http://dx.doi.org/10.3745/KIPSTC.2012.19C.3.179

SHA-3 최종 라운드 후보 Skein에 대한 부채널 공격 방법

박 애 선[†] · 박 종 연[††] · 한 동 국[†††] · 이 옥 연[††††]

요 약

NIST(National Institute of Standards and Technology)는 SHA-2의 대체 알고리즘 부재로 SHA-3 개발 프로젝트를 진행 되고 있는 중 이다. 2010년 최종 라운드 후보 5개가 발표되었고, SHA-3 최종 라운드 5개의 후보에 대한 부채널 공격 시나리오가 제안되었다. 본 논문에서는 SHA-3 최종 라운드 후보 중 Skein에 대한 부채널 공격 시나리오를 32비트 레지스터를 사용하는 ARM Chip을 이용하여, 8 비트의 블록단위로 Divide and Conquer 분석이 가능함을 실험을 통해 증명한다. 9700개의 파형으로 128비트 키의 모든 비트를 찾을 수 있음을 실험으로 검증하였다.

키워드 : SHA-3, Skein 해쉬함수, 부채널 공격, HMAC

Side-channel Attack on the Final Round SHA-3 Candidate Skein

Aesun Park[†] · Jong-Yeon Park[††] · Dong-Guk Han[†††] · Okyeon Yi[††††]

ABSTRACT

Due to the absence of an alternative algorithm SHA-2, NIST (National Institute of Standards and Technology) is proceeding to development project of SHA-3. NIST announced five candidates of the final round at the end of 2010. Side-channel attack scenarios of five candidates for SHA-3 final round have been proposed. In this paper, we prove the possibility of the analysis against 32-bit modular addition by 8-bit blocks from our experiment on ARM chip board with a register size of 32-bit. In total we required 9700 power traces to successfully recover the 128-bit secret key for the attack against.

Keywords : SHA-3, Skein Hash Function, Side Channel Attack, HMAC

1. 서 론

컴퓨터 하드웨어 기술의 발전으로 최근 스마트폰, 스마트카드, USIM, 전자 여권 등의 다양한 디바이스의 사용이 급증하고 있다. 이러한 디바이스는 사용자의 프라이버시를 보호하기 위해 대칭키 암호, 공개키 암호, 해쉬함수 등의 인증된 암호 알고리즘을 통해 중요 데이터를 암호화 및 인증하여 저장한다. 이러한 암호 알고리즘은 통계적, 대수적 분석 등 이론적으로 안전성이 검증 되어 사용되고 있지만 실제 디바이스가 구동되는 동안 암호 연산의 시간, 소비 전력 및 전자파 등의 부채널 정보가 발생되는데, 이러한 부채널 정보를 이용하여 비밀 데이터를 분석하는 공격기법을 부채널 분석(Side Channel Analysis, SCA) 공격이라 한다[1].

부채널 분석 공격은 다양한 방법이 존재하고, 그중 가장 강력한 공격 방법으로 알려진 전력 분석 공격은 Kocher등에 의해 소개된 방법으로 암호 장치에서 연산을 수행할 때 소모되는 전력을 측정하여 관찰함으로 비밀 정보를 알아내는 공격 방법이다[2]. 전력 분석 공격은 단 한 번의 연산 수행에 대한 전력 소모량을 관찰하여 키 값을 유도해 내는 단순 전력 분석(Simple Power Analysis, SPA) 공격과 비밀키에 대해 여러 번의 알고리즘 수행을 거쳐 수집된 전력 소모량 측정치들 사이의 통계적 특성을 이용해 키를 찾아내는 차분 전력 분석(Differential Power Analysis, DPA), 상관 전력 분석(Correlation Power Analysis, CPA)로 나눌 수 있으며 그중 차분 전력 분석과 상관 전력 분석은 소비되는 전력을 분석하는 방법으로서 강력한 분석 기법이다.

해쉬함수는 임의의 길이의 이진 문자열을 고정된 길이의 문자열로 매칭시키는 단방향 함수로 데이터에 대한 무결성을 점검하거나 사용자에 대한 인증을 위한 용도로 사용된다. 해쉬함수에 비밀 키를 사용하지 않는 경우 메시지와 해쉬값을 함께 저장할 경우 고의적인 공격자는 메시지뿐만 아

※ 이 논문은 2011년도 정부(교육과학기술부)의 재원으로 한국연구재단의 기초연구사업 지원을 받아 수행된 것임(2011-0026354).
† 준 회 원 : 국민대학교 수학과 석사과정
†† 준 회 원 : 국민대학교 수학과 이학석사
††† 정 회 원 : 국민대학교 수학과 조교수
†††† 정 회 원 : 국민대학교 수학과 교수
논문접수 : 2012년 2월 27일
수 정 일 : 1차 2012년 5월 14일
심사완료 : 2012년 5월 30일

니라 해쉬값도 함께 변경할 수 있기 때문에 무결성을 보장하기 어렵다. 안전한 해쉬값 생성을 위해 해쉬함수에도 일반적인 블록 암호와 같이 비밀 키를 사용하는데 이처럼 비밀 키를 사용하여 계산되는 해쉬값을 메시지 인증 코드(MAC, Message Authentication Code)라 한다. 특히 HMAC 구조에서 해쉬함수는 메시지 인증 코드를 구축하는데 매우 유용하다.

SHA-1에 대한 충돌쌍 공격이 발표된 이래로, NIST에서는 신규 해쉬 알고리즘 SHA-3 개발을 목표로 "SHA-3 개발 프로젝트(cryptographic hash project)"를 진행 중에 있다. 2010년 5개의 SHA-3 최종 라운드 후보 BLAKE, Grøstl, JH, Keccak, Skein를 발표했으며, 최근 최종 라운드 후보들에 대한 부채널 공격 시나리오가 제안되었다[3,7].

Zohner 등은 Skein에 대한 부채널 공격 시나리오를 제시한 후 8비트 레지스터를 사용하는 ATMega-256-1 마이크로 컨트롤러를 사용하여 제안한 공격 시나리오를 실험을 통해 증명하였다[7]. 8비트 레지스터를 사용하는 디바이스에서 8비트 분석은 캐리비트에 의해 영향을 받지 않으므로 자연스러운 연구 결과이지만, 8비트 보다 큰 레지스터를 사용하는 디바이스에서 Zohner 등[7]이 제안한 공격시나리오의 연구는 아직 없었다. 최근 스마트 폰, 스마트 태블릿 등에 사용되는 디바이스는 32비트 레지스터를 사용하는 것으로 본 논문에서는 캐리비트를 고려한 Divide and Conquer 분석을 이용하여 레지스터가 큰 디바이스에서도 공격시나리오를 적용 할 수 있음을 보인다. 32비트 레지스터를 사용하는 ARM Chip 보드의 덧셈연산 소비 전력과 제안된 Skein-MAC 공격방법을 이용하여 분석이 가능함을 실험을 통해 검증하였다.

본 논문의 구성은 다음과 같다. 우선 2장에서는 CPA 공격법을 소개하고, 3장에서는 Skein 해쉬함수를 살펴본다. 4장에서는 Skein-MAC에 대한 전력 분석 방법과 32비트를 8비트 블록으로 나누어 분석하는 Divide and Conquer 분석 흐름을 설명하고, 5장에서는 전력 분석 방법을 이용한 실험 결과를 다룬다. 마지막으로 6장에서 논문을 마무리 짓는다.

2. CPA 공격법

2.1 CPA

대부분의 부채널 분석은 하나의 전력 파형으로는 분석이 불가능하며 다수의 소비 전력을 수집해 통계치를 활용하여 키를 찾아낸다. CPA의 구현은 다음의 두 단계로 나눌 수 있다. 먼저 데이터 수집 단계로 디바이스가 암호학적 연산을 실행할 때 소비되는 전력을 표본화하여 그 데이터를 수집한다. 그 후, 데이터 분석 단계로 표본화한 데이터를 통계적인 방법으로 분석한다. 수집된 소모전력 데이터를 분석하기 위해서 먼저 정확한 비밀 키가 입력되었을 때 그 비밀 키와의 반응을 알 수 있는 선택함수를 정해야 한다. 이 선택함수는 특정 비트(bit)나 바이트(byte)의 해밍 웨이트(hamming weight , HW) 또는 해밍 디스턴스(hamming distance, HD)를 조사하여 수집한 데이터를 분류하는 함수이다. 이 함수로 데이터를 분류한 후 가능한 비밀 키의 그룹에서 키를 추측하여 비밀 키를 알아낸다.

CPA는 전력 파형에서 각 포인트들이 가지고 있는 상관관계에 주목한다. CPA는 공격비트를 추측하여 얻은 중간 결과 값을 디바이스의 전력 소비 모델에 따라 변환한 후, 이것과 실제 디바이스로부터 얻은 소비 전력과의 상관관계를 계산하는 공격 방법으로 계산을 통해 얻은 결과와 실제 측정을 통해 얻은 파형의 상관관계를 구하여 높은 상관성을 갖는 키 값을 키로 선택한다. CPA는 연산이 계산되는 위치뿐만 아니라, 비밀 정보인 실제로 연산되는 키의 값도 알아낸다. 전체적인 CPA의 흐름은 다음과 같이 정리된다.

Step 1. 공격하는 장치에 S개의 랜덤한 메시지를 입력하고 알고리즘연산에 대한 파형을 얻는다.

Step 2. L개의 랜덤한 메시지에 (Step 1.에서의 메시지와 동일)대하여 시뮬레이션을 수행하여 추측한 비트 값에 대한 중간 결과 값을 얻는다.

Step 3. 중간 결과 값을 장치의 전력 소모 모델(HW 모델 혹은 HD 모델)로 변환한다.

Step 4. Step 3.에서 계산을 통하여 얻은 결과에 Step 1.에서 실제 측정을 통하여 얻은 파형의 상관관계를 구한다.

Step 5. 높은 상관관계를 보이는 추측을 옳은 추측으로 간주한다.

2.2 해밍 웨이트 모델과 해밍 디스턴스 모델

d비트의 문자열 $M = (m_0 m_1 m_2 \cdots m_{d-1})_{(2)}$, $m_i \in \{0, 1\}$에 대하여 해밍 웨이트는 1의 개수이다. 즉, 다음과 같이 표현되어진다.

$$H(M) = \sum_{i=0}^{d-1} m_i.$$

만일 M이 각각의 독립적인 m_i에 대하여 균등한 분포의 확률 변수를 갖는다면, 즉 0 또는 1이 동일한 확률로 나온다면 $H(M)$은 $\frac{d}{2}$를 평균으로, $\frac{d}{4}$를 분산으로 갖게 된다.

M과 같은 비트의 문자열 M'에 대하여 M'에서 M으로 바뀌었을 때 1에서 0 또는 0에서 1로 바뀐 비트의 수를 M과 M'사이의 해밍 디스턴스라 하고, 다음과 같은 식으로 표현되어진다.

$$H(M \oplus M').$$

2.3 상관계수(Correlation Coefficient)

상관계수란 $X = \{X_1, X_2, \cdots, X_s\}$, $Y = \{Y_1, Y_2, \cdots, Y_s\}$ 로 이루어진 두 분포 X, Y에 대한 연관성을 판단하는 척도로 다음과 같은 연산을 통해 계산되어 진다. 상관계수는 두 분포의 선형성을 의미하며, 두 분포가 $Y = aX + b$의 형태와

유사한 관계로 양의 상관관계를 가지게 되면 상관계수는 1에, 음의 상관관계가 있으면 -1에 가까운 값을 갖게 되고 두 분포간의 연관성이 없고 독립적일수록 그 값은 0에 가까워진다.

$$corr(X,Y)=\frac{E(XY)-E(X)E(Y)}{\sqrt{VAR(X)VAR(Y)}}$$
$$=\frac{\sum_{i=1}^{s}(X_i-E(X))(Y_i-E(Y))}{\sqrt{\sum_{i=1}^{s}(X_i-E(X))^2\sum_{i=1}^{s}(Y_i-E(Y))^2}}$$

3. Skein

이 절에서는 Skein의 알고리즘에 대해 설명한다. Skein은 메시지를 UBI(Unique Block Interation) 체이닝 모드를 이용해 압축한다[4]. 본 논문에서는 Skein-512에 대하여 논하고, 가능한 경우 모든 salt(난수)값 입력은 0으로 간주한다.

3.1 Skein-MAC

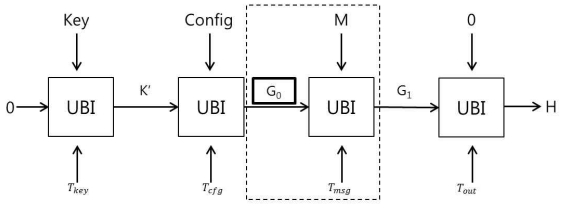
인증을 위한 해쉬함수 사용의 표준방법은 HMAC 구조를 사용하는 것이다[5,6]. Skein도 HMAC을 사용할 수 있으나 본 논문에서의 공격방법은 Skein-MAC 또는 HMAC을 적용한 알고리즘 모두에 일반적으로 적용 가능하므로 Skein-MAC 구조만 간단히 설명한다.

HMAC은 참고문헌 [5]를 참고한다.

Skein은 UBI 체이닝 모드를 이용하여 메시지를 압축한다. Skein-MAC은 메시지 M을 입력받아 아래와 같은 과정으로 MAC값을 생성한다. (그림 1)은 이 과정의 흐름도를 묘사하고 있다.

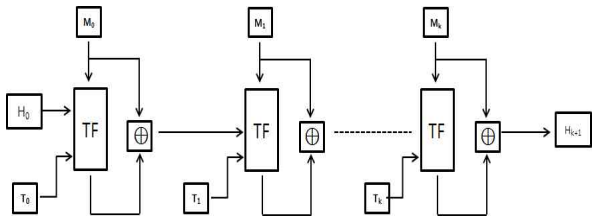
$$K' = UBI(0, Key, T_{key})$$
$$G_0 = UBI(K', Config, T_{cfg})$$
$$G_1 = UBI(G_0, M, T_{msg})$$
$$H = UBI(G_1, 0, T_{out}) = Output(G_1, N_0)$$

(여기에서 N_0 : $Skein$의 출력 비트수,
$T_{key}, T_{cfg}, T_{msg}, T_{out}$: $tweak$(상수),
$M = M_0 \| M_1 \| M_2 \| \cdots \| M_k$, $M_i \in \{0,1\}^{512}$, $\|$는 연접을 나타낸다.)



(그림 1) Skein-MAC 흐름도

3.2 UBI 체이닝 모드 (UBI Chaining Mode)



(그림 2) UBI 체이닝 모드

(그림 2)는 UBI 체이닝 모드를 보여주고 있는 그림으로, (그림 1)의 점선 부분을 자세히 보여준다. $H_0 = G_0$ 로 Threefish 블록암호(TF)의 키 역할을 하게 되어 공격자가 G_0를 복원하면 메시지 다이제스트 값을 만들 수 있기 때문에 해쉬값 계산이 가능하다. 즉, G_0를 복원하면 키(Key)를 모르고도 원하는 메시지에 대한 MAC값을 생성할 수 있게 된다.

Skein은 UBI 체이닝 모드를 식 (1)과 같이 사용하여 메시지를 압축한다.

$$H_{i+1} = TF[H_i, T_i, M_i] \oplus M_i \ , (0 \le i \le k) \qquad (1)$$

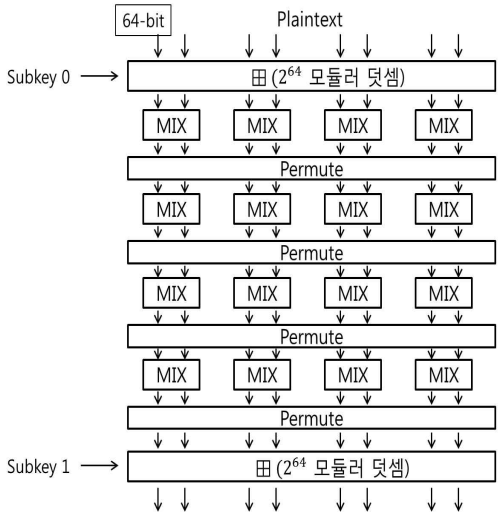
식 (1)은 H_{i+1}는 $i+1$번째 체이닝 변수로 Threefish 블록 암호에 i번째 체이닝 변수(H_i), Tweak(T_i) 값, 메시지(M_i)를 넣어 암호화를 시킨 후, 메시지(M_i)와 Exclusive or(XOR) 시행 후 얻을 수 있다. 여기에서의 Tweak 값은 알려진 상수이다.

3.3 Threefish-512비트 구조

Threefish 블록 암호는 키, tweak와 평문 세 개의 변수가 입력된다. (그림 3)은 Threefish-512 비트의 구조를 보여주며, Threefish는 64 비트 씩 연산되어지는데 평문 512 비트와 부분키는 모듈러 2^{64} 덧셈으로 계산된다.

키 $K(=k_0,k_1,\cdots,k_7)$와 평문 메시지 $M(=m_0,m_1,\cdots,m_7)$ 그리고 tweak $T(=t_0,t_1)$ 값을 입력으로 하고, T는 키 스케줄 사용 시 이용된다. 여기에서 k_i, m_i, t_i는 모두 64 비트이다.

Threefish는 다음과 같은 과정으로 진행된다. 첫째, 키 스케줄에 의해 부분 키들이 생성되고 $v_{0,i}$을 평문 m_i로 대체한다. 그 다음 각 라운드에 대해 내부 상태 $v_{d,i}$는 다음과 같은 흐름으로 업데이트 되는데, 만약 $d \equiv 0$ mod4 이면 $v_{d,i}$는 부분키($k_{d/4,i}$)와 모듈러 2^{64} 덧셈 되어 지고, 나머지는 그대로 유지된다. 다음 단계는 MIX함수와 $\pi(i)$ 함수(permutation)를 거치는 연산을 72회 시행한다. (단, 여기에서 d라운드 후의 64 비트 8개로 이루어진 내부 상태를 $v_{d,i}$, s번째 부분키를 $k_{s,i}$라 표시했으며, i는 0부터 7까지의 정수를 의미한다.)



(그림 3) Threefish-512 비트 구조

4. Skein 전력 분석 방법

4.1 Skein-MAC 공격 시나리오

앞서 소개한 바와 같이 (그림 1)의 G_0는 Threefish 알고리즘을 수행 할 때, 키로 작용하며, G_0를 알아내면 메시지 다이제스트 값을 만들 수 있기 때문에 공격자는 해쉬값을 계산할 수 있다. 그러므로 Threefish의 키 G_0를 찾는 공격 시나리오를 설명한다.

G_0와 M_0는 512 비트로 다음과 같이 64비트 8개로 다음과 같이 표현한다.

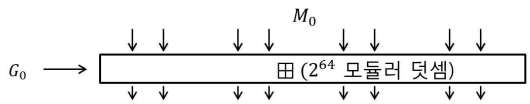
$$G_0 = g_{0,0} \| g_{0,1} \| \cdots \| g_{0,7},\ g_{0,j} \in \{0,1\}^{64},$$

$$M_0 = m_{0,0} \| m_{0,1} \| \cdots \| m_{0,7},\ m_{0,j} \in \{0,1\}^{64}.$$

아래의 (그림 4)에서 볼 수 있듯이 알고 있는 정보와 비밀 정보가 섞이는 곳은 (그림 3)의 Subkey 0 와 평문이 모듈러 2^{64} 덧셈 연산 되어지는 부분이다. 이곳은 (그림 4)에서 G_0와 M_0가 연산되어지는 부분으로 우리는 이곳에서 비밀 정보 G_0를 찾기 위해 선택함수

$$e_{0,i} = (m_{0,i} + g_{0,i}) \bmod 2^{64},\ (0 \le i \le 7) \quad (2)$$
$$(e_{0,i}, m_{0,i}, g_{0,i} \in \{0,1\}^{64})$$

를 사용한다.



(그림 4) 공격 지점

정확한 중간 값 계산을 위해서는 2^{64}개의 키를 추측하여 중간 값을 연산해야 하지만, 키 공간이 너무 크므로 현실적인 분석 방법이 되지 못한다. 하지만 덧셈 연산의 특성상, 블록 단위의 중간 값 연산이 가능하므로, 일정 크기의 블록 단위로 분석을 수행하는 Divide and Conquer 분석이 가능하다. 이를 위해 다음 수식 (3)과 같이 8비트 중간 값 계산 모델을 정의한다.

$$e_{0,i,j} = (m_{(0,i,j)} + g_{(0,i,j)}) \bmod 2^8,\ (0 \le i,j \le 7) \quad (3)$$

(여기에서, $e_{(0,i,j)}, m_{(0,i,j)}, g_{(0,i,j)}$ $(\in \{0,1\}^8)$는 각각 $e_{0,i}, m_{0,i}, g_{0,i}$의 하위 (j)-번째 8비트 블록을 나타낸다.)

식 (3)은 64비트를 8비트 블록 단위로 중간 값 계산한 식으로 j는 0부터 7까지 값을 가지게 된다. 이러한 방법으로 분석이 가능한 이유는 각각의 독립된 블록의 키 바이트는 덧셈 이후에 저장되는 레지스터에 독립적으로 영향을 주기 때문이다.

4.2 캐리비트를 고려한 Divide and Conquer 분석

8비트 레지스터에서 식 (3)의 중간 값 계산 모델이 합리적인 이유는 덧셈과정에서의 캐리비트가 없기 때문이다. 그러나 32비트 레지스터 사용에서 8비트 블록 단위로 공격할 경우 캐리비트의 문제를 배제할 수 없다. 본 논문의 실험은 32비트 레지스터를 사용하는 ARM Chip 보드를 통해 획득한 파형으로 8비트 블록 단위로 중간 값을 계산하여 실험하였다. 본 절에서는 캐리비트를 고려한 Divide and Conquer 분석의 과정을 설명한다. 실험의 과정은 다음의 Algorithm.1을 참고한다.

Algorithm.1은 평문 $P_0, P_1, \cdots, P_{n-1}$ $(P_i \in \{0,1\}^{32})$ n개와 전력 파형 $C_0, C_1, \cdots, C_{n-1}$ $(C_i \in \{0,1\}^{32})$을 입력 값으로 32비트 키 $K = k_3k_2k_1k_0$ $(k_i \in \{0,1\}^8)$을 얻는 과정을 나타낸다. 128비트를 모두 찾기 위해서는 Algorithm.1을 4번 시행하면 된다. k_i는 8비트이기 때문에 k_i의 후보는 0부터 255까지 256개이다.

32비트의 하위 8비트 k_0를 얻는 과정에서는 캐리비트를 고려해 줄 필요가 없다. 그러나 k_1, k_2, k_3는 그전 하위 8비트의 값에 따라 영향을 받기 때문에 캐리비트를 고려해야 한다. Algorithm.1의 5, 6에서 볼 수 있듯이 확정되어진 k_t $(t = 0, 1, 2)$를 이용하여 평문이 변함에 따라 캐리비트를 고려한다. H는 k_i를 0부터 255까지 선택하고 이 값을 입력 평문과 조합해서 얻은 2^{32}모듈러 덧셈 시 소모될 것으로 예상되는 데이터의 헤밍 웨이트로 구성된 가상의 전력량에 대한 행렬이다. 즉, 입력 평문 P_i에 대한 m포인트 전력 파형 $C_i (= c_{i0} \| c_{i1} \| \cdots \| c_{i(m-1)})$에서 모듈러 덧

셈이 이루어지는 시점의 전력을 c_{iw}라 하고, 실제 키를 k_c라 한다면, 두 분포

$$\{H_{k_c0}, H_{k_c1}, \cdots, H_{k_c(n-1)}\}, \{c_{0w}, c_{1w}, \cdots, c_{(n-1)w}\}$$

는 상당한 유사성을 지닌다.

Algoritm.1에서 P_s $(0 \le s \le (n-1))$는

$P_s = P_{s,0} \parallel P_{s,1} \parallel P_{s,2} \parallel P_{s,3}$ $(P_{s,j} \in \{0,1\}^8)$를 나타낸다.

Algorithm.1 2^{32}모듈러 덧셈 Divide and Conquer CPA

Input : 전력 파형 ($C_0, C_1, \cdots, C_{n-1}$), 평문 ($P_0, P_1, \cdots, P_{n-1}$)
Output : $K = k_3 k_2 k_1 k_0$

```
for i = 0 to 3 do
  for k_i = 0 to 255 do
    for j = 0 to (n − 1) do
      if i=0 : H_{k_i,j} = HW(P_{j,i} + k_i (mod 2^8))
      else   : if P_{j,i−1} + k_t ≥ 2^8 then carry = 1 , else carry = 0
               H_{k_i,j} = HW(P_{j,i} + k_i + carry (mod 2^8))
    end for
    for u=0 to (m-1) do
      Cr_{k_i,u} = Corr({H_{k_i0}, H_{k_i1}, ..., H_{k_i(n−1)}}, {c_0w, c_1w, ..., c_(n−1)u})
    end for
  end for
  k_i = j (Cr_jw: C_r의 최대값)
end for
Return K
```

5. 실 험

5.1 실험 환경

<표 1>에서와 같이 실험 환경은 디지털 오실로스코프로 ARM7 Chip 보드를 이용 하여, 1GS/s의 샘플링 속도(sampling rate)로 32비트 연산으로 이루어진 소비 전력 10,000개를 수집하였다. 수집된 소비 전력은 128 비트 키를 8비트씩 추측하여 CPA분석을 하였다.

〈표 1〉 실험 환경

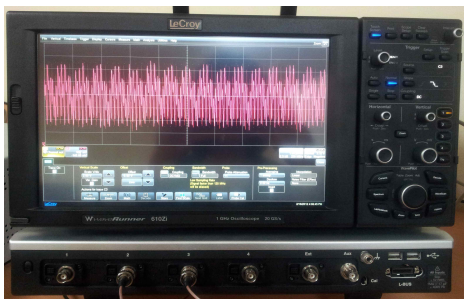
오실로스코프	디지털 오실로스코프
프로세서	ARM7 Chip 보드
연산비트	32 비트
샘플링 속도	1GS/sec
소비 전력 수	10,000
평문 비트 수	128비트

(그림 5)는 실험에 사용된 LeCroy사의 610zi 오실로스코프로, 각 보드에서 소모한 전력정보를 관찰하여 저장하는 장비이다.

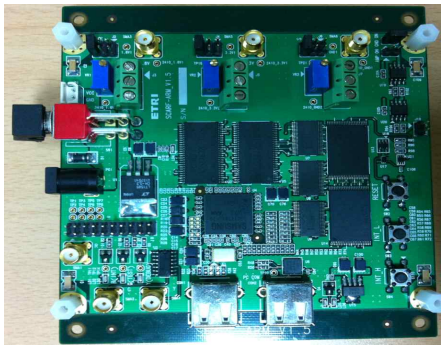
(그림 6)은 ARM Chip을 사용한 보드이다. 암호 알고리즘을 포팅(porting)하여 암호 알고리즘을 동작 시키는 장비이다. ARM Chip은 초전력 마이크로 칩으로 최근 스마트 폰, 스마트 태블릿 등에 사용되는 칩이다. 32비트 프로세서의 대표적인 칩으로 50 ~ 270MHz의 동작 주파수를 갖는다.

5.2 실험 결과

<표 2>는 8 비트씩 키를 추측 하였을 때 옳은 키가 최소



(그림 5) LeCroy 610zi 오실로스코프



(그림 6) ARM7 Chip 보드

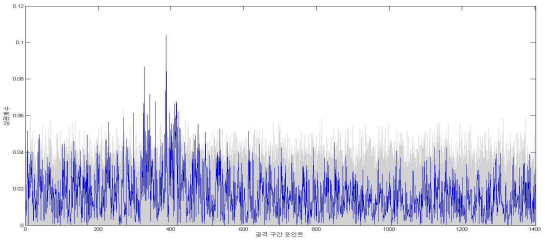
몇 개의 소비 전력을 가져야 하는지를 나타낸 것으로, <표 2>를 보면 32 비트 모듈로 덧셈 연산되어진 10,000개의 소비 전력으로 8비트씩 128비트의 상관계수 전력 분석을 하면 평문 9700개로 16개의 모든 키를 얻을 수 있다.

〈표 2〉 최소 평문수

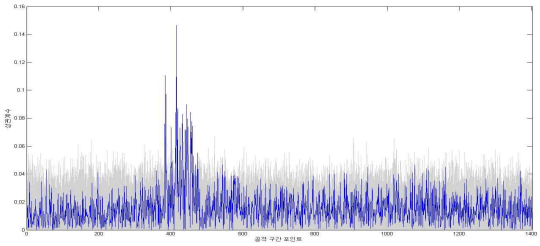
키 블록 순서	0	1	2	3	4	5	6	7
최소 평문수	1200	800	800	5200	5500	1200	5300	6000
키 블록 순서	8	9	10	11	12	13	14	15
최소 평문수	5700	7300	9700	8600	5000	6300	5800	9000

(그림 7)과 (그림 8)은 각 각 첫 번째와 세 번째 S-box의 8비트 키 후보에 대한 분석 결과를 중첩해서 나타낸 것이다. 짙은 색으로 나타난 피크는 옳은 키에 대한 높은 상관계수를 나타내는 구간으로써 덧셈 연산 결과가 저장되고 불려오는 부분이라는 것을 알 수 있다. 뒤에 옅은 색들은 옳은 키를 제외한 255개의 키 후보들의 상관계수를 나타낸다.

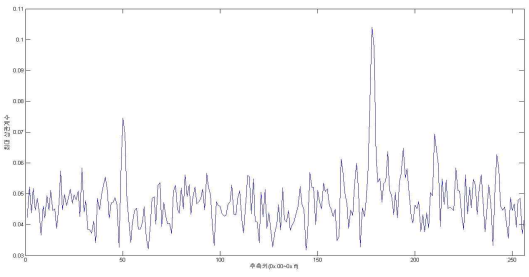
(그림 9)는 8 비트 추측키 256개의 가장 큰 상관계수를 비교하여 나타낸 것인데, 가장 높은 피크가 나타내는 것이 키라는 것을 확인 하였다. 주위에 다른 높은 피크들도 존재하는데 이것은 덧셈 연산이 수행되는 분석 지점의 특성상 필연적으로 나타나는 유사키의 형태이다. (가장 높은 피크의 상관계수를 두 번째 피크의 상관계수로 나눈 값이 1.5이상일 때 옳은 키로 간주하였다.)



(그림 7) 1st S-box의 옳은 키에 대한 상관계수



(그림 8) 3rd S-box의 옳은 키에 대한 상관계수



(그림 9) 1st S-box의 추측키에 대한 최대 상관계수

6. 결 론

본 논문에서는 SHA-3의 최종 라운드 후보 중 Skein의 키를 찾아내는 실제적인 공격 시나리오를 스마트 폰 또는 스마트 태블릿 등에 많이 사용되는 32비트 레지스터를 사용하는 ARM Chip 보드를 통해 수집된 소비 전력을 이용하여 Divide and Conquer 분석이 가능함을 실험을 통해 검증 하였다. 본 논문을 해쉬함수 또한 일반적인 블록암호의 경우처럼 부채널 공격에 대한 안전한 대응법이 같이 연구되어야 할 것이다.

향후 우리는 여러 가지 해쉬함수의 부채널 공격 방법뿐만 아니라 그에 대한 안전한 대응법 개발에 대해 더 연구해야 할 분야로 남아있다.

참 고 문 헌

[1] P. Kocher, J. Jaffe, and B. Jun, “Introduction to differential power analysis and related attacks” White Paper, Cryptography Research, 1998.
[2] P. Kocher, J. Jaffe, and B. Jun, “Differential power analysis” CRYPTO 1999, LNCS 1666, Springer-Verlag, pp.388-397, 1999.
[3] O. Benoît, T. Peyrin, “Side-Channel Analysis of Six SHA-3 Candidates” CHES 2010, LNCS 6225, Springer-Verlag, pp.140 - 157, 2010.
[4] N. Ferguson, S. Lucks, B. Schneier, D. Whiting, M. Bellare, T. Kohno, J. Callas, and J. Walker, The skein hash function family, http://www.schneier.com/skein.html, October.
[5] M. Bellare, R. Canetti and H. Krawczyk, “Keying hash functions for message authentication” CRYPTO '96, LNCS 1109, Springer-Verlag, pp.1-15, 1996.
[6] National Institute of Standards and Technology, “The Keyed-Hash Message Authentication Code (HMAC),” FIPS 198, 6 Mar, 2002.
[7] M. Zohner, M. Kasper, M. Stottinger, “Side channel analysis of the sha-3 finalists” Design, Automation & Test in Europe, DATE, 2012.

박 애 선

e-mail : aesons@kookmin.ac.kr
2011년 국민대학교 수학과(학사)
2011년~현 재 국민대학교 수학과 석사과정
관심분야 : 부채널 분석 및 대응법, 신호처리, 개인신원인증(PIV) 시스템 등

박 종 연

e-mail : flyfree@kookmin.ac.kr
2010년 국민대학교 수학과(학사)
2012년 국민대학교 수학과 이학석사
관심분야 : 부채널 분석 및 대응법, 화이트박스 암호, 암호 모듈 고속 구현, 무선 보안 등

한 동 국

e-mail : christa@kookmin.ac.kr
1999년 고려대학교 수학과(학사)
2002년 고려대학교 수학과(이학석사)
2005년 고려대학교 정보보호대학원 (공학박사)
2004년~2005년 일본 Kyushu Univ., 방문연구원
2005년~2006년 일본 Future Univ.-Hakodate, Post.Doc.
2006년~2009년 한국전자통신연구원 정보보호연구본부 선임연구원
2009년~현 재 국민대학교 수학과 조교수
관심분야 : 공개키 암호시스템 안전성 분석 및 고속 구현, 부채널 분석, RFID/USN 정보보호 기술 등

이 옥 연

e-mail : oyyi@kookmin.ac.kr
1988년 고려대학교 수학과(학사)
1990년 고려대학교 수학과(이학석사)
1996년 8월 University of Kentucky 이학박사
1999년 7월~2001년 8월 한국전자통신연구원 선임연구원
2000년 3월~2001년 8월 한국전자통신연구원 팀장
2001년 9월~현 재 국민대학교 수학과 교수
관심분야 : 스마트 그리드 보안, 무선보안, 4G보안, 스마트워크 보안 등