情報保護學會論文誌
第20卷 第6號, 2010. 12

AES에 대한 반복문 오류주입 공격

박 제 훈,[1†] 배 기 석,[1] 오 두 환,[2] 문 상 재,[1] 하 재 철[2‡]
[1]경북대학교, [2]호서대학교

A Fault Injection Attack on the For Statement in AES Implementation

JeaHoon Park,[1†] KiSeok Bae,[1] DooHwan Oh,[2] SangJae Moon,[1] JaeCheol Ha[2‡]
[1]Kyungpook National University, [2]Hoseo University

요 약

오류 주입 공격은 비밀키가 내장된 암호 장치에서 연산을 수행 시 공격자가 오류를 주입하여 비밀키를 찾아내는 공격으로서 암호시스템의 안전성에 심각한 위협이 되고 있다. 논문에서는 AES 암호를 수행하는 동안 마지막 라운드의 키를 더하는 반복문(for statement)에 대한 오류 주입을 통해 비밀키 전체를 공격할 수 있음을 보이고자 한다. 상용 마이크로프로세서에 AES를 구현한 후 한 번의 레이저 오류 주입 공격을 시도하여 128비트의 비밀키가 노출됨을 확인하였다.

ABSTRACT

Since an attacker can occur an error in cryptographic device during encryption process and extract secret key, the fault injection attack has become a serious threat in chip security. In this paper, we show that an attacker can retrieve the 128-bits secret key using fault injection attack on the for statement of final round key addition in AES implementation. To verify possibility of our proposal, we implement the AES system on ATmega128 microcontroller and try to inject a fault using laser beam. As a result, we can extract 128-bits secret key through just one success of fault injection.

Keywords: AES, Fault injection attack, Loop statement, Cryptographic device

I. 서 론

오류 주입 공격은 1997년 Boneh 등에 의해 RSA-CRT(Chinese Remainder Theorem) 서명 알고리즘 과정 중 오류를 주입하여 비밀정보를 추출하는 공격 방법으로 처음 소개되었다 [1]. 또한 Biham과 Shamir가 블록 암호에 대해서 차분 오류 공격이 가능함을 제안하였고 [2], 이를 차분 오류 분석(Differential Fault Analysis, DFA) 공격이라고 부른다. 차분 오류 주입 공격은 암호 디바이스의 실행 중에 생성한 정상 암호문과 오류를 주입하여 얻은 오류 암호문을 분석하여 해당 암호 디바이스에서 사용된 비밀키를 추출할 수 있는 공격 기법으로서 최근 위협적인 공격 방법으로 주목받고 있다. 특히, AES [3] 알고리즘에서의 DFA에 대한 연구도 많이 이루어졌는데, Piret와 Quisquater는 2쌍의 정상/오류 암호문을 이용하여 128비트 AES 비밀키를 찾는 방법을 제안하였고 [4], Giraud가 한 비트 오류 또는 키 스케줄상의 한 바이트 오류를 주입하여 전체 키를 찾아내는 방법을 제안하였다 [5]. 2008년에는 Kim과 Quisquater가 키 스케줄링 상에 오류를 주입함으로써 모두 8쌍의 정상/오류 암호문을 이용하여 키를 찾아내는 방법을 제안하였다 [6]. 위의 오류 주입 방법들은 키 스케줄링이나 암호화가 수행되는 동안

접수일(2010년 7월 19일), 수정일(2010년 10월 8일),
게재확정일(2010년 11월 2일)
† 주저자, jenoon65@ee.knu.ac.kr
‡ 교신저자, jcha@hoseo.edu

데이터를 저장하는 레지스터나 메모리에 오류를 주입하는 방식이다.

또한, AES 암호 알고리즘이 수행되는 동안 for 문과 같은 반복적인 연산을 하는 과정에 오류를 주입하여 수행되는 라운드 수를 10개에서 하나로 줄여 그 결과 값을 차분하여 공격하는 방법도 제안되었다 [7]. 이 방법은 AES에서 저장된 데이터가 아닌 프로그램 코드를 공격하는 새로운 공격 형태이다.

본 논문에서는 Choukri와 Tunstall의 공격 방식 [7]과 유사하게 반복적인 연산을 수행하는 동안 오류를 주입하여 비밀키를 추출하는 방법을 제시한다. 논문 [7]에서는 암호화 과정의 라운드를 줄인 것인데 논문에서는 마지막 라운드의 10라운드 키를 더하는 과정(AddRoundKey)에 오류를 주입하여 마지막 라운드 키를 찾는 공격 방법이다. AES에서는 마지막 라운드 키를 찾으면 역산으로 마스터 비밀키를 찾을 수 있다.

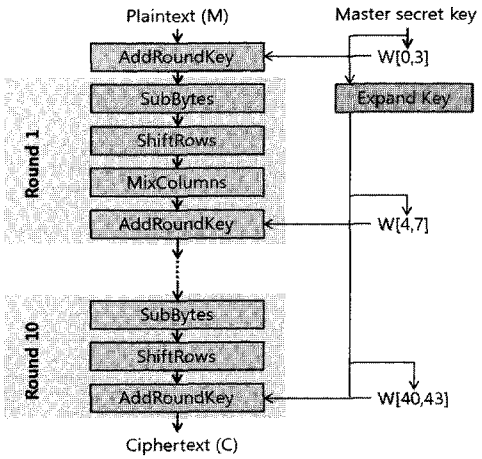
본 논문의 2장에서는 AES와 기존의 for문에 대한 오류 주입으로 라운드 축소를 이용한 비밀키 공격 방법을 소개한다. 제 3장에서는 논문에서 제시하는 마지막 라운드의 라운드 키를 더하는 과정에서 오류를 주입하여 비밀키를 공격하는 방법을 제시한다. 4장에서는 상용 마이크로프로세서에 AES를 구현하고 레이저 장비를 이용한 오류주입 공격을 통해 실제로 비밀키가 추출되는 실험을 보이고자 한다. 마지막으로 5장에서 결론을 맺는다.

II. AES에서의 오류주입 공격

2.1 AES 암호

AES는 2001년 미국 NIST에 의해 연방 정보처리 표준으로 지정된 대칭키 암호화 방식으로 128비트 블록과 128, 192, 256비트의 키를 사용한다. 128비트의 평문 입력 블록은 각 라운드의 연산 과정을 거쳐 128비트의 암호문으로 출력된다. 사용하는 키의 길이에 따라 수행되는 라운드 수가 달라지게 되는데 128비트 비밀키를 사용하는 경우에는 10라운드 연산을 거치게 된다. [그림 1]은 AES의 암호화 과정을 나타낸 것이다.

암호용 칩은 암호 연산을 수행하기 전에 128비트 비밀키를 확장하는 과정부터 수행한다. 키 확장 과정을 통해 128비트로 이루어진 11개의 라운드 키를 만들게 된다. 여기서 $W[0]$, $W[1]$, $W[2]$, $W[3]$는 라운드 함수가 시작되기 전 평문과 더해지는(실제로는 XOR 연산) 초기 라운드 키이다. 그리고 각 라운드는 SubBytes, ShiftRows, MixColumns 그리고 AddRoundKey 연산을 수행한다. 단 마지막 10라운드에는 MixColumns 연산을 수행하지 않는다. 본 논문에서의 공격은 마지막 라운드의 AddRoundKey 연산에 관한 것이며 이를 통해 마지막 라운드 키 $W[40]$, $W[41]$, $W[42]$, $W[43]$를 알아내고자 하는 것이 목표이다. 여기서 주목할 사실은 AES에서는 한 라운드 키 128비트만 노출되면 역계산에 의해 마스터 키를 알 수 있다는 점이다.



[그림 1] AES 암호화 과정

2.2 AES에서 라운드 축소 오류주입 공격

Choukri와 Tunstall는 AES 암호 알고리즘이 수행되는 동안 for 문과 같은 반복적인 연산을 하는 과정에 오류를 주입하여 라운드 수를 줄여 출력한 결과 값을 차분하여 공격하는 방법을 제안하였다 [7].

```
state = M;
AddRoundkey(state, W[0]..W[3]);
for( i=1 i<=10 i++) { //오류주입 위치
  SubBytes(state);
  ShiftRows(state);
  if( i != 10) MixColumns(state);
  AddRoundKey(state,W[i*1]..W[i*1+3]);
}
C=state;
```

[그림 2] AES 암호화 의사 코드

그들의 논문에서 사용한 AES 암호 과정의 의사 코드(pseudo code)는 [그림 2]와 같다.

이 공격에서는 for문이 수행될 때 오류를 주입하여 반복문을 한 번만 수행하고 벗어나도록 하였다. 그 결과 1라운드를 포함한 아래의 5개 함수만 수행한 오류 암호문을 출력한다.

```
state = M;
AddRoundkey(state, W[0]..W[3]);
SubBytes(state);
ShiftRows(state);
MixColumns(state);
AddRoundKey(state, W[4]..W[7]);
C'=state;
```

따라서 초기 라운드 키를 더하는 과정과 1라운드만 수행한 라운드 축소형 오류 암호문을 얻을 수 있다. 공격자는 2개의 평문 M_1과 M_2에 대한 오류 암호문 C_1'과 C_2'을 이용하여 다음 식을 만족하는 초기 라운드 키 $W[0], W[1], W[2], W[3]$를 찾게 된다.

$$InvShiftRows(InvMixColumns(C_1' \oplus C_2')) = SubBytes(M_1 \oplus W_0) \oplus SubBytes(M_2 \oplus W_0) \quad (1)$$

여기서 W_0는 128비트의 초기 라운드 키를 의미하며 $W_0 = W[0] \| W[1] \| W[2] \| W[3]$이다. 라운드 키를 찾기 위한 (1)식에서 우측 항은 바이트 단위로 연산할 수 있어 라운드 키를 바이트 단위로 키를 계산해 낼 수 있다. 여기서 오류주입 공격의 중요한 요소는 1라운드만 수행하고 for 문을 벗어나도록 프로그램 수행 코드에 오류를 주입하는 일이다.

이 공격은 10개의 라운드를 1개의 라운드로 축소하여 공격하는 형태이므로 공격이 다소 용이하였다. 그러나 알고리즘을 구현하는 개발자가 [그림 2]와 같이 구현하지 않고 10라운드 과정을 for문 바깥에서 수행하도록 하면 최소 2개 라운드를 수행해야 하므로 공격이 쉽지 않다. 실제로 AES 표준 문서인 FIPS 197에서는 10라운드는 for문 안에서 수행하는 것이 아니라 바깥에서 수행하도록 제안하고 있다. 자세한 내용은 표준문서 [3]에서 참조할 수 있다.

III. 제안하는 반복문 오류 주입 공격

본 논문에서는 Choukri와 Tunstall의 공격 방식과 유사한 가정으로 반복적인 연산을 수행하는 과정에서 오류를 주입하여 오류 주입 시점 이후의 연산을 건너뛰도록 하는 것이다. 논문에서 제안하는 공격이 시도되는 지점은 아래와 같은 마지막 10라운드의 키를 XOR 하는 과정이다.

```
AddRoundKey(state, W[40]..W[43]);
```

여기서 state는 128비트의 데이터를 바이트 단위로 나누어 4×4 배열 형태로 저장하고 처리한다. 따라서 워드 단위로 표시된 32비트 라운드 키 $W[i]$는 4개의 바이트 단위의 키로 분할되어 처리된다. 즉, 아래와 같이 표현할 수 있다.

$$W[40] = b[0] \| b[1] \| b[2] \| b[3]$$
$$W[41] = b[4] \| b[5] \| b[6] \| b[7]$$
$$W[42] = b[8] \| b[9] \| b[10] \| b[11]$$
$$W[43] = b[12] \| b[13] \| b[14] \| b[15]$$

따라서 AddRoundKey 연산은 state를 나타내는 16개 바이트의 중간 값과 16개 바이트의 라운드 키 값이 XOR 연산을 하는 것이다. 마지막 라운드의 키를 XOR하는 과정을 C언어로 된 의사 코드로 구체화하면 [그림 3]과 같다.

```
....
for( i=0 i<16 i++) { // 오류 주입 위치
    state[i] = state[i] XOR b[i] ;
}
C=state;
```

[그림 3] 마지막 라운드의 AddRoundKey 의사 코드

먼저 공격자는 메시지 M에 대해서는 정상 암호문 C을 가지고 있다. 그리고 그림 3에서 1번의 루프 연산만 수행하고 출력되는 오류 암호문 C'을 가지고 있다고 가정하자. 이때 메시지 M은 정상 암호문을 출력할 때와 동일한 것을 사용한다. 따라서 동일한 메시지에 대한 정상 암호문과 오류 암호문을 차분하면 (2)식과 같이 마지막 라운드 키를 구할 수 있다.

$$C \oplus C' = W[40] \| W[41] \| W[42] \| W[43] \quad (2)$$

그런데 for문에 대한 오류를 주입할 때 최소 한 번의 루프는 수행하므로 첫번째 바이트는 언제나 정상 암호문으로 출력이 된다. 따라서 정상 루프를 수행한 첫번째 바이트를 제외한 나머지 120비트의 마지막 라운드 키를 찾을 수 있다. 알지 못하는 8비트는 키에

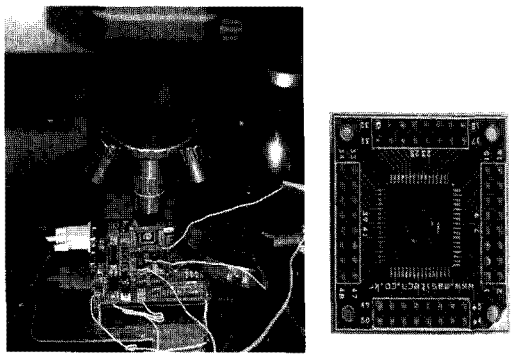
대한 전탐색을 시행하면 쉽게 키를 얻을 수 있다. 이를 위해 먼저 10라운드 키 중에서 마지의 8비트를 가정하여 마스터 비밀키를 역산한다. 그리고 이 비밀키를 이용하면 정상적인 암호문이 나오는지를 검사하는 방식으로 8비트에 대한 전탐색을 실시하면, 128비트의 10라운드 키를 찾고 결국 마스터 비밀키를 찾을 수 있다. 제안하는 공격이 Choukri와 Tunstall의 공격과 차이나는 점은 오류를 주입하여 라운드를 축소하는 것이 아니라 AddRoundKey 함수를 수행할 때 반복 회수를 줄이는 공격이라는 것이다. 또한 제안 방식에서는 하나의 평문에 대한 정상 암호문과 오류 암호문을 사용하는 것이 Choukri와 Tunstall의 공격과 다르다. 따라서 Choukri와 Tunstall의 공격에서는 동일한 위치에서 두 번의 오류 주입이 필요한 반면, 제안 방식에서는 1번의 오류 주입으로 키를 찾을 수 있다.

IV. 오류 주입 실험 및 고찰

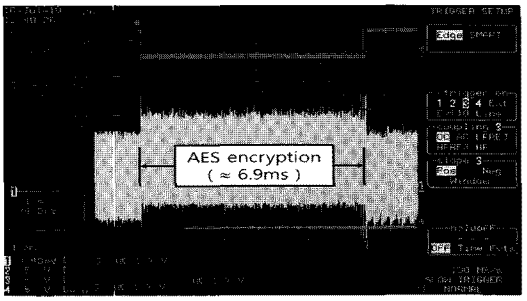
논문에서는 제안하는 반복문에 대한 오류 주입 공격이 실제로 가능한지를 실험하였다. 암호용 칩으로 사용할 수 있는 상용 마이크로프로세서에 AES를 구현하여 실험하였다. 암호용 칩은 ATmega128 [9]를 사용하였으며 오류 주입을 위해서는 EzLaze 3 레이저 장비를 사용하였다. 물론, 파형 관측을 위해 고성능 오실로스코프 장비를 사용하였으며 용이한 오류 주입을 위해 칩의 표면을 디캡핑(decapping)하여 사용하였다.

[그림 4]는 레이저 오류 주입 공격 장치와 디캡핑된 ATmega128 칩을 나타낸 것이다 [8]. 또한, 디캡핑한 칩에 공간적으로 정확한 오류 주입 위치를 잡기 위해 광학 현미경을 이용하였다.

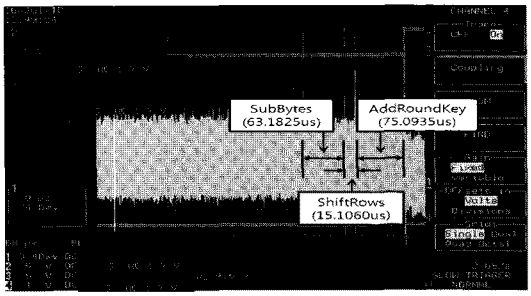
실험에서 실제 AES를 정상적으로 수행한 경우의



[그림 4] 레이저 오류 주입 실험 장치

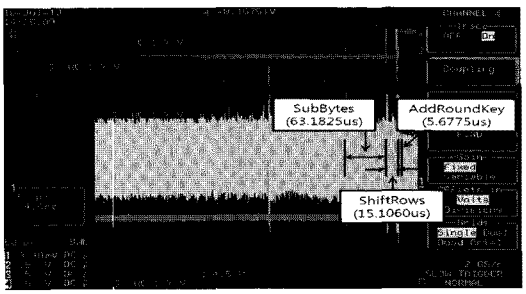


(a)



(b)

[그림 5] 정상 전력 파형: (a)AES 전체, (b)마지막 라운드 확대



[그림 6] 오류 주입에 의한 전력 파형 변화

전력 파형과 마지막 라운드의 라운드 키를 더하는 연산을 건너뛴 경우의 전력 파형을 나타낸 것이 [그림 5]와 [그림 6]이다.

실험에 사용된 128비트의 AES 알고리즘의 초기 비밀키는 "0x 2b 7e 15 16 28 ae d2 a6 ab f7 15 88 09 cf 4f 3c"이며, 10라운드 키는 "0x d0 14 f9 a8 c9 ee 25 89 e1 3f 0c c8 b6 63 0c a6"이다. 그리고 평문은 "0x 32 43 f6 a8 88 5a 30 8d 31 31 98 a2 e0 37 07 34"을 사용하였다. 논문에서는 실험의 객관적인 검증을 위해 비밀키와 평문은 AES 표준 문서인 FIPS 197의 테스트 벡터 [4]에서 제시한 것을 그대로 사용하였다.

AES의 전체 수행시간은 약 6.9ms이며 [그림 5]

의 (b)는 마지막 라운드를 확대한 그림이다. 마지막 라운드의 라운드 키를 더하는 반복 연산에 오류를 주입한 결과 [그림 6]과 같은 전력 파형과 [그림 7]과 같은 결과를 얻을 수 있었다.

[그림 6]에서 보면 정상인 경우 마지막 라운드의 AddRoundKey는 75μs정도 걸렸지만 오류가 발생했을 경우 약 5.6μs 정도 소요됨을 확인할 수 있다.

다음으로 정상 암호문과 실제 오류가 주입된 오류 암호문을 비교해 보자. [그림 7]은 정상적인 암호문을 수행하다 오류가 주입된 후 출력된 오류 암호문을 나타낸 것이다. 실험에서는 암호 연산을 반복하는 과정에서 오류를 주입하였으며 [그림 7]의 2, 3, 그리고 5번째 라인의 결과가 오류가 주입되었을 때의 오류 암호문이다.

그림에서 보는 바와 같이 정상 암호문과 오류 암호문은 첫 번째 바이트를 제외하고 모두 다름을 볼 수 있다. 따라서 이 두 결과를 XOR하면 첫번째 바이트 0xd0를 제외한 120비트의 10라운드 키를 찾을 수 있다.

```
39 25 84 1D 02 DC 09 FB DC 11 85 97 19 6A 0B 32
39 31 7D B5 CB 32 2C 72 3D 2E 89 5F AF 09 07 94
39 31 7D B5 CB 32 2C 72 3D 2E 89 5F AF 09 07 94
39 25 84 1D 02 DC 09 FB DC 11 85 97 19 6A 0B 32
39 31 7D B5 CB 32 2C 72 3D 2E 89 5F AF 09 07 94
39 25 84 1D 02 DC 09 FB DC 11 85 97 19 6A 0B 32
39 25 84 1D 02 DC 09 FB DC 11 85 97 19 6A 0B 32
39 25 84 1D 02 DC 09 FB DC 11 85 97 19 6A 0B 32
```

[그림 7] 정상 암호문과 오류 암호문

정상 암호문 ⊕ 오류 암호문 =
0x 39 25 84 1D 02 DC 09 FB DC 11 85 97 19 6A 0B 32 ⊕
0x 39 31 7D B5 CB 32 2C 72 3D 2E 89 5F AF 09 07 94
= 0x 00 14 f9 a8 c9 ee 25 89 e1 3f 0c c8 b6 63 0c a6

오류가 주입되는 이유를 알아보기 위해 [그림 3]의 의사 코드에 해당되는 어셈블리 코드를 분석해 보았다. [그림 3]의 for문을 어셈블리어로 구현된 것이 [그림 8]이다. 어셈블리어의 전반부는 인덱스 i를 초기화시키고 조건문($i<16$)을 수행한다. 그 후 라운드 키와 state 값을 XOR한 후 후반부에서는 i값을 1만큼 증가시킨 후 다시 조건문을 검사하는 곳으로 복귀한다.

[그림 8]의 어셈블리 코드에서 오류가 발생하여 for문을 벗어나는 경우는 인덱스(index)에 대한 로드 오류(LDD 명령어)나 저장 오류(STD 명령어), 갑작스런 증감(ADIW 명령어)을 예상할 수 있으며 비교 단계의 오류(CPI 명령어) 그리고 분기 구문의 생략(RJMP 명령어) 등도 고려할 수 있다. 실험에서는 위 어셈블리 명령어 중 어느 명령어에 의한 오류인지 구별하기 위해 어셈블리 명령어의 순서와 클럭 개수에 기반하여 정밀도를 높여 오류 위치를 분석해 보았다. 잡음 발생, 트리거 지터링 (jittering), 레이저 주입 장치 자체 지연 시간 등의 영향으로 정확한 공격 대상 지점을 찾아 옷하는 결과를 얻을 수 없었기 때문에, 어셈블리 코드 분석을 통하여 대상이 되는 연산 부근에서 레이저 주입 시간을 변화시켜 가면서 수십 번 반복 주입하여 원하는 오류 출력을 얻을 수 있었다.

논문에서는 비교 단계에서 오류가 일어날 경우, 이어지는 분기 명령으로 for문을 종료할 수 있으므로, 공격 실험에서는 [그림 8]의 CPI 명령어를 대상으로 하여 비교 단계에서 오류를 유도하였다. 여러 번의 시행착오를 거친 실험 결과, 공격 대상이 되는 CPI 비교 연산에 오류가 주입하고 출력되는 오류 값을 분석함으로써 한 번의 루프 연산만을 수행하고 프로그램을 마치는 결과를 확인할 수 있었다. 그리고 이 오류 암호문을 정상 암호문과 XOR 함으로써 120비트의 10라운드 비밀키를 직접 구할 수 있었으며 나머지 8비트 키도 전탐색을 통해 구할 수 있었다.

```
   for(i=0 ; i<16 ; i++){
STD       Y+7,R1  // for문의 초기화 (i=0 수행)
STD       Y+8,R1
LDD       R24,Y+7
LDD       R25,Y+8
CPI       R24,0x10 //for문의 조건 검사 (i<16 확인)
CPC       R25,R1
BRGE      PC+0x24  //for문 끝냄 (i≥16 경우)

   state[i]=state[i] XOR b[i]
   (어셈블리어 생략)
   ...

LDD       R24,Y+7 // for문의 끝부분 (i++ 수행)
LDD       R25,Y+8
ADIW      R24,0x01
STD       Y+7,R24
STD       Y+8,R25
RJMP      PC-0x0027 // for문의 비교부분으로 되돌림
```

[그림 8] for 문에 대한 어셈블리 코드

V. 결 론

오류 주입 공격은 AES와 같은 암호 알고리즘을 수행하는 동안 데이터에 대해 직접적인 변형을 가져오는 공격이 대부분이었다. 오류 주입을 통해 얻은 오류 값을 이용하여 공격자는 암호 칩에 내장된 비밀키를 찾아낼 수 있었다. 본 논문에서는 데이터가 아닌 프로그램 코드에 오류를 주입하여 잘못된 연산을 유도하는 오류 주입공격을 제안하였다. 그리고 실제 오류 주입 공격의 가능성을 검증하기 위해 상용 마이크로프로세서에 AES를 구현하고 레이저 빔을 이용하여 오류를 주입하였다. 그 결과 한 번의 오류 주입 공격으로도 128비트 비밀키를 모두 찾아낼 수 있음을 확인하였다. 따라서 암호용 칩 개발자들은 오류 주입 공격에 대비한 물리적 보완 대책을 강구하는 것은 물론 데이터에 대한 마스킹과 같은 소프트웨어적인 대응책을 강구할 필요가 있다.

참고문헌

[1] D. Boneh, R. DeMillo, and R. Lipton, "On the Importance of Checking Cryptographic Protocols for Faults," EUROCRYPTO'97, LNCS 1233, pp. 37-51, 1997.

[2] E. Biham and A. Shamir, "Differential Fault Analysis of Secret Key Cryptosystems," CRYPTO'97, LNCS 1294, pp. 513-525, 1997.

[3] National Institute of Standards and Technology, "Advanced Encryption Standards," NIST FIPS PUB 197, 2001.

[4] G. Piret and J. Quisquater, "A differential fault attack technique against SPN structures, with application to the AES and KHAZAD," CHES'03, LNCS 2779, pp. 77 - 88, 2003.

[5] C. Giraud, "DFA on AES," Advanced Encryption Standard-AES'04, LNCS 3373, pp. 27 - 41, 2005.

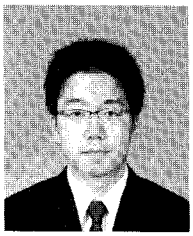
[6] C. Kim and J. Quisquater, "New Differential Fault Analysis on AES Key Schedule: Two Faults are enough," CARDIS'08, LNCS 5189, pp. 48-60, 2008.

[7] H. Choukri and M. Tunstall, "Round reduction using faults," FDTC'05, pp. 13-24. 2005,

[8] 박제훈, 문상재, 하재철, "CRT-RSA 암호시스템에 대한 광학적 오류 주입 공격의 실험적 연구," 한국정보보호학회논문지, 19(3), pp. 51-60, 2009년 6월.

[9] Atmel사 홈페이지, http://www.atmel. com/atmel/acrobat/ doc2467.pdf

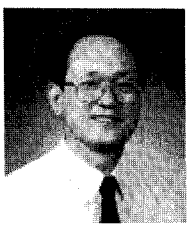
〈著者紹介〉



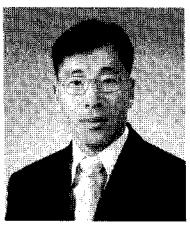
박 제 훈 (JeaHoon Park) 학생회원
2004년 2월: 경북대학교 전자·전기공학부 졸업
2006년 2월: 경북대학교 전자공학과 석사
2006년 3월~현재: 경북대학교 전자공학과 박사과정
〈관심분야〉 정보보호, 네트워크 보안, 스마트카드 보안

배 기 석 (KiSeok Bae) 학생회원
2006년 8월: 경북대학교 전자·전기공학부 졸업
2008년 8월: 경북대학교 전자공학과 석사
2009년 3월~현재: 경북대학교 전자공학과 박사과정
〈관심분야〉 정보보호, 네트워크 보안, 스마트카드 보안

오 두 환 (Doo Hwan Oh) 학생회원
2010년 2월: 호서대학교 정보보호학과 학사
2010년 3월~현재: 호서대학교 정보보호학과 석사과정
〈관심분야〉 암호 알고리즘, 하드웨어 보안



문 상 재 (SangJae Moon) 종신회원
1972년 2월: 서울대학교 공업교육(전자전공)과 학사
1974년 2월: 서울대학교 전자공학과 석사
1984년 6월: 미국 UCLA 전기공학과 박사
1984년 7월~1985년 6월: UCLA Postdoctor 근무
1984년 7월~1985년 6월: 미국 OMNET 컨설턴트
1997년 9월~1998년 8월: 경북대학교 전자전기공학부 학부장
1974년 12월~현재: 경북대학교 IT대학 전자공학과 교수
2000년 8월~현재: 경북대학교 이동네트워크 정보보호기술 연구센터장
2002년 2월~현재: 한국정보보호학회 명예회장
〈관심분야〉 정보보호, 디지털 통신, 이동 네트워크



하 재 철 (JaeCheol Ha) 종신회원
1989년 2월: 경북대학교 전자공학과 학사
1993년 8월: 경북대학교 전자공학과 석사
1998년 2월: 경북대학교 전자공학과 박사
1998년 3월~2007년 2월: 나사렛대학교 정보통신학과 부교수
2006년 7월~2006년 12월: QUT in Australia 연구 교수
2007년 3월~현재: 호서대학교 정보호호학과 부교수
2002년 3월~현재: 한국정보보호학회 이사
2009년 1월~현재: 한국산학기술학회 이사
〈관심분야〉 정보보호, 네트워크 보안, 스마트카드 보안