$GF(2^n)$에서 안전한 S-box의 구성과 효율적인 구현방법

박 난경†, 이 필중‡

†포항공대 정보통신연구소, ‡포항공대 전자전기공학과

요 약

블록암호의 설계에서 S-box는 가장 중요한 요소이다. S-box의 크기는 기존공격에 대한 안전도와 수행 시 필요한 메모리량, 수행속도가 동시에 고려되어 선택되어야 한다. 일반적으로 S-box의 입출력의 크기가 커지면 안전도와 메모리 소요량은 크게 증가하며 수행속도는 저하된다. 한편, $GF(2^n)$에서의 멱함수(power permutation)는 DC, LC에 강하므로 여러 암호에 적용되었으나 최근 고계차분공격(higher order differential attack)과 보간공격(interpolation attack)에 의해 공격된 바 있다. 본문에서는 DC, LC, 고계차분공격, 보간공격에 안전한 S-box로서 $GF(2^n)$에서의 멱함수인 $x^{-2^k}(k<n)$를 구성하고, n이 짝수일 때 메모리 소요량이 보다 적은 구현방법으로서 $GF(2^{n/2})$의 연산을 이용하는 방법을 제시한다.

제 1 절 서론

블록암호에 사용되는 S-box의 크기가 커지면서 기존의 난수적인 생성방법외에 구조적인 S-box를 생성하는 사례로서 $GF(2^n)$에서 정의된 멱함수에 대한 성질이 고찰되었다[20, 21, 22, 23, 6, 2, 7, 9]. $GF(2^n)$위에서의 멱함수는 입출력길이가 같은 S-box의 구성에 유용하며 차분공격(differential attack[8], DC), 선형공격(linear attack[16], LC)에 대해 안전하다고 알려져 있어 여러가지 암호의 설계에 적용되었다. SHARK[26], SQUARE[10], SNAKE[15], RIJDAEL[27] 등에는 $GF(2^8)$의 연산이 사용되었고 $\mathcal{KN}$ 암호와 $\mathcal{PURE}$ 암호에는 $GF(2^{33})$에서의 x^3과 $GF(2^{32})$에서의 x^3 연산이 각각 사용되었다. 특히, $\mathcal{KN}$암호와 $\mathcal{PURE}$암호는 DC, LC에 대해 증명가능한 안전도(provable security)를 가지는 prototype으로서 제시되었다[19].

그러나 이 중 $\mathcal{KN}$ 암호와 SNAKE는 DC개념을 확장한 고계차분공격(higher order differential attck)에 약하며 SHARK, SNAKE는 암호문과 평문의 관계를 보간(interpolation)하여 키를 구하고자 하는 보간공격(interpolation attack)에 취약하다고 알려졌다[12, 28]. 따라서 $GF(2^n)$에서의 멱함수로 S-box를 설계하려면 고계차분공격과 보간 공격에 대한 특성이 반드시 고려되어야 한다.

한편, $GF(2^n)$에서의 연산을 사용한 S-box의 안전도는 n의 크기와 밀접한 관련이 있다. DC, LC에 대한 안전도는 n이 커짐에 따라 지수적으로 증가한다. 그 외 고계차분공격이나 보간공격에 대한 안전도도 n이 커짐에 따라 증가한다. 더 안전한 암호를 설계하기 위해서는 큰 n을 선택해야 한다. 그러나 일반적으로 n이 커지면 S-box의 구현에 필요한 메모리의 소요량도 급격히 증가하며 계산효율은 저하된다. 따라서 S-box는 안전도, 메모리량, 계산효율간의 균형관계(trade-off)를 고려하여 선택되어야 한다.

만약 $n=16, 32$와 같이 큰 유한체에서 정의되는 멱함수를 더 적은 크기의 메모리로 빠르게 구현할 수 있다면 더욱 안전한 S-box를 사용할 수 있다. 또한 각 라운드에 대한 안전도가 크게 증가하면 암호의 라운드 수가 감소하게 되어 동일한 암호의 수행효율에 비해 상대적으로 더 높은 암호강도를 확보할 수 있게 될 것이다. 그러면 암호의 안전도와 메모리량, 효율성간의 균형관계(trade-off)도 개선될 수 있다.

이에 본문에서는 DC, LC를 비롯하여 고계차분공격, 보간공격에 강한 S-box로서 $GF(2^n)$에서의 $x^{-2^k}(k<n)$를 제시하고 n이 큰(가령, $n=16,\cdots,32$) $GF(2^n)$에서의 S-box를 메모리의 소요량과 구현속도면에서 보다 효과적으로 구현하는 방법을 제시한다. 본문의 구성은 다음과 같다. 제 2절에서는 기존 공격방법에 강하기 위해 S-box가 가져야 할 특성에 대해 기술하고 $GF(2^n)$의 구현방법에 관해 소개한다. 제 3절에서는 널리 쓰이는 멱함수와 암호학적인 성질을 정리하고 보다 안전한 S-box로서 $GF(2^n)$에서의 $x^{-2^k}(k<n)$의 특성을 고찰한다. 제 4 절에서는 $GF(2^n)$(n:짝수)에서의 x^{-2^k}을 subfield인 $GF(2^{n/2})$에서의 연산을 이용한 구현방법을 설명한다.

제 2 절 Preliminary

1 기존 공격방법에 대한 고찰

DC는 주어진 암호에서 입력비트에 대한 변화와 출력비트에 대한 변화의 특성을 이용하여 키를 구하는 선택평문공격이다. 반면에 LC는 입력의 특정 비트와 출력의 특정비트간의 관계를 키의 관련비트들로 선형근사시켜 키의 몇비트를 구하는 기지평문 공격이다. DC와 LC에 대한 강도는 각 라운드의 암호함수의 차분확률(differential probability)과 선형확률(linear probability)에 기초한다[1].

정의 1(differential probability와 linear probability) $F:GF(2^n)\rightarrow GF(2^n)$에 대해 F의 차분확률 DP와 선형확률 LP는 다음과 같이 정의된다.

$$\begin{aligned} DP^F(\Delta x\rightarrow\Delta y) &= Pr_x\{F(x)\oplus F(x\oplus\Delta x)=\Delta y\} \\ LP^F(\Gamma y\rightarrow\Gamma x) &= |2\cdot Pr_x\{x\cdot\Gamma x=F(x)\cdot\Gamma y\}-1|^2 \end{aligned}$$

또한 F에 대한 DP와 LP의 최대값인 DP^F_{max}, LP^F_{max}를 다음과 같이 정의한다.

$$\begin{aligned} DP^F_{max} &= \max_{\Delta x(\neq 0),\Delta y} DP^F(\Delta x\rightarrow\Delta y) \\ LP^F_{max} &= \max_{\Gamma x,\Gamma y(\neq 0)} LP^F(\Gamma y\rightarrow\Gamma x) \end{aligned}$$

$DP^F, LP^F, DP^F_{max}, LP^F_{max}$ 사용 시 혼동할 우려가 없으면 F를 생략하기로 한다.

고계차분공격은 DC를 변형한 공격방법으로 고계차 도함수(higher order derivative)에 기초한 선택평문 공격이다[14, 13]. 즉, 암호함수의 차수가 낮으면 암호전체를 암호함수를 중첩한 형태의 다항식으로 표현할 수 있다. 이 때 표현된 다항식에 대한 higher order differential이 0 또는 상수가 되도록 하여 키를 구할 수 있다. 그러므로 고계차분공격에 대해 강하려면 사용되는 암호함수가 가능한 높은 차수를 가지도록 설계되어야 한다.

$F:GF(2^n)\rightarrow GF(2^n)$를 n-비트 입력 $x=(x_{n-1},\cdots,x_0)\in GF(2^n)$에 대해 n-비트 출력 $y=(y_{n-1},\cdots,y_0)\in GF(2^n)$을 내는 함수라 하자. 출력의 각 비트는 $y_i=F_i(x)$와 같이 $F_i:GF(2^n)\rightarrow GF(2)$인 부울함수로 나타낼 수 있다. 이 때 F_i를 좌표함수(coordinate function)라 한다.

정의 2(nonlinear degree of F) i번째 좌표함수 F_i의 X에 대한 차수를 $deg_X F_i$라 하면 X에 대한 F의 차수는 다음과 같이 정의된다.

$$deg_X F=\min_{0\le i\le n-1}(deg_X F_i)$$

[1] 실제로 DC와 LC에 대한 암호의 강도는 모든 라운드에 대해 구성된 특성(iterated characteristic이나 differential)과 effective linear expression의 확률로 나타내지며 각 확률값은 차분확률과 선형확률을 바탕으로 계산된다. 암호함수가 전단사(permutation)인 Feistel 구조의 암호에서 암호함수의 DP_{max}와 LP_{max}가 p이고 라운드 수가 3이상이면, 전체 암호의 DP_{max}와 LP_{max}가 p^2이하임이 증명되어 있다[17].

보간 공격은 고정된 키에 대한 암호문과 평문의 관계를 다항식이나 유리식으로 보간하여 암호를 해독하는 방법이다[12]. 주어진 키에 대한 보간식의 계수는 키에 의존하는 미지수가 된다. 만약 계수의 개수가 n이면 n개의 평문과 암호문쌍으로 모든 계수를 구할 수 있으므로 주어진 키에 대해 암호문과 평문의 관계를 보간식으로 나타낼 수 있다. 그러므로 보간식을 구하기 어렵게 하기 위해서는 미지수의 개수가 선택가능한 평문과 암호문의 개수보다 많도록 설계되어야 한다.

정의 3($deg_X^{(pol)}F$, $deg_X^{(rat)}F$) F를 X에 대한 다항식으로 나타낸 경우의 차수를 $deg_X^{(pol)}F$라 하고, 유리식으로 나타냈을 때 분모의 차수와 분자의 차수의 합을 $deg_X^{(rat)}F$라고 한다.

이 때, $deg_X^{(pol)}F$과 $deg_X^{(rat)}F$이 클수록 찾아야 할 계수의 개수가 많아지므로 보간공격에 대해 더 안전하다고 할 수 있다.

2 $GF(2^n)$에서의 연산

$GF(2^n)$에서 정의되는 연산을 수행하는 알고리즘에는 다항식 기저(polynomail basis)나 정규 기저(normal basis)를 사용하거나 좌표함수를 사용하는 방법, 테이블 검색(table-lookup), n이 소수가 아닌 경우 subfield를 확장하는 방법(successive extention) 등과 여러 알고리즘을 조합하는 방법이 있다.

1. 다항식 기저: n차의 기약다항식(irreducible polynomial) $f(x)$에 대해, $GF(2^n)$과 $GF(2)/(f(x))$은 isomorphic하므로 $GF(2^n)$의 원소를 차수가 n보다 작은 다항식으로 표현할 수 있으며 $\{x^{n-1}, x^{n-2}, \cdots, x, 1\}$를 다항식기저라 한다. $GF(2^n)$의 원소 $A = (a_{n-1}, ..., a_0)$는 $a_{n-1}x^{n-1} + \cdots + a_0$와 같이 표현된다. 두 원소간의 덧셈은 비트별 exclusive OR에 의해 계산되고 A, B의 곱셈은 $AB \bmod f(x)$이며 역원은 extended Euclidean algorithm에 의해 계산된다. $GF(2^n)$에서의 k승은 $A^k \bmod f(x)$로 구해진다.

2. 정규 기저: $GF(2^n)$상에서는 $\{\beta, \beta^2, \beta^{2^2}, \cdots, \beta^{2^{n-1}}\}$가 기저(basis)가 되도록 하는 원소 β가 항상 존재하며 이 기저를 정규기저라 한다. 정규기저를 사용하면 회전이동(cyclic shift)만으로 제곱을 계산할 수가 있으나 곱셈과 역원의 계산은 다항식기저를 사용하는 것보다 더 복잡하다 . 정규기저에서 보다 효과적인 곱셈과 역원의 계산방법으로 [24, 25]의 알고리즘이 제안되었다.

   기저를 사용하여 $GF(2^n)$의 연산을 하는 경우 n이 커짐에 따라 연산의 수행속도가 매우 느려지므로 n이 작을 때 더 효과적이다.

3. 테이블 검색: 함수 $GF(2^n)$에서 정의된 함수 F가 주어지면 모든 x에 대한 (a). $F(x)$를 저장하거나, (b). 역원을 저장하거나[18], (c). 지수-로그(exp-log)값을 저장하여[29] F를 구한다. F가 고정되면 (a)가 가장 효과적이며 (b)와 (c)는 곱셈, 멱함수, 역원 등의 연산을 수행할 수 있으므로 보다 광범위한 용도로 쓰일 수 있다.

   일반적으로 속도면에서 테이블 검색이 가장 효과적이지만 n이 커지면 테이블의 크기가 지수적으로 증가한다. 그러므로 n이 크면 자원이 제한적인 스마트카드 등에서 사용하기가 어렵다.

4. 좌표함수 사용: $GF(2^n)$의 함수 F를 $(F_{n-1}, \cdots, F_1)$과 같이 좌표함수로 나타내서 각 F_i를 계산하여 F를 구하는 것이다. $deg_X F$가 높거나 n이 커지면 F_i를 찾기가 어려워지나 n이 큰 경우에 테이블 검색방법과 적절히 병용하면 효과적일 수 있다.

5. subfield 확장(successive extention): 메모리 소요량은 줄이되 연산속도는 되도록 빠르게 하기 위한 방법이라고 할 수 있다. $n = n_1 \times n_2$이고 n_1과 n_2가 서로 소이면 $GF(2)$상에서 차수가 n_2인 기약다항식이 $GF(2^{n_1})$에서도 차수가 n_2인 기약다항식이 된다. 이 사실을 이용하면 $GF(2^n)$은 $GF(2)$를 n_1차원 확장하여 $GF(2^{n_1})$을 만든 후 다시 n_2 차원만큼 확장하여 $GF((2^{n_1})^{n_2})$를 만들 수 있다. 이와 같이 $GF(2^n)$을 subfield인 $GF(2^{n_1})$의 원소로 표현하면 다항식 기저와 테

F	$deg_X F$	DP_{max}	LP_{max}	조건
x^{2^k+1}	2	2^{s-n}	N/A	$s=\gcd(k,n)$
			2^{s-n}	$s=\gcd(k,n)$, n/s: 홀수
$x^l=(x^{2^k+1})^{-1}$	$\frac{n+1}{2}$	2^{1-n}	2^{1-n}	$1=\gcd(k,n)$, n : 홀수, $l=\sum_{i=0}^{\frac{n-1}{2}} 2^{ik} \bmod 2^n-1$
x^{-1}	$n-1$	2^{1-n}	2^{1-n}	n:홀수
		2^{2-n}	2^{2-n}	n:짝수

표 1: $GF(2^n)$에서의 멱함수 예제들

이블 검색방법을 병용하여 각각의 단점을 보완할 수 있다[11, 29]. 즉, 차수가 n_2인 다항식 기저를 사용하고 $GF(2^{n_1})$에서의 연산을 $GF(2^{n_1})$에 대한 역원이나 지수-로그 테이블을 검색하여 계산하면 n대신 n_1만큼의 테이블을 저장하므로 메모리 소요량이 감소하며 기저사용시 차수가 n_2로 낮아지므로 구현속도가 향상된다.

반면에 n_1과 n_2가 서로소가 아닌 경우에는 $GF(2^{n_1})$상에서 차수가 n_2인 기약다항식을 찾기가 쉽지 않다. 그러나, Aoki 등은 $GF(2^n)$상에서 차수가 2인 기약다항식을 구성하는 방법을 제시하고 구성된 2차 기약다항식과 $GF(2^n)$의 연산을 이용하여 $GF(2^{2n})$의 연산을 수행하는 알고리즘과 그에 따른 복잡도를 계산하였다[1].

일반적으로, $GF(2)$에서의 기저를 사용하는 경우에, n이 커짐에 따라, 연산의 수행속도가 매우 느려지며 테이블 검색방법을 사용하는 경우에는, n에 따라, 메모리 소요량이 급격히 증가한다. 그러므로 subfield를 확장하는 방법은 상기의 단점을 보완할 수 있다. 또한 저장용량이 제한된 스마트카드 등에서 S-box를 구현할 때 더욱 효과적이다. 참고로, [5]에서는 세가지 방법((a). $GF(2)$에 대한 다항식기저, (b). 지수-로그 테이블 검색, (c). $n=n_1\times n_2$, $\gcd(n_1,n_2)=1$인 경우 $GF(2^{n_1})$의 지수-로그 테이블 이용)으로 $GF(2^n)$에서 x^3 연산에 가장 효과적인 방법 및 속도를 $n=3,\cdots,65$인 모든 n에 대해 실험하였다. [5]에 의하면 $n\leq 19$이면 (b)가 가장 효과적이며 $n>19$이면 $n=n_1\times n_2$, $\gcd(n_1,n_2)=1$인 n_1,n_2가 있을 경우엔 (c)가 효과적이다.

제 3 절 $GF(2^n)$에서 $x^{-2^k}(0\leq k\leq n-1)$의 안전성 고찰

이절에서는 지금까지 안전하다고 알려져 있는 $GF(2^n)$에서 정의된 멱함수의 예제와 각각의 암호학적인 특성을 정리하고 이를 이용하여 x^{-2^k}에 대한 안전성을 살펴보기로 한다.

1 $GF(2^n)$위에서 멱함수의 예제

표 1에서는 $GF(2^n)$에서의 대표적인 멱함수와 각각에 대한 암호학적 성질을 정리하였다.

n이 홀수일 때 더 좋은 DC, LC 특성을 가지며 x^{-1}인 경우에 최대 차수인 $deg_X F=n-1$를 가진다.

한편, x^3이 암호함수로 사용된 KN 암호의 경우 $deg_X x^3=2$임을 이용하여 고계차분공격이 가능하였고 최대차수를 가지는 x^{-1}가 적용된 SNAKE와 SHARK의 경우에는 $deg_X^{(rat)} x^{-1}$가 낮기 때문에 유리식에 의한 보간공격이 가능했다. 이에 x^{-1}와 같이 최대 차수를 가지며 보간공격에 대한 취약점을 보완하는 멱함수의 일례로서 x^{-2^k}를 살펴보기로 한다.

2 $GF(2^n)$에서의 $x^{-2^k}(0\leq k\leq n-1)$의 안전성 고찰

$F: GF(2^n)\rightarrow GF(2^n)(0\leq k\leq n-1)$를 다음과 같이 정의하자.

$$F(x) = \begin{cases} x^{-2^k} & \text{if } x \neq 0, \\ 0 & \text{if } x = 0. \end{cases}$$

보조정리 1 $GF(2^n)$에서의 $F(x) = x^{-2^k}(k = 0, 1, ..., n-1)$는 가역적인(invertible) 선형행렬 A에 대해 $F(x) = A \circ x^{-1}$과 같이 나타낼 수 있다. 단, $\circ$는 함수의 합성을 나타낸다.

증명 $GF(2^n)$의 원소 x, y를 $GF(2)$상의 정규기저를 사용하여, $x = (x_{n-1}, x_{n-2}, ..., x_0), x_j \in GF(2), y = (y_{n-1}, y_{n-2}, ..., y_0), y_j \in GF(2)$로 나타내자. 그러면 $GF(2^n)$상에서 $y = x^2$은 다음과 같은 행렬 B에 대해 $y = Bx$로 표현될 수 있다.

$$B = \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & & 0 \\ \vdots & & \ddots & & \\ 0 & & & & 1 \\ 1 & 0 & 0 & \dots & 0 \end{pmatrix}$$

이 때 B는 정규기저 사용시 제곱연산을 위한 선형변환이라고 할 수 있다. 한편 $GF(2^n)$은 $GF(2)$의 벡터공간이고 B의 rank가 n이므로 기저가 다른 경우라도 기저변화에 따른 선형변환을 B'라 하면 B'의 rank는 역시 n이 된다.

그러므로 임의의 기저에 대한 선형변환 B'를 사용하면,

$$x^{-2^k} = (x^{-1})^{2^k} = \underbrace{B' \cdots B'}_{k-times} \circ x^{-1}$$

이 된다.

한편, $2^n - 2^k - 1(k = 0, 1, \cdots, n-1)$과 $2^n - 1$이 서로 소이므로 x^{-2^k}는 전단사(permutation)이다. 그러므로 라운드 함수 F에 대한 DP_{max}와 LP_{max}로서 암호의 DC와 LC에 대한 강도를 비교해 볼 수 있다.

Nyberg는 암호학적으로 좋은 성질을 가지는 S-box를 여러가지 대수적 방법으로 조합하거나 변형하여 새로운 S-box를 구성하고 DC와 LC에 대한 성질을 고찰하였다[23]. 다음 정리는 주어진 함수의 입출력값을 선형변환에 의해 변환시킨 후의 DP와 LP에 대한 결과이다.

정리 2 [23] $f : GF(2^n) \to GF(2^m)$라 하고 A, B를 각각 $A : GF(2^m) \to GF(2^m), B : GF(2^n) \to GF(2^n)$인 선형(또는 affine) 변환이라 하자. 그러면 다음이 성립한다.

$$\begin{aligned} DP^{(A \circ f \circ B)} &= DP^f, \\ LP^{(A \circ f \circ B)} &= LP^f. \end{aligned}$$

그러면 보조정리 1과 정리 2에 의해 다음을 구할 수 있다.

따름정리 3 $F(x) = x^{-2^k}$(단, $0 \leq k \leq n-1$)에 대해, n이 홀수이면 $DP_{max} = LP_{max} = 2^{1-n}$이고, n이 짝수이면 $DP_{max} = LP_{max} = 2^{2-n}$이다.

$w_2(k)$를 양의 정수 k의 Hamming weight라 하자. 다음 정리는 $GF(2^n)$에서 정의되는 임의의 멱함수의 차수(nonlinear degree)에 관한 결과로서 Carlet에 의해 증명되었다.

정리 4 [3] $F : GF(2^n) \to GF(2^n)$이고 $x \longmapsto x^e$(e : 정수)가 $GF(2^n)$의 전단사(permutation)이면 다음이 성립한다.

$$deg_X F = w_2(e).$$

정리 4에 의해, $x^{-2^k} = x^{2^n-2^k-1}$이므로 $deg_X x^{-2^k}$는 $GF(2^n)$상에서 최대 차수인 $n-1$을 가진다. 그러므로 고계차분공격에 보다 안전하기 위해서는 x^{-2^k}과 되도록 큰 n을 사용하면 된다.

또한 $deg_X^{(pol)} F = 2^n - 2^k - 1, deg_X^{(rat)} F = 2^k$이므로 k가 $n-1$에 가까운 값일 수록 보간공격에 대해 더 안전하다. 가령 $n = 8, k = 7$이고 x^{-2^7}이 암호함수이면 $deg_X^{(pol)} F = 127, deg_X^{(rat)} F = 128$이므로 1라운드만을 다항식이나 유리식에 의해 보간하는 경우에 모두 127개이상의 평문과 암호문쌍이 필요하므로 보간공격에 대해 더 큰 k값을 사용하는 것이 효과적이다. 결론적으로 n과 k가 클수록 $GF(2^n)$에서의 $x^{-2^k}(k = 0, \ldots, n-1)$는 DC, LC, 고계차분공격, 보간공격에 대해 더 안전하다. 구현의 편의상 $n = 8, 16, 32$와 같은 2의 승수를 사용하고 위의 멱함수를 복합적으로 사용(가령, $x^{-2^{n-1}}, x^{-2}, x^{-1}$를 같이 사용)하는 것도 x^{-1}을 반복적으로 쓰는 것보다 암호의 안전도를 개선시킬 수 있다. 참고로, [4]에서는 $GF(2^8)$에서의 모든 멱함수(x^k, $k = 1, \cdots, 255$)에 대해 전단사성질과 DC, LC에 대한 성질, 최대차수에 대해 실험적으로 조사하였다.또한 DC, LC, 고계차분공격과 보간공격에 강한 S-box로서 다음 함수를 제시하였다.

$$F(x) = \sum_{k_i \subset C} a_i x^{k_i},\ a_i \in GF(2),\ x \in GF(2^8)$$

단, $C = \{127, 191, 223, 239, 247, 251, 253, 254\}$이고 $\sum a_i$ = 홀수이어야 한다. 그 예로 SNAKE에서 x^{-1}대신 $x^{-1} + x^{-2} + x^{-4}$를 사용한다면 보간공격에 충분히 강해짐을 보였다.

제 4 절 $GF(2^{2n})$에서 정의된 $x^{-2^k}(k < 2n)$의 구현

이 절에서는 [1]에서 구성한 2차 기약다항식(irreducible polynomial)을 이용하여 $GF(2^{2n})$에서의 x^{-2^k}를 $GF(2^n)$의 연산을 이용하여 구현하는 방법을 소개하고자 한다.

1 표수가 2인 유한체의 2차 확장체(Quadratic Extension) 구성

보조정리 5 [1] X에 대한 2차 다항식 $X^2 + X + z, z \in GF(2^n)$이 $GF(2^n)$상에서 기약다항식라고 하자. $X^2 + X + z$의 한 근을 ω라 하면, $X^2 + X + z\omega$는 $GF(2^{2n}) = GF(2^n)(\omega)$ 상에서 기약다항식이다.

위 정리에 의해, $X^2 + X + a = 0, (a \in GF(2^n))$의 한 근 α에 대해, $GF(2^{2n}) = GF(2^n)(\alpha)$가 된다. $(\alpha+1)^2 + (\alpha+1) + a = \alpha^2 + \alpha + a = 0$이므로 $\alpha+1$이 위 2차식에 대한 α의 켤레근이 된다. 한편, $((\alpha+1)^2 + (\alpha+1) + a)^{2^n} = (\alpha^{2^n})^2 + \alpha^{2^n} + a = 0$이므로 α^{2^n}은 α이거나 $\alpha+1$이 된다. $\alpha^{2^n} = \alpha$이면, $\alpha \in GF(2^n)$ 이므로, $\alpha^{2^n} = \alpha + 1$이 된다. 즉, α가 정규기저의 generator이면, $[\alpha\ \alpha^{2^n}] = [\alpha\ \alpha+1]$가 된다.

2 다항식기저에 의한 x^{-2^k}의 계산

이 절에서는 다항식기저 $[1\ \alpha]$를 사용하여 x^{-2^k}를 계산하고자 한다. $x \in GF(2^{2n})$이면 x는 $x_1 + x_2\alpha$ $(x_1, x_2 \in GF(2^n))$와 같이 표현될 수 있다. $\alpha + 1$이 α의 켤레근이므로

$$(x_1 + x_2\alpha)(x_1 + x_2(\alpha + 1)) = x_1(x_1 + x_2) + ax_2^2$$

이 된다. 그러므로 x의 역원은 다음과 같다.

$$(x_1 + x_2\alpha)^{-1} = (x_1(x_1 + x_2) + ax_2^2)^{-1}(x_1 + x_2(\alpha + 1)). \quad (1)$$

한편, 표수가 2이므로

$$x^{2^k} = (x_1 + x_2\alpha)^{2^k} = x_1^{2^k} + x_2^{2^k}\alpha^{2^k}.$$

여기서 $s = a + a^2 + \cdots + a^{2^k-1}$라 놓으면 $\alpha^{2^k} = \alpha + s$이므로 다음과 같이 계산된다.

$$x^{2^k} = x_1^{2^k} + x_2^{2^k}(\alpha + s).$$

식 (1)에 의해 x^{-2^k}는 다음과 같이 표현된다.

$$(x_1 + x_2\alpha)^{-2^k} = A^{-1}(B + C\alpha). \tag{2}$$

단,

$$\begin{aligned} A &= (x_1^{2^k+1} + x_2^{2^k}(x_1^{2^k} + a^{2^k}x_2^{2^k})), \\ B &= x_1^{2^k} + (s+1)x_2^{2^k}, \quad C = x_2^{2^k}. \end{aligned}$$

3 정규기저에 의한 x^{-2^k}의 계산

이 절에서는 정규기저 $[\alpha\ \alpha+1]$를 사용하여 x^{-2^k}를 계산하고자 한다. $x \in GF(2^{2n})$이면 x는 $x_1\alpha + x_2(\alpha+1)(x_1, x_2 \in GF(2^n))$와 같이 표현될 수 있다. norm의 성질에 의해

$$(x_1\alpha + x_2(\alpha+1))(x_2\alpha + x_1(\alpha+1)) = a(x_1 + x_2)^2 + x_1x_2$$

이 된다. 그러므로 x의 역원은 다음과 같이 구해진다.

$$(x_1\alpha + x_2(\alpha+1))^{-1} = (a(x_1 + x_2)^2 + x_1x_2)^{-1}(x_2\alpha + x_1(\alpha+1)) \tag{3}$$

한편, 표수가 2이므로

$$\begin{aligned} x^{2^k} &= (x_1\alpha + x_2(\alpha+1))^{2^k} \\ &= (x_1^{2^k} + x_2^{2^k})\alpha^{2^k} + x_2^{2^k}. \end{aligned}$$

여기서 $s = a + a^2 + \cdots + a^{2^k-1}$라 놓으면 $\alpha^{2^k} = \alpha + s$이므로

$$x^{2^k} = (x_1^{2^k} + s(x_1^{2^k} + x_2^{2^k})\alpha + (x_2^{2^k} + s(x_1^{2^k} + x_2^{2^k}))(\alpha+1).$$

이다. 그러므로 식 (3)에 의해 x^{-2^k}는 다음과 같이 계산된다.

$$(x_1\alpha + x_2(\alpha+1))^{-2^k} = A^{-1}(B\alpha + C(\alpha+1)). \tag{4}$$

단,

$$\begin{aligned} A &= a^{2^k}(x_1^{2^k} + x_2^{2^k})^2 + x_1^{2^k}x_2^{2^k}, \\ B &= s(x_1^{2^k} + x_2^{2^k}) + x_2^{2^k}, \quad C = s(x_1^{2^k} + x_2^{2^k}) + x_1^{2^k}. \end{aligned}$$

4 $GF(2^{2n})$에서 $x^{-2^k}(k < 2n)$의 구현에 관하여

a^{2^k}, s, $s+1$를 사전계산한다고 하면, $GF(2^{2n})$에서 x^{-2^k}을 계산하기 위해서 필요한 연산의 수를 기저의 종류에 따라 비교하면 표 2와 같다. 괄호안의 값은 $GF(2^{2n})$에서의 x^{-1}에 필요한 $GF(2^n)$의 연산의 개수이다.

기저	2^k승	제곱	곱셈	역원	덧셈
다항식기저	3(0)	1(1)	5(4)	1(1)	3(2)
정규기저	3(0)	1(1)	5(4)	1(1)	4(2)

표 2: $GF(2^{2n})$에서의 $x^{-2^k}(x^{-1})$의 계산 시 필요한 $GF(2^n)$ 연산 수 비교

$GF(2^n)$의 원소를 지수-로그 테이블로 저장한다면 x^{-2^k}를 수행하기 위해서는 대략 25회[2]의 테이블 검색이 필요하다. 반면에 x^{-1}를 수행하기 위해서는 약 16회의 테이블 검색이 필요하다. 한 번의 S-box연산을 위해 소요되는 검색 횟수로서는 다소 많지만 n이 충분히 크면 라운드 함수의 안전도가 매우 높아지므로 전체 암호의 라운드수를 줄일 수 있다. 결국 전체 암호수행 시 필요한 S-box 연산의 회수가 감소되며 테이블의 크기가 작으면 테이블검색에 소요되는 단위시간이 줄어들게 되므로 암호의 효율을 감소시키지 않을 것이다. 또한 스마트카드와 같이, 안전한 암호가 사용되어야 하나 저장용량이 제한된 경우에 제안된 방법이 효과적으로 적용될 수 있다. 가령, 입출력길이 $n=8$인 $GF(2^8)$에서의 x^k를 예로 들어보자. 이 연산을 테이블로 바로 저장하려면 256~512바이트가 필요하다. 반면에 제시된 구현방법을 사용한다면 $GF(2^4)$에 대한 지수-로그값을 저장하기 때문에 16바이트가 필요하다. 그러므로 SQUARE과 같은 기존의 블럭암호를 저장용량이 극히 제한된 스마트카드 등에서도 쉽게 구현할 수가 있다.

또한 보다 안전한 시스템을 설계하기 위해 $n=32$와 같이 입출력 길이가 큰 S-box를 사용하는 경우에 $GF(2^{32})$에 대한 테이블 검색을 사용하여 S-box를 구현하면 $2^{32}\times 32\times 2=32$Gbyte가 필요하므로 일반적인 시스템에서는 사용이 불가능하다. 그러나 $32=2\times 16$이므로 제시된 구현방법을 사용하면 $2^{16}\times 16\times 2=256$Kbyte가 필요하므로 Pentium정도의 PC에서도 쉽게 사용할 수 있다.

제 5 절 결론

이상으로 DC, LC, 고계차분공격, 보간공격에 대해 안전한 S-box로서 $GF(2^n)$에서의 $x^{-2^k}(k<n)$을 구성하고 암호학적인 성질을 고찰하였다. 또한 제안된 S-box의 구현을 위해 $GF(2^n)$에서의 연산에 대해 개괄적으로 살펴보았고 n이 짝수일 때 subfield인 $GF(2^{n/2})$의 연산을 이용한 구현방법을 소개하였다.

일반적으로, n이 클 수록 S-box의 안전도는 우수하지만 구현상의 효율은 크게 저하한다. 가령, $GF(2^n)$의 연산을 수행하기 위해 $GF(2)$에서의 기저를 사용하면, n이 커짐에 따라, 연산의 수행속도가 매우 느려지며 테이블 검색을 사용하면, n에 따라, 메모리 소요량이 지수적으로 증가한다. 반면 본문에서 소개된 subfield 연산을 이용한 계산방법은 기저 사용과 테이블 검색방법의 단점을 서로 보완할 수 있다. 그러므로 제안된 구현방법은 S-box의 안전도를 높이기 위해 16, 32와 같이 큰 n을 선택하더라도 $n=8$, 16정도의 적은 메모리만으로 효율적인 S-box를 구현할 수 있으며 PC 등의 계산능력을 고려하여 개발된 기존의 S-box를 저장용량이 제한적인 스마트카드 등에서 구현할 때 더욱 유용하다.

참고 서적

[1] K. Aoki, K. Ohta, Fast Arithmetic Operations in $F(2^n)$ for Software Implementation, Proc. of SAC'97-Workshop on Selected Areas in Cryptography, 1997.

[2] 지수-로그방식을 사용하면 곱셈 연산 시 $xy=\exp^{\log x+\log y}$이므로 3번의 테이블 검색이 필요하다. 마찬가지로 역원과 제곱을 구하기 위해서는 2번의 테이블 검색이 필요하다.

[2] T. Beth, C. Ding, On Almost Perfect Nonlinear Permutations, Advances in Cryptology-Eurocrypt'93, Springer-Verlag, 1994.

[3] C. Carlet, *Codes de Reed-Muller, codes de Kerdock et de Preparata*, Thesis, Publication of LITP, *Institute Blaise Pascal, Univ. Paris 6*, 90.59(1990).

[4] S. Moriai, How to Design secure S-boxes against Differential, Linear, Higher Order Differential, and Interpolation Attacks, The 1998 Symposium on Cryptography and Information Security, Japan, Jan 28-31, 1998 (in Japanese).

[5] S. Moriai, T. Shimoyama, Performance and Security of Block Ciphers using Operations in $GF(2^n)$, Proc. of SAC'97-Workshop on Selected Areas in Cryptography, 1997.

[6] J. P. Peiprzyk, Non-linearity of Exponent Permutations, Advances in Cryptology-Eurocrypt'89, Springer-Verlag, pp.80-92, 1989.

[7] J. P. Peiprzyk, Bent Permutations, Technical Report CS91/11; The University of New South Wales, Dept. CS, Presented at Int. Conf. on Finite Fields, Coding Th. and Adv. on Comm. & Comp., Las Vegas, 1991.

[8] E. Biham, A. Shamir, Differential Cryptanalysis of DES-like Cryptosystems, J. of Cryptology, vol. 4, no. 1, pp.3-72, 1991.

[9] F. Chabaud, S. Vaudenay, Links between Differential and Linear Cryptanalysis, Advances in Cryptology-Eurocrypt'94, Berlin, Springer-Verlag, pp.356-365, 1994.

[10] J. Daemen, L. Knudsen, V. Rijmen, The Block Cipher SQUARE, Fast Software Encryption Workshop, Springer-Verlag, pp.137-151, 1997.

[11] G. Harper, A. Menezes, S.A. Vanstone, Public-Key Cryptosystems with very small Key Lengths, Advances in Cryptology-Eurocrypt'92, Springer-Verlag, pp.163-173, 1993.

[12] T. Jacobson, L. Knudsen, The Interpolation Attack on Block Cipher, Proc. of 5th Fast Software Encryption, FSW'97, Springer-Varlag, pp.28-40, 1997.

[13] L. Knudsen, Truncated and Higher Order Differentials, Proc. of 2nd Fast Software Encryption Workshop, Springer Verlag, Leuven, Belgium, pp.196-211, 1995.

[14] X. Lai, Higher Order Derivatives and Differential Cryptanalysis, Proc. of Symposium on Communication, Coding, and Cryptography in honor of James L. Massey on the occasion of his 60'th birthday, Monte-Verita, Ascona, Switzerland, 1994.

[15] C. Lee, Y. Cha, The Block Cipher:SNAKE with Provable Resistance against DC and LC Attacks, In Proc. of 1997 Korea-Japan Joint Workshop on Information Security and Cryptology(JW-ISC'97), pp.3-17, 1997.

[16] M. Matsui, Linear Cryptanalysis Method for DES Cipher, Advances in Cryptology-Eurocrypt'93, Springer-Verlag, vol. 765, pp.386-397, 1994.

[17] M. Matsui, New Structures of Block cipher with Provable Security against Differential and Linear Cryptanalysis, Proc. of 3rd Fast Software Encryption Workshop, Cambridge, U.K, Springer-Verlag, Berlin, pp.205-218, 1996.

[18] T. Matsumoto, Y. Takashima, J.W. Machar, H. Imai, Inverter-based Multiplier for Ciphers and Codes, in Proc. of the 1988 Conf. of IEICE, SA-7-5, A-1, pp.173-174, 1988.

[19] K. Nyberg, L. Knudsen, Provable Security Against a Differential Attack, J. of Cryptology, vol.8, pp.27-37, 1995.

[20] K. Nyberg, Perfect Nonlinear S-boxes, Advances in Cryptology-Eurocrypt'91, Springer-Verlag, pp.378 - 386, 1991.

[21] K. Nyberg, On the Construction of highly Nonlinear Permutations, Advances in Cryptology-Eurocrypt'92, Springer-Verlag, pp.92-98, 1993.

[22] K. Nyberg, Differentially Uniform Mapping for Cryptography, Advances in Cryptology-Eurocrypt'93, Springer-Verlag, pp. 55-64, 1994.

[23] K. Nyberg, S-boxes and Round Functions with Controllable Linearity and Differential Uniformity, Fast Software Encryption Workshop, Springer-Verlag, pp.111-130, 1994.

[24] A. Pincin, A New Algorithm for Multiplications in Finite Fields, IEEE Trans. on Computers, vol. 38, no. 7, pp.1045-1049, 1989.

[25] C. Paar, A New Architecture for a Parallel Finite Field Multiplier with Low Complexity Based on Composite Fields, IEEE Trans. on Computers, vol. 45, no. 7, pp.856-861, 1996.

[26] V. Rijmen, J. Daemen, The Cipher SHARK, Proc. of 4th Fast Software Encryption, FSW'96, pp.99-112, Springer-Varlag, 1996.

[27] V. Rijmen, J. Daemen, AES Proposal:RIJDAEL, AES Submission, 1998.

[28] T. Shimoyama, S. Moriai, T. Kaneko, Improving the Higher Order Differential Attack and Cryptanalysis of KN Cipher, Preproc. of Information Security Workshop'97, Japan, pp.1-8, 1997.

[29] E. De Win, A. Bosselaers, S. Vandenberghe, P.D. Gersen, J. Vanderwalle, Fast Software Implementation for Arithmetic Operations in $GF(2^n)$, Advances in Cryptology-Asiacrypt'96, pp.65-76, Springer-Verlag, 1996.