Fast signing method in RSA with high speed verification

Kim Gyu-Chol*, Jong Yong-Bok
Kim Chaek University of Technology, Pyongyang, DPR of Korea
* Corresponding Author. Tel.: 0085023811811; Fax: 0085023814410;
E-mail address: kgc841110@star-co.net.kp; P.O. Box: 60 Kyogu Pyongyang

***Abstract*— In this paper, we propose the method to speed up signature generation in RSA with small public exponent. We first divide the signing algorithm into two stages. One is message generating stage and the other is signing stage. Next, we modify the RSA signature so that the bulk of the calculation cost is allocated to message generating stage. This gives the possibility to propose the RSA signature schemes which have fast signature generation and very fast verification. Our schemes are suited for the applications in which a message is generated offline, but needs to be quickly signed and verified online.**

***Keywords*—RSA, rebalanced RSA, signature generation, signature verification**

1. INTRODUCTION

RSA[1] is a public key cryptosystem based on the difficulty of integer factorization problem, where n-bit composite number $N(=pq: p$ and q are the $n/2$-bit prime numbers) is used as a modulus number. The public key e and private key d of RSA satisfy the following equation.

$$e\,d \equiv 1\ mod\ (p-1)(q-1) \tag{1}$$

To sign a message $m \in Z_N^*$, the signer calculates $s = m^d\ mod\ N$. Verification of a signature s on a message m is carried out by checking whether or not $m = s^e\ mod\ N$.

It is easy to increase the signature verification speed by using small public exponent (e.g., $e = 3$ or $e = 65537$) in RSA In this case, d is full sized (on the order of N) and so, most of calculation costs are allocated to signature generation. CRT (Chinese Remainder Theorem) can be used[2] for the speed up of signature generation, but signature generation is still not so fast.

From this, many researches have been done to increase the signature generation speed by transferring the calculation costs to signature verification.

In [3], [6], [8] and [9], it is proved that RSA with small private exponent d is insecure and so, rebalanced RSA that reduces the CRT exponents $d_p(= d\ mod\ (p-1))$ and $d_q(= d\ mod\ (q-1))$ instead of d has been proposed in [10].

More recent researches [12, 13, 16, 17] have occurred to propose the variants of rebalanced RSA which allow the cost of signature generation and verification to be balanced. However, for the following problems, there is no significant improvement to speed up RSA signature generation in practice.

First, the key generation schemes of [12], [13], [16] and [17] are not practical, because they cannot use the typical prime generation module.(mentioned in Section2.2.)

Second, it becomes to be difficult to reduce the CRT exponents even if public exponent is full sized (on the order of N), because powerful lattice based attacks[26, 27, 28] have been proposed in recent years.

Note. In RSA, it is difficult to reduce both public exponent and CRT exponents after selecting the proper prime numbers[12, 13, 16, 17]. And referring to [10], it is possible to use 224-bit CRT exponents, but referring to [27], it is not possible to use CRT exponents below 250-bit for 2048-bit modulus in rebalanced RSA with full sized public exponent

From the above considerations, we focused on making both signature generation and verification high speed in RSA.

For this purpose, we modified RSA with $e = 65537$(noted as typical RSA) as follows.

We converted the private exponent $d = 65537^{-1}\ mod\ (p-1)(q-1)$ into h, d_1 and d_0 by modifying the RSA equation to

$$e(hd_1 + d_0) \equiv 1\ mod\ (p-1)(q-1) \tag{2}$$

Note. In this case, $hd_1 + d_0 \equiv d\ mod\ (p-1)(q-1)$ is satisfied and so, $h_p d_{1p} + d_{0p} \equiv d_p\ mod\ (p-1)$ and $h_q d_{1q} + d_{0q} \equiv d_q\ mod\ (q-1)$ are satisfied for $d_p = d\ mod\ (p-1), d_q = d\ mod\ (q-1), d_{0p} = d_0\ mod\ (p-1), d_{0q} = d_0\ mod\ (q-1), d_{1p} = d_1\ mod\ (p-1), d_{1q} = d_1\ mod\ (q-1), h_p = h\ mod\ (p-1)$ and $h_q = h\ mod\ (q-1)$.

In the signing stage, public parameter h is additionally used and signature generation protocol is changed as follows.

First, message generator calculates $m_1 = H(m)^h mod\ N$ from plaintext m and sends $m||m_1$ to signer. (H denotes full domain hash function and $||$ denotes concatenation.)

Next, signer calculates signature $s = H(m)^{d_0} {m_1}^{d_1}\ mod\ N(= H(m)^{(hd_1+d_0)}\ mod\ N)$ by using private key d_0 and d_1 instead of original private key d and return $m||s$. In this case, calculation of s can be done fast by CRT as follows.

Step1. Calculate $m_{0p} = H(m) mod\ p, m_{0q} = H(m) mod\ q, m_{1p} = m_1 mod\ p$ and $m_{1q} = m_1 mod\ q$.

Step2. Calculate $s_p = {m_{0p}}^{d_{0p}} {m_{1p}}^{d_{1p}} mod\ p(= {m_{0p}}^{h_p d_{1p} + d_{0p}} mod\ p)$ and $s_q = {m_{0q}}^{d_{0q}} {m_{1q}}^{d_{1q}} mod\ q(= {m_{0q}}^{h_q d_{1q} + d_{0q}} mod\ q)$.

Step3. Calculate $s = \left(\left((s_p - s_q)(q^{-1} mod\ p)\right) mod\ p\right) q + s_p$.

Calculation of $s = {m_0}^{d_0} {m_1}^{d_1}\ mod\ N, s_p = {m_{0p}}^{d_{0p}} {m_{1p}}^{d_{1p}}\ mod\ p$ and $s_q = {m_{0q}}^{d_{0q}} {m_{1q}}^{d_{1q}}\ mod\ q$ can be done fast by following Algorithm1[5, Algorithm14.88].

Algorithm1. Simultaneous multiple exponentiation algorithm

Input: $a, b \in G$ and a positive integers $x = (x_t x_{t-1} \cdots x_1 x_0)_2$ and $y = (y_t y_{t-1} \cdots y_1 y_0)_2$.

Output: $a^x b^y$.

Step1. $z = x\&y, u = x \oplus z, v = y \oplus z$.

Step2. $A = a, B = b, C = ab\ , D = 1, i = t + 1$.

Step3. $i = i - 1, D = D \cdot D$.

Step4. if $z_i = 1$ then $D = D \cdot C$ else if $u_i = 1$ then $D = D \cdot A$ else if $v_i = 1$ then $D = D \cdot B$.

Step5. if $i \geq 1$ then go to 3.

Step6. return D.

(x & y denotes the bitwise AND of x and y, $x \oplus y$ denotes the bitwise XOR of x and y.)

The verification (checking whether or not $s^e mod\ n = H(m)$) is identical to typical RSA.

The above modification gives the possibility to speed up signature generation in the typical RSA when message generator and signer are not identical(Message generator is not the signer in some applications such as X.509 public key certificate). In this case, signature generation speed and security are influenced by the selection of h, d_0 and d_1. (In [20], an attack to find d from h, e and N when d_0 and $d_1 (d = hd_1 + d_0)$ are the small unknown integers has already been presented.) Hence, many considerations are needed for the selection of h, d_0 and d_1.

In this paper, we describe two selection schemes of h, d_0 and d_1. One is to increase signature generation speed without effect on the security. (If CRT is not used in signature generation, $d = hd_1 + d_0, h = 2^{\lceil n/2 \rceil}, 0 < d_0, d_1 < h$; else $d_p = hd_{1p} + d_{0p}, d_q = hd_{1q} + d_{0q}, h = 2^{\lceil n/4 \rceil}, 0 < d_{0p}, d_{1p}, d_{0q}, d_{1q} < h$).

Note. $\lceil x \rceil$ denotes the smallest integer greater than or equal to x.

And the other is the scheme ($d \equiv hd_1 + d_0\ mod\ (p-1)(q-1)$: d_0 and d_1 are the full sized private exponents which have small CRT exponents) to obtain extremely fast signature generation.

This paper is organized as follows. In Section 2, we briefly review the rebalanced RSA and small CRT exponent attacks. In Section 3, we propose two fast variants of RSA signature(Scheme1 and Scheme2) and analyze their security. In section 4, we present the performance comparison between the proposed schemes and the other RSA variants. Finally we conclude this paper in Section 5.

2. REBALANCED RSA AND ITS VARIANTS

2.1. Rebalanced RSA

Rebalanced RSA[3, 10] is a variant of RSA designed to speed up signature generation by reducing CRT exponents d_p and d_q instead of private exponent d. In key generation of rebalanced RSA, d which has the small CRT exponents (i.e., $d_p, d_q < n^\delta, \delta < 1/2$) is first selected and then, e is selected as a modular inverse of d. Hence, e is usually full sized (i.e., $\alpha = log_N e, \alpha \approx 1$) [10].

Signature generation and verification are identical to CRT-RSA and so, RSA –FDH[4] and RSA-PSS[4] can be straightly applied to rebalanced RSA under the assumption that it is computationally secure (i.e., δ is enough large to be secure) from the small CRT exponent attacks.

2.2. Variants of rebalanced RSA and related attacks

Rebalanced RSA has achieved the fast signature generation at the cost of a significant loss of verification performance because public exponent e is full sized. Hence, the schemes to speed up signature generation without paying high price for verification (i.e., rebalanced RSA schemes with small public exponent) have been proposed with the small CRT exponent attacks related to α[12, 13, 16, 17]. However, their key generation cannot use the typical prime generation module, because primes p and q are generated based on the selection of e and d. For such a reason, the key generation schemes of [12], [13], [16] and [17] are not widely used in practice and usually used to describe the small CRT exponent attacks.

Note. Much of RSA key generation time is consumed in generating prime numbers. Hence, fast and safe prime generation modules have been developed and used typically in RSA key generation. See the Open SSL RSA key generation for more details.

In other words, many researches [12, 13, 15, 16, 17, 19, 26, 27, 28] for rebalanced RSA have focused on the small CRT exponent attacks rather than practical key generation. Especially, in recent years, more efficient attacks based on Coppersmith's lattice reduction technique have been proposed [21, 22, 23, 24, 25, 26, 27, 28].

Among the above attacks, we considered the attacks applicable to rebalanced RSA variants which have two balanced primes as follows. (That is, attack of [11], [21], [22], [23], [24], [25], [26, Section3, 5], [27, Section3, 5] and [28] are not considered in this paper.)

In RSA, following equations are satisfied for the CRT exponents d_p and d_q.

$$ed_p \equiv 1\ mod\ (p-1) \tag{3}$$

$$ed_q \equiv 1\ mod\ (q-1) \tag{4}$$

From Equation (3), for any plaintext $m\left(m \in Z_p^*\right)$, $m^{ed_p} \equiv m \bmod p$ is satisfied and so, $gcd(M,n) = p$ is satisfied for $M = (m^{ed_p} - m) \bmod n$.

Hence, Attack1 is resulted in finding small root x such that $gcd(A^x \bmod n - m, n) > 1$ where $A = m^e \bmod n$ in rebalanced RSA. Attack1 is not related to α and can factor the modulus n in time $O(z^{1/2} logz)$ [10, 18, Attack8.1]. In this case, $z = min\left(d_p, d_q\right)$.

In RSA, $ed_p + k_p - 1 = k_p p$ and $ed_q + k_q - 1 = k_q q$ are satisfied from Equation (3) and (4). Multiplying above equations together, following Equation (5) is yielded.

$$e^2 d_p d_q + ed_p\left(k_q - 1\right) + ed_q\left(k_p - 1\right) - (n-1)k_p k_q - k_p - k_q + 1 = 0 \tag{5}$$

And reducing Equation (5) modulo e, following Equation.(6) is yielded.

$$(n-1)k_p k_q + k_p + k_q - 1 = 0 \ \bmod e \tag{6}$$

Hence, Attack2 factors the modulus by finding small root (x_1, x_2, x_3, x_4) of the equation $f(x_1, x_2, x_3, x_4) = 0$ given by

$$f = e^2 x_1 x_2 + e x_1 x_4 - e x_1 + e x_2 x_3 - e x_2 - (n-1) x_3 x_4 - x_3 - x_4 + 1 \tag{7}$$

with monomials $1,\ x_1,\ x_2,\ x_3,\ x_4,\ x_1x_2, x_1x_4, x_2x_3, x_3x_4$ and small root

$$\left(x_1^{(0)}, x_2^{(0)}, x_3^{(0)}, x_4^{(0)}\right) = \left(d_p, d_q, k_p, k_q\right), \text{ with } \begin{cases} |x_1^{(0)}| < X_1 = n^{\delta}, \\ |x_2^{(0)}| < X_2 = n^{\delta}, \\ |x_3^{(0)}| < X_3 = n^{\alpha+\delta-1/2}, \\ |x_4^{(0)}| < X_4 = n^{\alpha+\delta-1/2}, \end{cases}$$

for some known upper bounds X_j, for $j = 1, \ldots, 4$.

And Attack3 factors the modulus by finding small root (x, y) of the modular equation $f_e(x, y) = 0$ given by

$$f_e = (n - 1)x\, y + x + y - 1 \ (mod\ e) \tag{8}$$

with monomials $1, x, y, xy$ and small root

$$(x^{(0)}, y^{(0)}) = \left(k_p, k_q\right), \text{ with } \begin{cases} |x^{(0)}| < X = n^{\alpha+\delta-1/2}, \\ |y^{(0)}| < Y = n^{\alpha+\delta-1/2}, \end{cases}$$

for some known upper bounds X and Y.

Note. It is trivial that $p(q)$ can be found from d_p and $k_p(d_q$ and $k_q)$. And if $e > n^{1/4}$, p can be found from k_p(more precisely, from $\tilde{p}(= 1 - k_p^{-1} \bmod e)$)[13,Thorem1], because $p \equiv 1 - k_p^{-1}\ (mod\ e)$ is satisfied.

Lattice based method [14, 16] is used to solve the Equation(7), while there are two kinds of methods to solve the Equation (8). One is lattice based method [13,Section5.2, 16,AppendixB,18] and the other is continued fraction based method[13, Section5.1]. And continued fraction based method yields the same asymptotic security requirement as the lattice based method[13,Section5.2,16,AppendixB].

For the success of the Attack2, it is necessary to satisfy the following Equation (9) and (10)[16][18].

$$\delta < \frac{(5 + 20\tau + 18\tau^2) - 4\alpha(1 + 4\tau + 3\tau^2)}{2(1+\tau)(7 + 21\tau + 12\tau^2)} - \epsilon, \qquad (1/2 \le \alpha \le 1) \tag{9}$$

$$\delta < \frac{(5 + 20\tau + 27\tau^2 + 12\tau^3) - \alpha(4 + 16\tau + 30\tau^2 + 24\tau^3)}{2(1+\tau)(7 + 21\tau + 12\tau^2)} - \epsilon, \quad (1/6 \le \alpha < 1/2) \tag{10}$$

And for the success of the Attack3, it is necessary to satisfy the following Equation (11) [13, 16, Appendix B].

$$\delta < \frac{1}{2} - \frac{2}{3}\alpha - \epsilon, \qquad (1/4 \le \alpha \le 3/4) \tag{11}$$

From Equation (9), (10) and (11), for the small public exponent, Attack2 is stronger and for the large public exponent, Attack3 is stronger with a crossover point at $\alpha \approx 0.375$(See [16, Section7.1, 18, Figure8.1] for more details.). And, when α approaches to 1, Attack1 is the strongest among the attacks above for the 2048 bits modulus. (Of course, referring to Theorem1 of [16], Attack2 becomes stronger when the size of the modulus increases.) Until following Attack4 was proposed, Attack 1, 2 and 3 had been known to be stronger [26, 27] than other small CRT exponent attacks and so, we described only Attack 1, 2 and 3.

From Equation(3) and (4), following Equations are obtained.

$$ed_q p = p + k_q(n - p) = n + \left(k_q - 1\right)(n - p)$$
$$ed_p q = q + k_p(n - q) = n + (k_p - 1)(n - q)$$
$$\left(k_q - 1\right)\left(k_p - 1\right) = k_q k_p n\ (mod\ e)$$
$$(n-1)k_q\left(k_p - 1\right) + nk_q + \left(k_p - 1\right) = 0\ (mod\ e)$$
$$(n-1)k_p\left(k_q - 1\right) + nk_p + \left(k_q - 1\right) = 0\ (mod\ e)$$

Hence, Attack4 is resulted in finding the root $(x_{p,1}, x_{q,1}, x_{p,2}, x_{q,2}, y_p, y_q)$ of the modular equations $f_{p,1}\left(x_{p,1}, y_p\right) = f_{q,1}\left(x_{q,1}, y_q\right) = f_{p,2}\left(x_{p,2}, y_p\right) = f_{q,2}\left(x_{q,2}, y_q\right) = h\left(x_{p,1}, x_{q,1}, x_{p,2}, x_{q,2}\right) = 0$ given by

$$f_{p,1}\left(x_{p,1}, y_p\right) = n + x_{p,1}\left(n - y_p\right) \bmod e \tag{12}$$

$$f_{q,1}(x_{q,1}, y_q) = 1 + x_{q,1}(y_q - 1) \bmod e \tag{13}$$
$$f_{p,2}(x_{p,2}, y_p) = 1 + x_{p,2}(y_p - 1) \bmod e \tag{14}$$
$$f_{q,2}(x_{q,2}, y_q) = n + x_{q,2}(n - y_q) \bmod e \tag{15}$$
$$\begin{aligned} h(x_{p,1}, x_{q,1}, x_{p,2}, x_{q,2}) &= (n-1)x_{p,1}x_{p,2} + x_{p,1} + nx_{p,2} \bmod e \\ &= (n-1)x_{q,1}x_{q,2} + nx_{q,1} + x_{q,2} \bmod e \end{aligned} \tag{16}$$

with monomials $1,\ x_{p,1},\ x_{p,2},\ x_{q,1},\ x_{q,2},\ x_{p,1}y_p, x_{p,2}y_p, x_{q,1}y_q, x_{q,2}y_q, x_{p,1}x_{p,2}, x_{q,1}x_{q,2}$ and small root

$$(x_{p,1}^{(0)}, x_{q,1}^{(0)}, x_{p,2}^{(0)}, x_{q,2}^{(0)}\ y_p^{(0)}, y_q^{(0)}) = (k_q - 1, k_q, k_p, k_p - 1, p, q), \text{ with } \begin{cases} |x_{p,1}^{(0)}| < X_{p,1} = n^{\alpha+\delta-1/2}, \\ |x_{q,1}^{(0)}| < X_{q,1} = n^{\alpha+\delta-1/2}, \\ |x_{p,2}^{(0)}| < X_{p,2} = n^{\alpha+\delta-1/2}, \\ |x_{q,2}^{(0)}| < X_{q,2} = n^{\alpha+\delta-1/2}, \\ |y_p^{(0)}| < Y_p = n^{1/2}, \\ |y_q^{(0)}| < Y_q = n^{1/2}. \end{cases}$$

Attack4 also used the lattice based method and for the success of the Attack4, it is necessary to satisfy the following Equation (17) [27, Theorem5].

$$\delta < \frac{1}{2} - \sqrt{\frac{\alpha}{7}}, \qquad (7/16 \le \alpha \le 1) \tag{17}$$

When $\alpha \approx 1$, Attack4 works for $\delta < 0.122 < 1/2 - 1/7$ ($z < 2^{250}$ for the 2048bits modulus N) which is better than Attack2 ($\delta < 0.073$) and Attack1($z < 2^{224}$ for the 2048bits modulus N).

Since no improvements have been introduced so far, Attack4 is known to be the state-of-the-art small CRT exponent attack.

Note. Attack4 was generalized in [28]. However, if p and q are balanced then generalized attack equals Attack4[28].

3. The Proposed scheme

In rebalanced RSA, signature generation speed up is obtained by transferring calculation costs to signature verification. In the proposed schemes, we speed up signature generation by shifting calculation costs to message generator and so, verification still remains fast($e = 65537$).

3.1. Scheme1

In this scheme, we shift the signature generation costs to message generator without effect on the security of typical RSA.

Key generation: The key generation algorithm takes a security parameter n(typically $n = 2048$).

Step1. Generate two distinct $(n/2)$-bit primes p and q such that $\gcd(65537, (p-1)(q-1)) = 1$ and calculate $N = pq$.

Step2. Select $e = 65537$ and calculate $d = e^{-1} mod\ (p-1)(q-1), d_p = d\ mod\ (p-1), d_q = d\ mod\ (q-1)$ and $h = 2^{\lceil n/4 \rceil}$.

Step3. Find d_{0p}, d_{0q}, d_{1p} and d_{1q} such that $d_p = hd_{1p} + d_{0p}, d_q = hd_{1q} + d_{0q}$ and $0 < d_{0p}, d_{0q}, d_{1p}, d_{1q} < h$.

Step4. Public key is (e, N, h) and private key is $(d_{0p}, d_{0q}, d_{1p}, d_{1q}, p, q)$.

Signature generation is similar to Section1. The only difference is that the signer acts as prover in Step3 and 4, which are needed to be secure against the active attack.

Signature generation: Message generator calculates hash value $m_1 = H(m)^h mod\ N$ from plaintext m and sends $m||m_1$ to signer who returns signature s or special symbol $\perp$ (which means reject) as follows.

Step1. Calculate $m_{0p} = H(m) mod\ p, m_{0q} = H(m) mod\ q, m_{1p} = m_1 mod\ p$ and $m_{1q} = m_1 mod\ q$.

Step2. Calculate $s_p = {m_{0p}}^{d_{0p}} {m_{1p}}^{d_{1p}} mod\ p (= {m_{0p}}^{h_p d_{1p} + d_{0p}} mod\ p)$ and $s_q = {m_{0q}}^{d_{0q}} {m_{1q}}^{d_{1q}} mod\ q (= {m_{0q}}^{h_q d_{1q} + d_{0q}} mod\ q)$.

Step3. Calculate $t_p = {s_p}^e mod\ p$ and $t_q = {s_q}^e mod\ q$.

Step4. If $t_p \neq m_{0p}$ or $t_q \neq m_{0q}$ then return $\perp$.

Step5. Return $s = \left(\left((s_p - s_q)(q^{-1} mod\ p)\right) mod\ p\right) q + s_p$.

Signature verification is identical to typical RSA which has the fastest verification among all the standardized signature schemes.

Security: In Scheme1, $h(= 2^{\lceil n/4 \rceil})$ does not provide any information except for the bit size of private exponent, which has been known to be approximately equal to n. Compared to RSA-FDH, in Scheme1, some operation $(H(m)^h mod\ n)$ of signature generation $(H(m)^d mod\ n)$ has been only pre-calculated by message generator. In RSA-FDH, unlike RSA-PSS, both message generator and signer can obtain the padding result by calculating hash function without random salt and so, padding by message generator does not influence the security of RSA-FDH. Hence, RSA-FDH can be straightly applied to this scheme and following theorem is satisfied.

Theorem1. In the random oracle model, Scheme1 is $(t, q_{hash}, q_{sig}, \varepsilon)$-secure under the assumption that RSA-FDH is $(t, q_{hash}, q_{sig}, \varepsilon)$-secure where q_{hash} and q_{sig} are the number of hash queries and signature queries performed by forger and where ε is the probability to break the scheme in time t.

3.2. Scheme2

In this scheme, we shift the signature generation costs to message generator to obtain extremely fast signature generation.

Key generation: The key generation algorithm takes two security parameters n and k where $k \leq n/2$(typically n=2048 and k=112).

Step1. Generate two distinct $(n/2)$-bit primes p and q such that $\gcd(p-1, q-1) = 2$ and $\gcd\big(65537, (p-1)(q-1)\big) = 1$ and calculate $N = pq$.

Step2. Select $e = 65537$ and calculate $d = e^{-1} mod\ (p-1)(q-1), d_p = d\ mod\ (p-1)$ and $d_q = d\ mod\ (q-1)$.

Step3. Select k-bit numbers d_{0p}, d_{0q}, d_{1p} and d_{1q} such that $\gcd\big(d_{1p}, p-1\big) = 1, \gcd\big(d_{1q}, q-1\big) = 1$ and $d_{0p} \equiv d_{0q}\ mod\ 2$.

Step4. Calculate $h_p = \big(d_p - d_{0p}\big){d_{1p}}^{-1} mod\ (p-1)$ and $h_q = \big(d_q - d_{0q}\big){d_{1q}}^{-1} mod\ (q-1)$.

Step5. Find h such that $h_p = h\ mod\ (p-1), h_q = h\ mod\ (q-1)$ and $0 < h < (p-1)(q-1)$.

Step6. Public key is (e, N, h) and private key is $(d_{0p}, d_{0q}, d_{1p}, d_{1q}, p, q)$.

Note. Refer to [10, Section4] for the method to find h in Step5.

Signature generation and verification are the same as in Section3.1. The only issue is that signature generation can be done faster than other RSA variants, because d_{0p}, d_{0q}, d_{1p} and d_{1q} are extremely small.

Security: Similar to Scheme1, some operation $(H(m)^h mod\ n)$ of signature generation $(H(m)^d mod\ n)$ has been only pre-calculated in Scheme2, too. However, in Scheme2, h provides some information (i.e., d_{0p}, d_{1p}, d_{0q} and d_{1q} such that $d_p \equiv h_p d_{1p} + d_{0p} mod\ (p-1)$ and $d_q \equiv h_q d_{1q} + d_{0q} mod\ (q-1)$, are small) about private exponent $d(= 65537^{-1} mod\ (p-1)(q-1))$. Hence, unlikely Scheme1, the effect by h must be considered in security analysis.

For d_0 and d_1 such that $d_{0p} = d_0\ mod\ (p-1), d_{0q} = d_0\ mod\ (q-1), d_{1p} = d_1\ mod\ (p-1), d_{1q} = d_1\ mod\ (q-1)$ and $0 < d_0, d_1 < (p-1)(q-1)$, $hd_0 + d_1 \equiv d\ mod\ (p-1)(q-1)$ is satisfied. And h, d, d_0 and d_1 are usually large (on the order of N) [10]. From this, attack of [20] cannot be applied to Scheme2. Then, let's consider the small CRT exponent attacks. In the small CRT exponent attacks, Equation (20) and (21) are used instead of (18) and (19).

$$ed_p = k'_p(p-1) + 1 \tag{18}$$

$$ed_q = k'_q(q-1) + 1 \tag{19}$$

$$e\big(hd_{1p} + d_{0p}\big) = k_p(p-1) + 1 \quad (hd_{1p} + d_{0p} > d_p) \tag{20}$$

$$e\big(hd_{1q} + d_{0q}\big) = k_q(q-1) + 1 \quad (hd_{1q} + d_{0q} > d_q) \tag{21}$$

It is an open problem whether there exists any efficient small CRT exponent attack to Scheme2. However, the only clear thing is that known small CRT exponent attacks to RSA such as Attack1, 2, 3 and 4 cannot be applied to Scheme2 because Equation (20) and (21) are used.

The best known attack can be described in the following theorem.

Theorem2. Let (N, e) be an RSA public key with $N = pq$ (p and q are primes such that $\gcd(p-1, q-1) = 2$ and $\gcd\big(65537, (p-1)(q-1)\big) = 1$) and $e = 65537$.

Further, let $d, d_p, d_q, h, d_0, d_1, h_p, h_q, d_{0p}, d_{0q}, d_{1p}, d_{1q}$ and r be the integers such that $d = e^{-1} mod\ (p-1)(q-1), d_p = d\ mod\ (p-1), d_q = d\ mod\ (q-1), d \equiv hd_1 + d_0 mod\ (p-1)(q-1), d_{0p} = d_0 mod\ (p-1), d_{0q} = d_0 mod\ (q-1), d_{1p} = d_1 mod (p-1), d_{1q} = d_1 mod\ (q-1), h_p = h\ mod\ (p-1), h_q = h\ mod\ (q-1)$ and $r = \min(\max(d_{0p}, d_{1p}), \max(d_{0q}, d_{1q}))$.

Then given (N, e, h) an adversary can expose the private key d in time $O(rlogr)$.

Proof. Let $r = \max(d_{0p}, d_{1p})$. From $d_p \equiv h_p d_{1p} + d_{0p} \equiv hd_{1p} + d_{0p}\ mod\ (p-1)$, $m^{e(h_p d_{1p} + d_{0p})} \equiv m^{e(hd_{1p} + d_{0p})} \equiv m\ mod\ p$ is satisfied and so, $\gcd({c_0}^{d_{0p}} {c_1}^{d_{1p}}\ mod\ N - m, N) = p$ is satisfied for $c_0 = m^e mod\ N$ and $c_1 = m^{eh} mod\ N$. That is, attack is resulted in finding small roots i and j such that $\gcd({c_0}^i {c_1}^j mod\ N - m, N) > 1$ and $0 \leq i, j \leq r$. Referring to [10, 18, Attack8.1], the attack can succeed in time $O(rlogr)$. (end of proof)

Theorem2 shows that, to obtain 2^{112} security, r must be at least 112 bits long. Consequently, k should not be less than 112 in Scheme2. Under the assumption that Scheme2 is computationally secure, RSA-FDH can be safely applied to Scheme2 as in Scheme1.

Note. To the best of our knowledge the attacks applicable to the rebalanced RSA$(\alpha \approx 1)$ are only the Attack1, 2 and 4. Referring to Attack1, we proposed Theorem2. It still remains an open problem whether it is possible to propose efficient lattice based attacks referring to Attack2 and 4. However, it would not be easy to propose lattice based attacks which are better than Theorem2. See Appendix A for more details.

4. PERFORMANCE COMPARISON

TABLE 1. SIGNATURE GENERATION AND VERIFICATION TIME COMPARISON

	Typical RSA	Scheme1	Rebalanced RSA	Scheme2
Public Exponent	17bits($2^{16}+1$)	17bits($2^{16}+1$)	2048 bits	17bits($2^{16}+1$)
Num of Multiplication in signature verification (module size)	17 (2048)	17 (2048)	2048×1.5=3072 (2048)	17 (2048)
Secret Exponent	2048 bits	2048 bits	2048 bits	2048 bits
CRT-Exponent	1024 bits	512 bits	250 bits	112 bits
Num of Multiplication in signature generation (module size)	1024×1.5×2+2=3074 (1024)	512×1.75×2+4+17×2 =1830 (1024)	250×1.5×2+2=752 (1024)	112×1.75×2+4+17×2 =430 (1024)
Signature generation Time	31ms	18.5ms	7.1ms	4.1ms
Signature verification Time	0.8ms	0.8ms	122ms	0.8ms
Total Processing Time= Max(Signature generation, Signature verification)	31ms	18.5ms	122ms	4.1ms

Note. Timings were made on 3.5GHz Core i3-4150 desktop using NTL with GMP Library and can be treated as a relative guideline. And delays by hash function were not considered in timing because they can be ignored compared to modular exponentiation of big integers.

Table 1 shows the signature generation and verification time comparison of typical RSA, rebalanced RSA, Scheme1 and Scheme2. As shown in Table 1, Scheme1 and Scheme2 are approximately 1.67 and 7.56 times faster than typical RSA, respectively, in total processing.

5. CONCLUSION

We have described the method to increase the signature generation speed in RSA which has the small public exponent for the fast verification. The basic idea is to transfer the calculation costs from signer to message generator. Hence, proposed schemes are suited for the signature applications where message is not generated by signer (i.e., message is generated offline, but signed online.).

In security, Scheme1 is identical to typical RSA, but it is an open problem that Scheme2 is identical to typical RSA. Especially, it remains to be further clarified whether there is any efficient small CRT exponent attack to Scheme2 and so, further studies are needed. Proposed schemes can also be applied to RSA signature with other small public exponents such as 3 or 17.

REFERENCES

[1] R.L. Rivest, A. Shamir, L. Adleman , A method for obtaining digital signatures and public – key cryptosystems, Communications of ACM 21(2)(1978) 120-126.

[2] J. J. Quisquater , C. Couvreur, Fast Decipherment Algorithm for RSA Public-Key Cryptosystem, IEEE Electronics Letters 18(1982) 905-907.

[3] H.Wiener, Cryptanalysis of Short RSA Secret Exponents, IEEE Transactions on Information Theory 36(3) (1990) 553-558.

[4] M. Bellare, P. Rogaway, Exact security of digital signatures-How to sign with RSA and Rabin, EUROCRYPT'96, LNCS1070(1996), 399-416.

[5] A.Menezes , P.van Orschot , and S. Vanstone, Handbook of Applied Cryptography, CRC Press, 1996, pp.287, 617-618.

[6] E.Verheul, H.van Tilborg, Cryptanalysis of less short RSA secret exponents, Applicable Algebra in Engineering, Communication and Computing 8 (1997) 425-435.

[7] D. Coppersmith, Small solutions to polynomial equations and low exponent RSA vulnerabilities, Journal of Cryptology, 10(4) (1997), 233-260.

[8] D.Boneh, Twenty Years Attacks on the RSA Cryptosystem, Notices of the American Mathematical Society 46 (1999) 203-213.

[9] D.Boneh, G.Durfee, Cryptanalysis of RSA with Private Key d less than $N^{0.292}$, IEEE Transactions on Information Theory 46(4) (2000) 1339-1349.

[10] D.Boneh , H.Shacham., Fast variants of RSA, CryptoBytes (The Technical Newsletter of RSA Laboratories)5(1) (2002) 1–9.

[11] A.May, Cryptanalysis of unbalanced RSA with small CRT–exponent, CRYPTO2002, LNCS2442 (2002), 242–256.

[12] H.M. Sun, M.E. Wu, An Approach Towards Rebalanced RSA-CRT with Short Public Exponent, Cryptology ePrint Archive, Report 2005/053, 2005. http://eprint.iacr.org/.

[13] S.D.Galbraith, C.Heneghan, J.F.McKee, Tunable balancing of RSA, ACISP 3574(2005), 280-292.

[14] E. Jochemsz, A. May, A Strategy for finding Roots of Multivariate Polynomials with New Applications in Attacking RSA Variants, ASIACRYPT2006(LNCS4284) (2006),267-282.

[15] D.Bleichenbacher and A.May, New attacks on RSA with small secret CRT-exponents, In International Workshop on Public Key Cryptography (2006), 1-13.

[16] E. Jochemsz, A. May, A polynomial time attack on RSA with private CRT-exponents smaller than $N^{0.073}$, In A. Menezes, editor, volume 4622 of Lecture Notes in Computer Science, Springer, 2007, pp 395-411.

[17] H.M.Sun, M.E.Wu, M.Jason.Hinek, Trading decryption for speeding encryption in Rebalanced-RSA, The Journal of Systems and Software 82 (2009), 1503-1512.

[18] M.Jason Hinek , Cryptanalysis of RSA and its variants, CRC Press ,2010, pp. 23-27,139-155.

[19] N.Shinohara, T.Izu, N.Kunihiro, Small secrete CRT-exponent attacks on Takagi's RSA, IEICE Transactions 94-A(1) (2011), 19-27

[20] A.Nitaj, M.O.Douh, A new attack on RSA with a composed decryption exponent, Cryptology ePrint Archive, Report 2014/035, 2014. http://eprint.iacr.org/.

[21] S.Sakar, Small secret exponent attack on RSA variant with modulus $N = p^r q$, Designs Codes and Cryptography, 73(2) (2014), 130-159

[22] A.Takayasu, N.Kunihiro, Cryptoanalysis of RSA with multiple small secret exponents, ACISP2014, LNCS8544 (2014), 176-191

[23] L.Peng, Y.Lu, S.Sakar, J.Xu, Z.Huang, Cryptanalysis of variants of RSA with multiple small secret exponents, INDOCRYPT2015, LNCS9462 (2015), 105-123
[24] S.Sakar, Revisiting prime power RSA, Discrete Applied Mathematics, 203 (2016), 127-133
[25] A.Takayasu, N.Kunihiro, How to generalize RSA cryptanalyses, PKC2016, LNCS 9615(2016), 67-97
[26] A.Takayasu, Y.Lu, L.Peng, Small CRT-exponent RSA revisited, EUROCRYPT2017, LNCS10211 (2017), 130-159
[27] A.Takayasu, Y.Lu, L.Peng, Small CRT-exponent RSA revisited, Journal of Cryptology, 32(4) (2019), 1337-1382 (full version of [26])
[28] L.Peng, A.Takayasu, Generalized cryptanalysis of small CRT-exponent RSA, Theoretical Computer Science, 795(2019), 432-458

APPENDIX A. POSSIBILITY OF LATTICE BASED SMALL CRT EXPONENT ATTACK TO SCHEME2

If h is not used in the attacks, Scheme2 is identical to typical RSA in the security. Hence, the efficient lattice based small CRT exponent attack to Scheme2 (noted as Attack X) should use Equation (20) and (21), if it (i.e., Attack X) exists.

And Attack X can also be applied to rebalanced RSA by using $h(=\left\lfloor \frac{\delta}{2} log_2 N \right\rfloor$, N and δ are publicized) which makes d_{0p}, d_{1p}, d_{0q} and d_{1q} to be small (i.e., $d_p = hd_{1p} + d_{0p}$ and $d_q = hd_{1q} + d_{0q}$).

Meanwhile, when α approaches 1, Attack 2 had been known as the state-of-the-art lattice based attack to rebalanced RSA before the proposal of Attack4[26, 27] and so, it can be seen that Attack X is not stronger than Attack 2(and 4) in rebalanced RSA.

On the basis of such facts, we considered the relationship between Attack X to Scheme2 and Theorem2 as follows.

First, we considered the general scenario of lattice based small CRT exponent attacks using Coppersmith technique. Coppersmith lattice based attacks are mounted in two stages. First stage is constructing the modular (or integer) equations, which have the small roots, from CRT equations. Second stage uses Coppersmith's method to solve the equations. In [7], Coppersmith describes the techniques to find small integer roots of polynomials in a single variable modulo N, and polynomials in two variables over the integers. These methods can be extended to more variables and Coppersmith's ideas of finding modular roots are reformulated by Howgrave-Graham as follows.

Theorem3 (Howgrave-Graham). Let $h(x_1, \dots, x_k)$ be a polynomial in k variables with ω monomials. Suppose that $h\left(x_1^{(0)}, \dots, x_k^{(0)}\right) = 0 \; mod \; \phi$ where $x_i^{(0)} \leq X_i, i = 1, \dots, k$ and $\|h(x_1 X_1, \dots, x_k X_k)\| < \phi/\sqrt{\omega}$. Then $h\left(x_1^{(0)}, \dots, x_k^{(0)}\right) = 0$ holds over the integers. $\|h(x_1, \dots, x_k)\|$ denotes a Euclid norm of the coefficient vector of polynomial h.

L^3-reduction algorithm is often used to find the Equations which have the coefficients small enough to satisfy Theorem3.

Theorem4 (Lenstra, Lenstra, Lovàsz). Given a basis $B = \{b_1, \dots, b_w\}$, L^3-reduction algorithm outputs reduced basis vectors $\{v_1, \dots, v_w\}$ that satisfy

$$\|v_j\| \leq 2^{w(w-1)/4(w-j+1)} \det(L(B))^{1/(w-j+1)} \;\; for \; 1 \leq j \leq w,$$

in time polynomial in w and in the bit-size of the entries in B.

In the lattice based attacks that use the Coppersmith's technique (i.e., Theorem3 and 4), the possibility that L^3-algorithm outputs vectors which satisfy Howgrave-Graham's theorem relates to the upper bounds of variables.

Second, we considered the Attack X to rebalanced RSA when α approaches 1(more precisely,$\alpha \approx 0.95$) and compared it with Attack X to Scheme2 which is secure against Theorem2 in the parameters (including the upper bounds of variables) that can be used in small CRT exponent attacks.

Then, what is the reason to make α approache 1?

Let $w = eh$. Then, following Equation (A.1), (A.2), (A.3) and (A.4) are obtained from Equation (20) and (21).

$$wd_{1p} + ed_{0p} = k_p(p-1) + 1 \quad (A.1)$$
$$wd_{1q} + ed_{0q} = k_q(q-1) + 1 \quad (A.2)$$
$$k_p(p-1) - ed_{0p} + 1 = 0 \; mod \; w \quad (A.3)$$
$$k_q(q-1) - ed_{0q} + 1 = 0 \; mod \; w \quad (A.4)$$

And these equations would be used as the basis equations in Attack X. In the case of Scheme2, h is full sized and e is small, so we considered the case of rebalanced RSA ($\alpha \approx 0.95$) in which e is large and h is small in order to adjust the balance of bit size of w approximately between both cases.

We first proved that rebalanced RSA($\alpha \approx 0.95, \delta > 0.109$) is secure against AttackX as follows.

From [16],[18] and [27], it can be seen that rebalanced RSA($\alpha \approx 0.95, \delta > 0.109$) is secure against Attack2, but is not secure against Attack4. And Attack2 is known to be stronger than other small CRT exponent lattice based attacks except for Attack4 when α approaches 1. From this, it can be seen that rebalanced RSA($\alpha \approx 0.95, \delta > 0.109$) is also secure against Attack X. (If not, Attack X that uses $h = \left\lfloor \frac{\delta}{2} log_2 N \right\rfloor$ would be introduced as a small CRT exponent attack stronger than Attack2)

Next, we compared rebalanced RSA($\alpha \approx 0.95, \delta > 0.109$) with Scheme2 which is secure against Theorem2. Table 2 shows the comparison of parameters between rebalanced RSA($\alpha \approx 0.95, \delta > 0.109$) and Scheme2.($D_{0p} = D_{1p} = D_{0q} = D_{1q} \geq 2^{112}$) for the 2048bits modulus N. ($P, Q, D_p, D_q, D_{0p}, D_{1p}, D_{0q}, D_{1q}, K_p$ and K_q mean the upper bounds of $p, q, d_p, d_q, d_{0p}, d_{1p}, d_{0q}, d_{1q}, k_p$ and k_q, respectively.)

As shown in Table 2, Scheme2 is approximately same to rebalanced RSA in parameters and from the property of Coppersmith's lattice based attacks, Scheme2 which is secure against Theorem2 would also be secure from Attack X (i.e., Attack X would not be better than Theorem2 in Scheme2) with high probability.

From all the facts above, it can be seen that it would not be easy to propose the lattice based small CRT exponent attacks better

TABLE 2. COMPARISON OF PARAMETERS THAT CAN BE USED IN ATTACK X FOR 2048 BITS MODULUS

	e	h	$w = eh$	P, Q	D_p, D_q	$D_{0p}, D_{1p}, D_{0q}, D_{1q}$	$\boldsymbol{K_p}, \boldsymbol{K_q}$
rebalanced RSA ($\alpha \approx 0.95, \delta > 0.109$)	1953 bits	112 bits	2065 bits	1024 bits	224 bits	112 bits	1153 bits
Scheme2 (Theorem2)	17 bits	2048 bits	2065 bits	1024 bits	1024 bits	112 bits	1153 bits

Note. In rebalanced RSA, $d_p = hd_{1p} + d_{0p}$ and $d_q = hd_{1q} + d_{0q}$ are satisfied but, in Scheme2, $d_p = hd_{1p} + d_{0p}\ mod\ (p-1)$ and $d_q = hd_{1q} + d_{0q}\ mod\ (q-1)$ are satisfied. Scheme2 is superior to rebalanced RSA($\alpha \approx 0.95, \delta > 0.109$) in all parameters except for e.

than Theorem2 in Scheme2 under the assumption that, except for Attack4, no attacks to rebalanced RSA($\alpha \approx 0.95, \delta > 0.109$) have been introduced thus far.