A New Exponentiation Algorithm Resistant to Combined Side Channel Attack

HyungDong Kim[1], YongJe Choi[2], DooHo Choi[2] and JaeCheol Ha[1]*
[1] Hoseo University, Asan, ChungNam, Republic of Korea
karuceace@nate.com, jcha@hoseo.edu
[2] Electronics and Telecommunications Research Institute (ETRI), Daejeon, Republic of Korea
{choiyj, dhchoi}@etri.re.kr

Abstract

Since two different types of side channel attacks based on passive information leakage and active fault injection are independently considered as implementation threats on cryptographic modules, most countermeasures have been separately developed according to each attack type. But then, Amiel *et al.* proposed a combined side channel attack in which an attacker combines these two methods to recover the secret key in an RSA implementation. In this paper, we show that the BNP (Boscher, Naciri, and Prouff) algorithm for RSA, which is an SPA/FA-resistant exponentiation method, is also vulnerable to the combined attack. In addition, we propose a new exponentiation algorithm resistant to power analysis and fault attack as well as the combined attack. The proposed secure exponentiation algorithm can be employed to strengthen the security of CRT-RSA.

Keywords: Side Channel Attack, Fault attack, Combined Attack, Exponentiation, CRT-RSA Algorithm

1 Introduction

Although a cryptographic algorithm is considered very secure against traditional cryptanalysis, it can be vulnerable to side channel attacks due to an implementation flaw. An attacker should monitor the characteristics of power signal or electromagnetic radiation in order to try these side channel attacks. Furthermore, these attacks are attempted under the assumption that an attacker already knows about cryptographic algorithms, implementation conditions, device specifications and so on. From this perspective, an evil inside attacker who can easily manipulate other cryptographic devices would be scarier than outsider.

The RSA [20], one of the most widely used public key cryptographic algorithms, is also susceptible to a variety of different side channel attacks. Among the side channel attacks on RSA, the Power Analysis(PA) attack ,which is divided into Simple Power Analysis (SPA) and Differential Power Analysis (DPA), and Fault Attack (FA) are still threats. Indeed, an attacker measures the power dissipation as a single trace in SPA, manipulates the collected power signals in DPA and extracts the secret information [17, 8, 19]. On the other hand, the goal of an FA is to disturb a hardware module during cryptographic algorithm operation, analyze the faulty information leaked from the target module and finally recover the secret information [5, 18, 12, 4, 16].

Many researchers have developed solutions to defeat PA attacks and FAs on RSA. These protections for secure RSA operation are focused on the modular exponentiation method, which is composed of

Journal of Internet Services and Information Security (JISIS), volume: 3, number: 3/4, pp. 17-27

*Corresponding author: Deptartment of Information Security, Hoseo University, 165 Sechul-Ri, Baebang-Eup, Asan-Si, Chungnam, 336-795 Korea, Tel: +82-(0)41-540-5991, Web: `http://islab.hoseo.ac.kr/jcha` hundreds of multiplications. Among these countermeasures, counteracting SPA and FA at the same time are preferentially taken into account. The BNP(Boscher, Naciri, and Prouff) exponentiation algorithm [6] is a well known countermeasure to simultaneously defeat SPA and FA such as a C-safe Error Attack [23].

While the threats described above are often considered separately, Amiel *et al.* recently proposed a passive and active combined attack on an exponentiation implementation using the Square and Multiply method [2]. To counteract this combined attack, Schmidt *et al.* presented an efficient exponentiation algorithm [21]. Despite their clever design for protecting against the combined attack, Feix and Venelli show that Schmidt *et al.*'s countermeasure has a flaw and propose an improved version [10].

In this paper, we focus on the problem of finding a secure exponentiation algorithm resistant to the many types of side channel attacks, including a combined attack. We first show that the BNP algorithm has a vulnerability under the existing combined attack. Furthermore, we propose a new exponentiation algorithm that is resistant to PA and FA, as well as the combined attack. The proposed exponentiation algorithm can be applied to a CRT-RSA(Chinese Remainder Theorem based RSA) [9] and efficiently used to guarantee security against a CRT-based attack such as the Bellcore attack [5].

The paper is organized as follows. In the next section, we briefly recall the classical RSA cryptosystem, the CRT-RSA version, and several side channel attacks on them. Then, in section 3, we present a new exponentiation algorithm resistant to PA, FA, and the combined attack and give it a security analysis. The CRT-RSA implementation method with a new exponentiation algorithm is given in Section 4. Section 5 concludes the paper.

2 RSA and Side Channel Attacks

2.1 RSA Cryptosystem

The RSA cryptosystem involves a public modulus N, which is the product of two secret primes, p and q. The public exponent e is relatively prime with $(p-1)(q-1)$, and the private key d is the inverse of e modulo $(p-1)(q-1)$. The classical RSA signature of message M is $S = M^d \mod N$.

To speed up this exponentiation, we usually adopt CRT-RSA [9]. In CRT-RSA, we first compute $S_p = M^{d_p} \bmod p$ and $S_q = M^{d_q} \bmod q$, where $d_p = d \bmod (p-1)$ and $d_q = d \bmod (q-1)$. Then, the signature S of message M is computed by the following Garner's recombination procedure:

$$S = CRT(S_p,\ S_q) = ((S_p - S_q) \cdot I_p\ mod\ q) \cdot p + S_p. \tag{1}$$

Here, $I_p = p^{-1} \bmod q$ and can be precomputed to reduce the computational load.

2.2 Power Attack on RSA Algorithm

Among side channel attacks, SPA on RSA retrieves the secret information by measuring the power consumption trace of one operation of the exponentiation algorithm, whereas DPA collects many power traces and applies statistical techniques to get secret information. In this paper, we mainly focus on SPA attacks. More details on DPA attacks and their countermeasures can be found in [19].

We usually implement a modular exponentiation algorithm known as the Square and Multiply algorithm, in which the exponent bits are bit-wisely scanned. However, this algorithm has been known that the secret exponent d can easily be retrieved when execution power consumption of squaring and multiplication are different [1].

To counteract an SPA attack, the Square and Multiply Always algorithm was proposed by Coron [8]. Added to this, Joye and Yen proposed the Montgomery ladder exponentiation algorithm for SPA resistance [14]. However, the Montgomery algorithm is susceptible to the Relative Doubling Attack(RDA) [26] which is a kind of SPA as a special case of the Doubling Attack(DA) [11].

2.3 Fault Attack on RSA Algorithm

The Square and Multiply Always algorithm ensures protection against SPA, but has a weakness with respect to another type of FA, well known as the C-safe Error Attack [23]. In the C-safe Error Attack, if an error is induced on the dummy multiplication that is used when an exponent bit equals 0, an attacker extracts the value of the bit, depending on the correctness or the incorrectness of the final exponentiation result. That is, if the result is correct, then the bit of the secret exponent was 0. On the other hand, if it is not correct, then the bit was 1.

To thwart SPA and FA such as a C-safe Error Attack at the same time, the BNP exponentiation algorithm [6] is proposed. This is a typical Right-to-Left modular exponentiation and uses a coherence test at the end to defeat a CRT-based fault attack on CRT-RSA, the so-called Bellcore attack [5].

The Bellcore attack was originally presented by Boneh *et al.* in 1997. More precisely, when an attacker injects a fault during first exponentiation and erroneous output $\widehat{S_p}$ occurs, the faulty signature $\widehat{S}$ is computed as:

$$\widehat{S} = ((\widehat{S_p} - S_q) \cdot I_p \ mod \ q) \cdot p + \widehat{S_p}. \tag{2}$$

Knowing a correct signature S and a faulty $\widehat{S}$ on the same message M, the attacker can find the secret prime q by calculating the following simple manipulation, which enables the factorization of N:

$$q = gcd((S - \widehat{S}), N). \tag{3}$$

In an enhanced attack proposed by Lenstra, the attacker can succeed with the CRT-based fault attack by using only one faulty signature [18] and public exponent e. That is, the attacker can extract q by computing as follows:

$$q = gcd((\widehat{S}^e - M) \bmod N, N). \tag{4}$$

To defeat these fault attacks, Shamir proposed a redundancy way to compute S_p and S_q and verified the correctness of them before the RSA recombination procedure [22]. In this countermeasure, both $p^* = p \cdot r$ and $q^* = q \cdot r$ are first computed where r is a b-bit random prime(typically, $b = 32$). Two partial signatures, modulo p^* and q^*, are evaluated as follows:

$$S_p{}^* = M^{d_p{}^*} \bmod p^*, \quad S_q{}^* = M^{d_q{}^*} \bmod q^* \tag{5}$$

Here, $d_p{}^* = d \bmod (p-1)(r-1)$ and $d_q{}^* = d \bmod (q-1)(r-1)$. We then check whether $S_p{}^* \equiv S_q{}^* \bmod r$ before returning the final signature. However, this method requires the direct usage of private key d which is not usually known in CRT-RSA and cannot detect faults occurred during RSA recombination procedure [3].

In 2001, Yen *et al.* proposed a different kind of countermeasure using *Fault Infective Computation* method [25]. Unfortunately, their countermeasure was proven to be insecure against another type of fault attack [24]. Thereafter, Giraud proposed a novel CRT-RSA method to defeat SPA and FA. Since this algorithm has regular square and multiply operations based on the Montgomery ladder property [14], we can prevent SPA attack. However, Giraud's algorithm is not secure against Kim and Quisquater fault attack [15] in which an attacker inserts double faults in a CRT-RSA exponentiation.

2.4 Combined Attack on RSA Algorithm

In 2007, Amiel *et al.* first proposed a combined attack on an SPA-resistant implementation of exponentiation using combination of a FA and SPA attack [2]. The authors targeted a classical side channel resistant implementation of RSA using an atomicity-based Square and Multiply algorithm as shown Fig. 1. In this work, if an attacker inserts a fault to bypass a step 2, the $S[0]$ is kept a initial value, typically $S[0] = 0$. With a such fault effect, two different power signal patterns appear during the main loop of exponentiation (step 6 of Algorithm 1). The first one is for squaring $S[0] = S[0] \cdot S[0] \bmod (r_2 \cdot N) = 0 \cdot 0 = 0$ which is performed in case $k = 0$ and the second is for multiply $S[0] = S[0] \cdot S[1] \bmod (r_2 \cdot N) = 0 \cdot M = 0$ in case $k = 1$. Since two different power signal patterns appear according to the exponent bit, the adversary can simply retrieve the secret information d by observing the power trace. Even though the authors propose a countermeasure called *Detect and Derive* based on the infective computation principle, it was shown vulnerable in [21].

```
    Algorithm 1 : SPA-EXP ($M$, $d$, $N$)
    Input: $M$, $d = (d_{n-1}, ..., d_0)$, $N$
    Output: $S = M^d \bmod N$
1.  Get $r_1$, $r_2$ two non zero small random values
2.  $S[0] = 1 + r_1 \cdot N$
3.  $S[1] = M + r_1 \cdot N \bmod (r_2 \cdot N)$
4.  $k = 0$
5.  for $i$ from $n - 1$ to 0 do
6.      $S[0] = S[0] \cdot S[k] \bmod (r_2 \cdot N)$
7.      $k = k \oplus d_i$
8.      $i = i - \neg k$
9.  $S = S[0] \bmod N$
10. return($S$)
```

Figure 1: Atomicity-based Square and Multiply Algorithm Protected against SPA.

In this vein, Schmidt *et al.* presented an efficient exponentiation algorithm to resist this combined attack [21] also based on infective computation. Their countermeasure diffuses the secret exponent when a fault injection is detected and masks intermediate values during exponentiation computation. However Feix and Venelli showed that Schmidt *et al.*'s countermeasure has a flaw and proposed a more secure version of this algorithm [10]. Their fault attack exploits a skip of instruction on the conditional test where the infective calculation replaced the entire exponent by 1. They present a simple countermeasure in which this fixed value is replaced by random values for each words of exponent when a fault injection is detected. Despite the clever improvement, their exponentiation algorithm is somewhat complex and needs some additional computation loads.

3 Exponentiation Resistant to Side Channel Attacks

3.1 Threat of Combined Attack on BNP Exponentiation

The BNP exponentiation method described in Fig. 2 was presented to ensure that no fault has been induced during the execution of the algorithm using SPA-resistant regularity and some coherence checking methods [6]. We now show that the BNP algorithm is also vulnerable to combined attack by the same fault model applied in the atomicity-based Square and Multiply algorithm [7].

Indeed, the fault injection tries to modify the computations in step 1 or 2 in Algorithm 2. If step 1 is bypassed due to fault injection, the $S[0]$ value is kept as an initial value in memory, typically $S[0] = 0$. By such a fault effect, two different computations are performed during the exponentiation loop at step 5: one is $S[0] = 0 \cdot A = 0$, and the other is $S[1] = S[1] \cdot A \neq 0$. In most implementations, two power consumption patterns dependent on the exponent are easily identified by a simple power trace. Therefore an attacker can simply recover the private exponent d by analyzing a power signal collected after injecting an error at step 1. And the attacker can also try at step 2 by skipping the initialization operation on S[1]. In this case, two computations are performed at step 5: one is $S[0] = S[0] \cdot A \neq 0$, and the other is $S[1] = 0 \cdot A = 0$. Furthermore this attack can similarly be applied to a CRT-RSA signature in order to recover the exponent d_p or d_q, because a fault injection check routine is performed after injecting a fault and getting a power trace. The weakness of the BNP exponentiation algorithm against the combined attack originated from the fact that renewal $S[d_i]$ at step 5 is dependent on only the exponent bit d_i; that is, all A values are independent of d_i.

```
     Algorithm 2 : SPA/FA-EXP ($M$, $d$, $N$)
     Input: $M$, $d = (d_{n-1}, ..., d_0)$, $N$
     Output: $S = M^d$ mod $N$ or "Error"
1.   $S[0] = 1$
2.   $S[1] = 1$
3.   $A = M$
4.   for $i$ from 0 to $n-1$ do
5.       $S[d_i] = S[d_i] \cdot A$ mod $N$
6.       $A = A^2$ mod $N$
7.   if ($M \cdot S[0] \cdot S[1] \equiv A$ mod $N$) and ($A \neq 0$) then
8.       return($S[1]$)
9.   else
10.      return("Error")
```

Figure 2: SPA/FA-Resistant BNP Exponentiation.

To prevent this combined attack, we consider a method to generate A related with d_i. In that vein, let us review $S[d_i]$ at step 5. Here we denote d'_j the binary value composed to the $(j-1)$-th bit from the least significant one when $j \geq 1$, $d'_j = (d_{j-1}, ..., d_0)$. In the main loop operation when index i is $(j-1)$, $S[0]$ is the result of exponentiation of M by the complement of d'_j denoted $\overline{d'_j}$, that is, $\overline{d'_j} = 2^j - d'_j - 1$. In addition, $S[1]$ at step 5 equals $M^{d'_j} \bmod N$ and the value of A is $M^{2^j} \bmod N$. We can now find an important result that the product of $S[0]$ and $S[1]$ is always $M^{2^j-1} \bmod N$. Therefore, if the initialization value of S[0] at step 1 becomes M, then we can make a product of $S[0]$ and $S[1]$ to $M^{2^j} \bmod N$ in the $(j-1)$-th loop.

3.2 Proposed Exponentiation Resistant to Combined Attack

The aim of our idea is to replace squaring at step 6 in Algorithm 2 with multiplication of $S[0]$ and $S[1]$. By doing so, we can protect against the combined attack even though one of $S[0]$ and $S[1]$ becomes 0 by fault injection. Our approach is somewhat similar to the idea of the multiplication-only algorithm, the ADD-only scalar multiplication algorithm in Elliptic Curve Cryptography(ECC) [13]. Even though the multiplication-only exponentiation algorithm does not involve squaring, what is needed is a division operation; that is, subtraction in the ECC version. On the other hand, our proposed exponentiation algorithm removes the division operation, which is an annoying and inefficient computation.

Now we propose a secure exponentiation algorithm to resist SPA, FA such as the C-safe Error Attack, and a combined attack as shown Fig. 3. Our algorithm is still resistant to SPA because two multiplications are regularly performed per exponent bit. To detect the fault induction, we use a small random number r (typically 32-bit). We first compute a reference check value c:

$$c = M^{2^n} \bmod r. \tag{6}$$

After performing all n loops for steps 7 and 8, the result of $S[2]$ is used to verify that computations before step 10 are correctly performed. If reference check value c equals $S[2] \bmod r$, and $S[0] \cdot S[1] \bmod N$ is $S[2] \bmod N$, then we return $S[1]$ as a correct signature S.

```
     Algorithm 3 : SPA/FA/CA-EXP ($M$, $d$, $N$, $r$)
     Input: $M$, $d = (d_{n-1}, ..., d_0)$, $N$, $r$
     Output: $S = M^d \bmod N$ or "Error"
1.   $c = M^{2^n} \bmod r$
2.   $S[0] = M$
3.   $S[1] = M$
4.   $S[2] = 1$
5.   $N^* = N \cdot r$
6.   for $i$ from 0 to $n-1$ do
7.       $S[d_i] = S[d_i] \cdot S[2] \bmod N^*$
8.       $S[2] = S[0] \cdot S[1] \bmod N^*$
9.   $S[1] = S[1] \bmod N$
10.  if (($c \equiv S[2] \bmod r$) and ($S[0] \cdot S[1] \equiv S[2] \bmod N$) ) then
11.      return($S[1]$)
12.  else
13.      return("Error")
```

Figure 3: Proposed Side Channel Resistant Exponentiation.

Correctness. We now consider the correctness of the final result of the proposed exponentiation algorithm. Let us review the contents of the main loop of Algorithm 3. The value $S[1]$ is the result of exponentiation of M using d, $M^d \bmod N^*$. On the other hand, $S[0]$ is the result of exponentiation of M by $\overline{d}+1$, $M^{\overline{d}+1} \bmod N^*$. Here, we denoted $\overline{d}$ as the complement of d; that is, $\overline{d} = 2^n - d - 1$. Since the value $S[2]$ is the product of $S[0]$ and $S[1]$, it satisfies:

$$S[2] = S[0] \cdot S[1] = M^{2^n - d} \cdot M^d = M^{2^n} \bmod N^*. \tag{7}$$

Therefore, $S[2] \bmod r$ in step 10 equals $M^{2^n} \bmod r$. To ensure that none of the modular multiplications at steps 7 and 8 were corrupted, we perform a check operation between two values, c and $S[2] \bmod r$, after all loop iterations. Since $S[2] \bmod r$ equals the reference check value c, we can verify that $S[0]$ and $S[1]$ are correctly computed without injecting any fault, and return $S[1]$.

Security Analysis. Two modular multiplication structures in the main loop of the proposed algorithm make it resistant to the SPA attack. Furthermore, we adopt a message randomization to defeat a DPA attack by computing multiplication modulo N^*. Concerning the C-safe Error Attack, even though an attacker inserts a fault in a dummy operation at step 7 in the case of $d_i = 0$, the attacker can't obtain the computational result due to a coherence test in step 10. So, our algorithm resists against the C-safe Error Attack.

The security of Algorithm 3 with respect to a combined attack is deduced from the coherence test in step 10. On the basis of the same attack model used in Amiel *et al.*'s proposal, we assume that a fault is inserted to bypass the instruction at step 2. Then, since $S[0]$ is initialized by zero, the $S[0]$ at step 7 in the main loop is always zero. By this effect, $S[2]$ at step 8 and $S[d_i]$ in the next loop become zero. Since $S[2]$ is consequently zero after the final loop operation, the coherence test in step 10 is not established. As our proposed algorithm finally returns an error message to the attacker, it can also defeat the combined attack. The attack model on $S[1]$ at step 3 gives the same result as the case of $S[0]$ at step 2. Here, we can carefully assume that a powerful adversary is able to induce a fault in the exponent d. Considering this case in our exponentiation algorithm, verification of the correct usage of d should be performed using an extra checking method.

Efficiency. Compared to the BNP exponentiation, this method requires adding only a small modular exponentiation in step 1. This is a computing procedure of a reference check value. The main computations of our algorithm are $2n$-modular multiplications in step 7 and 8. The number of modular multiplications in these steps is same with one of BNP algorithm. However, the multiplication of modulo N^* instead of N may increase the time complexity. Nevertheless, the message randomization is absolutely required in most exponentiation methods. Indeed, the DPA countermeasure in the BNP exponentiation algorithm was not adopted. So, when the r having about 32-bits is very small compared with modulus N at about 1024-bits, the additional time complexity is negligible. Since the modular multiplication algorithm takes $O(n^2)$ bit operations for two n-bit integers, the proposed algorithm additively requires operation time of about $((1024+32)/1024)^2 \approx 6.3\%$ compared to the BNP algorithm without a DPA countermeasure. And, the additional memory consumption for storing a reference check value c and a random number r is required, compared with the BNP exponentiation.

4 Application to CRT-RSA

Even if the two modular exponentiations in the CRT-RSA operation have not been compromised, the entire correctness of CRT-RSA is not guaranteed. A secure modular exponentiation described above (see Fig. 3) is used to prevent faults during two exponentiations of a CRT-RSA algorithm. To defeat a CRT-based Bellcore attack, we need to prove that the Garner's recombination step is correct.

The proposed CRT-RSA algorithm shown as Fig. 4 uses a similar principle to the method presented in [6]. Here, we denote l by the bit-length of the secret exponent p and q. Instead of returning one result of exponentiation in the SPA/FA/CA-EXP algorithm of Fig. 3, it returns three variables, $S[0]$, $S[1]$ and $S[2]$. On the basis of Garner's algorithm, the recombination procedure of the CRT-RSA is performed at step 4 using the values S_p and S_q, which are returned through a variable $S[1]$. Then we perform two coherence tests with respect to modulo p and q to check that the recombination in step 4 was not disturbed. In coherence tests in steps 5 and 7, the values S'_p, S'_q, T_p, and T_q returned through two variables $S[0]$ and $S[2]$ are used to verify whether the final signature S is correctly computed or not.

Correctness. We now consider the correctness of the coherence tests in steps 5 and 7 of the proposed CRT-RSA algorithm. Due to the Chinese Remainder Theorem, the result of the recombination is $S = M^d \mod N$ in step 4. The value S'_p and T_p are $M^{2^l - d_p} \mod p^*$ and $M^{2^l} \mod p^*$ respectively, where $p^* = p \cdot r$. Thus, multiplying S'_p with modulo p by the signature S in step 5, we get $S \cdot S'_p = M^{d_p} \cdot M^{2^l - \bar{d}_p} \mod p$, that is, $S \cdot S'_p = M^{2^l} \mod p$. The value of $T_p \mod p$ in step 5 is M^{2^l}. Consequently, if no error occurs

```
    Algorithm 4 : Secure CRT-RSA ($M$, $p$, $q$, $d_p$, $d_q$, $I_p$)
    Input: $M$, $p$, $q$, $d_p$, $d_q$, $I_p$
    Output: $S = M^d \bmod N$ or "Error"
1.  Generate a 32-bit random number $r$
2.  $(S'_p, S_p, T_p) = SPA/FA/CA-EXP(M, d_p, p, r) = (M^{2^l - d_p} \bmod p^*,\ M^{d_p} \bmod p,\ M^{2^l} \bmod p^*)$
3.  $(S'_q, S_q, T_q) = SPA/FA/CA-EXP(M, d_q, q, r) = (M^{2^l - d_q} \bmod q^*,\ M^{d_q} \bmod q,\ M^{2^l} \bmod q^*)$
4.  $S = ((S_q - S_p) \cdot I_p \bmod q) \cdot p + S_p$
5.  if $(S \cdot S'_p \neq T_p \bmod p)$
6.      return("Error")
7.  if $(S \cdot S'_q \neq T_q \bmod q)$
8.      return("Error")
9.  return($S$)
```

Figure 4: CRT-RSA Signature Resistant to Bellcore Attack.

during the operation of the CRT-RSA signature algorithm, then the three values S, S'_p, and T_p must satisfy the equality of step 5. In a similar way, the verification of step 7 is clearly a coherence test with modulo q.

Security Analysis. To guarantee correctness of the recombination step, we assume that the value of S_q is corrupted by a fault injection. Then the faulty signature is as follows:

$$\widehat{S} = ((\widehat{S_q} - S_p) \cdot I_p \bmod q) \cdot p + S_p. \tag{8}$$

In this case, since $\widehat{S} \cdot S'_p \bmod p$ in step 5 equals $T_p \bmod p$, this coherence test is passed. However, considering that the value $\widehat{S} \bmod q$ is not S_q, the coherence test $\widehat{S} \cdot S'_q \equiv T_q \bmod q$ in step 7 does not satisfy the equality.

On the other hand, if the value of S_p is corrupted, the faulty signature is as follows:

$$\widehat{S} = ((S_q - \widehat{S_p}) \cdot I_p \bmod q) \cdot p + \widehat{S_p}. \tag{9}$$

So, considering that the value $\widehat{S} \bmod p$ is not S_p, the coherence test $\widehat{S} \cdot S'_p \equiv T_p \bmod p$ in step 5 does not satisfy the equality. Consequently, the fault injected in S_p or S_q is detected in two coherence tests of step 5 and 7.

To ensure the validity of the input parameters such as secret exponent d_p, d_q, message M, secret primes p, q and random number r used in the SPA/FA/CA-EXP algorithm of Fig. 4, the extra check mechanism may be used. The validation check methods of these input parameters are outside the scope of this paper.

Efficiency. Our method requires a Garner's recombination and two modular multiplications. This time complexity is less than the BNP RSA signature using CRT. The BNP CRT-RSA method requires adding two recombinations and some modular multiplication according to modulo N instead of p or q. Furthermore, the check reference values at steps 2 and 3 of Algorithm 4 are the same when the random number r is commonly used. So, the number of the computing check reference value can be reduced to one in our CRT-RSA algorithm.

5 Conclusion

In this paper, we present a novel exponentiation algorithm that is resistant to SPA, DPA, the C-safe Error Attack, and especially a combined attack. By generating a low-overhead reference check value and comparing it with a computational result, we can verify the correctness of performing an exponentiation algorithm. Compared to the previous BNP exponentiation algorithm that is vulnerable to the combined side channel attack, the overhead of timing and memory added by strengthening side channel security is reasonable.

In addition, we show that our proposed exponentiation algorithm can be applied in a CRT-RSA signature. The additional values returned after two exponentiation operations are used to check the recombination procedure of CRT-RSA. Only one Garner's recombination and two modular multiplications are needed aside from two exponentiations in the CRT-RSA algorithm. As a result, our exponentiation algorithm is well suited to implementing the RSA signature system in low-resource devices

Acknowledgments

This work was supported by the KLA-SCARF project, the ICT R&D program of ETRI (Research on Key Leakage Analysis and Response Technologies)

References

[1] F. Amiel, B. Feix, M. Tunstall, C. Whelen, and W. Marnane. Distinguishing Multiplications from Squaring Operations. In *Proc. of the 15th International Workshop on Selected Areas in Cryptography (SAC'08), Sackville, New Brunswick, Canada, LNCS*, volume 5381, pages 346–360. Springer-Verlag, August 2009.

[2] F. Amiel, K. Villegas, B. Feix, and L. Mercel. Passive and Active Combined Attacks: Combining fault attacks and side channel analysis. In *Proc. of the 4th Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC'07), Vienna, Austria*, pages 92–102. IEEE, September 2007.

[3] C. Aumüller, P. Bier, W. Fischer, P. Hofreiter, and J. Seifert. Fault attacks on RSA with CRT: concrete results and practical countermeasures. In *Proc. of the 4th International Workshop on Cryptographic Hardware and Embedded Systems (CHES '02), Redwood Shores, CA, USA, LNCS*, volume 2523, pages 260–275. Springer-Verlag, August 2002.

[4] J. Blömer, M. Otto, and J. Seifert. A new CRT-RSA algorithm secure against Bellcore attacks. In *Proc. of the 10th ACM Conference on Computer and Communications Security (CCS'03), Washington, DC, USA*, pages 331–320. ACM, October 2003.

[5] D. Boneh, R. DeMillo, and R. Lipton. On the importance of checking cryptographic protocols for faults. In *Proc. of the International Conference on the Theory and Application of Cryptographic Techniques (EUROCRYPT '97), Konstanz, Germany, LNCS*, volume 1233, pages 11–15. Springer-Verlag, May 1997.

[6] A. Boscher, R. Naciri, and E. Prouff. CRT-RSA algorithm protected against fault attacks. In *Proc. of the International Workshop on Information Security Theory and Practices. Smart Cards, Mobile and Ubiquitous Computing Systems (WISTP '07), Heraklion, Crete, Greece, LNCS*, volume 4462, pages 237–252. Springer-Verlag, May 2007.

[7] B. Chevallier-Mames, M. Ciet, and M. Joye. Low-cost solutions for preventing simple side-channel analysis: side-channel atomicity. *IEEE Trans. on Computers*, 53(6):760–768, June 2004.

[8] J. Coron. Resistance against differential power analysis for elliptic curve cryptosystems. In *Proc. of the 1st International Workshop on Cryptographic Hardware and Embedded Systems (CHES '99), Worcester, MA, USA, LNCS*, volume 1717, pages 292–302. Springer-Verlag, August 1999.

[9] C. Couvreur and J. Quisquater. Fast decipherment algorithm for RSA public-key cryptosystem. *IEE Electronics Letters*, 18(21):905–907, October 1982.

[10] B. Feix and A. Venelli. Defeating with fault injection a combined attack resistant exponentiation. In *Proc. of the 4th Constructive Side-Channel Analysis and Secure Design (COSADE'13), Paris, France, LNCS*, volume 7864, pages 32–45. Springer-Verlag, March 2013.

[11] P. Fouque and F. Valette. The doubling attack- why upwards is better than downwards. In *Proc. of the 5th International Workshop on Cryptographic Hardware and Embedded Systems (CHES '03), Cologne, Germany, LNCS*, volume 2779, pages 269–280. Springer-Verlag, September 2003.

[12] C. Giraud. Fault resistant RSA implementation. In *Proc. of the 2nd Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC'05), Edinburgh, Scotland, UK*, pages 142–151, September 2005.

[13] M. Joye. Highly regular right-to-left algorithms for scalar multiplication. In *Proc. of the 9th International Workshop on Cryptographic Hardware and Embedded Systems (CHES '07), Vienna, Austria, LNCS*, volume 4727, pages 135–147. Springer-Verlag, September 2007.

[14] M. Joye and S. Yen. The Montgomery powering ladder. In *Proc. of the 4th International Workshop on Cryptographic Hardware and Embedded Systems (CHES '02), Redwood Shores, CA, USA, LNCS*, volume 2523, pages 291–302. Springer-Verlag, August 2002.

[15] C. Kim and J. Quisquater. Fault attacks for CRT based RSA: New attacks, new results, and new countermeasures. In *Proc. of the International Workshop on Information Security Theory and Practices. Smart Cards, Mobile and Ubiquitous Computing Systems (WISTP '07), Heraklion, Crete, Greece, LNCS*, volume 4462, pages 215–228. Springer-Verlag, May 2007.

[16] C. Kim and J. Quisquater. How can we overcome both side channel analysis and fault attacks on RSA-CRT. In *Proc. of the 4th Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC'07), Vienna, Austria*, pages 21–29. IEEE, September 2007.

[17] P. Kocher, J. Jaffe, and B. Jun. Differential power analysis. In *Proc. of the 19th International Cryptology Conference (CRYPTO '99), Santa Barbara, California, USA, LNCS*, volume 1666, pages 388–397. Springer-Verlag, August 1999.

[18] A. Lenstra. Memo on RSA signature generation in the presence of faults. Manuscript, September 1996.

[19] T. Messerges, E. Dabbish, and R. Sloan. Power analysis attacks on modular exponentiation in smart cards. In *Proc. of the 1st International Workshop on Cryptographic Hardware and Embedded Systems (CHES '99), Worcester, MA, USA, LNCS*, volume 1717, pages 144–157. Springer-Verlag, August 1999.

[20] R. Rivest, A. Shamir, and L. Adelman. A method for obtaining digital signature and public key cryptosystems. *Communications of ACM*, 21(2):120–126, February 1978.

[21] J. Schmidt, M. Tunstall, R. Avanzi, I. Kizhvatov, and D. Oswald. Combined Implementation attack resistant exponentiation. In *Proc. of the 1st International Conference on Cryptology and Information Security in Latin America (LATINCRYPT'10), Puebla, Mexico, LNCS*, volume 6212, pages 305–322. Springer-Verlag, August 2010.

[22] A. Shamir. Method and apparatus for protecting public key schemes from timing and fault attack. United State Patent 5,991,415, November 1999.

[23] S. Yen and M. Joye. Checking before output may not be enough against fault-based cryptanalysis. *IEEE Trans. on Computers*, 49(9):967–970, September 2000.

[24] S. Yen, D. Kim, and S. Moon. Cryptanalysis of two protocols for RSA with CRT based on fault infection. In *Proc. of the 3rd International Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC'06), Yokohama, Japan, LNCS*, volume 4236, pages 53–61. Springer-Verlag, 2006.

[25] S. Yen, S. Kim, S. Lim, and S. Moon. RSA speedup with residue number system immune against hardware fault cryptanalysis. In *Proc. of the 4th International Conference on Information Security and Cryptology (ICISC'01), Seoul, Korea, LNCS*, volume 2288, pages 397–413. Springer-Verlag, December 2001.

[26] S. Yen, L. Ko, S. Moon, and J. Ha. Relative doubling attack against Montgomery ladder. In *Proc. of the 8th International Conference on Information Security and Cryptology (ICISC'05), Seoul, Korea, LNCS*, volume 3935, pages 117–128. Springer-Verlag, December 2005.

Author Biography

HyungDong Kim is currently a master course student working in department of information security at Hoseo University, Asan, Rep. of Korea. His main research areas are smartphone security, side channel analysis, and wireless network security. He received the bachelor in dept. of information security from Hoseo University in 2012.

YongJe Choi received his BSEE and MS from Cheonnam National University, Kwangju, Rep. of Korea in 1996 and 1999, respectively. He is a senior member of technical staff at ETRI, Deajeon, Rep. of Korea. His research interests include side channel VLSI design, crypto processor design, side channel analysis, and information security.

DooHo Choi received the BS in mathematics from Sungkyunkwan University, Seoul, Korea, in 1994, and the MS and PhD in mathematics from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 1996, 2002, respectively. He has been a senior researcher at ETRI, Daejeon, Korea, since January 2002. His current research interests are side-channel analysis and its resistant crypto design, security technologies of RFID and wireless sensor network, lightweight cryptographic protocol/module design, and cryptography based on non-commutativity. He was an editor of the ITU-T Rec. X.1171.

JaeCheol Hareceived the BE, ME, and PhD in electronics engineering from Kyungpook National University, Rep. of Korea, in 1989, 1993, and 1998, respectively. He is currently a full professor of the department of information and security at Hoseo University, Asan, Rep. of Korea. During 1998 to 2006, he also worked as a professor in the department of information and communication at Korea Nazarene University, Cheonan, Korea. In 2006, he was a visiting researcher at the Information Security Institute of Queensland University of Technology, Australia. His research interests include network security, smart card security, crypto chip design, and side-channel attacks.