Rijndael 암호알고리듬에 대한 Hamming weight 모델의 DPA공격

전영환*,곽동진*,이훈재**,문상재*

*경북대학교, 전자공학과

**경운대학교, 컴퓨터전자정보공학부

A DPA attack using hamming weight model on Rijndael algorithm

Young-hwan Jeon*,Dong-Jin Kwak*,Hoon-Jae Lee**,Sang-Jae Moon*

*School of Electronic and Electrical Eng., Kyungpook National University

**School of Computer, Electronics, and Information Communication Engneering, Kyungwoon University,

요 약

부-채널 공격 중에서 가장 핵심이 되는 전력분석 공격은 여러 가지 암호알고리듬이 장착된 스마트 카드 시스템에 대해 공격이 이루어졌으며, 대부분 이 전력분석 공격에 취약한 것으로 알려져 있다. 본 논문에서는 AES로 채택된 Rijndael 알고리듬에 대하여 스마트 카드 구현시 고려되는 전력분석 공격중에서 hamming weight 모델을 이용한 세 가지의 DPA 공격을 제시하고 그 대응방안을 설명한다.

I. 서론

스마트 카드는 마이크로 프로세서와 메모리를 통한 데이터 연산 처리 기능과 데이터 저장 기능을 바탕으로 1990년 이후 본격적으로 전자상거래, 이동 통신, 금융 결재, 교통, 의료 등 다양한 응용 분야에 사용되고 있다. 이와 같이 스마트 카드의 이용 분야가 급격히 확대되어 감에 따라 고도의 보안성과 안전성이 필요하게 되었다.

암호학자들은 기존의 수학이론들을 배경으로 가능한 수학적 공격방법들에 대해 안전할 수 있는 알고리듬들을 개발해 왔다. 하지만 암호시스템의 구현 시 암호알고리듬의 설계에 고려되지 못한 비밀정보의 누출을 발생시킬 수 있다. 특히 스마트 카드의 구현 시 비밀키에 대한 연산이 빈번히 일어남으로 이 비밀정보의 누출이 시스템의 안전도에 큰 영향을 끼칠 수 있다.

전력분석 공격은 Crypto'99에서 Paul Kocher에 의해 DES에 적용시킴으로서 제안된 방법으로, 만약 공격자가 스마트 카드 비밀키의 일부 hamming weight를 알고 있다면 전탐색(brute force search)의 가능한 키 영역을 줄일 수 있고 충분한 hamming weight가 주어진다면 전체 비밀키에 대한 정보를 알 수가 있다[1][2][3]. 전력분석 공격은 크게 스마트 카드 내부의 비밀키의 연산 시 직접 소비전력신호의 특성을 파악하여 비밀키에 대한 정보를 알아내는 SPA(simple power analysis)와 SPA에 통계적인 분석방법과 에러 정정 기술을 첨가한 DPA(differential power analysis), 그리고 IPA(inferential power analysis)로 나누어 질 수 있다[4].

본 논문에서는 DES를 대신해 그 활용 범위가 확대되어 가고 있는 Rijndael 암호알고리듬에 대하여 DPA 공격이 가능한 부분에 대해 분석하고 hamming weight 모델을 적용한 DPA 공격을 제안하고 그 대응방법에 대해 간단히 설명한다.

II. 전력분석공격

실제 암호학자들은 암호알고리듬의 개발 시 입력 평문과 출력 암호문에 대해서는 공격자가 접근할 수 있지만 비밀키에 대해서는 접근할 수 없다고 보고 프로토콜을 개발한다. 그러나 암호알고리듬을 바탕으로 한 암호시스템의 구현은 그림 1과 같이 구동 시 그 알고리듬 개발자에 의해 고려되지 못하여 생기는 정보의 누출로 공격에 취약점을 가질 수 있다.

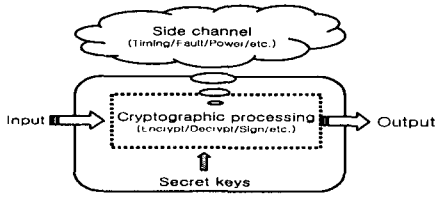

그림 1: 부-채널 정보의 누출

Kelsey는 이런 정보의 누출을 부-채널(side-channel)이라 정의하고 부-채널을 이용한 공격방법을 부-채널 공격(side-channel attack)이라 언급했다[5].부-채널 공격은 크게 시간공격(timing attack), 결함 주입공격(fault insertion attack), 전자기 누출공격(electromagnetic emission attack), 그리고 전력분석 공격(power analysis attack)등으로 나눌 수 있다[6][7][8][9].

부-채널 정보 중 소비전력의 측정은 시스템 개발자가 가장 제어하기 어려운 항목이다. 스마트 카드 내부에서 이루어지는 모든 연산은 디지털화된 이진 데이터의 연산으로 이루어지므로 이진 데이터 연산을 행할 때 비밀키의 hamming weight에 따른 전력 소비의 차이점은 개발자가 제어하기 힘든 부분이다. 소비전력의 부-채널에 대한 정보를 개발자가 줄일 수 있다하더라도 완전히 막을 수는 없다. 위의 소비 전력을 분석하는 부-채널 공격방법을 전력분석 공격이라 한다.

1. SPA(simple power analysis)

SPA 공격은 스마트 카드에서 연산되는 암호 프로세서의 전력소비를 직접 관찰하여 카드내부의 저장되어 있는 비밀키를 알 수 있는 공격방법이다. 실제로, 스마트 카드 내에서 소비되는 전력을 측정하기 위해서는 스마트 카드의 접지 부분과 전력 공급기 접지사이에 10~50Ω정도의 작은 값을 갖는 저항을 직렬로 연결하여 소비되는 전력을 측정할 수 있다[10]. 스마트 카드내의 암호 프로세서는 수행되어 지는 명령(instruction)에 따라 저마다 서로 다른 소비 전력 신호 특성을 가진다. 특히, 다음 명령들은 매우 서로 다른 소비 전력 신호 형태를 가진다[11].

- 사칙연산
- 비트형 부울(boolean) 연산
- 레지스터의 값을 RAM에 저장
- 레지스터의 값을 EEPROM에 저장
- RAM/EEPROM의 값을 레지스터에 로딩 (loading)
- 기타.

공격자는 이런 정보를 이용하여 스마트 카드 내부의 비밀키에 따라 수행되는 명령의 특성을 파악하여 그 명령의 순서를 역추적 함으로써 내부 비밀키를 알아낼 수 있다.

2. DPA(differential power analysis)

DPA 공격은 SPA 공격보다 방어하기 더 어려운 강력한 공격방법으로 기존의 SPA의 소비 전력을 관찰하는 것에 더하여 비밀키와 정확히 상관관계(correlation)를 가지는 정보를 추출하기 위해 통계적인 분석(statistical analysis)과 에러 정정(error correction) 기술을 사용한다.

DPA의 구현은 다음의 두 단계로 나눌수 있다. 먼저 데이터 수집 단계로 스마트카드가 암호학적 연산을 실행 시에 소비되는 전력을 표본화(sampling)하여 그 데이터를 수집한다. 데이터 수집 후 데이터 분석 단계로 그 표본화한 데이터를 잡음신호 감소와 차분(differential)신호의 명확성을 위해 디지털 신호 해석과 통계적인 방법으로 분석한다.

III. Rijndael 암호알고리듬

NIST(National Institute of Standards and Technology)에서 1997년 9월 DES를 대체할 후속 대칭키 암호 알고리듬인 AES를 공개 모집한 후 3번에 걸쳐 관련 컨퍼런스가 열렸다. 그리고 2000년 10월 2일 5개의 후보 알고리듬 중 최종적으로 Rijndael이 선정되었다[12]. Rijndael은 128, 192, 256비트의 입력에 대한 암호화를 모두 지원하며, 현재까지 알려진 공격방법에 강하도록 설계되면서

IC 카드 적용도 고려한 하드웨어와 소프트웨어 구현 측면에서도 효율적으로 설계된 것으로 평가되었다. Rijndael은 4개의 바이트(byte)로 이루어진 워드(word)단위로 구성된 열을 입력받아 암·복호문을 생성한다. CipherKey와 암·복호문의 크기는 각각 4, 6, 8 워드 중 하나를 선택 가능하며, CipherKey와 암·복호문의 크기는 서로 독립으로 연관이 없다.

1. 암호화 과정

암호화 과정은 initial round key addition을 한 후에 Nr - 1 번의 라운드를 거치고 나서 최종 라운드를 거친다. 각 라운드는 ByteSub, ShiftRow, MixColumn, AddRoundKey 4개의 서로 다른 변환 함수로 구성되며 최종 라운드에서는 MixColumn이 제외된 상태에서 수행이 이루어진다. 그림 2는 Rijndael의 암호화 과정을 블록선도로 나타낸 것이다.

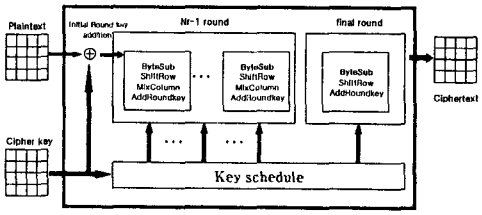

그림 2: 암호화 과정의 블록 선도

각각의 변환 함수는 그림 3에 나타내었다. 각 라운드에서 ByteSub은 바이트 단위로 역 변환이 가능한 행렬형태의 S-box에 의해서 계산하여 변환한다. ShiftRow는 그림에서와 같이 일정한 규칙에 따라서 좌측회전이동을 하며 MixColumn은 행의 요소간에 GF(2^8)상 모듈라 곱셈 연산을 한다. 그리고 AddRoundKey 는 CipherKey가 전 라운드에 걸쳐 영향을 미치게 그림에서와 같이 비트별 더하기를 한다.

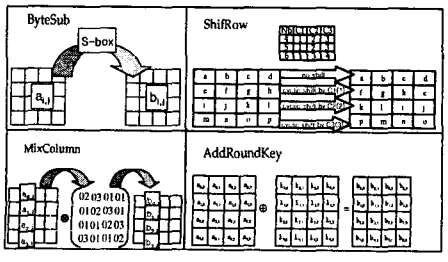

그림 3: 4개의 변환 함수

IV. Hamming weight 모델을 이용한 Rijndael의 DPA 공격

본 장에서는 Rijndael 알고리듬을 사용하여 스마트 카드내 구현 시 고려되는 전력분석 공격 중에서 hamming weight 모델을 사용한 DPA 공격을 설명한다. Rijndael 알고리듬에 대한 전력분석 공격은 세 부분에서 DPA 공격이 가능하며, 각각의 공격에 대하여 설명하고자 한다.

1. 공격-I

스마트 카드내의 프로세서는 데이터가 처리되어지는 과정에서 데이터의 hamming weight에 대한 정보를 누설할 수 있으며, 높은 hamming weight의 값을 갖는 데이터는 낮은 hamming weight의 값을 갖는 데이터보다 데이터 처리과정에서 더 많은 전력을 소비하게 된다. 그림 4는 스마트 카드내에서 8비트 프로세스를 사용하여 Rijndael 알고리듬의 AddRoundKey 변환과정을 수행할 경우에 대한 DPA 공격이다.

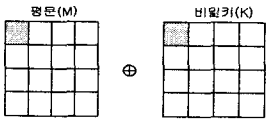

그림 4: DPA 공격-I

Rijndael의 initial round key addition 부분과 마지막 라운드의 AddRoundKey 변환 과정에서 실제 DPA 공격이 가능하며, 비밀키와 메시지가 연산될 때 소비되는 전력을 이용하여 비밀키를 알아낼 수 있다. 여기서는 Rijndael 알고리듬의 Nb와 Nk를 각각 4로 두고, initial round key addition부분에 DPA 공격을 하였다.

■1단계: 전체 구하려는 스마트 카드의 N비트 비밀키를 $K(k_0, k_1, \cdots, k_{N-2}, k_{N-1})$라 정의하고 최하위 비트의 순서로 키의 일부가 입력되며 K의 ℓ 번째 비트(k_ℓ)를 공격한다고 가정한다.

■2단계: 임의의 N비트 평문 $M_i(m_0, m_1, \cdots, m_{N-2}, m_{N-1})$을 선택하여 연산을 수행한 후 소비전력신호 $S_i[j]$을 구한다.

■3단계: 각각의 평문에 대한 ℓ 번째 비트의 값이 "0" 과 "1"로 분류한 후 그 평문에 해당하는 소비전력신호 데이터를 분류한다.

S_0=[$S_i[j]$ | 평문 ℓ번째 비트의 값 : "0"]

S_1=[$S_i[j]$ | 평문 ℓ번째 비트의 값 : "1"]

■4단계: 양분한 전력신호 데이터를 각각 평균하여 평균에 대한 차분신호 D[j]를 구한다.

$$D[j]=\frac{1}{|S_0|}\sum_{S_i[j]\in S_0}S_i[j]-\frac{1}{|S_1|}\sum_{S_i[j]\in S_1}S_i[j]$$
$$=\overline{S_0}[j]-\overline{S_1}[j]$$

■5단계: 비밀키 K의 ℓ번째 비트의 값 k_ℓ을 결정한다.

$k_\ell=1$ if D[j] = " Positive "

$k_\ell=0$ if D[j] = " Negative "

5단계의 비밀키 K의 ℓ번째 비트의 값 k_ℓ을 결정하는 과정에 대한 설명이다. 비밀키 k_ℓ와 m_ℓ의 배타적 논리합에 대한 hamming weight의 기대값은 식(1)과 같이 정의된다.

$$E[d\mid k_\ell\oplus m_\ell=0]=\frac{P-1}{2},\ E[d\mid k_\ell\oplus m_\ell=1]=\frac{P+1}{2} \quad (1)$$

여기서 d는 hamming weight이고, P는 스마트 카드내의 프로세서 크기를 나타낸다. 비밀키 k_ℓ의 값이 "0"일 경우 평균 소비 전력신호 데이터의 기대값은 다음과 같다.

$$\overline{S_0}[j]=E[S\mid k_\ell=0, m_\ell=0]=\frac{P-1}{2}\times\Delta h \quad (2)$$

$$\overline{S_1}[j]=E[S\mid k_\ell=0, m_\ell=1]=\frac{P+1}{2}\times\Delta h \quad (3)$$

위의 식(2)에서 식(3)의 차를 구하면 식(4)과 같으며 Δh는 하나의 hamming weight에 대한 소비되는 전력을 나타낸다.

$$D[j]=\overline{S_0}[j]-\overline{S_1}[j]=-\Delta h \quad (4)$$

같은 방법으로, 비밀키 k_ℓ의 값이 "1" 일 경우 차분신호는 식(5)과 같다.

$$D[j]=\overline{S_0}[j]-\overline{S_1}[j]=\Delta h \quad (5)$$

그러므로, 비밀키 k_ℓ값이 "0" 일 경우는 차분신호 D[j]의 값이 음의 값을 가지고, k_ℓ값이 "1" 일 경우는 양의 값을 가지게 된다.

2. 공격-II

공격-I의 경우는 8-비트 프로세서를 사용할 경우 1-비트의 영향으로도 충분히 DPA가 가능하지만, 16-비트 이상의 프로세서를 가진 스마트 카드에서는 1-비트의 영향으로는 상당히 많은 수의 소비 전력 신호가 필요하게 되므로 공격-I의 경우로 공격하기엔 너무나 비효율적이다. 그래서 공격-II는 16-비트 이상의 프로세서를 가진 스마트 카드를 좀 더 효율적으로 공격하는 방법이다.

공격-I는 차분 소비전력 신호의 값에 따라 한 비트씩 비밀키의 값을 알아내는 방법이지만 공격-II는 공격-I와는 달리 비밀키의 일부분을 추측하여 그 추측한 값이 실제 비밀키의 값인지를 검증하는 방법으로 여러 비트씩 공격이 가능하다. 만약 추측한 키의 값이 옳지 않다면 분류함수를 이용하여 분류한 소비 전력신호는 실제 비밀키와 아무런 상관관계를 갖고 있지 않아 차분신호는 랜덤한 특성을 나타낸다. 반면에 올바른 키라면 분류함수와 실제 키가 상관관계를 가져 차분신호는 spike를 띠게 된다. 따라서 공격-II는 실제 소비 전력신호를 분류하는 분류함수를 어떻게 설정하는가하는 문제가 제일 중요하다.

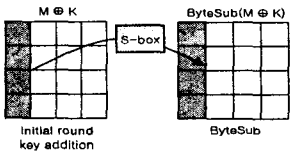

그림 5: DPA 공격-II

그림 5는 스마트 카드 내에서 32비트 프로세서를 사용하여 Rijndael 알고리듬을 수행할 경우에 대한 DPA 공격이다. 공격-II에서 가장 중요한 분류함수는 initial round key addition부분과 ByteSub 변환 함수에 hamming weight 모델을 이용하여 분류함수를 정의하였다.

■1단계: 임의의 평문 M_i을 선택하여 연산을 수행한 후 소비전력신호 $S_i[j]$을 구한다.

■2단계: 키의 일부부인 8-비트에 대하여 $K(k_0, k_1, \cdots, k_7)$값을 추측한다.

■3단계: 추측한 키와 전력신호 데이터를 구할 때 입력으로 사용한 평문을 이용하여 사전 계산한 후 ByteSub 변환함수를 수행한 결과 값에 대해 분류함수(distribution function)를 이용하여 전력신호데이터를 분류한다

S_{high}=[$S_i[j]$ | $D[ByteSub(K\oplus M)]$="high hamming weight"

S_{low}=[$S_i[j]$ | $D[ByteSub(K\oplus M)]$="low hamming weight"

■4단계: 양분한 전력신호 데이터를 각각 평균하여 평균에 대한 차분신호 D[j]를 구한다.

$$D[j] = \overline{S_{high}}[j] - \overline{S_{low}}[j]$$

■5단계: 추측한 K에 대하여 검증한다.

K is correct if D[j] = " Spike "

K is wrong if D[j] = " Non-spike "

■6단계: 추측한 키가 옳지 않다면 다시 2단계로 간다.

3. 공격-III

공격-I·II의 경우 메시지에 대한 입력 또는 연산 후의 출력 값을 공격자가 알고 있어야 하지만 공격-III에서는 입·출력에 대한 값을 알 필요가 없다. 즉 메시지에 독립(independent)한 공격 방법이다. Rijndael은 메시지를 암·복호하는 과정에서 각 라운드마다 다른 키를 사용하며, 이 라운드 키는 비밀키에서 키 생성(key scheduling) 과정을 통하여 각 라운드에 해당하는 키를 생성하게 된다. 그래서 낮은 계산 능력과 적은 메모리를 가진 스마트 카드에서는 전체 라운드에 대한 키를 생성할 수 없고, 각 라운드마다 해당되는 키를 생성하여 메모리에 저장하게 된다. 스마트 카드에서는 메모리에 "1"의 값을 저장하는데 소비되는 전력과 "0"의 값을 저장하는데 소비되는 전력은 차이가 난다. 이 개념으로부터 각 라운드에서 생성된 키를 메모리에 저장하는 과정에서 소비되는 전력을 이용하면 각 라운드 키에 대한 hamming weight 값을 알 수 있게 된다. 공격-III에서는 키 생성과정 부분을 찾아내는 방법이 중요하며, 다음과 같은 방법으로 찾아낼 수 있다.

■1단계: 하나의 스마트 카드에 임의의 많은 수의 평문을 선택하여 연산을 수행하여, 각 평문에 해당하는 소비전력신호를 구한다.

■2단계: 1단계에서 구한 소비전력신호를 각 클럭 사이클(Clock cycle)에 대하여 소비전력신호를 비교하여 많은 변동을 갖는 클럭 사이클은 제거한다. 이 부분은 처리되는 데이터 값에 영향을 많이 받는 부분이며, 제거되고 남은 클럭 사이클은 입력되는 데이터에 독립적인 부분이다.

■3단계: 다른 비밀키를 가진 여러 개의 스마트 카드를 이용하여 1단계의 과정을 되풀이 한 후 데이터에 독립된 공통 부분을 찾아낸다. 그리고 이 공통 부분에 대하여 각 카드사이에서 작은 변동을 갖는 클럭 사이클은 제거한다. 제거된 이 부분은 기본적인 시스템과 관련된 소비전력 부분이다.

8-비트 프로세서를 가진 스마트 카드 내에서 비밀키 $K(k_1, k_2, \cdots, k_{127}, k_{128})$라고 정의하며 키 생성과정에서 생성된 각 라운드 키 $K^{(i)} = (k_1^i, k_2^i, \cdots, k_{128}^i)$로 나타낸다. 여기에서 i는 라운드를 의미한다.

그림 6은 Rijndael의 키 생성과정을 나타낸것으로 비밀키를 이용하여 첫 라운드 키를 생성하며, 첫 라운드의 키를 이용하여 둘째 라운드 키를 생성하듯이 이전 라운드 키를 이용하여 다음 라운드 키를 생성하는 단계를 수행한다.

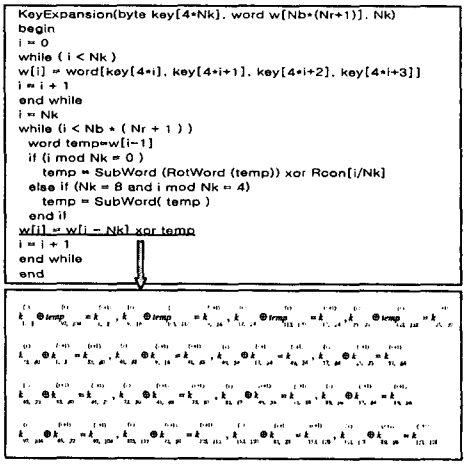

그림 6: 키 생성과정

키 생성과정을 통하여 생성된 각 라운드 키들을 메모리에 저장하는 과정에서 hamming weight의 값을 알 수 있다. 공격자는 각 라운드 키에 대한 hamming weight의 정보를 알고 있으므로 생각할 수 있는 진짜 라운드 키일 경우는 ${}_8C_j$ 이다. 여기서 j는 hamming weight 값이다. 만약 라운드 키를 생성하는 과정에서 hamming weight의 값이 크거나 작은 경우는 쉽게 라운드 키를 찾을 수 있다. 라운드 키 $K^{(i)}$에 대하여 각 바이트(byte)의 hamming weight값을 알 수 있으므로 hamming weight를 통하여 각각의 바이트 당 식(6)과 같은 정보를 누설하게 된다.

$$-\sum_{j=0}^{8} \frac{\binom{8}{j}}{2^8} \lg \frac{\binom{8}{j}}{2^8} \approx 2.54\ [bit] \qquad (6)$$

그러므로, 전체 128-비트 비밀키에 대하여 $16 \times 2.54 \approx 40.64[bit]$의 정보를 누설하게 된다. 이를 이용하면 키의 전 탐색(brute force search) 사이즈도 2^{128}에서 2^{87}로 줄일 수 있다. 이와 같은 방법으로 공격자가 한 라운드 키를 찾을 수 있다면

키 생성과정을 역으로 계산하여 비밀키 K를 찾을 수 있다.

4.대응방안

DPA는 SPA를 이용하여 얻은 소비전력 신호를 가지고 통계적인 분석방법으로 비밀키에 대한 정보를 알아내는 방법이므로, DPA에 대한 근본적인 대응 방안은 이 통계적인 데이터 분석을 어렵게 하는 것이다. 통계적 분석 방법을 어렵게 하는 DPA 대응 방안으로 크게 두 가지 범주(category)로 분류할 수 있다.

첫 번째 방법으로 DPA 차분 소비전력 신호를 줄이는 것과 소비전력 측정에 잡음(noise)을 첨가하여 신호 대 잡음비를 줄이는 것이다. 차분 소비전력 신호를 줄이는 방법은 물리적으로 시스템을 차폐시켜 소비 전력신호자체를 줄이는 방법과 "0"과"1"의 상태 변화를 거의 같게 하여 hamming weight값을 일정하게 하는 것이다. 하지만 이 방법들은 차분 소비전력 신호를 어느 정도 줄일 수는 있어도 없앨 수는 없다. 그러므로 공격자가 상당히 많은 수의 소비전력 신호를 이용하여 DPA 공격을 한다면 막을 수가 없다. 잡음을 첨가시도 마찬가지로 많은 소비전력 신호를 이용하면 DPA 공격이 가능하다.

두 번째 방법으로 처음부터 암호 알고리듬을 스마트 카드내에 구현 시 DPA 공격을 고려하여 DPA 공격이 봉쇄되도록 구현하는 것이다. 알고리듬 상이나 구현시 약간의 비용증가와 덧붙여지는 잉여(redundancy)가 있을 수 있으나 DPA 공격을 효과적으로 막을 수 있다. 이 방법은 실제 DPA 공격의 분류함수를 비밀키에 대한 정보와 아무런 상관관계도 없게 하여 비밀키에 대한 정보가 누출되지 않게 봉쇄하는 것이다.

V. 결론

본 논문에서는 AES인 Rijndael 알고리듬에 대하여 공격이 가능하도록 hamming weight 모델을 적용한 세 가지의 DPA 공격을 제안 및 분석하였으며, 분석 결과 Rijndael 알고리듬은 세 가지의 DPA 공격에 모두 취약한 것으로 밝혀졌다. DPA 공격에 대한 대응책이 마련되지 않을 경우 시스템의 안전도에 큰 영향을 끼칠 수 있기 때문에 스마트 카드 시스템에 Rijndael 알고리듬을 적용할 경우에는 필히 이러한 DPA 공격을 막을 수 있는 구체적인 대응방안이 필요하다.

참고문헌

[1] P. Kocher, J. Jaffe, and B. Jun, "Introduction to Differential Power Analysis and Related Attacks,"http://www.cryptography.com/dpa/technical, 1998.

[2] P. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis,"CRYTO '99, pp. 388-397, 1999.

[3] National Bureau of Standards,"Data Encryption Standard," FIPS 46, 1977.

[4] Paul N. Fahn and Peter K. Pearson, "IPA: A New Class of Power Attack", CHES'99, pp. 144-157.

[5]J. Kelsey, B. Schneier, D. Wagner, and C.Hall,"Side Channel Cryptanalysis of Product Cipher," in Proceedings of ESORICS '98, pp. 97-110, 1998.

[6] P. Kocher, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems," CRYPTO '96, pp.104-113, 1996.

[7] E. Biham and A. Shamir, "Differential Fault Analysis of Secret Key Cryptosystems," CRYPTO'97, pp. 513-525, 1997.

[8] W. van Eck, "Electromagnetic Radiation from Video Display Units:An Eavesdropping Risk," Computers and Security, v. 4, pp. 269-286, 1985.

[9] K. Gandolfi, C. Mourtel, and F. Olivier, "Electromagnetic Analysis: Concrete Results," CHES'2001, pp. 255-265.

[10] T. S. Messerges, E. A. Dabbish, and R. H. Slan, "Investigations of Power Analysis Attacks on Smartcards," Proceeding of USENIX Workshop Smartcard Technology, May 1999, pp.151-161.

[11] Joan Daemen and Vincent Rijmen, "Resistance Against Implementation Attacks: A Comparative Study of the AES proposals," Second Advanced Encryption Standard(AES) Candidate Conference, March, 1999.

[12] Joan Daemen and Vincent Rijmen, "Rijndael Block Cipher," 1999.