdot X_1 \oplus \beta_1 \cdot X_{r-1} \oplus b_1 = \delta_1 \cdot \mathbb{W} = z_1,\\ &\quad\vdots \qquad\qquad\qquad\quad \vdots\\ &\alpha_m \cdot X_1 \oplus \beta_m \cdot X_{r-1} \oplus b_m = \delta_m \cdot \mathbb{W} = z_m.\end{aligned} \tag{8}$$

이 경우 $(\boldsymbol{z}_I\boldsymbol{z}_O)$에 대한 분할-정복 공격이 가능할 만큼 많은 평문-암호문 쌍을 이용한다면 $\boldsymbol{z}_O$에 대한 것은 여전히 모호하지만, $\boldsymbol{z}_I$에 대한 정보는 거의 확실하게 얻을 수 있다. 많은 평문-암호문 쌍을 이용할수록 알고리즘에 의하여 만들어지는 $(\boldsymbol{z}_I\boldsymbol{z}_O)$ 목록의 맨 첫 순위에 대응되는 값이 $(\boldsymbol{z}_I\boldsymbol{z}_O)$에 대한 옳은 값일 확률이 높아지기 때문이다. 문제는, 선형키스케줄을 갖는 블록암호에 적용되었을 때와는 달리, 평문-암호문 쌍을 많이 이용해야만 $\boldsymbol{z}_I$에 대한 정보를 얻을 수 있다는 것이다. 많은 선형근사식과 많은 평문-암호문 쌍을 공격에 이용하는 것은 공격의 복잡도를 너무 크게 만들 우려가 있다. 그렇다고 Algorithm MK2에 적은 평문-암호문쌍을 이용하면 각 $\boldsymbol{z}_I$값에 대하여 생성되는 $\boldsymbol{z}_O$ 목록들이 제공하는 정보의 모호성은 $\boldsymbol{z}_I$에도 확산되기 때문에 공격의 효율은 크게 떨어진다.

Ⅲ. 새로운 다중선형공격법

3.1. 공격 알고리즘 개요

이 절에서는 비선형 키스케줄을 갖는 블록암호에도 적용할 수 있는 새로운 다중선형공격 알고리즘을 소개한다. 이것을 Algorithm MK3라 하자. Algorithm MK3는 다음과 같은 m개의 식을 이용한다.

$$\begin{aligned}&\alpha_1 \cdot X_1 \oplus \beta_1 \cdot X_{r-1} = 0,\\ &\quad\vdots\\ &\alpha_m \cdot X_1 \oplus \beta_m \cdot X_{r-1} = 0.\end{aligned} \tag{9}$$

$\alpha_j \cdot X_1 \oplus \beta_j \cdot X_{r-1}$의 계산에 필요한 서브키 비트열을 $\boldsymbol{z}$라 하자. 그리고 공격의 효율을 높이기 위해, 모든 식은 같은 활성(active) S박스를 갖는다고 가정하자. N개의 평문-암호문 쌍이 이용된다고 하자. 공격의 첫 단계인 통계량 추출과정에서는 가능한 모든 $\boldsymbol{z}$값에 대하여 (9)의 식들을 만족시키는 평문-암호문 쌍의 개수의 벡터 $\boldsymbol{t}_{\boldsymbol{z}}=(t_{\boldsymbol{z},1},t_{\boldsymbol{z},2},\cdots,t_{\boldsymbol{z},m})$들을 구한다. (9)의 각 식들의 우변이 서브키비트에 관한 어떤 식으로 표현될 수 있는지는 관심의 대상이 아니라는 것이 Algorithm MK1, MK2와의 차이점이다. 이것은 본질적으로 큰 차이를 가져온다. Algorithm MK1과 MK2에서 이용되는 식들은 선형근사식이라 불리는 반면, Algorithm MK3에서 이용되는 것은 linear hull이다. 그러므로 j번째 식의 임밸런스의 제곱은 α_j와 β_j를 X_1과 X_{r-1}의 비트마스크로 갖는 모든 선형근사식들의 임밸런스의 제곱들의 합이기 때문에, 확률적 관점에서 큰 이득을 얻을 수 있다.

키 후보 정렬과정에서는 이전단계에서 생성된 벡터 $\boldsymbol{t}_z$들을 다음과 같은 값으로 변환한다.

$$u_z = \frac{4}{N}\sum_{j=1}^{m}\left(t_{z,j} - \frac{N}{2}\right)^2 \quad (10)$$

그 다음, u_z 값에 따라 대응되는 서브키 비트열의 후보들을 내림차순으로 정렬한다. Algorithm MK1과 MK2가 서브키들의 후보들을 오름차순으로 정렬한 반면, Algorithm MK3이 내림차순으로 정렬하는 이유는 옳은 서브키 비트열인 $\boldsymbol{z}^*$에 대하여 N이 커질수록 $|\hat{\boldsymbol{c}} - \boldsymbol{c}_{\boldsymbol{z}^*}|^2$의 값이 0으로 수렴할 것으로 기대되지만, 반대로 $u_{\boldsymbol{z}^*}$의 값은 0으로부터 점점 멀어질 것으로 기대되기 때문이다.

마지막으로 키 검색과정에서는 이전 단계에서 작성된 목록을 이용하여 서브키 또는 마스터키의 옳은 값을 검색한다.

3.2. 성공 확률

$\boldsymbol{z}^*$를 옳은 서브키 비트열이라 하자. (9)의 j번째 식의 임밸런스를 $c_j > 0$라 하자. 벡터 $\boldsymbol{t}_{\boldsymbol{z}^*}$의 각 성분 $t_{\boldsymbol{z}^*,j}$ 는 정규분포 $\mathrm{N}(N(1+c_j)/2, N/4)$ 또는 $\mathrm{N}(N(1-c_j)/2, N/4)$를 따른다. 반면, $\boldsymbol{z} \neq \boldsymbol{z}^*$인 $\boldsymbol{z}$에 대하여 $t_{\boldsymbol{z},j}$ 는 정규분포 $\mathrm{N}(N/2, N/4)$을 따른다. 그러므로, $u_{\boldsymbol{z}}$는 자유도가 m인 전형적인 카이제곱분포를 따른다. 이 때의 $u_{\boldsymbol{z}}$의 확률밀도함수를 $f_{\boldsymbol{z}}(x;m)$라 하자. $f_{\boldsymbol{z}}(x;m)$은 다음과 같은 형태를 갖는다.

$$f_{\boldsymbol{z}}(x;m) = \frac{(1/2)^{m/2}}{\Gamma(m/2)} x^{\frac{m}{2}-1} e^{-\frac{x}{2}} \quad (11)$$

반면에, $u_{\boldsymbol{z}^*}$는 자유도가 m이며 중심매개변수

$$\lambda = N\sum_{j=1}^{m} c_j^2 = N\bar{c}^2 \quad (12)$$

를 갖는 비중심(noncentral) 카이제곱분포를 따른다. 이 때의 $u_{\boldsymbol{z}}$의 확률밀도함수를 $f_{\boldsymbol{z}^*}(x;m,\lambda)$라 하자. $f_{\boldsymbol{z}^*}(x;m,\lambda)$는 다음과 같은 형태를 갖는다.

$$f_{\boldsymbol{z}^*}(x;m,\lambda) = \frac{1}{2} e^{-\frac{x+\lambda}{2}} \left(\frac{x}{\lambda}\right)^{\frac{m}{4}-\frac{1}{2}} I_{\frac{m}{2}-1}(\sqrt{\lambda x}) \quad (13)$$

여기서 $I_a(y)$는 Bessel I 함수라 불리며 다음과 같은 형태를 갖는다.

$$I_a(y) = \left(\frac{y}{2}\right)^a \sum_{j=0}^{\infty} \frac{(y^2/4)^j}{j!\Gamma(a+j+1)} \quad (14)$$

이와 같은 통계학적 논리를 바탕으로 Matsui가 Algorithm 2에 대하여 계산한 것과 비슷한 논리로 Algorithm MK3의 성공확률을 계산해보자. $\boldsymbol{z}$의 비트길이를 k라 하자. 옳은 서브키 비트열에 대응되는 $u_{\boldsymbol{z}^*}$을 u_0로, 나머지 $u_{\boldsymbol{z}}$들을 $u_1, \cdots, u_{2^k-1}$로 재정렬 하도록 하자. Algorithm MK의 '성공'을 u_0이 가장 큰 값을 갖는 경우로 정의할 때, 성공확률을 다음과 같이 정리할 수 있다.

$$\Pr[u_0 > 0, u_0 > u_i \text{ for } 1 \le i \le 2^k - 1] = \int_0^{\infty} f_{\boldsymbol{z}^*}(x;m,\lambda) \prod_{\boldsymbol{z} \neq \boldsymbol{z}^*} \int_0^x f_{\boldsymbol{z}}(y;m)\, dy\, dx. \quad (15)$$

블록암호에 대한 공격을 구상할 때 찾고자하는 서브키 비트 수나 이용하는 선형근사식의 개수는 미리 결정되는 것이므로 성공확률에 영향을 미치는 유일한 변수는 평문-암호문의 개수 $N = \lambda\bar{c}^{-2}$ 또는 $\lambda = N\bar{c}^2$이다. 따라서 공격자는 시간복잡도와 성공확률을 감안하여 λ를 결정하고 이를 통하여 데이터 복잡도 즉, 평문-암호문 쌍의 개수를 결정할 수 있다.

이러한 성공확률 계산방식은 공격 알고리즘에 의해 작성되는 키후보 목록의 최상위권만을 고려하는 분할-정복 전략을 가정한 것이다. 그러나 분할-정복 전략만을 고집하는 것은 옳지 않다. 다중선형공격에서 많은 선형근사식을 이용하다보면 분할-정복의 목적을 달성할 수 있을 만큼 많은 평문-암호문 쌍을 이용하는 것이 전수조사 못지않은 부담을 안겨줄 수 있기 때문이다. 반대로, 이용하는 평문-암호문 쌍의 개수에 따라, 대상이 되는 서브키 비트들에 대한 몇 비트의 정보를 높은 확률로 얻어낼 수 있는지 예측할 수 있는 방법으로 알고리즘의 '성공'을 정의하고 성공확률을 계산해보자. N 개의 평문-암호문 쌍을 이용했을 때, 옳은 서브키 비트열 $\boldsymbol{z}^*$가 알고리즘에 의하여 작성되는 키 후보 목록

의 상위 2^{-a} 내에 있게 되는 사건을 매개변수 a에 관한 '성공'으로 정의하자. 이것은 [11]에서 제시되었던 Algorithm 2의 성공확률 계산법에 적용된 '성공'의 개념이기도 하다. 이 개념을 Algorithm MK3에 적용하여 성공확률을 계산해보고자 한다.

$u_{1,}\cdots,u_{2^k-1}$를 $u_1<\cdots<u_{2^k-1}$를 만족하도록 재배열시키자. $u_j(0<j<2^k)$들은 모두 자유도 m인 카이제곱분포를 따른다. 자유도 m인 카이제곱분포의 누적분포함수와 확률밀도함수를 각각 F와 f로 간단히 나타내도록 하자. [11]에 따르면 $q=1-2^{-a}$라 할 때, $u_j(0<j<2^k)$ 중 $\bar{r}=2^k-2^{k-a}=\lfloor(2^k-1)q\rfloor+1$번째 통계량 $u_{\bar{r}}$의 확률분포는 다음과 같은 평균 μ_q와 표준편차 σ_q를 갖는 정규분포에 근사된다.

$$\mu_q=F^{-1}(q),\ \sigma_q=\frac{1}{f(\mu_q)}2^{-\frac{k+a}{2}} \tag{16}$$

평문-암호문 쌍의 개수 N이 충분히 커지면, u_0의 분포인, 자유도 m과 비중심 매개변수 λ를 갖는 비중심 카이제곱분포는 중심극한정리에 의해 평균 $\mu_0=m+\lambda$, 분산 $\sigma_0^2=2(m+2\lambda)$를 갖는 정규분포에 근사된다. 그러므로, 옳은 서브키 비트열 z^*이 Algorithm MK3에 의해 작성되는 목록의 상위 2^{-a} 내에 있을 확률, 즉 새로운 개념의 성공확률은 다음과 같이 계산된다.

$$\begin{aligned}\Pr[u_0>u_{\bar{r}}]&=\Pr[u_0-u_{\bar{r}}>0]\\&=\int_{-\frac{\mu_0-\mu_q}{\sqrt{\sigma_0^2+\sigma_q^2}}}^{\infty}\phi(x)dx\end{aligned} \tag{17}$$

Ⅳ. Algorithm MK3에 대한 실험

4.1. 실험용 블록암호 Marmotte

우리는 실험용으로 설계된 블록암호 Marmotte에 Algorithm MK3을 적용하여보았다. Marmotte는 16비트의 블록 길이를 갖고, 전형적인 6 라운드 SPN 구조로 이루어져 있다. 블록암호 Marmotte는 16비트 평문 P를 다음과 같은 과정을 거쳐 16비트 암호문 C로 변환한다. 임의의 16비트 변수 A는 4비트 값을 성분으로 갖는 벡터로 고려할 수 있다고 하자: $A=(A_3,A_2,A_1,A_0)$.

[표 1] . 블록암호 Marmotte의 암호화 과정

입력	평문 P, 서브키 $w_0,\cdots,w_6$ (각 16비트)
출력	암호문 C (16비트)
1라운드	$x_0\leftarrow P$ $y_0\leftarrow(S(x_{0,3}\oplus w_{0,3}),S(x_{0,2}\oplus w_{0,2}),$ $S(x_{0,1}\oplus w_{0,1}),S(x_{0,0}\oplus w_{0,0}))$ $x_1\leftarrow(y_{0,2}\oplus y_{0,1}\oplus y_{0,0},y_{0,3}\oplus y_{0,1}\oplus y_{0,0},$ $y_{0,3}\oplus y_{0,2}\oplus y_{0,0},y_{0,2}\oplus y_{0,1}\oplus y_{0,0})$
2 ~ 5 라운드 $1\le j\le 4$	$y_j\leftarrow(S(x_{j,3}\oplus w_{j,3}),S(x_{j,2}\oplus w_{j,2}),$ $S(x_{j,1}\oplus w_{j,1}),S(x_{j,0}\oplus w_{j,0}))$ $x_{j+1}\leftarrow(y_{j,2}\oplus y_{j,1}\oplus y_{j,0},y_{j,3}\oplus y_{j,1}\oplus y_{j,0},$ $y_{j,3}\oplus y_{j,2}\oplus y_{j,0},y_{j,2}\oplus y_{j,1}\oplus y_{j,0})$
6라운드	$y_5\leftarrow(S(x_{5,3}\oplus w_{5,3}),S(x_{5,2}\oplus w_{5,2}),$ $S(x_{5,1}\oplus w_{5,1}),S(x_{5,0}\oplus w_{5,0}))$ $C\leftarrow(y_{5,3}\oplus w_{6,3},y_{5,2}\oplus w_{6,2},$ $y_{5,1}\oplus w_{6,1},y_{5,0}\oplus w_{6,0})$

$S(\cdot)$는 다음과 같은 4비트 입출력표를 갖는 S박스이다.

입력값	0	1	2	3	4	5	6	7
출력값	7	A	2	C	4	8	F	0
입력값	8	9	A	B	C	D	E	F
출력값	5	9	1	E	3	D	B	6

서브키 $w_{0,}\cdots,w_6$은 16비트 마스터키 K로부터 다음과 같은 높은 비선형성을 갖는 키 스케줄 알고리즘에 의해 생성된다. 블록암호 Marmotte의 암호화 과정의 평문 대신 K를 대입하고, 서브키 자리에는 모두 0을 대입하여 7번의 OFB연산을 수행한다. OFB 체인의 첫 번째, ... , 여섯 번째 결과값들을 각각 서브키 $w_0,\cdots,w_3$으로 정한다.

[표 2] . 블록암호 Marmotte에 대한 Algorithm MK3의 성공확률(%,회수)

a	$SR_{\lambda=2}$		$SR_{\lambda=4}$		$SR_{\lambda=8}$		$SR_{\lambda=16}$		$SR_{\lambda=32}$		$SR_{\lambda=64}$		$SR_{\lambda=128}$	
	이론	실험	이론	실험	이론	실험	이론	실험	이론	실험	이론	실험	이론	실험
1	55.0	53	58.6	61	65.3	66	76.9	80	91.3	97	99.2	100	100	100
2	28.9	31	32.3	35	39.3	48	53.7	61	77.4	84	97.0	96	100	100
3	14.6	9	17.0	18	22.5	22	35.3	47	62.1	74	93.1	94	100	99
4	7.1	12	8.7	9	12.4	12	22.1	32	47.7	66	87.5	91	99.9	100
5	2.7	3	3.5	7	5.5	10	11.6	20	32.2	53	78.4	86	99.7	100
6	1.6	4	2.1	1	3.4	8	7.9	19	25.2	48	72.5	82	99.5	97
7	0.7	2	1.0	5	1.7	5	4.5	11	17.5	36	63.9	74	99.1	100
8	0.3	0	0.5	1	0.9	1	2.5	10	11.9	29	55.3	69	98.5	95
9	0.1	2	0.2	1	0.4	1	1.4	3	7.9	23	46.8	63	97.7	94
10	0.07	0	0.1	0	0.2	0	0.8	3	5.2	13	39.0	58	96.5	94
11	0.03	0	0.05	1	0.1	0	0.5	1	3.5	12	32.1	57	94.9	90

4.2. 블록 암호 Marmotte에 대한 공격

우리가 Marmotte의 다중선형공격에 이용한 linear hull은 다음과 같이 라운드2의 입력 x_1과 암호문 C에 대하여 성립하는 형태를 갖는다.

$$\alpha \cdot x_{1,0} \oplus \beta \cdot C_0 = 0. \quad (18)$$

Marmotte에는 (18)과 같은 형태를 갖고 임밸런스가 0이 아닌 linear hull이 225개가 존재하며, 임밸런스의 제곱의 합 $\tilde{c}^2$은 약 $2^{-5.48}$이다. 이러한 linear hull들을 이용하는 Algorithm MK3의 타겟이 되는 것은 첫 번째 서브키 w_0의 상위 12비트 즉, $w_{0,3}, w_{0,2}, w_{0,1}$이다. 100개의 랜덤하게 선택된 키들에 대하여 다양하게 Algorithm MK3을 적용한 결과, 실제의 성공확률이 이론적인 것과 거의 일치 하였다. (표 2). 이것은 이론적인 성공확률을 이용하여 실제 성공확률을 매우 잘 예측할 수 있음을 의미한다. 또한, 목표가 되는 정보량(즉, a)에 따라 평문-암호문쌍의 개수를 조절함으로써 시간 복잡도를 감안한 공격이 가능하다.

V. 결 론

본 논문에서는 비선형 키스케줄을 갖는 블록암호에 Birukov의 다중선형공격법이 적용될 수 없는 이유를 구체적으로 밝히고, 비선형 키스케줄을 갖는 블록암호에 효과적으로 적용될 수 있는 새로운 다중선형공격법을 제시하였다. 실험용 16비트 블록암호 Marmotte에 대한 Algorithm MK3의 실험결과는 새로운 다중선형공격법에 관한 이론이 실제로도 잘 적용된다는 것을 보여준다. 이 새로운 공격법은 선형 키스케줄을 갖는 블록암호에도 적용될 수 있으며, 기존에 제시된 블록암호들에 대한 선형공격에 대한 안전성의 재평가에 필요한 도구로서 활용될 수 있을 것으로 기대된다.

참고문헌

[1] E. Biham, A. Shamir, *Differential Cryptanalysis of the Data Encryption Standard*, Springer-Verlag, 1993.

[2] A. Biryukov, C. De Cannière, M. Quisquater, "On Multiple Linear Approximations", *CRYPTO 2004*, LNCS 3152, pp. 1-22, Springer-Verlag, 2004.

[3] P. Junod, "On the Optimality of Linear, Differentia l, and Sequential Distinguishers", *EUROCRYPT 2003*, LNCS 2656, pp. 17-32, Springer-Verlag, 2003.

[4] P. Junod, S. Vaudenay, "Optimal Key Ranking Procedures in a Statistical Cryptanalysis", *FSE 2003*, LNCS 2887, pp. 235-246, Springer-Verlag, 2003.

[5] B. S. Kaliski, M. J. Robshaw, "Linear Cryptanaly sis Using Multiple Approximations", *CRYPTO '94*, LNCS 839, pp. 26-39, Springer-Verlag, 1994.

[6] L. R. Knudsen, J. E. Mathiassen, "A Chosen-

Plaintext Linear Attack on DES", *FSE 2000*, LNCS 1978, pp. 262-272, 2001.

[7] M. Matsui, "Linear Cryptanalysis Method for DES Cipher", *EUROCRYPT'93*, LNCS 765, pp. 386-397, Springer-Verlag, 1993.

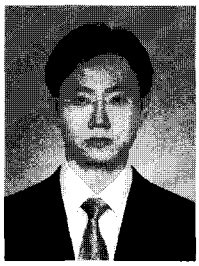
[8] M. Matsui, "The First Experimental Cryptanalysis of the Data Encryption Standard", *CRYPTO '94*, LNCS 839, pp. 1-11, Springer-Verlag, 1994.

[9] NIST, FIPS 46-3: Data Encryption Standard, 1999.

[10] NIST, FIPS 197: Advanced Encryption Standard, 2001.

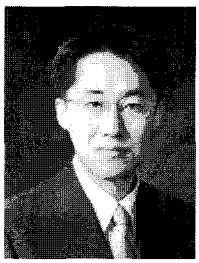
[11] A. A. Selcuk, A. Bicak, "On Probability of Success in Linear and Differential Crypt analysis", SCN'02, LNCS 2576, pp. 177-188, Springer-Verlag, 2002.

〈著者紹介〉



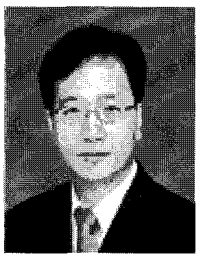
홍 득 조 (Deukjo Hong) 정회원

1999년 8월 : 고려대학교 수학과 졸업
2001년 8월 : 고려대학교 수학과 석사
2006년 2월 : 고려대학교 정보보호대학원 박사
2006년 3월~2006년 8월 : 고려대학교 정보보호기술연구센터 박사후 연구원
2006년 9월~현재 : 고려대학교 정보보호기술연구센터 연구조교수
<관심분야> 대칭키 암호의 분석 및 설계



성 재 철 (Jaechul Sung) 정회원

1997년 8월 : 고려대학교 수학과 학사
1999년 8월 : 고려대학교 수학과 석사
2002년 8월 : 고려대학교 수학과 박사
2002년 7월 ~ 2004년 1월 : 한국정보보호진흥원 선임연구원
2004년 2월 ~ 현재 : 서울시립대학교 수학과 조교수
<관심분야> 대칭키 암호의 분석 및 설계



홍 석 희 (Seok-Hie Hong) 정회원

1995년 2월 : 고려대학교 수학과 학사
1997년 2월 : 고려대학교 수학과 석사
2001년 2월 : 고려대학교 수학과 박사
1999년 8월 ~ 2004년 2월 : (주) 시큐리티 테크놀로지스 선임연구원
2003년 2월 ~ 2004년 2월 : 고려대학교 정보보호기술연구센터 선임 연구원
2004년 4월 ~ 2005년 2월 : K.U.Leuven 박사후연구원
2005년 3월 ~ 현재 : 고려대학교 정보경영공학전문대학원 조교수
<관심분야> 대칭키 암호의 분석 및 설계, 컴퓨터 포렌식

이 상 진 (Sangjin Lee) 정회원

1987년 2월 : 고려대학교 수학과 학사
1989년 2월 : 고려대학교 수학과 석사
1994년 8월 : 고려대학교 수학과 박사
1989년 10월 ~ 1999년 2월 : 한국전자통신연구원 선임연구원
2001년 3월 ~ 현재 : 한국정보보호학회 편집위원 및 감사
2006년 3월 ~ 현재 : 한국정보보호학회 암호연구회 회장
1999년 3월 ~ 현재 : 고려대학교 정보경영공학전문대학원 교수
<관심분야> 대칭키 암호의 분석 및 설계, 컴퓨터 포렌식