情報保護學會論文誌
第 22 卷 第 3 號, 2012. 6

오류 주입 공격에 안전한 전자서명 대응법*

김 태 원,[1†] 김 태 현,[1] 홍 석 희,[1] 박 영 호[2‡]
[1]고려대학교, [2]세종사이버대학교

A new digital signature scheme secure against fault attacks*

TaeWon Kim,[1†] TaeHyun Kim,[1] Seokhie Hong,[1] Young-Ho Park[2‡]
[1]Korea University, [2]Sejong Cyber University

요 약

오류 주입 공격은 공격자가 암호 알고리즘이 수행되는 동안 물리적으로 오류를 주입한 후, 그 결과를 분석하여 비밀 정보를 알아내는 강력한 부채널 분석 기법이다. 본 논문에서는 국제 표준 서명 알고리즘인 DSA (Digital Signature Algorithm)에 대하여 그 동안 제안되었던 오류 주입 공격과 대응법을 소개한 후, 기존 오류 주입 공격에 안전하다고 제안된 알고리즘의 취약점을 분석하였다. 또한 오류주입공격에 안전한 새로운 서명알고리즘을 제안한다. 제안하는 방법은 오류확산기법과 2개의 난수를 사용하여 서명 하도록 설계하였고, 현재까지 소개된 모든 오류 주입 공격에 안전하다.

ABSTRACT

Fault attacks are a powerful side channel analysis extracting secret information by analyzing the result after injecting faults physically during the implementation of a cryptographic algorithm. First, this paper analyses vulnerable points of existing Digital Signature Algorithm (DSA) schemes secure against fault attacks. Then we propose a new signature algorithm immune to all fault attacks. The proposed DSA scheme is designed to signature by using two nonce and an error diffusion method.

Keywords: DSA, Fault Attacks, Countermeasures

I. 서 론

이론적으로 안전한 암호 알고리즘도 실제 구현 단계에서 암호 설계자가 고려하지 못한 부가적인 정보의 노출로 인해 비밀 정보가 노출될 수 있음이 알려졌다. 부채널 공격 (side channel attack)[1]은 오류 주입 공격 (fault attack)[2,3], 시간 공격 (timing attack)[1,4], 전력 분석공격 (power analysis attack)[5,6,7] 등이 있다. 부채널 공격 중 가장 강력한 분석 기법은 오류 주입 공격이다. 오류 주입 공격은 1996년 Boneh 등에 의해 처음 소개되었고, 정보보호 디바이스에서 암호 알고리즘이 구동될 때 오류를 주입하여 비밀 정보를 찾아내는 공격 방법이다[3]. 이후 공개키 암호 알고리즘인 RSA 암호 알고리즘[1,8,9], ElGamal 서명, DSA(Digital Signature Algorithm)서명[10-17], 타원곡선 DSA 서명[20,21] 및 블록암호 알고리즘인 DES(Data Encryption Standard)와 AES(Advanced Encryption Standard)[18,19] 등을 대상으로 오류 주입 분석에 관한 연구 및 대응법 연구가 활발하게 진행되었다.

DSA 서명에 대한 오류 주입 공격은 Bao 등이 처음 제시하였고[11], 개인키의 한 비트 플립 오류(filp

접수일(2012년 3월 7일), 게재확정일(2012년 3월 16일)
* 본 연구는 2011년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업입니다. (No. 2011-0004395)
† 주저자, finitefield@korea.ac.kr
‡ 교신저자, youngho@sjcu.ac.kr

error)를 가정하여, 한 번의 오류 주입으로 개인키의 한 비트를 찾아낼 수 있다. 2004년 Giraud와 Knudsen은 바이트 단위의 공격으로 확장하였고, 2300개의 오류 서명을 통해 개인키를 찾을 수 있었다[14].

개인키에 대한 오류 주입 공격을 방어하기 위하여 Nikodem은 오류 확산 방법을 통해 안전한 DSA서명 방식[12,13]을 제안하였고, Bae 등은 비선형 함수인 역원 연산을 통해 1비트 오류가 확산되도록 하여 오류 주입 공격을 방어하는 대응법을 제안하였다[16].

DSA 서명 알고리즘에 대한 오류 주입 공격은 개인키 뿐만 아니라 서명 시 매 번 생성하는 난수에 대해서도 가능하다. 난수에 대한 오류 주입 공격법은 난수의 부분 정보를 알아냄으로써 최종적으로 개인키를 복구한다. Naccache등은 난수가 생성될 때 최하위 바이트를 0으로 강제리셋 시키는 오류를 주입하고, 27개의 오류 서명으로 개인키 1바이트를 추출하였다[15]. 또한 Schmidt는 ECDSA를 대상으로 난수에 대한 새로운 오류 주입 공격을 소개하였다[21]. 생성원과 난수를 이용해 스칼라 곱셈을 수행 할 때 특정 루프의 2배 연산과정을 생략하여 오류 서명을 출력하고, 이러한 서명 쌍을 모아 lattice 공격으로 개인키를 알아낸다. 이 공격법을 변형시켜 Jung 등은 DSA 서명 알고리즘의 취약점을 분석하였다[17].

본 논문에서는 기존에 제한된 오류 주입 공격에 대한 대응방법들의 취약점을 분석한 후, 새로운 DSA 서명 알고리즘을 제안한다. 제안하는 서명 알고리즘은 Nikodem의 대응법과 유사하게 서명 도중 오류 주입 여부를 검사하는 단계를 넣었고, 기존 서명 기법과는 다르게 난수를 2개 사용함으로써 모든 공격을 방어하도록 설계하였다. 제안 서명 알고리즘의 안정성은 다양한 오류 모델을 가정하여 증명하였다.

본 논문의 구성은 다음과 같다. 2장에서는 기존 오류 주입 공격과 대응법에 대해 살펴본다. 그리고 3장에서는 기존 대응법의 취약점을 분석하였다. 이런 취약점을 보안하여 모든 공격에 안전한 DSA 서명 알고리즘을 4장에 제안하였다. 마지막으로 5장에서 결론을 맺는다.

II. DSA 서명에 대한 오류 주입 공격 및 대응법 분석

본 장에서는 DSA 서명 알고리즘에 대한 설명과 오류 주입 공격 방법 및 기존의 대응 방법에 대해 살펴본다.

2.1 DSA 서명 알고리즘

현재 국제 전자서명 표준으로 사용되고 있는 DSA는 1991년 NIST(National Institute of Standard and Technology)에 발표되었다[10]. DSA는 ElGamal 서명과 마찬가지로 이산대수 문제에 안전성을 기반하고 있다. 전자 서명 알고리즘에 사용되는 파라미터들은 다음과 같다.

a. 파라미터 생성

$-p$: 512비트 이상의 소수.
$-q$: 160비트 소수로서 $p-1$의 약수.
$-g$: Z_p^*의 원소로 위수 q를 가지는 생성원.
$-h()$: 해쉬 함수.
$-a$: Z_p^*의 원소로서 사용자의 개인키.
$-y$: $y=g^a \bmod p$로 사용자의 공개 키.

b. 서명

메시지 m을 받은 서명자는 다음과 같은 과정으로 서명 값 (r,s)를 생성한다.
단계 1) 난수 $k(0<k<q)$ 선택.
단계 2) $r=(g^k \bmod p) \bmod q$ 계산.
단계 3) $s=k^{-1}(h(m)+ar) \bmod q$ 계산.

c. 서명검증

수신자는 메시지 m과 서명 값(r,s)를 다음 절차에 의해 검증한다.
단계 1) $w=s^{-1} \bmod q$를 계산.
단계 2) $v_1=g^{h(m)w} \bmod p$, $v_2=y^{rw} \bmod p$를 계산.
단계 3) $v_1 v_2 \bmod p \bmod q=r$인지 확인.

2.2 오류 주입 공격

2.2.1 개인키에 대한 오류

- Bit flip 오류

Bao 등이 제안한 공격 방법[11]에서 공격자는 DSA 서명 알고리즘이 수행되는 동안 개인키 a가 저장되어 있는 레지스터에 오류를 주입하여 개인키의 임의의 한 비트가 비트 플립(flip error)되었다고 가정하고, 다음과 같은 과정을 통해 개인키를 복구할 수

있다.

단계 1) 개인키 a의 i번째 비트에 오류 발생, 오류가 주입된 개인키는 $\bar{a}=a\pm 2^i$라 표현.

단계 2) 공격자는 오류가 주입된 개인키 $\bar{a}$를 이용하여 다음과 같이 오류 서명 값 $(r,\bar{s})$를 얻을 수 있다.

$$r=(g^k \bmod p) \bmod q$$
$$\bar{s}=k^{-1}(h(m)+\bar{a}r) \bmod q$$

단계 3) 오류 서명을 받은 공격자는 다음과 같이 T를 계산할 수 있다.

$$\bar{w}=\bar{s}^{-1} \bmod q$$
$$\bar{v_1}=g^{h(m)\bar{w}} \bmod p,\ \bar{v_2}=y^{r\bar{w}} \bmod p$$
$$T=\bar{v_1}\cdot\bar{v_2} \bmod p$$
$$=g^{h(m)\bar{w}}\cdot y^{r\bar{w}} \bmod p$$
$$=g^{(\bar{w}(h(m)+ar) \bmod q)} \bmod p$$

단계 4) 공격자는 검사 값 R_i를 계산한다.

$$R_i=(g^{r\bar{w}2^i}) \bmod p,\ (i=0,1,\cdots,t-1) \quad (1)$$

단계 5) 공격자는 다음 두 식을 확인해 봄으로써 오류가 발생했던 개인키의 한 비트를 찾을 수 있다.

$TR_i=r \bmod p \bmod q$ 이면 $a_i=0$

$T/R_i=r \bmod p \bmod q$ 이면 $a_i=1$

비트 플립 오류는 랜덤한 비트에 오류가 주입되므로 공격자는 서로 다른 i에 대해 반복적으로 단계 5를 수행함으로써 개인키 a의 i번재 비트를 찾을 수 있다. 이러한 공격이 가능한 이유는 만약 개인 a의 i번째 비트가 0에서 1로 바뀌었다면 r은 $TR_i=g^{\bar{w}(h(m)+r(a+2^i)) \bmod q} \bmod p$ 와 같아질 것이고 1에서 0으로 바뀌었다면 r은 $T/R_i=g^{\bar{w}(h(m)+r(a-2^i)) \bmod q} \bmod p$와 같아지기 때문이다.

- Random byte 오류

Giraud와 Kundsen〔14〕은 Bao 등의 공격 기법을 바이트 단위 오류 공격으로 확장하였다. 이 방법은 오류가 주입된 개인키 값과 주입되지 않은 개인키 값의 차분을 계산하고 이러한 과정을 여러 번 반복하여 키 후보가 될 수 있는 키 후보군을 전수조사가 가능한 수준까지 줄여 키를 찾았다. 이 오류 공격에서 필요한 오류 서명 쌍은 약 2300개였다. 이 방법은 $\bar{a}$를 $a+j\times 2^i,\ 0\le i\le 19,\ -255\le j\le 255$로 표현하여 Bao 등의 공격과 동일한 분석 과정을 거치므로 자세한 수식과 설명은 생략한다. (자세한 내용은 〔14〕참조)

2.2.2 난수에 대한 오류

- 바이트 리셋 오류

서명 시 매번 생성하는 난수의 부분 정보를 이용하여 개인키를 복구하는 기법이 Naccache 등에 의해 소개되었다〔15〕. 논문에서 난수가 생성되는 동안 glitch attack을 통해 난수의 최하위 몇 바이트를 0으로 만드는 오류 모델을 가정한다. 난수의 부분정보를 통해 개인키를 복구 하는 문제는 lattice 공격으로 해결된다〔22,23〕. 이 문제는 Boneh과 Venkatesan에 의해 hidden number problem 이라〔25〕 소개된 것과 매우 유사하다. 이 문제는 다음과 같이 간단히 설명한다.

$$s=\frac{h(m)+ar}{k} \bmod q$$

위의 식은 DSA 서명 중 서명 값 s를 생성하는 과정이다. 서명식에서 공격자는 $s,h(m),r$을 알 수 있다. 이 때 k의 정보 없이는 a를 알아낼 수 없으나, k의 부분정보를 알고 있는 s값을 충분히 많이 가지고 있다면 a값을 복구 할 수 있다. 이론적으로 160비트의 a를 복구하기 위해서 k의 l비트를 알고 있는 서명 값 s가 $160/l$개 필요하다. 예를 들면, 개인키를 추출하기 위해서 최하위 11개의 바이트에 대해 오류를 주입하면 2개의 서명 값이 필요하고, 최하위 1개의 바이트에 대해 오류를 주입하면 27개의 오류 서명 쌍이 필요하다고 제안하였다.

- 제곱 연산 과정을 생략하는 오류

Schmidt 등이 ECDSA를 대상으로 처음 제안하였고〔21〕, Jung 등이 이를 변형하여 DSA에 적용하

입력 : 난수 $k=(k_{t-1},\cdots,k_0)_2$
생성원 $g\in Z_p^*$
출력 : $g^k \bmod p$

1. $A=1$
2. for $i=t-1$ *downto* 0 do
3. $A=A\cdot A \bmod p$
4. if $(k_i=1)$ then
$A=A\cdot g \bmod p$
end if
end for
5. return A

〔그림 1〕 Left-to-Right 이진 지수승 알고리즘

였다[17]. Jung 등이 사용한 공격 가정은 서명 값 r을 생성 시 "[그림 1]"과 같이 Left-to-Right 이진 지수승을 수행하게 되는데 이때 $(t-i)$번째 제곱 연산을 생략하는 것이다. 즉, k에 대한 지수승 연산 시, $g^k \bmod p$, 연산 과정을 잘못하게 유도하여 정상적이지 않은 서명 값을 얻어낸다. 이 경우 난수 k의 i번째 이후 비트 값을 $k' = (k_i, \cdots, k_0)_2$로 표기한다. 공격을 통해 k'를 복구한 뒤 lattice 공격으로 개인키를 얻을 수 있다. k'을 얻는 과정을 다음과 같다.

단계 1) 공격자는 서명 값 $g^k \bmod p$ 생성 과정에서 $t-i$번째 제곱 연산을 건너뛰는 오류를 주입하여 오류 서명 값을 얻는다.

$(\bar{r} = (g^{\bar{k}} \bmod p) \bmod q, \bar{s} = k^{-1}(h(m) + a\bar{r}) \bmod q)$

단계 2) 공격자는 오류 서명 값 $(\bar{r}, \bar{s})$을 통해 $\sqrt{g^k} \bmod p$를 계산한다.

$$\sqrt{g^k} \bmod p = \sqrt{g^{\bar{s}^{-1}h(m)} \bmod p} \cdot \sqrt{y^{\bar{s}^{-1}\bar{r}} \bmod p}\,.$$

단계 3) 단계1,2에서 얻은 $\bar{r}$과 $\sqrt{g^k}$를 통해 다음 등식을 만족하는 k'를 구한다.

$$\bar{r} = (\sqrt{g^k} \bmod p \cdot \sqrt{g^{k'}} \bmod p) \bmod q, \qquad (2)$$

여기서 $\bar{s}^{-1}h(m) \bmod q$, $\bar{s}^{-1}\bar{r} \bmod q, k$는 모두 짝수라는 제한 조건을 갖는다.

2.3 오류 주입 공격 대응법

위에서 언급한 오류 주입 공격에 대한 대응법은 다음과 같다.

Bit flip/Random byte 오류 : Bit flip/Random byte 오류 주입 공격은 공격자가 오류 서명 값 $\bar{s}$와 공개된 파라미터들을 이용해 서명 값 r을 복원한다. 즉, 공격자는 서명 값 r과 비교하기 위해 필요한 식 (1)의 R_i를 공개된 정보로만 계산한다. 따라서 대응법은 비밀정보 없이는 비교 값 R_i를 계산할 수 없도록 설계되어야 한다.

바이트 리셋 오류 : 공격자는 오류를 주입하여 난수 k의 부분 정보를 알고 있는 여러 개의 서명 쌍을 가지고 있음을 가정한다. 여러 개의 서명 쌍을 이용해 lattice 공격으로 개인키를 추출한다. 따라서 대응법은 lattice 공격에 필요한 조건을 만족시킬 수 없도록 설계되어야 한다.

제곱 연산 과정을 생략하는 오류 : 공격자는 난수의 부분정보를 알아내기 위해 식 (2)를 만족시키는 $\sqrt{g^k} \bmod p$ 를 계산해야 한다. 즉, 오류서명 값 $\bar{r}$을 알고 있는 공격자는 공개된 정보만으로 $\sqrt{g^k} \bmod p$를 계산한다. 따라서 대응법은 비밀정보 없이 $\sqrt{g^k} \bmod p$를 계산할 수 없도록 설계되어야 한다.

2.3.1 Nikodem의 방법

입력 : 메시지 m, 개인 키 a, 공개키 y 모듈러스p,q 생성자 g 출력 : 서명 값 (r,s)
1. 랜덤 수 $k(0<k<q)$를 선택한다. 2. $r = (g^k \bmod p) \bmod q$ 를 계산한다. 3. $v = k + ar \bmod q$ 를 계산한다. 4. $V = ((g^v y^{-r} \bmod p) \bmod q) - r \bmod q$를 계산한다. 5. $k' = (k \oplus V)^{-1} \bmod q$ 를 계산한다. 6. $s = k'(h(m) + v) - 1 \bmod q$ 를 계산한다.

[그림 2] 오류 주입 공격에 대응 하는 Nikodem의 DSA 서명 알고리즘

Nikodem은 bit flip 오류 주입 공격에 안전한 새로운 DSA 서명 알고리즘을 제안하였다[13]. 서명 과정중 개인키에 오류가 주입되면 그 오류가 서명 문 전체에 확산되는 성질을 이용하였다. "[그림 2]"는 Nikodem이 제안한 새로운 DSA 서명 알고리즘을 나타내고 있다.

오류 주입 여부를 확인하는 과정을 통해 공격을 방어할 수 있다. "[그림 2]"의 단계 3에서 개인키를 사용한 후 단계 4를 통해서 오류 주입 여부를 검증하게 된다. 만약 오류가 주입 되지 않았다면 $V=0$이고, 그렇지 않으면 $V \neq 0$이다. 따라서 단계 7의 서명 값 s는 추측하기 힘든 랜덤 값이 될 것이다. 오류 서명 값을 얻은 공격자는 식 (1)에서 공격에 필요한 R_i를 계산해야 되지만 서명 값 s가 랜덤 값이므로 비밀 정보 없이는 계산할 수 없다. 따라서 bit flip 오류 주입 공격을 적용할 수 없게 된다. 위의 서명 알고리즘은 기존 DSA서명 방법에 비해 약 3배 정도의 연산이 필요하다는 단점이 있다.

이 밖에도 Nikodem이 제안한 다른 기법도 있지만 이는 "[그림 2]"에서 연산량을 줄이기 위한 변형이므로 생략하도록 한다.(자세한 내용은 [12],[13]참조)

2.3.2 Bae 등의 방법

Bit flip 오류 주입 공격에 안전하면서 효율적으로 구현할 수 있는 DSA 서명 알고리즘이 Bae 등에 의

입력 : 메시지 m, 개인키 a, 공개키 y
모듈러스p,q 생성자 g
출력 : 서명 값 (r,s)

1. 난수 $k(0<k<q)$를 선택한다.
2. $r=(y^k \bmod p) \bmod q$ 를 계산한다.
3. $s=k^{-1}(a^{-1}h(m)+r) \bmod q$ 를 계산한다.

〔그림 3〕 오류 주입 공격에 대응하는 Bae 등의 DSA 서명 알고리즘

해 소개되었다〔16〕. DSA 서명 방식에서 사용되는 파라미터나 함수를 그대로 사용하면서 1회의 역원 연산만 추가하므로 타 대응방식에 비해 효율적으로 구현할 수 있다. 이는 Nikodem과 같이 오류를 검출하여 서로 비교하는 방식이 아니기 때문에 가능하다. 이 방식에서 가장 큰 특징 중 하나는 서명 값 r을 생성할 때 공개키 y를 사용하는 것과 서명 값 s를 생성할 때 개인키 a의 역원이 사용되도록 설계된 것이다. 개인키 a에 오류가 발생하더라도 a의 역원을 구하는 과정에서 오류가 확산된다. 역원의 비선형성(nonlinearity) 때문에 bit flip 오류 주입 공격에서 필요한 비교 값 R_i를 개인키에 대한 정보 없이 구할 수 없으므로 공격에 안전하다고 주장하였다. "〔그림 3〕"은 Bae 등의 알고리즘이다.

2.3.3 Jung 등의 방법

〔17〕에서 DSA에 대한 새로운 오류 주입 공격뿐만 아니라 이에 대한 대응법도 소개하였다. 또한 제안한 알고리즘이 bit flip 오류 주입 공격에도 안전하다고 주장하였다. "〔그림 4〕"는 Jung 등이 제안한 DSA 서명 알고리즘과 개선된 이진 지수승 연산 알고리즘이다. 먼저 bit flip 오류의 발생 여부를 확인하는 절차는 개인키 a의 역수를 사전에 저장한 후, T를 계산하는 단계 4에서 오류 주입 여부를 검증한다. 만약 오류가 주입 되었을 경우 a와 a^{-1}가 소거되지 않기 때문에 $T \neq 0$이 되며 최종 서명은 추측하기 어려운 랜덤 값이 된다. 따라서 공격자는 식 (1)에서 공격에 필요한 R_i를 계산해야 되지만 서명 값 s가 랜덤 값이므로 비밀정보 없이는 계산할 수 없으므로 개인키 오류 주입 공격에 안전하다. 제곱 연산 과정을 생략하는 난수에 대한 오류 주입 공격의 대응법으로 단계 2와 같이 새로운 이진 지수승 연산을 제안하였다. 단계 2를 통해 r을 계산할 때 사용된 k와 같은 값을 k_r로 복원시킨다. 이 후 서명 과정에서 k 대신 k_r을 사용한다.

입력 : 메시지 m, 개인키 a, 공개키 y
모듈러스p,q 생성자 g
출력 : 서명 값 (r,s)

1. 난수 $k(0<k<q)$를 선택한다.
2. $r=(g^k \bmod p) \bmod q$ 를 계산한다.
   2.1 $r=1, k_r=0$
   2.2 for $i=t-1$ *downto* 0 do
   2.3 $r=r \cdot r \bmod p$ with $k_r=2k_r$
   2.4 if $(k_i=1)$ then
   2.5 $r=r \cdot g \bmod p$ with $k_r=k_r+1$
   2.6 end if
   2.7 end for
   2.8 $r=r \bmod q$
   2.9 return(r,k_r)
3. $s_t=k_r^{-1}(h(m)+ar) \bmod q$ 를 계산한다.
4. $T=(s_t k_r-h(m))a^{-1} \bmod q \oplus r$ 를 계산한다.
5. $s=(s_t \oplus T) \bmod q$ 를 계산한다.

〔그림 4〕 오류 주입 공격에 대응하는 Jung등의 DSA 서명 알고리즘

따라서 공격자는 단계 2,3에서 쓰인 난수가 서로 같으므로 식 (2)을 만족하는 $\sqrt{g^k} \bmod p$를 계산할 수 없기 때문에 공격에 안전하다.

III. 기존 대응법의 취약성 분석

본 장에서는 지금까지 제안된 오류 주입 공격에 안전한 DSA 서명 알고리즘의 취약성을 분석하였다. 분석 대상은 Nikodem, Bae 등, Jung 등이 제안한 알고리즘이다. 각 논문에서 안전하다고 주장한 공격 모델 외에도 다양한 공격 기법을 적용하여 분석 하였다. 세 가지 알고리즘 모두 개인키 비트 플립 오류에 안전하게 설계하였다. 또한 개인 키 비트 플립 오류에 안전하면 랜덤 바이트 오류에도 안전한 것은 쉽게 확인할 수 있으므로 본 장에서는 난수에 대한 오류 주입 공격만을 고려한다.

3.1 Nikodem의 대응법

- 바이트 리셋 오류

"〔그림 2〕"의 알고리즘에서 난수 k 생성 시, 오류를 주입하여 최하위 몇 바이트를 0으로 고정시킬 수 있다. 이렇게 일부 값을 알고 있는 난수 k를 사용해 서명 값 s를 얻는다. 서명 값 s는 다음과 같은 과정을 통해 변형 할 수 있다.

공격자는 위와 같은 1 차식을 얻을 수 있다. 또한 최종 식에서 $s, h(m), r$ 값을 알 수 있고 k의 부분정보

를 알기 때문에 이러한 서명 쌍을 여러 개 모아 lattice 공격을 통해 개인키 a를 추출할 수 있다.

$$\begin{aligned} s &= k^{-1}(h(m)+v)-1 \bmod q \\ &= k^{-1}(h(m)+k+ar)-1 \bmod q \\ &= k^{-1}(h(m)+ar) \bmod q \\ ks &= h(m)+ar \bmod q \end{aligned}$$

- 제곱 연산 과정을 생략하는 오류

"[그림 2]"의 단계 2에서 이진 지수승을 계산할 때 제곱 연산이 한 번 생략되었다고 가정하자. 오류로 인한 잘못된 서명 값 r을 이용하여 다음가 같은 계산이 이루어진다.

$$\begin{aligned} \bar{v} &= k+a\bar{r} \\ V &= ((g^{\bar{v}}y^{-\bar{r}} \bmod p) \bmod q)-\bar{r} \bmod q \\ &= ((g^{k+a\bar{r}}g^{-a\bar{r}} \bmod p) \bmod q)-\bar{r} \bmod q \\ &= ((g^k \bmod p) \bmod q-\bar{r}) \bmod q \\ &= r-\bar{r} \bmod q \end{aligned}$$

위의 마지막 식에서 $r \neq \bar{r}$ 이므로 $V \neq 0$. 따라서 공격자는 식 (2)를 만족하는 $\sqrt{g^k} \bmod p$ 를 계산할 수 있어야 하지만 비밀 정보를 알지 못하면 구할 수 없으므로 공격에 안전하다.

3.2 Bae 등의 대응법

- 바이트 리셋 오류

Naccache 등이 [15]에서 자신들의 공격에 대한 대응법을 제시하였는데, Bae등은 그 대응법을 자신들이 제안한 알고리즘에 적용하면 오류 주입 공격을 방어할 수 있다고 주장하였다. 하지만 [15]에서 제안한 대응법은 공격을 막는 것이 아닌 공격에 더 많은 시간을 필요하게 만드는 것이다. 또한 오류 검출 방법은 많은 오버헤드가 발생할 뿐만 아니라 오류가 검출되기 전 공격을 성공할 수 있기 때문에 오류 주입 공격에 안전하지 않다. 따라서 알고리즘 단계에서 바이트 리셋 오류에 안전한 대응법이 필요하다.

- 제곱 연산 과정을 생략하는 오류

단계 1) 공격자는 서명 값 $y^k \bmod p$ 생성 과정에서 $t-i$번째 제곱 연산을 건너뛰는 오류를 주입하여 오류 서명 값을 얻는다.

$(\bar{r}=(y^{\bar{k}} \bmod p) \bmod q,\ \bar{s}=(ak)^{-1}(h(m)+a\bar{r}) \bmod q)$

단계 2) 공격자는 오류 서명 값 $(\bar{r},\bar{s})$을 통해 $\sqrt{y^k} \bmod p$를 계산한다.

$$\sqrt{y^k} \bmod p = \sqrt{g^{\bar{s}^{-1}h(m) \bmod p}} \cdot \sqrt{y^{\bar{s}^{-1}\bar{r}} \bmod p}$$

단계 3) 단계1,2에서 얻은 $\bar{r}$과 $\sqrt{y^k}$를 이용해 다음 등식을 만족하는 k'를 구한다.

$$\bar{r}=(\sqrt{y^k} \bmod p \cdot \sqrt{y^{k'}} \bmod p) \bmod q.$$

이렇게 k의 i번째 이후 비트 정보를 알고 있는 여러 개의 서명쌍을 통해 lattice 공격을 적용하면 개인키를 복구할 수 있다. Jung 등의 논문에서 처럼 $\bar{s}^{-1}h(m) \bmod q$, $\bar{s}^{-1}r \bmod q$, k이 짝수라는 가정이 있어야 공격에 성공할 수 있다.

3.3 Jung 등의 대응법

- 바이트 리셋 오류

"[그림 4]"의 알고리즘에서 난수 k 생성 시 오류를 주입하여 최하위 바이트를 0으로 고정시킬 수 있다. 알고리즘의 단계 2부터 단계 4까지 쓰인 난수 k 와 k_r이 모두 같기 때문에 $T=0$ 이다. 여기서 유의해야 할 점은 난수 k_r이 정상적으로 복구되었기 때문에 k의 최하위 바이트가 0인 것과 마찬가지로 k_r도 동일한 바이트의 위치 값이 0임을 알 수 있다. 따라서 공격자 일부 정보를 알고 있는 난수 k_r에 대한 1차식을 다음과 같이 얻을 수 있다.

$$\begin{aligned} s_t &= k_r^{-1}(h(m)+ar) \bmod q \\ k_r s_t &= h(m)+ar \bmod q \end{aligned}$$

공격자는 lattice 공격에 필요한 1차 식을 위 식을 통해서 얻을 수 있으므로 이러한 서명 쌍을 여러 개 모아 개인키 a를 복원할 수 있다.

IV. 제안하는 서명 기법

본 장에서는 위에서 언급한 4가지 오류 주입 공격을 동시에 방어할 수 있는 새로운 DSA 서명 알고리즘을 제안한다. 제안하는 방식은 Nikodem의 오류 주입 공격에 안전한 DSA 서명 알고리즘[13]을 변형시킨 것이다. 서명 과정 중 오류가 주입되면 그 오류가 서명문 전체에 확산되도록 설계하였다. 이렇게 확산된 오류는 서명 중 검증 단계를 통해 확인할 수 있다. 또한 기존에 하나만 사용했던 난수 k를 2개 사용함으로써 난수에 대한 오류 주입 공격도 방어할 수 있다.

4.1 제안하는 DSA 서명 기법

제안하는 오류 주입 공격에 안전한 DSA 서명 알고

입력 : 메시지 m, 개인키 a, 공개키 y
모듈러스p,q 생성자 g
출력 : 서명 값 (r,s)

1. 난수 k_1,k_2($\left\lceil \frac{q}{2} \right\rceil < k_1,k_2 < q$)를 선택한다.
2. $k = k_1 k_2 \bmod q$ 를 계산한다.
3. $r = (g^k \bmod p) \bmod q$ 를 계산한다.
4. $v = k_1 + ar \bmod q$ 를 계산한다.
5. $V = ((g^v y^{-r} \bmod p)^{k_2} \bmod q) - r \bmod q$ 를 계산한다.
6. $k' = (k \oplus V)^{-1} \bmod q$ 를 계산한다.
7. $s = k'(h(m)+v) - k_2^{-1} \bmod q$ 를 계산한다.

〔그림 5〕 제안하는 DSA 서명 알고리즘

리즘은 "〔그림 5〕"에서 기술한다. 알고리즘 단계 1에서 난수 2개를 선택하여 새로운 난수 k를 생성한다. 이 때, 모듈러 연산이 반드시 수행되도록 $k_1 \bullet k_2 > q$를 만족하는 k_1,k_2를 선택해야 한다. 단계 4,5는 오류 주입 여부를 검증하는 단계이다. 만약 단계 5 이전에 개인키 또는 난수에 오류가 주입되면 $V \neq 0$이 되어 서명 값 s는 추측하기 어려운 난수가 된다. 또한 "〔그림 5〕"와 같은 서명 알고리즘을 사용하더라도 검증 방법은 일반적인 DSA 서명 알고리즘과 동일하다. 즉 m에 대한 서명 값 (r, s)를 받은 수신자는 다음과 같이 계산하여 서명의 유효함을 확인할 수 있다.

$$w = s^{-1} \bmod q$$
$$v_1 = g^{h(m)w} \bmod p,\ v_2 = y^{rw} \bmod p$$
$$(v_1 \bullet v_2 \bmod p) \bmod q = ?\ r$$

(증명) DSA 검증 알고리즘

서명식
$$\begin{aligned} s &= k'(h(m)+v) - k_2^{-1} \bmod q \\ &= k^{-1}(h(m)+v) - k^{-1}k_1 \bmod q \\ &= k^{-1}(h(m)+v-k_1) \bmod q \\ &= k^{-1}(h(m)+ar) \bmod q \end{aligned}$$

따라서 $w = k(h(m)+ar)^{-1} \bmod q$

검증식
$$\begin{aligned} &(v_1 \bullet v_2 \bmod p) \bmod q \\ &= (g^{h(m)w} y^{rw} \bmod p) \bmod q \\ &= (g^{h(m)w} g^{arw} \bmod p) \bmod q \\ &= (g^{w(h(m)+ar)} \bmod p) \bmod q \\ &= (g^{k(h(m)+ar)^{-1}(h(m)+ar)} \bmod p) \bmod q \\ &= (g^k \bmod p) \bmod q \\ &= r \end{aligned}$$

4.2 안전성 분석

제안하는 서명 알고리즘의 안전성은 앞에서 언급한 4가지 오류 공격 모델을 가정하여 분석한다.

4.2.1 개인키에 대한 오류

먼저 서명 알고리즘이 수행되는 동안 개인키 a에 오류가 발생했을 경우에는 오류가 서명문 전체에 확산되기 때문에 오류 주입 여부 검증 단계를 거쳐 오류의 주입 여부를 검사 할 수 있다. 그 과정은 다음과 같이 이루어진다. 이 과정은 bit flip오류뿐만 아니라 random byte 오류에도 동일하게 적용된다. 개인키 a에 오류가 주입되었다고 가정하자. 오류가 주입된 후 서명 과정을 살펴보면 단계 4는 $v = k_1 + \bar{a}r \bmod q$가 되고 이를 이용해 단계 5를 계산하면 $V = ((g^{k_1 + \bar{a}r - ar} \bmod p)^{k_2} \bmod q) - r \bmod q$ 와 같이 표현된다. 따라서 $V = 0$을 만족시키기 위한 필요충분조건인 $g^v y^{-r} \bmod p \bmod q = r$가 성립되지 않으므로 $V \neq 0$이다. 서명 값 s는 랜덤 값이 되기 때문에 식 (1)의 R_i를 비밀정보 없이 구할 수 없다. 따라서 개인키에 대한 오류 주입 공격으로부터 안전하다.

4.2.2 랜덤수에 대한 오류

- 제곱 연산 과정을 생략하는 오류

"〔그림 4〕"의 단계 3에서 $t-i$번째 제곱 연산을 생략했다고 가정하자. 그러면 다음과 같은 과정을 통해 서명 값을 얻는다.

1. $\bar{r} = g^{\bar{r}} \bmod q$ 를 계산한다.
2. $\bar{v} = k_1 + a\bar{r} \bmod q$ 를 계산한다.
3. $V = ((g^{\bar{v}} y^{-\bar{r}} \bmod p)^{k_2} \bmod q) - \bar{r} \bmod q$
   $= (g^{(k_1 + a\bar{r} - a\bar{r})k_2} \bmod q) - \bar{r} \bmod q$
   $= r - \bar{r} \bmod q$ 를 계산한다.
4. $k' = (k \oplus (r - \bar{r}))^{-1} \bmod q$ 를 계산한다.
5. $\bar{s} = (k \oplus (r - \bar{r}))^{-1}(h(m)+v) - k_2^{-1} \bmod q$
   를 계산한다.

서명 값 $(\bar{r}, \bar{s})$를 얻은 공격자는 난수의 부분정보를 알아내기 위해 식 (2)를 만족하는 $\sqrt{g^k} \bmod p$를 다음과 같이 계산한다.

$$\sqrt{g^{\bar{s}^{-1}h(m)} \bmod p} \times \sqrt{y^{\bar{s}^{-1}\bar{r}} \bmod p}$$

하지만 서명과정에서 $k_1 \times k_2 \neq \bar{k}$ 이므로 $V \neq 0$가 된다. 따라서 $\bar{s}$가 추측하기 힘든 랜덤 값이 되고, $\sqrt{g^k} \bmod p$ 와 $\sqrt{g^{\bar{s}^{-1}h(m)} \bmod p} \times \sqrt{y^{\bar{s}^{-1}\bar{r}} \bmod p}$ 는 같지 않기 때문에 공격자는 공격에 필요한 $\sqrt{g^k} \bmod p$ 을 계산할 수 없다. 그러므로 제곱 연산 과정을 생략하는 오류 모델에 안전하다.

〔표 1〕 오류 주입 공격에 대한 안전성 분석

	개인키에 대한 오류 주입		난수에 대한 오류 주입	
	Bao 등의 분석기법	Giraud 등의 분석기법	Naccache 등의 분석기법	Jung 등의 분석기법
Nikodem 등의 대응법	안전	안전	취약	취약
Bae 등의 대응법	안전	안전	취약	취약
Jung 등의 대응법	안전	안전	취약	안전
제안하는 대응법	안전	안전	안전	안전

- 바이트 리셋 오류

위의 오류 모델에서 여러 개의 서명 값 s가 lattice 공격에 필요한 1차 식을 만족할 수 있다면 개인키 a를 복원할 수 있다. 만약 k_1(또는 k_2)을 생성 시 오류를 주입하여 최하위 바이트가 0이 되었다고 가정하자. 이렇게 생성된 k_1을 가지고 서명을 생성하면 단계 5의 V값은 0이 되고, 서명 값 $s = k'(h(m)+v) - k_2^{-1} \bmod q$를 얻는다. 서명 값 s를 정리하면 다음과 같다.

$$sk = (h(m) + ar) \bmod q \quad (3)$$

비록 1차 식을 얻었지만 공격자는 k의 부분정보를 알 수 없기 때문에 lattice 공격에 적용할 수 없다. 이는 난수 k를 계산 할 때 모듈러 연산을 하기 때문에 k의 최하위 바이트가 0이 아닌 값을 가지게 된다. 만약 $k_1 \times k_2 <$ 모듈러 q 이면 모듈러 연산을 하지 않으므로 k의 최하위 바이트도 0임을 알 수 있다. 이 경우에는 식(3)에 의해 lattice 공격으로 개인키 a를 복구 할 수 있다. 따라서 난수 k_1, k_2를 선택할 때 모듈러 연산을 하도록 $\left\lceil \frac{q}{2} \right\rceil < k_1, k_2 < q$ 에서 선택해야 한다. k_1, k_2 둘 다 최하위 바이트를 0으로 만들어도 마찬가지로 모듈러 연산으로 인해 k의 부분정보를 알 수 없기 때문에 이 또한 안전하다.

V. 결론

지금까지 DSA 서명 알고리즘에 대한 오류 주입 분석 기법과 기존 대응법을 알아보았고, 여러 가지 공격 모델을 적용하여 기존 대응법을 분석하였다. 또한 모든 오류 주입 공격에 안전한 새로운 DSA 서명 알고리즘을 제안하고 안정성을 분석하였다. "〔표 1〕"에서 기존 대응법과 안전성을 비교하였다. 제안 알고리즘을 제외한 다른 대응법들은 오류 주입 공격에 모두 취약함을 알 수 있다.

제안하는 DSA 서명 알고리즘은 오류 확산 기법을 이용하여 오류 주입 여부를 검증하고 기존에 1개만 사용하였던 난수를 2개 사용하여 서명하도록 설계하였다. 비록 제안하는 서명 기법이 모든 오류 주입 공격에 안전하나 연산량이 크다는 단점이 있으므로, 앞으로 이를 개선해 나가야 할 것이다.

참고문헌

〔1〕 P. Kocher, J. Jaffe, and B. Jun, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS,and Others Systems," CRYPTO-1996, LNCS vol.1109, pp. 104-113, 1996.

〔2〕 E. Biham, A. Shamir, "Differential Fault Analysis of Secret Key Cryptosystems," CRYPTO-1997, LNCS vol. 1294, pp. 513-525, 1997.

〔3〕 D. Boneh, R. A. DeMillo and R. J. Lipton, "On the Importance of Checking Cryptographic Protocols for Faults," EUROCRYPT-1997, LNCS vol. 1233, pp. 37-51, 1997.

〔4〕 P. Kocher, J. Jaffe, and B. Jun, "Introduction to differential power analysis and related attacks," White Paper, Cryptography Research, http://www.cryptography.com/dpa/technical 1998.

〔5〕 P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," CRYPTO-1999, LNCS vol. 1666, pp. 388-397, 1999.

〔6〕 S. Chari, J.R. Rao, and P. Rohatgi. "Template Attacks," CHES-2002, LNCS vol. 2523, pp. 13 - 28, 2002.

〔7〕 E. Brier, C. Clavier, and F. Olivier, "Cor-

relation power analysis with a leakage model," CHES-2004, LNCS vol. 3156, pp. 16-29, 2004.

〔8〕 Bellcore Press Release, "New threat model breaks crypto codes," Sep. 1996.

〔9〕 S. Yen, S. Kim, S. Lim, and S. Moon, "RSA speedup with Chinese Remainder Theorem ImmuneAgainst Hardware Fault Cryptanalysis," IEEE Transaction on Computer, Special issue on CHES, vol. 52, no. 4, pp. 461-472, Apr. 2003.

〔10〕 "National institute of standards and technology," FIPS PUB 186-2: Digital Signature Standard, 2000.

〔11〕 F. Bao, R. H. De6ng, Y. Han, A Jeng, A. D. Narasimhalu, T. Ngair, "Breaking Public Key Cryptosystems on Tamper Resistant Devices in the Presence of Transient Faults," International Workshop on Security Protocols-1997, LNCS, vol. 1361, pp. 115-124, 1997

〔12〕 M. Nikodem,"Error Prevention, Detection and Diffusion Algorithms for Cryptographic Hardware," 2nd International Conference on Dependability of Computer Systems - DepCos-RELCOMEX'07, pp. 127-134, June. 2007.

〔13〕 M. Nikodem, "DSA Signature Scheme Immune to the Fault Cryptanalysis," CARDIS-2008, LNCS, vol. 5189. pp. 61-73, 2008

〔14〕 C. Giraud and E. Knudsen, "Fault Attacks on Signature Schemes," ACISP-2004, LNCS vol. 3108, pp. 478-491, 2004.

〔15〕 D. Naccache, P. Nguyen, M. Tunstall and C.Whelan, "Experimenting with Faults, Lattices and the DSA," PKC-2005, LNCS vol. 3386, pp. 16-28, 2005.

〔16〕 K. S. Bae, Y. R. beak, S. J. Moon, J. C. ha, "An Efficient DSA Signature Scheme Resistance to the Fault Analysis Attack," Journal of The Korea Institute of Infomation Security and Cryptology, vol. 20, no. 5 pp. 49-57, Oct. 2010.

〔17〕 Chul-Jo Jung, Doo-Hwan Oh, Doo-Sik Choi, Hwan-Koo Kim and Jae-Cheol Ha, "Cryptanalysis using Fault Injection and Countermeasures on DSA," Journal of The Korea Academia -industrial Cooperation Society, Vol. 11, no. 8 pp. 3045-3052, Aug. 2010.

〔18〕 E. Biham and A. Shamir. "Differential Fault Analysis of Secret Key Cryptosystem," CRYPTO-97, LNCS vol. 1294, pp. 513 - 525, 1997.

〔19〕 G. Piret and J.-J. Quisquater. "A Differential Fault Attack Technique Against SPN Structures, with Application to the AES and Khazad," CHES-2003, LNCS vol. 2779, pp. 77 - 88, 2003.

〔20〕 I. Biehl, B. Meyer, and V. Müller. "Differential Fault Analysis on Elliptic Curve Cryptosystems. In M. Bellare," CRYPTO-2000, LNCS vol. 1880, pp. 131 - 146, 2000.

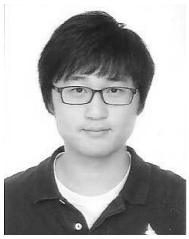
〔21〕 J. Schmidt, M. Medwed, "A Fault Attack on ECDSA," FDTC-2009, pp. 93-99, Sep. 2009.

〔22〕 N. A. Howgrave-Graham and N. P. Smart, "Lattice Attacks on Digital Signature Schemes," Design, Codes and Cryptography, vol. 23 no 3, pp. 283 - 290, Aug. 2001.

〔23〕 P. Q. Nguyen and I. E. Shparlinski, "The Insecurity of the Digital Signature Algorithm with Partially Known Nonces," Journal of Cryptology, Springer, vol. 15, no. 3, pp. 151 - 176, Mar. 2002.

〔24〕 D. Boneh and R. Venkatesan, "Hardness of Computing the Most Significant Bits of Secret Keys in Diffie-Hellman and Related Schemes," CRYPTO-1996, LNCS 1109, pp. 126 - 142, 1996.

〈著者紹介〉



김 태 원 (TaeWon Kim) 학생회원
2010년 2월: 광운대학교 수학과 학사
2010년 8월~현재: 고려대학교 정보보호대학원 석사과정
〈관심분야〉 부채널 공격, 스마트 카드 보안, 암호시스템 안전성 분석 및 고속구현

김 태 현 (TaeHyun Kim) 학생회원
2002년 2월: 서울 시립대학교 수학과 이학사
2004년 8월: 고려대학교 정보보호대학원 공학석사
2010년 2월: 고려대학교 정보보호대학원 공학박사
2010년 8월: ETRI 부설연구소
2011년 2월~현재: 고려대학교 정보보호대학원 연구교수
〈관심분야〉 부채널 공격, 공개키 암호 알고리즘, 암호칩 설계 기술

홍 석 희 (Seokhie Hong) 정회원
1995년 2월: 고려대학교 수학과 학사
1997년 2월: 고려대학교 수학과 석사
2001년 8월: 고려대학교 수학과 박사
1999년 8월~2004년 2월: (주) 시큐리티 테크놀로지스 선임연구원
2003년 8월~2004년 2월: 고려대학교 정보보호기술연구센터 선임연구원
2004년 4월~2005년 2월: K.U.Leuven, ESAT/SCD-COSIC 박사후연구원
2005년 3월~현재: 고려대학교 정보보호대학원 부교수
〈관심분야〉 대칭키·공개키 암호 분석 및 설계, 컴퓨터 포레식

박 영 호 (YoungHo Park) 종신회원
1990년 2월: 고려대학교 수학과 학사
1993년 2월: 고려대학교 수학과 석사
1997년 2월: 고려대학교 수학과 박사
2002년 3월~현재: 세종 사이버대학교 부교수
〈관심분야〉 암호이론, 부채널 공격, 금융보안, 정보보호시스템