情報保護學會論文誌
第18卷 第4號, 2008. 8

비교연산을 사용하지 않는 오류주입 공격에 안전한 CRT 기반의 RSA*

김 성 경[1†], 김 태 현[1], 한 동 국[2], 박 영 호[3], 홍 석 희[1]
[1]고려대학교 정보경영공학전문대학원, [2]한국전자통신연구원, [3]세종사이버대학교

Secure RSA with CRT Protected Against Fault Attacks without using Checking Procedure*

Sung-Kyoung Kim[1†], Tae Hyun Kim[1], Dong-Guk Han[2], Young-Ho Park[3], SeokHie Hong[1]
[1]Graduate School of Information Management and Security, Korea University,
[2]Electronics and Telecommunications Research Institute, [3]Sejong cyber University

요 약

중국인의 나머지 정리(Chinese Remainder Theorem)를 이용한 RSA 암호 시스템(RSA CRT)은 모듈러 지수승 연산이 기존의 제곱 연산을 반복하는 것보다 빠르게 계산될 수 있기 때문에 표준으로 권장하고 있다. 그러나 1996년 Bellcore가 RSA CRT의 오류주입 공격에 대해서 발표한 이래로 RSA CRT의 안전성 문제가 대두되었다. 1997년 Shamir가 오류 주입을 확인하는 비교 연산을 이용한 대응 방법을 소개하였고, 곧이어 이러한 비교연산도 안전하지 않다고 알려졌다. 최근 Yen이 오류주입 공격에 안전한 두 가지의 CRT 연산 프로토콜을 제안하였으며 이 프로토콜은 오류 주입을 확인하는 비교연산이 존재하지 않는다. 그러나 FDTC 2006에서 Yen의 두 CRT 연산 프로토콜에 대한 공격 방법이 소개되었다. 본 논문에서는 FDTC 2006에서 제시된 공격 방법에도 안전한 두 CRT 연산 프로토콜을 제안한다. 제안하는 방법은 비트연산(AND)의 특성을 이용하며 추가적인 연산을 고려하지 않아도 된다.

ABSTRACT

Because Chinese Remainder Theorem based RSA (RSA CRT) offers a faster version of modular exponentiation than ordinary repeated squaring, it is promoting with standard. Unfortunately there are major security issues associated with RSA CRT, since Bellcore announced a fault-based cryptanalysis against RSA CRT in 1996. In 1997, Shamir developed a countermeasure using error free immune checking procedure. And soon it became known that the this checking procedure can not effect as the countermeasures. Recently Yen proposed two hardware fault immune protocols for RSA CRT, and this two protocols do not assume the existence of checking procedure. However, in FDTC 2006, the method of attack against the Yen's two protocols was introduced. In this paper, the main purpose is to present a countermeasure against the method of attack from FDTC 2006 for CRT-RSA. The proposed countermeasure use a characteristic bit operation and dose not consider an additional operation.

Keywords : CRT-RSA. Fault attack, checking procedure

Ⅰ. 서 론

RSA 암호 알고리즘의 대표적인 연산은 비밀키를 사용한 지수승 연산이다. 스마트카드나 센서 노드 등 계산 능력이나 메모리 같은 자원이 제한된 환경에서 지수승 연산의 효율성을 증대시키기 위한 많은 방법들이 제안되었다. 그 중에서 중국인의 나머지 정리(Chinese Remainder Theorem, CRT)는 RSA 암호시스템에 대하여 전자서명 생성 시에 약 4배 정도의 향상된 속도를 얻을 수 있기 때문에 효율적으로 구현할 수 있다. 따라서 속도를 향상하기 위하여 많은 표준들에서 권장하고 있지만[1] CRT 기반의 RSA 암호시스템(RSA CRT)이 공격에 매우 취약하다. 1996년 Bellcore[2]가 RSA CRT가 오류주입 공격에 안전하지 않다는 것을 소개한 후, 1년 뒤인 1997년에 D.Boneh[3] 등이 공격 방법에 대한 자세한 내용을 언급하였다. 소개한 공격 방법은 암호 시스템이 동작하는 스마트카드와 같은 환경에 오류를 주입하여 RSA 암호 시스템의 경우 비밀 값인 소수 값을 공격자가 알아내는 방법이다. 오류주입 공격에 대한 연구가 활발해지면서 스마트카드와 같은 장치에서 안전한 암호학적 알고리즘을 구현하는 문제가 매우 중요해졌다. 특히 RSA는 현실적으로 가장 널리 쓰는 알고리즘이고 CRT를 RSA에 적용할 경우 효율적으로 연산을 수행할 수 있기 때문에, RSA CRT를 오류주입 공격으로부터 안전하게 구현하려는 노력이 지속되어 왔다[4-10]. RSA CRT 암호시스템에 대한 오류주입 공격 방법에 대한 대응책은 Shamir가 1997년에 CRT 연산 마지막 단계에 비교 연산을 추가함으로 오류 주입 여부를 확인하였고[9,10], 그 방법을 응용하여 많은 대응 방법이 제안되었다[5-7]. 그러나 이러한 비교 연산으로 구성된 대응 방법이 오류 주입 공격에 안전하지 않다는 것이 알려졌다[11].

2003년에 Yen이 두 가지의 CRT 연산 프로토콜(CRT-1, CRT-2)을 제안하였다[11]. Yen 방법은 비교 연산을 사용하지 않고 오류가 발생하였을 경우 공격자가 비밀 정보를 알아내지 못하게 하는 방법이다. Yen의 방법은 기존의 비교연산을 사용하는 타 방법보다 연산량은 많지만, 비교연산을 사용함으로 취약할 수 있는 타 방법보다 안전한 방법이다. 그러나 FDTC 2006에서 CRT-1, CRT-2 연산 프로토콜이 안전하지 않음으로 보여줬다. 두 CRT 연산 프로토콜에서 비교 연산을 사용하지 않기 위하여 추가 되었던 계산 과정과 메모리 접근 단계에서 오류 주입공격이 가능하여 비밀 정보를 공격자가 얻을 수 있다.

본 논문에는 FDTC 2006에서 제시된 공격 방법에도 안전한 두 CRT 연산 프로토콜을 제안한다. 제안하는 방법은 다른 두 값에 대한 비트 연산의 결과가 랜덤 한 값이라는 비트 연산의 특성, 즉 $x \wedge y = random,\ x \neq y$, 을 이용하여 공격에 대응하고, 기존의 Yen의 CRT-1 연산 프로토콜보다 한 번의 비트 연산(AND)만을 추가적으로 요구하며 CRT-2의 경우에는 한번의 AND 연산이 추가되나 덧셈 연산이 한번 줄어들게 된다. Yen의 장점을 그대로 취하며 모든 레지스터에 대한 오류주입에도 안전한 방법이므로 스마트카드와 같은 내장형 장치(embedded device)에 안전하게 사용할 수 있다.

본 논문의 구성은 다음과 같다. 2장에서는 CRT 기반의 RSA 암호시스템에 대한 설명과 오류 주입공격에 대한 설명이, 3장에서는 2003년에 Yen이 제안한 두 CRT 프로토콜에 대한 설명과 FDTC 2006에서 소개 한 두 CRT 프로토콜에 대한 공격방법을 소개한다. 제안하는 두 CRT 프로토콜의 대응 방법을 4장에서 소개한다.

Ⅱ. RSA-CRT 알고리즘과 오류주입 공격

RSA 암호 시스템에서는 개인키를 이용하여 연산하는 서명 생성 과정의 속도 향상을 위해 중국인의 나머지 정리(Chinese Remainder Theorem, CRT)를 많이 이용한다[1]. CRT 기반의 RSA 암호 시스템에서 사용되는 파라미터들은 RSA 모듈러 값인 $n = p \cdot q$와 공개키 e 그리고 개인키 d이다.

$z = CRT(x,y) \bmod n$는 $z \equiv x \bmod p,\ z \equiv x \bmod q$을 만족하는 유일한 값($z(\in Z/nZ)$)이다. CRT 기반의 RSA 암호 시스템에서 $m^d \bmod n$의 계산은

$$s_p \equiv m^{d \bmod (p-1)} \bmod p$$
$$s_q \equiv m^{d \bmod (q-1)} \bmod q$$

을 계산한 후 $s = CRT(s_p, s_q)$연산을 시행한다.

접수일 : 2007년 12월 5일; 수정일 : 2008년 4월 16일;
채택일 : 2008년 5월 16일
* 본 연구는 지식경제부 및 정보통신연구진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음 (IITA-2008-(C1090-0801-0025))
† 주저자, likesk@cist.korea.ac.kr

Algorithm 1. RSA CRT

Input : A message $m \in Z_n$, p, q, d_p, d_q, $A(=p^{-1})$
Output : $Sig := m^d \bmod n$.

1. $s_p \leftarrow m^{d_p} \bmod p$
2. $s_q \leftarrow m^{d_q} \bmod q$
3. $s \leftarrow ((s_q - s_p) \cdot A \bmod q) \cdot p + s_p$
4. Output s.

Algorithm 1은 CRT를 이용한 RSA 서명 알고리즘(RSA CRT)이다. 이 때 $d_p = d \bmod p-1$, $d_q = d \bmod q-1$ 이다.

RSA-CRT 알고리즘의 경우 서명 생성 속도는 향상되나 [2]의 논문에 의해서 오류주입공격에 안전하지 않다는 것이 소개되었다. 공격 방법은 다음과 같고, 이 때, 계산의 중간 값 x에 오류가 주입된 경우를 x'라고 표기하고, x'의 값은 랜덤 한 값으로 $x+t$ (t: 랜덤 값)으로 정의한다.

가. 단계 1의 오류가 주입 된 중간 결과 값 : s_p'
나. 단계 2의 중간 결과 값 : s_q
다. 오류가 주입 된 CRT 연산 결과 값 :
$s' = ((s_q - s_p') \cdot A \bmod q) \cdot p + s_p'$

$\gcd(s-s', n)$을 계산하면 $s = s' \bmod q, s \neq s' \bmod p$이므로 비밀 값인 q를 얻을 수 있다.

이러한 공격 방법이 소개 된 후 Shamir가 Eurocrypt'97[9]와 1999년 등록 된 특허[10]에서 CRT 기반의 RSA 암호 시스템에 대한 오류주입 공격에 대한 대응 방법을 제안하였다. 제안하는 방법은 32 비트 정도의 랜덤 한 값 r을 선택한 후

$$s_p \equiv m^{d \bmod (p-1)(r-1)} \bmod pr,$$
$$s_q \equiv m^{d \bmod (q-1)(r-1)} \bmod qr$$

을 계산하고, CRT 연산 전에 $s_p \equiv s_q \bmod r$를 확인한다. 만약 s_p(또는 s_q)에서 오류가 발생하였을 경우 $s_p \equiv s_q \bmod r$의 연산 단계를 통과하지 못하기 때문에 오류 주입 여부를 확인할 수 있다.

2.1 비교연산의 안전성

오류주입 공격에 대한 첫 대응 법 인 Shamir의 방법 [9,10]이 소개된 후 이를 응용하여 많은 대응 방법이 소개되었다[4-10]. 그러나 Shamir의 방법에는 여러 가지 문제점이 있다. 가장 큰 문제점은 비교 연산이다. 비교 연산의 안전성 문제는 다음과 같다.

비교연산 $(a=b)$의 연산 과정은 $c=a-b$의 단계와 $c=0$인지를 확인하는 두 단계로 나눠진다. 만약 공격자가 다음과 같은 두 개의 오류를 CRT 기반의 RSA 서명 생성 과정에 주입하게 되면 공격자는 비밀 정보를 알 수 있다[11].

i. 첫 번째 오류는 서명 값을 생성 하는 중간 과정에서 발생하는 오류이다.
ii. 두 번째 오류는 $c=0$를 확인하는 과정에 주입한다. $c=0$을 확인하는 과정을 건너뛰게 하는 것이다.

WISTP 2007의 [12]논문에서 실험을 통하여 두 번째 오류 주입이 가능함을 보여주었다. 비교 연산을 이용한 알고리즘의 경우 첫 번째 오류 확인을 비교 연산을 통하여 확인하게 되는데, 두 번째 오류 과정을 통하여 첫 번째 오류가 무조건 통과하게 되므로 공격자는 비밀 정보를 얻을 수 있게 된다.

Ⅲ. CRT-Protocol에 대한 분석

오류주입에 대한 공격 방법이 소개 된 후 CRT 기반의 RSA 암호시스템에서 오류주입 공격에 대한 다양한 대응 방법이 제안되었다. 2003년 Yen은 오류주입 공격에 안전한 두 가지 방법의 CRT 연산 프로토콜을 제안하였다[11]. [11]에서 제안하는 두 프로토콜에서는 서명 값 생성 시 오류가 주입되었을 경우 오류 확인을 위한 비교 연산 없이 공격자가 비밀 정보를 알 수 없도록 제안하였다. 그러나 FDTC 2006에 발표된 [8] 논문에서 두 CRT 연산 프로토콜에 대한 공격방법이 소개되었다. 본 장에서는 두 가지 CRT 연산 프로토콜에 관한 설명과 공격자가 수행 할 수 있는 공격 방법에 대해서 설명한다.

Yen의 프로토콜에서 사용되는 기호의 정의는 다음과 같다.

- $e_r = d_r^{-1} \bmod \Phi(n)$
- $d_r = d - r$ 이 때, r의 값은 작은 랜덤한 값이다.

Algorithm 2은 첫 번째 CRT 연산 프로토콜(CRT-1)이다. Algorithm 2에서 오류주입 공격이 없을 경우, 단

계 2에서 $\hat{m}$의 값은

$$\begin{aligned}\hat{m} &= ((s_p^{e_r} \bmod p) + k_p \cdot p) \bmod q \\ &= ((m \bmod p) + (m - (m \bmod p))) \bmod q \\ &= m \bmod q\end{aligned}$$

$$\begin{aligned}\hat{m} &= ((s_p^{e_r} \bmod p) + k_p \cdot p) \bmod q \\ &= ((m \bmod p) + (m - (m \bmod p))) \bmod q \\ &= m \bmod q\end{aligned}$$

이고, $\tilde{m}$의 값은

$$\begin{aligned}\tilde{m} &= (s_q^{e_r} \bmod q) + k_q \cdot q \\ &= (m \bmod q) + ((m - (m \bmod q)) = m\end{aligned}$$

을 만족한다. 따라서 올바른 서명 값을 생성할 수 있다.

Algorithm 2의 연산 과정에서 오류가 발생한 경우를 살펴보면 다음과 같다[8,10]. 2장에서도 표기하였듯이 x에 오류가 주입된 경우를 $x'(=x+t, t$=랜덤)이라고 한다.

(ㄱ) 단계 2에서 s'_p와 s_q의 경우, $\hat{m}$의 값은 s'_p의 오류의 영향으로 틀린 값이 된다. 따라서 $\hat{m}$을 이용하여 계산 되는 s_q의 값도 틀려지므로 공격자가 틀려진 서명 값을 가지고는 인수분해가 되지 않는다.

(ㄴ) s'_q, s_p의 경우, 단계 3의 $\tilde{m}$의 값이 틀려지고, s를 구하는 CRT와 $\tilde{m}$ 모두 올바르지 않는 값이 되기 때문에 인수분해로 비밀 값을 얻을 수 없다.

(ㄷ) k'_p와 k_q의 경우, $\hat{m}'$이고 s'_q는 $s'_q = \hat{m}'^{d_r \bmod (q-1)} \bmod q = s_q + R_1$가 되고, 이때 R_1의 값은 랜덤 한 값이다. 단계3에서 계산 되는 $\tilde{m}'$는 $\tilde{m}' = ((s_q + R_1)^{e_r} \bmod q) + k_q \cdot q = \tilde{m} + R_2$이다. 그러므로 서명 값인 s'은

$$\begin{aligned}s' &= CRT(s_p, s'_q) \cdot (\tilde{m} + R_2)^r \bmod n \\ &= CRT(s_p, s'_q) \cdot \tilde{m}^r + R_3 \bmod n\end{aligned}$$

가 되므로 공격자는 $\gcd(s' - m, n)$을 이용하여 비밀 정보를 알 수 없다.

(ㄹ) k_p, k'_q의 경우, 단계 3에서 $\tilde{m}$은 $\tilde{m}' = (s_q^{e_r} \bmod q) + k'_q \cdot q = \tilde{m} + t \cdot q$이므로, 서명 값은

$$\begin{aligned}s' &= CRT(s_p, s_q) \cdot \tilde{m}'^r \bmod n \\ &= CRT(s_p, s_q) \cdot (m + t \cdot q)^r \bmod n \\ &= CRT(s_p, s_q) \cdot m^r + CRT(s_p, s_q) \cdot R_1 \cdot q \bmod n \\ &= m^d + R_2 \cdot q \bmod n\end{aligned}$$

이다.

$\gcd(s' - m, n) = \gcd((m^d + R_2 \cdot q)^e - m, n) = q$를 만족하게 된다. 그러므로 CRT-1 프로토콜의 경우 k_q 값을 사용하기 위해 메모리에 접근 할 경우 발생한 오류나 k_q값을 계산할 때 발생하는 오류의 경우에는 CRT-1 프로토콜은 안전하지 않다. 유사한 방법으로 CRT-2 프로토콜(Algorithm 3)도 공격할 수 있다.

CRT-2 프로토콜의 단계 1의 k_p, k_q를 계산하는 과정에서 오류가 발생하거나, 단계 3에서 $\hat{m}$을 계산하기 위해 k_p, k_q 메모리를 참조 할 경우 오류가 발생하여 k'_q, k_p인 경우에 단계 2에서 s_p, s_q 값은 올바른 값이고, 단계 3에서

$$\begin{aligned}\hat{m}' &= \left\lfloor \frac{((s_p^{e_r} \bmod p) + k_p \cdot p) + ((s_q^{e_r} \bmod q) + k_q \cdot q)}{2} \right\rfloor \\ &= \left\lfloor \frac{m + (m + t \cdot q)}{2} \right\rfloor = m + \left\lfloor \frac{t \cdot q}{2} \right\rfloor = m + R \cdot q\end{aligned}$$

이 계산되고, 이 때 R은 $2|R$을 만족하는 정수(즉, 짝수)

Algorithm 2. CRT-1 프로토콜

Input : A message $m \in Z_n$,
Output : $Sig := m^d \bmod n$.

1. Compute both
   $k_p = \left\lfloor \frac{m}{p} \right\rfloor$ and $k_q = \left\lfloor \frac{m}{q} \right\rfloor$.
2. Compute $m^{d_r} \bmod n$ via a conventional CRT speedup as
   $$s_p = m^{d_r \bmod (p-1)} \bmod p,$$
   $$s_q = \hat{m}^{d_r \bmod (q-1)} \bmod q$$
   where $\hat{m} = ((s_p^{e_r} \bmod p) + k_p \cdot p) \bmod q$.
3. A CRT recombination operation and some additional manipulation are employed to compute the required signature as
   $$s = CRT(s_p, s_q) \cdot \tilde{m}^r \bmod n$$
   where $\tilde{m} = (s_q^{e_r} \bmod q) + k_q \cdot q$.
4. Output s.

Algorithm 3. CRT-2 프로토콜

Input : A message $m \in Z_n$,
Output : $Sig := m^d \bmod n$.

1. Compute both
   $k_p = \left\lfloor \frac{m}{p} \right\rfloor$ and $k_q = \left\lfloor \frac{m}{q} \right\rfloor$.
2. Compute $m^{d_r} \bmod n$ via a conventional CRT speedup as
   $$s_p = m^{d_r \bmod (p-1)} \bmod p,$$
   $$s_q = m^{d_r \bmod (q-1)} \bmod q.$$
3. A CRT recombination operation and some additional manipulation are employed to compute the required signature as
   $$s = CRT(s_p, s_q) \cdot \hat{m}^r \bmod n$$
   where
   $$\hat{m} = \left\lfloor \frac{((s_p^{e_r} \bmod p) + k_p \cdot p) + ((s_q^{e_r} \bmod q) + k_q \cdot q)}{2} \right\rfloor$$
4. Output s.

이다. 그러므로 생성되는 오류가 있는 서명 값은

$$\begin{aligned} s' &= CRT(s_p, s_q) \cdot \hat{m}'^r \bmod n \\ &= CRT(s_p, s_q) \cdot m^r + CRT(s_p, s_q) \cdot R \cdot q \bmod n \\ &= m^d + R_1 \cdot q \bmod n \end{aligned}$$

이다. $\gcd(s'-m, n) = \gcd((m^d + R_1 \cdot q)^e - m, n) = q$을 만족한다. k_p의 방법도 유사하게 공격가능하다.

Ⅳ. 제안하는 대응 방법

본 장에서는 본 논문에서 제안하는 두 개의 CRT 연산 프로토콜에 대한 오류주입 공격의 대응 방법에 대해서 살펴본다. 제안하는 대응 방법은 비트연산인 AND 연산(∧)을 이용하여 오류주입 시 올바르지 않은 서명 값을 통해 비밀 정보를 공격자가 알 수 없게 한다. AND 연산의 특징 중 "$x \wedge x = x$"와 "$x \wedge y =$랜덤 값"의 특징을 이용한다. 덧셈의 경우 $(13 \cdot 3) + (14 \cdot 3) = 27 \cdot 3$ 이고, AND 연산의 경우$(13 \cdot 3) \wedge (14 \cdot 3) = 34 \neq 3k$ $(k \in Z)$ 이다. 즉, 비트 연산의 경우 다른 두 값의 경우 특징이 연산 후에는 유지 되지 않음을 알 수 있다.

4.1 CRT-1 프로토콜의 대응방법

Algorithm 4는 본 논문에서 제안하는 CRT-1 프로토콜에 대한 대응방법이다. 단계 2는 기존의 방법과 동일하고, s_p에 오류가 주입 되지 않았다면 $\hat{m}$의 값은 $m \bmod q$의 값을 가지게 된다. 단계 3에서 서명 값 s를 계산하기 위해 $\tilde{m}$을 계산하게 된다.

$$\begin{aligned} \tilde{m} &= m \wedge (s_q^{e_r} \bmod q + k_q \cdot q) \\ &= m \wedge (m \bmod q + m - m \bmod q) \\ &= m \end{aligned}$$

그러므로 오류가 주입이 되지 않았을 경우 올바른 서명 값 s를 생성할 수 있다.

연산 과정에서 오류가 주입되었을 경우는 다음과 같다. 이 때, R_i의 값들은 랜덤 값이다.

a. s_p에 오류가 주입되었을 경우(s'_p)

: 각 단계의 결과 값은 아래와 같다.

가) 단계 2 : $s'_p = s_p + t$

$$\begin{aligned} \hat{m}' &= ((s'^{e_r}_p \bmod p) + k_p \cdot p) \bmod q \\ &= (((s_p + t)^{e_r} \bmod p) + k_p \cdot p) \bmod q \\ &= \hat{m} + R_1 \end{aligned}$$

Algorithm 4. CRT-1 프로토콜의 대응방법

Input : A message $m \in Z_n$.

Output : $Sig := m^d \bmod n$.

1. Compute both
   $k_p = \left\lfloor \frac{m}{p} \right\rfloor$ and $k_q = \left\lfloor \frac{m}{q} \right\rfloor$.
2. Compute $m^{d_r} \bmod n$ via a conventional CRT speedup as
   $s_p = m^{d_r \bmod (p-1)} \bmod p$,
   $s_q = \hat{m}^{d_r \bmod (q-1)} \bmod q$
   where $\hat{m} = ((s_p^{e_r} \bmod p) + k_p \cdot p) \bmod q$.
3. A CRT recombination operation and some additional manipulation are employed to compute the required signature as
   $s = CRT(s_p, s_q) \cdot \tilde{m}^r \bmod n$
   where $\tilde{m} = m \wedge (s_q^{e_r} \bmod q + k_q \cdot q)$.
4. Output s.

$$\begin{aligned} s'_q &= \hat{m}'^{d_r \bmod (q-1)} \bmod q \\ &= (\hat{m} + R_1)^{d_r \bmod (q-1)} \bmod q \\ &= s_q + R_2 \end{aligned}$$

나) 단계 3 :

$$\begin{aligned} \tilde{m}' &= m \wedge (s'^{e_r}_q \bmod q + k_q \cdot q) \\ &= m \wedge ((s_q + R_2)^{e_r} \bmod q + k_q \cdot q) \\ &= R_3 \end{aligned}$$

$$\begin{aligned} s' &= CRT(s'_p, s'_q) \cdot \tilde{m}'^r \bmod n \\ &= CRT(s'_p, s'_q) \cdot R_3^r \bmod n \end{aligned}$$

랜덤 한 서명 값을 얻은 공격자는 서명 값을 통하여 비밀 정보를 알 수 없다.

b. s_q에 오류가 주입되었을 경우(s'_q)

: 각 단계의 결과 값은 아래와 같다

가) 단계 2 : s_p

$$\begin{aligned} \hat{m} &= ((s_p^{e_r} \bmod p) + k_p \cdot p) \bmod q \\ &= m \bmod q \end{aligned}$$

$$s'_q = s_q + t$$

나) 단계 3 :

$$\begin{aligned} \tilde{m}' &= m \wedge (s'^{e_r}_q \bmod q + k_q \cdot q) \\ &= m \wedge ((s_q + t)^{e_r} \bmod q) + k_q \cdot q) = R_1 \end{aligned}$$

$$\begin{aligned} s' &= CRT(s_p, s'_q) \cdot \tilde{m}'^r \bmod n \\ &= CRT(s_p, s'_q) \cdot R_1^r \bmod n \end{aligned}$$

s_q에 오류가 주입 되었을 경우에도 공격자는 랜덤 한 서명 값을 얻게 된다.

c. k_p에 오류가 주입되었을 경우(k'_p)

: 각 단계의 결과 값은 아래와 같다

가) 단계 1 : $k'_p = k_p + t$

나) 단계 2 : s_p

$$\hat{m} = ((s_p^{e_r} \bmod p) + (k_p + t) \cdot p) \bmod q$$
$$= \hat{m} + R_1$$
$$s'_q = (\hat{m} + R_1)^{d_r \bmod (q-1)} \bmod q$$
$$= s_q + R_2$$

다) 단계 3 :

$$\tilde{m}' = m \wedge (s'^{e_r \bmod q}_q + k_q \cdot q)$$
$$= m \wedge ((s_q + R_2)^{e_r \bmod q} + k_q \cdot q)$$
$$= R_3$$
$$s' = CRT(s_p, s'_q) \cdot \tilde{m}'^r \bmod n$$
$$= CRT(s_p, s'_q) \cdot R_3^r \bmod n$$

d. k_q에 오류가 주입되었을 경우(k'_q)

: 각 단계의 결과 값은 아래와 같다

가) 단계 1 : $k'_q = k_q + t$

나) 단계 3 :

$$\tilde{m}' = m \wedge (s_q^{e_r} \bmod q + k'_q \cdot q)$$
$$= m \wedge (s_q^{e_r \bmod q} + (k_q + t) \cdot q)$$
$$= m \wedge (m + t \cdot q) = R_1$$
$$s' = CRT(s_p, s_q) \cdot \tilde{m}'^r \bmod n$$
$$= CRT(s_p, s_q) \cdot R_1^r \bmod n$$

이 외의 입력 값과 중간 결과 값에 오류가 주입된 경우 [11]의 논문에서의 안전성 분석과 동일하다.

4.2 CRT-2 프로토콜의 대응방법

Algorithm 5의 경우도 Algorithm 4와 동일한 방법으로 k_p 또는 k_q의 오류주입 공격에 대응한다. 기존 CRT-2의 방법에서는 단계 3에서 $\hat{m}$을 계산하기 위해 $\lfloor \frac{((s_p^{e_r} \bmod p) + k_p \cdot p) + ((s_q^{e_r} \bmod q) + k_q \cdot q)}{2} \rfloor$의 연산을 수행하면서 발생 한 문제점을 AND 연산($\wedge$)을 통하여 계산하도록 하였다. 오류가 주입되지 않았을 경우에는

$$\hat{m} = ((s_p^{e_r} \bmod p) + k_p \cdot p) \wedge ((s_q^{e_r} \bmod q) + k_q \cdot q)$$
$$= m \wedge m = m$$

을 가지므로 올바른 서명 값이 생성된다. 또한 기존의 방법과 동일하게 s_p 또는 s_q에 오류가 주입되었을 경우 단계 3에서 $\hat{m}$의 값이 랜덤 한 값이 생성 되므로 공격자는 gcd연산을 통하여 비밀 값을 얻을 수 없다. 만약 k_p

Algorithm 5. CRT-2 프로토콜의 대응방법

Input : A message $m \in Z_n$,
Output : $Sig := m^d \bmod n$.

1. Compute both $k_p = \lfloor \frac{m}{p} \rfloor$ and $k_q = \lfloor \frac{m}{q} \rfloor$.
2. Compute $m^{d_r} \bmod n$ via a conventional CRT speedup as
$$s_p = m^{d_r \bmod (p-1)} \bmod p,$$
$$s_q = m^{d_r \bmod (q-1)} \bmod q.$$
3. A CRT recombination operation and some additional manipulation are employed to compute the required signature as
$$s = CRT(s_p, s_q) \cdot \hat{m}^r \bmod n$$
where
$$\hat{m} = ((s_p^{e_r} \bmod p) + k_p \cdot p) \wedge ((s_q^{e_r} \bmod q) + k_q \cdot q)$$
4. Output s.

또는 k_q에 오류가 주입되었을 경우도 Algorithm 4와 동일한 방법으로 $\hat{m}$의 값이 공격자가 비밀 정보를 알 수 없는 랜덤 한 값으로 결과가 도출된다.

4.3 안전성과 효율성 비교

본 장에서는 기존에 오류주입 공격의 대응 방법으로 제안 되었던 방법과 본 논문에서 제안하는 방법에 대한 안전성과 효율성을 비교한다. 먼저 기존에 소개되었던 논문들의 안전성을 비교하면 아래 표와 같다. 공격 방법은 최근 많이 고려되고 있는 3가지의 공격 방법인 메모리 오류, Modify opcode, Wanger 공격이다. 메모리 오류는 연산 단계의 메모리에 일시적 오류나 지속적 오류가 발생한 경우이다. 메모리의 오류는 랜덤한 값으로 변환하는 것과 메모리를 '0'으로 초기화시키는 것을 고려한다. modify opcode는 연산 단계를 수행하지 않고 넘어가는 오류가 발생하였을 경우를 의미하고, Wanger 공격은 [13] 논문에서 소개한 공격방법이다. (- : 해당사항 없음)

[표 1]에서 알 수 있듯이 본 논문에서 제안하는 방법은 메모리에 랜덤한 값이나, '0'으로 초기화를 시켰을 때나 Modify opcode의 공격에 모두 안전하다.

[표 2]는 연산량을 비교한 것이다. D는 나눗셈, E는 모듈러 지수승 (E_p는 p, q길이에 해당하는 지수승, E_r은 r길이에 해당하는 지수승 연산), C는 CRT, M은 모듈러 곱셈, A는 덧셈 그리고 $\wedge$는 AND , S는 shift 연산을 나타낸다.

두 표에서 볼 수 있듯이 Yen의 방법은 타 방법보다 연산량이 많다. 하지만 Yen 논문이 가지는 장점은 비교

〔표 1〕

방법	안전성		
	메모리 오류	Wagner 공격[13]	Modify opcode
BNP [14]	X1	-	X
BOS [15]	O	X	O
ABFHS [5]	X	-	X
Giraud [16]	O	-	X
Ciet [17]	O	-	X
Kim [12] : modify Ciet	X2	-	O
Kim [12] : modify Giraud	O	-	X
Yen [11]	X	-	O
제안하는 방법	O	-	O

1. d_p, d_q에 랜덤한 오류 주입
2. 단계1, 2에서 오류 발생 + 단계 3에서 a 값 '0'으로 초기화

〔표 2〕

방법	연산량
BNP [14]	$2E_p + E_r + 2C + 2M$
BOS [15]	$2E_p + 3E_r + C$
ABFHS [5]	$2E_p + 2E_r + C$
Giraud [16]	$2E_p + 2C + M$
Ciet [17]	$2E_p + 3E_r + C + 2M + 6A + S$
Yen의 CRT-1[11]	$2D + 4E_p + E_r + C + 3M + 2A$
Yen의 CRT-2[11]	$2D + 4E_p + E_r + C + 3M + 3A$
제안하는 CRT-1	$2D + 4E_p + E_r + C + 3M + 2A + \wedge$
제안하는 CRT-2	

연산을 하지 않는 것과 중간 연산을 넘어가는 오류가 발생하여도 공격자는 비밀 정보를 알 수 없다는 것이다. 그러나 기존의 Yen의 CRT-1, 2 연산 프로토콜의 경우 k_p, k_q에서 오류가 주입 되는 경우에는 공격자의 공격이 가능했지만, 본 논문에서 제안하는 연산 프로토콜의 경우에는 입력 값과 중간 결과 값의 모든 경우에 오류가 주입 되더라도 안전 한 프로토콜이다. 또한 제안하는 대응방법은 기존의 연산량에서 비트 연산인 AND 연산만을 추가적으로 요구한다.

V. 결 론

2003년 Yen이 제안한 오류주입에 안전한 RSA CRT 연산 프로토콜을 FDTC 2006에서 안전하지 않음을 소개하였다. 본 논문에서 제안하는 대응방법은 다른 두 값의 비트 연산의 결과가 랜덤 값이 되는 비트 연산의 특징을 이용하여 오류주입 공격에 대응한다. 2003년 비교연산 대신에 오류 확산을 통한 Yen의 방법의 장점을 그대로 이용하면서 CRT-1의 경우는 추가 메모리와 연산량은 AND 연산 하나만을 추가하여 수행하며, CRT-2의 경우는 AND 연산이 한번 추가되나 덧셈 연산이 한 번 줄어든다. 입력 값과 서명 생성 과정의 중간 결과 값 모두에 오류가 주입되더라도 본 논문에서 제안하는 대응 방법에서는 공격자의 공격이 성공하지 못한다.

참고문헌

[1] J.-J. Quisquater and C. Couvreur, "Fast decipherment algorithm for RSA publickey cryptosystem," Electronics Letters, Vol. 18, No. 21, pp. 905-907, 1982.

[2] Bellcore Press Release, "New threat model breaks crypto codes", Sept 1996, http://www.bellcore.com/PRESS/ADVSRY96/facts.html

[3] D. Boneh, R.A. DeMillo, R.J. Lipton, "On the importance of checking cryptographic protocols for fault", EUROCRYPT'97, Springer-Verlag, LNCS 1233, pp. 37-51, 1997.

[4] ChangKyun Kim, JaeCheol Ha, Sung-Hyun Kim, Seokyu Kim, Sung-Ming Yen and SangJae Moon, "A Secure and Practical CRT-Based RSA to Resist Side Channel Attacks", ICCSA 2004, LNCS 3043, pp. 150-158, 2004.

[5] C. Aumüller, P. Bier, W. Fischer, P. Hofreiter, and J.-P. Seifert, "Fault attacks on RSA with CRT : Concrete results and practical countermeasures," Proceedings of Cryptographic Hardware and Embedded Systems - CHES 2002, LNCS 2523, pp. 260-275, Springer-Verlag, 2003.

[6] M.Joye, A.K.Lenstra and J.-J.Quisquater, "Chinese reamindering based cryptosystem in the presence of faults", Journal of cryptology, 12(4), pp. 241-245, 1999.

[7] A.J. Menezes, P.C. van Oorschot, and S.A.

Vanstone. Handbook of applied cryptography. CRC Press, 1997.

[8] Sung-Ming Yen, Dongryeol Kim, and SangJae Moon, "Cryptanalysis of Two Protocols for RSA with CRT Based on Fault Infection," FDTC 2006, LNCS 4236, pp. 53-51, Springer-Verlag, 2006.

[9] A. Shamir, "How to Check Modular Exponentiation", presented at the rump session of EUROCRYPT'97, May 1997.

[10] A. Shamir, "Method and Apparatus for Protecting Public Key Schemes from Timing and Fault Attacks" US Patent 5991415, 23 Nov. 1999.

[11] S.M. Yen, S.J. Kim, S.G. Lim, and S.J. Moon, "RSA speedup with Chinese remainder theorem immune against hardware fault cryptanalysis", IEEE Trans. on Computers . Special issue on CHES, Vol. 52, No. 4, pp. 461.472, April 2003.

[12] C. Kim, and J.-J.Quisquater "Fault attacks for CRT based RSA : new attacks, new result and new countermeasures", WISTP 2007, LNCS 4462, pp 215-228, Springer-Verlag, 2007.

[13] David Wagner, "Cryptanalysis of a provably secure CRT-RSA algorithm", Proceedings of the 11th ACM conference on Computer and communications security, pp. 92-97, Washington DC, USA, October 25-29, 2004.

[14] A. Boscher, R. Naciri, and E. Prouff, "CRT-RSA Algorithm Protected Against Fault Attacks," Workshop in Information Security Theory and practices-WISTP'07, LNCS Vol. 4462, pp. 237-252, 2007.

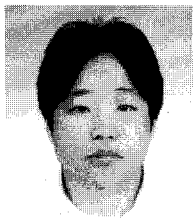
[15] J. Blömer, M. Otto, and J. P. Seifert, "A new CRT-RSA algorithm secure against Bellcore attacks," 10th ACM Conference on Computer and Communications Security, pp. 311-320, 2003.

[16] C. Giraud, "Fault resistant RSA implementation," Fault Diagnosis and Tolerance in Cryptography-FDTC'05, pp. 142-151, 2005.

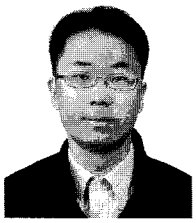
[17] M. Ciet and M. Joye, "Practical fault countermeasures for Chinese Remaindering based RSA," Fault Diagnosis and Tolerance in Cryptography - FDTC'05, pp. 124-131, 2005.

〈著者紹介〉

김 성 경 (Sung-Kyoung Kim) 학생회원
2005년 2월 : 동의대학교 수학과 학사
2005년 3월~2007년 8월 : 고려대학교 정보경영공학전문대학원 석사과정
2007년 8월~현재 : 고려대학교 정보경영공학전문대학원 박사과정
<관심분야> 부채널 공격, 공개키 암호, 암호칩 설계 기술

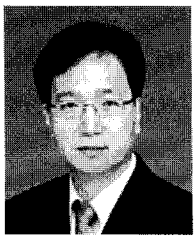


김 태 현 (Tae Hyun KIM) 학생회원
2002년 2월 : 서울 시립대학교 수학과 이학사
2004년 8월 : 고려대학교 정보보호 대학원 공학석사
2005년 2월~현재: 고려대학교 정보경영공학전문대학원 박사과정
<관심분야> 부채널 공격, 공개키 암호 알고리즘, 암호칩 설계 기술



한 동 국 (Dong-Guk Han) 정회원
1999년 : 고려대학교 수학과 졸업(학사)
2002년 : 고려대학교 수학과 석사 (이학석사)
2005년 : 고려대학교 정보보호대학원 박사 (공학박사)
2004년 4월~2005년 4월 : 일본 Kyushu Univ., 방문연구원
2005년 4월~2006년 4월 : 일본 Future Univ.-Hakodate, Post.Doc.
2006년 6월~현재 : 한국전자통신연구원 정보보호연구본부 선임연구원
<관심분야> 공개키 암호시스템 안전성 분석 및 고속 구현, 부채널 분석, RFID/USN 정보보호 기술

박 영 호 (Young-Ho Park) 종신회원
1990년 : 고려대학교 수학과 이학사
1993년 : 고려대학교 수학과 이학석사
1997년 : 고려대학교 수학과 이학박사
2006년 2월~현재 : 세종 사이버 대학교 정교수
<관심분야> 정수론, 공개키 암호, 암호 프로토콜, 부채널 공격



홍 석 희 (SeokHie Hong) 종신회원
1995년 : 고려대학교 수학과 학사
1997년 : 고려대학교 수학과 석사
2001년 : 고려대학교 수학과 박사
1999년 8월~2004년 2월 : (주)시큐리티 테크놀로지스 선임연구원
2003년 3월~2004년 2월 : 고려대학교 시간강사
2004년 4월~2005년 2월 : K.U. Leuven 박사후연구원
2005년 3월~현재 : 고려대학교 정보경영전문대학원 조교수
<관심분야> 대칭키 암호 알고리즘, 공개키 암호 알고리즘, 포렌식