Design of Rebalanced RSA-CRT for Fast Encryption

Hung-Min Sun and Mu-En Wu
Department of Computer Science
National Tsing Hua University, Hsinchu, Taiwan 300
Email:hmsun@cs.nthu.edu.tw; mn@is.cs.nthu.edu.tw

Abstract

Based on the Chinese Remainder Theorem (CRT), Quisquater and Couvreur proposed an RSA variant, RSA-CRT, to speed up RSA decryption. Then, Wiener suggested another RSA variant, Rebalanced RSA-CRT, to further accelerate RSA-CRT decryption by shifting decryption cost to encryption cost. However, such an approach makes RSA encryption very time-consuming because the public exponent \( e \) in Rebalanced RSA-CRT is of the same order of magnitude as \( \phi(N) \). In this paper we study the following problem: does there exist any secure variant of Rebalanced RSA-CRT, whose public exponent \( e \) is much shorter than \( \phi(N) \)? We solve this problem by designing two variants of Rebalanced RSA-CRT, Scheme A and Scheme B. In Scheme A, we focus on designing a variant in which \( d_p \) and \( d_q \) are of 160 bits, and \( e = 2^{567} + 1 \). Thus the encryption time is reduced to about \( \frac{1}{2.7} \) of the time required by the original Rebalanced RSA-CRT. Scheme B is a variant in which \( d_p \) and \( d_q \) are of 198 bits, and \( e = 2^{511} + 1 \). Thus its encryption is about 3 times faster than that of Rebalanced RSA-CRT, but the decryption is a little slower than that of Rebalanced RSA-CRT.

Keywords: RSA, RSA-CRT, CRT, Rebalanced RSA-CRT, Lattice Basis Reduction.

1 Introduction

Many practical issues have been considered when implementing RSA[17], such as how to reduce the storage requirement for RSA modulus, how to reduce the encryption time (or signature–verification time), how to reduce the decryption time (or signature–generation time)[3], how to balance the encryption and decryption time[18][19], and so on.

The encryption and decryption in RSA require taking heavy exponential multiplications modulus of a large integer \( N \) which is the product of two large primes \( p \) and \( q \). Without loss of generality, we assume \( N \) is of 1024 bits, and \( p \) and \( q \) are of 512 bits. In general, the RSA encryption and decryption time are roughly proportional to the number of bits in public and secret exponents respectively. To reduce the encryption time (or the signature–verification time), one may wish to use a small public exponent \( e \). The smallest possible value for \( e \) is 3, however, it has been proven to be insecure against some small public exponent attacks[11]. Therefore, a more widely accepted and used public exponent is \( e = 2^{16} + 1 = 65537 \). On the other hand, to reduce the decryption time (or the signature–generation time), one may also wish to use a short secret exponent \( d \). However, the use of short secret exponent encounters a more serious security problem due to some powerful short secret exponent attacks[21][20][2]. For instance, Wiener[21] announced an attack on short secret exponent, called continued fraction attack. He showed that RSA system can be totally broken if

This research was supported in part by the National Science Council, Taiwan, under contract NSC-93-2213-E-007-102.
the secret exponent \( d < \frac{1}{4}N^{0.25} \). Boneh and Durfee[2] further proposed a new attack on the use of short secret exponent. They improved Wiener’s bound up to \( N^{0.292} \), i.e., RSA system can be totally broken if the secret exponent \( d < N^{0.292} \).

Another well-known technique[16] to reduce the decryption time is to employ the Chinese Remainder Theorem (CRT) for RSA decryption. Using this technique, two half-sized modular exponentiations are required. Let \( N = pq \) be an RSA modulus and \((e, d)\) be a pair of public exponent and secret exponent. The first modular exponentiation gives the result \( C_p \equiv C^{d_p} \pmod{p} \), where \( d_p \equiv d \pmod{p-1} \); the second gives the result \( C_q \equiv C^{d_q} \pmod{q} \), where \( d_q \equiv d \pmod{q-1} \). These two results can be easily combined[15] to obtain the final result \( M \equiv C^d \pmod{N} \) by using CRT. Such an approach, called RSA-CRT, achieves 4 times faster in decryption compared to the standard RSA system. In addition, Wiener[21] suggested one can further reduce the decryption time by carefully choosing \( d \), such that both \( d_p \equiv d \pmod{p-1} \) and \( d_q \equiv d \pmod{q-1} \) are small. That is, in the key generation phase, one first selects two small CRT-exponents \( d_p \) and \( d_q \), and then these two CRT-exponents are combined to get the secret exponent \( d \) satisfying \( d_p \equiv d \pmod{p-1} \) and \( d_q \equiv d \pmod{q-1} \). At last, he computes the corresponding public exponent \( e \) satisfying \( ed \equiv 1 \pmod{\phi(N)} \). Such a variant of RSA-CRT, called Rebalanced RSA-CRT[21][1][3], enables us to rebalance the difficulty of encryption and decryption. In other words, we can speed up the CRT decryption by shifting the decryption cost to the encryption cost. Note that in Rebalanced RSA-CRT, both \( d \) and \( e \) will be of the same order of magnitude as \( \phi(N) \). The decryption time depends on the bit-size of \( d_p \) and \( d_q \), while not on the bit-size of \( d \). But the encryption time depends on the bit-size of \( e \). This will make the encryption for the original Rebalanced RSA-CRT very time-consuming. Due to Wiener’s suggestion[21], a raised open problem [21][1][2] is whether there exists any efficient attack on Rebalanced RSA-CRT. So far, the best known attack [1] on Rebalanced RSA-CRT runs in time complexity \( O(\min\{\sqrt{d_p} \log_2 d_p, \sqrt{d_q} \log_2 d_q\}) \) which is exponentially in the bit-size of \( d_p \) or \( d_q \). Boneh[1][3] suggested to use \( d_p \) and \( d_q \) of 160 bits in order to defend against this attack. Under such parameters, the decryption in the original Rebalanced RSA-CRT will be about \( \frac{512}{160} = 3.2 \) times faster than that in RSA-CRT. It is still an open problem whether Rebalanced RSA-CRT using \( d_p \) and \( d_q \) of 160 bits is secure.

According to the key generation in the original Rebalanced RSA-CRT, if we first select the small CRT-exponents \( d_p \) and \( d_q \), the public exponent \( e \) will be of the same bit-size as modulus \( \phi(N) \). This causes heavy encryption cost. If we can make the public exponent \( e \) much shorter than \( \phi(N) \), it will be more convenient and practical in many applications. In this paper, we are interested in studying the following problem: does there exist any secure variant of Rebalanced RSA-CRT, whose public exponent \( e \) is much shorter than \( \phi(N) \)? We solve this problem by designing two variants of Rebalanced RSA-CRT, Scheme A and Scheme B. In Scheme A, we focus on designing a variant in which \( d_p \) and \( d_q \) are of 160 bits, and \( e = 2^{567} + 1 \). Thus the encryption time is reduced to about \( \frac{1}{2^{77}} \) of the time required by the original Rebalanced RSA-CRT. Scheme B is a variant in which \( d_p \) and \( d_q \) are of 198 bits, and \( e = 2^{511} + 1 \). Thus its encryption is about 3 times faster than that of Rebalanced RSA-CRT, but the decryption is a little slower than that of Rebalanced RSA-CRT.

The remainder of this paper is organized as follows. In Section 2, we briefly review some well-known RSA variants. In Section 3, we review some well-known attacks on RSA variants. In Section 4, we propose and analyze our Scheme A. In Section 5, we propose another variant Scheme B. In Section 6, we show the experimental results of our implementations for our schemes and make comparisons of various RSA variants. Finally we conclude this paper in Section 7.
2 Overview of Some RSA Variants

2.1 RSA-Basic, RSA-Short-D, and RSA-CRT

We first review the original RSA and CRT decryption\[17\]. Depending on different choices for parameters and different decryption algorithms used, we classify them as RSA-Basic, RSA-Short-D, and RSA-CRT.

Key Generation in RSA

Let $N$ be the product of two large primes $p$ and $q$. Let $e$ and $d$ be two integers satisfying $ed \equiv 1 \pmod{\phi(N)}$, where $\phi(N) = (p-1)(q-1)$ is the Euler totient function of $N$. In general, $N$ is called the RSA modulus, $e$ is the public exponent, and $d$ is the secret exponent.

Encryption and Decryption in RSA

To encrypt a message (plaintext) $M$, one computes the corresponding ciphertext $C \equiv M^e \pmod{N}$. To decrypt the ciphertext $C$, the legitimate receiver computes $M \equiv C^d \pmod{N}$.

In general $d$ is a positive number which is smaller than $\phi(N)$. In fact, the secret exponent that can used to decrypt is not unique. For instance, we can let $d' = d - \phi(N)$. This $d'$ is negative and equivalent to $d \pmod{\phi(N)}$. That means we can use $d'$ as another secret exponent to decrypt. (Indeed, $C^{d'} \pmod{N} \equiv M^{e(d'-\phi(N))} \pmod{N} \equiv M^{ed} (M^\phi(N))^{-e} \pmod{N} \equiv M^{ed} \pmod{N} \equiv M \pmod{N} = M$). Note that in general we will not use such a negative exponent $d'$ because it is more time-consuming in decryption due to one more inverse operation required.

CRT Decryption

Based on CRT, Quisquater and Couvreur\[16\] proposed a fast decryption algorithm, called CRT decryption. Let $d_p \equiv d \pmod{p-1}$ and $d_q \equiv d \pmod{q-1}$. The CRT decryption is as follows:

1. Compute $C_p \equiv C^{d_p} \pmod{p}$.
2. Compute $C_q \equiv C^{d_q} \pmod{q}$.
3. Compute $\overline{C} \equiv (C_q - C_p) p^{-1} \pmod{q}$.
4. Compute $M = C_p + \overline{C} p$.

Note that the decrypter can compute $p^{-1} \pmod{q}$ in advance. Thus the main cost for CRT decryption is in Step 1 and Step 2. Therefore, CRT decryption is approximately 4 times faster than the decryption in standard RSA\[3\].

Parameters for RSA-Basic, RSA-Short-D, and RSA-CRT

Here we consider three types of RSA system, called RSA-Basic, RSA-Short-D, and RSA-CRT. In RSA-Basic, one first selects the public exponent $e = 2^{16} + 1$, and thus the secret exponent $d$ is about of 1024 bits. If CRT decryption is applied to RSA-Basic, we call it as RSA-CRT. In RSA-Short-D, one first selects a 512-bit secret exponent, and thus the public exponent is about of 1024 bits. Note that so far RSA can be totally broken if $d < N^{0.292}\[2\]$, but as mentioned by Boneh and Durfree the small inverse problem in [2] is very likely to have a unique solution when $d < N^{0.5}$. Therefore we choose a 512-bit $d$ in RSA-Short-D for achieving high-level security. All these three RSA variants will be compared with other RSA variants in Table 1.
2.2 Rebalanced RSA-CRT

Wiener [21] suggested an RSA variant, Rebalanced RSA-CRT, to further speed up decryption by shifting the work to the encrypter. One version of this variant, which is similar to Boneh and Shacham’s version [3], is described in the following.

Key Generation in Rebalanced RSA-CRT

Step 1. Randomly select two 512-bit primes \( p = 2p_1 + 1 \) and \( q = 2q_1 + 1 \) such that \( \gcd(p_1, q_1) = 1 \).

Step 2. Compute \( p_1^{-1} \pmod{q_1} \) satisfying \( p_1p_1^{-1} \equiv 1 \pmod{q_1} \).

Step 3. Randomly select two distinct odd numbers \( d_p \) and \( d_q \) of 160 bits such that \( \gcd(d_p, p - 1) = 1 \) and \( \gcd(d_q, q - 1) = 1 \).

Step 4. Compute \( d \equiv (d_q - d_p)p_1^{-1} \pmod{q_1} \).

Step 5. Compute \( d = d_p + d_p p_1 \). (Note that \( \gcd(d_p, p - 1) = 1 \) and \( \gcd(d_q, q - 1) = 1 \) imply \( \gcd(d, (p - 1)(q - 1)) = 1 \))

Step 6. Compute the public exponent \( e \) satisfying \( ed \equiv 1 \pmod{(p - 1)(q - 1)} \).

Step 7. The RSA modulus is \( N = pq \), the secret key is \( (d_p, d_q, p, q) \), and the public key is \( (N, e) \).

Encryption and Decryption in Rebalanced RSA-CRT

The encryption is the same as the encryption in standard RSA, that is \( C \equiv M^e \pmod{N} \). The decryption is the same as the CRT decryption. The main difference is that in Rebalanced RSA-CRT, the CRT-exponents \( d_p \) and \( d_q \) are only of 160 bits which are much shorter than the CRT-exponents of 512 bits in RSA-CRT. Thus, the decryption in Rebalanced RSA-CRT is about \( \frac{512}{160} = 3.2 \) times faster than that in RSA-CRT.

3 Related Attacks on RSA Variants

3.1 Short Secret Exponent Attacks

We present some short secret exponent attacks, including Wiener’s continued fraction attack [21], and some lattice attacks against on RSA [2][10].

Theorem 3.1 [21] In RSA system, let \( N = pq \) be an RSA modulus and \( (e, d) \) be a pair of public exponent and secret exponent satisfying \( ed \equiv k\phi(N) + 1 \) for some integer \( k \). Let \( |e| = N^\alpha \) and \( |d| < N^\gamma \) for some \( \alpha \) and \( \gamma \). If \( \gamma < \frac{1}{2} \), then we can efficiently recover \( d \).

From the theorem 3.1 we know if both \( \frac{e}{N} \) and \( \frac{k}{q} \) are close enough, then we can obtain the value \( d \) from one of the values of the continued fraction expansion of \( \frac{e}{N} \). Besides, the extension of Wiener attack was proposed by Verheul and Tilborg [20]. When \( d > N^{0.25} \), their attack needs to do an exhaustive search for about \( 2t+8 \) bits, where \( t \approx \log_2(\frac{d}{N^{0.25}}) \). Here we omit reviewing this extension.

Theorem 3.2 [2] In RSA system, let \( N = pq \) be an RSA modulus and \( (e, d) \) be a pair of public exponent and secret exponent satisfying \( ed \equiv k\phi(N) + 1 \) for some integer \( k \). Let \( |e| = N^\alpha \) and \( |d| < N^\gamma \) for some \( \alpha \) and \( \gamma \). If \( \gamma < \frac{1}{6} - \frac{1}{3}(1 + 6\alpha)^{1/2} \), then we can heuristically factor the RSA modulus \( N \).
Durfee and Nguyen[10] generalized Boneh-Durfee attack to the case when the difference between the primes $p$ and $q$ is large. They showed that the more unbalanced the prime factors are, the more insecure the RSA system is. Following we review Boneh’s[1] factoring attack based on CRT-exponents.

**Theorem 3.3**[1] In RSA system, let $(N,e)$ be the public key with $N = pq$. Let $d$ be the corresponding secret exponent satisfying $d_p \equiv d (\mod p - 1)$ and $d_q \equiv d (\mod q - 1)$ with $d_p < d_q$. Then given $(N,e)$, an adversary can expose the secret exponent $d$ in time complexity $O(\sqrt{d_p \log_2 d_p})$.

For current security level, we suppose $2^{80}$ is a safe complexity which makes an exhaustive search infeasible. Therefore, in order to achieve $2^{80}$ complexity for $O(\sqrt{d_p \log_2 d_p})$, we need use $d_p$ and $d_q$ of 160 bits.

### 3.2 Partial Key Exposure Attacks

Boneh, Durfee, and Frankel[4] showed that for low public exponent RSA, given a fraction of the secret exponent bits, an adversary can recover the entire secret exponent. This kind of attack is called the partial key exposure attack. Here we focus only on those attacks for the most significant bits (MSBs) known. Note that all these partial key exposure attacks can still work when $d$ is negative.

**Theorem 3.4**[4] In RSA system, let $N = pq$ be a 1024-bit RSA modulus and $(e,d)$ be a pair of public exponent and secret exponent satisfying $ed \equiv 1 (\mod \phi(N))$.

1. Suppose $e \in [2^t, \ldots, 2^{t+1}]$ is the product of at most $r$ distinct primes with $256 \leq t \leq 512$. Then given the factorization of $e$ and the $t$ MSBs of $d$, there is an algorithm to compute all of $d$ in time complexity $O(2^r \log_2 N)$.

2. When the factorization of $e$ is unknown, $e$ is in the range $[2^t, \ldots, 2^{t+1}]$ with $t \in 0, \ldots, 512$, and $d > \epsilon N$ for some $\epsilon > 0$. Then given the 1024 $- t$ MSBs of $d$, there is an algorithm to compute all of $d$ in time complexity $O(\frac{1}{\epsilon} \log_2 N)$.

Based on the above theorem, we know that if $e$ is a 512-bit number of $r$ distinct prime factors, then given the 512 MSBs of a 1024-bit $d$, an adversary can recover the entire $d$ in time complexity $O(2^r \log_2 N)$. On the other hand, Blömer and May[6] proposed further result about the partial key exposure attack for MSBs of $d$. Their result works for public exponent $e$ in the interval $[N^{0.5}, N^{0.725}]$.

### 4 The Proposed Schemes for Rebalanced RSA-CRT with Medium Public Exponent

Without loss of generality, we assume $N$ is about of 1024 bits. Our Scheme A produces a 568-bit public exponent, e.g., $e = 2^{567} + 1$, and two 160-bit CRT-exponents $d_p, d_q$. The encryption time is therefore reduced to about $\frac{1}{27}$ of the time required by the original Rebalanced RSA-CRT. In the following, we first introduce a fundamental theorem in number theory[15] as the basis of our construction.

**Theorem 4.1**[15] If $a$ and $b$ are relatively prime, i.e. $\gcd(a,b) = 1$, then we can find an unique
pair \((u_h, v_h)\) satisfying \(au_h - bv_h = 1\), where \((h - 1)b < u_h < hb\) and \((h - 1)a < v_h < ha\), for any integer \(h \geq 1\).

### 4.1 The Proposed Scheme (Scheme A)

The key generation of the proposed scheme is as follows:

**Key Generation in Scheme A**

1. Randomly select an odd number \(e\) of 568 bits.
2. Randomly select a number \(k_{p1}\) of 160 bits, such that \(\gcd(k_{p1}, e) = 1\).
3. Based on Theorem 4.1, we can uniquely determine two numbers \(d_p\), \(k_{p1} < d_p < 2k_{p1}\), and \(P\), \(e < P < 2e\), satisfying \(ed_p = k_{p1}P + 1\).
4. Factor \(P = k_qq' \cdot p'\) such that \(k_{p2}\) is a number of 56 bits and \(p \equiv p' + 1\) is a prime number. If this is infeasible, then go to Step 2.
5. Randomly select a number \(k_{q1}\) of 160 bits, such that \(\gcd(k_{q1}, e) = 1\).
6. Based on Theorem 4.1, we can uniquely determine two numbers \(d_q\), \(k_{q1} < d_q < 2k_{q1}\), and \(Q\), \(e < Q < 2e\), satisfying \(ed_q = k_{q1}Q + 1\).
7. Factor \(Q = k_qq' \cdot q'\) such that \(k_{q2}\) is a number of 56 bits and \(q \equiv q' + 1\) is a prime number. If this is infeasible, then go to Step 5.
8. The public key is \((N, e)\); the secret key is \((d_p, d_q, p, q)\).

In Scheme A, let \(k_p = k_{p1}k_{p2}\) and \(k_q = k_{q1}k_{q2}\). Based on Steps 2 to 4 and Step 5 to Step 7, we know that \(ed_p = k_p(p-1)+1\) and \(ed_q = k_q(q-1)+1\). This implies \(\gcd(e, \phi(N)) = 1\). Furthermore, we multiply these two equations, and hence obtain: \((ed_p - 1) \cdot (ed_q - 1) = k_p(p-1) \cdot k_q(q-1)\).

**Special Public Exponent**

In Scheme A, the public exponents \(e\) is of 568 bits. Therefore, 852 (= \(1.5 \times 568\)) modular multiplications are required for encryption. Instead of selecting a random \(e\), we can select a special public exponent \(e = 2^{567} + 1\). Thus only 568 modular multiplications are required for encryption. Compared to a 1024-bit public exponent, which can not be arbitrarily selected, in the original Rebalanced RSA-CRT, the encryption in Scheme A is about 2.7 times faster than that in the original Rebalanced RSA-CRT. As an example, we generate an instance for Scheme A in Appendix A.1.
4.2 The Expected Number of Iterations for Loop: Step 2 to Step 4 Scheme A

In Scheme A, there are two loops running from Step 2 to Step 4 and from Step 5 to Step 7 respectively. We try to find the upper bound of the expected number of iterations for the loop running from Step 2 to Step 4. The loop number we estimated is 380064. Due to the limit of space, we omit the process here and give the details in the full version.

4.3 Security Analysis for the Proposed Scheme

Defending against Attacks on Short Secret Exponent

First we consider Wiener’s continued fraction attack. In our scheme, the RSA modulus $N$ is of 1024 bits, the public exponent $e$ is of 568 bits, the secret exponent $d$ is of 888 bits, and the parameter $k$ is of 432 bits. Note that both $d$ and $k$ are negative and the intervals of these parameters are as follows: $2^{511} < p, q < 2^{513}, 2^{568} < e < 2^{569}, -2^{888} < d < -2^{887}, -2^{432} < k < -2^{431}$. Following Wiener’s continued fraction attack[21], we get:

$$
\left| \frac{e}{d} - k \right| = \left| \frac{ed - Nk}{N} \right| = \left| \frac{p - q + \frac{1}{k}}{N} \right| = \frac{k}{d} \frac{1}{q} > \frac{1}{2\sigma} \frac{1}{2^{332}} \gg \frac{1}{2\sigma} \approx \frac{1}{2\times (2^{332})}. 
$$

Therefore we know Wiener’s attack can not be applied to our scheme.

Secondly, we consider the Boneh-Durfee attack[2]. From the parameters constructed by the proposed scheme, we can get $\alpha \approx \frac{568}{1024}$ and $\gamma \approx \frac{888}{1024}$, where $\alpha$ and $\gamma$ satisfy $|e| = N^\alpha$ and $|d| < N^\gamma$ respectively. It is clear that $\gamma \approx \frac{888}{1024} > \frac{7}{6} - \frac{1}{6}(1 + 6\alpha)^{1/2} \approx 0.46$. So, the Boneh-Durfee attack cannot succeed.

Defending against Another Lattice Attack

Here we consider another attack based on the lattice basis reduction. Because $ed_p = k_p(p - 1) + 1$ and $ed_q = k_q(q - 1) + 1$, we can obtain the following two modular equations:

1. $k_p \cdot p \equiv k_p - 1 \pmod{e}$
2. $k_q \cdot q \equiv k_q - 1 \pmod{e}$

Combine (1) and (2), we can obtain the following equation with two unknown variables $k_p k_q$ and $k_p + k_q$:

3. $k_p k_q (N - 1) \equiv -(k_p + k_q) + 1 \pmod{e}$

According to Coppersmith’s technique[7] of finding the small root of a modular equation[8], the sufficient condition to solve the equation (3) is $|k_p k_q| \cdot |k_p + k_q| < e$. Obviously the proposed scheme makes $|k_p k_q| \cdot |k_p + k_q| \approx 2^{432} \cdot 2^{217} = 2^{649} \gg 2^{568} \approx e$. It has 81 bits more than 568-bit public exponent. Thus our scheme has $2^{81}$ complexity to defend exclusive search upon this boundary condition. Note that in Scheme A if $k_p$ and $k_q$ are known, then one can compute $p = k_p^{-1}(k_p - 1) \pmod{e}$ and $q = k_q^{-1}(k_q - 1) \pmod{e}$ from equations (1) and (2). Since the length of $e$ is 568-bit, which is longer than $p$ and $q$, one can get $p$ and $q$ immediately. Therefore we should keep the privacy of information $k_p$ and $k_q$ in the proposed scheme.

Defending against Partial Key Exposure Attacks

Here we consider the partial key exposure attacks for MSBs of secret exponent $d$[4][6]. The proposed scheme has $e$ of 568 bits and $d$ of 888 bits. Such a $d$ can be regarded as exposing the 136 MSBs of a 1024-bit $d$ (all the 136 MSBs are zero). Apply the theorem 3.4, our scheme is secure against these attacks.
4.4 More Security Considerations for Rebalanced RSA-CRT and the Proposed Scheme

In this section, we examine the difference between our scheme and the original Rebalanced RSA-CRT from the viewpoint of security. Further we consider the security of these two RSA variants by examining if the additional leaked information in these variants is helpful for an adversary to break these two. Recall that the proposed scheme has a secret exponent: $d = -ed_p d_q + d_p + d_q$, where $e$ is of 568 bits, $d_p$ and $d_q$ are of 160 bits, and $d$ is about of 888 bits. Thus $d \not\equiv d_p + d_q$ is about of 161 bits. This is obviously different from the standard RSA of which $d \equiv d_p + d_q$ is the size of order $e$. Considering the original Rebalance RSA-CRT, we say its secret exponent $d$ is computed from $d_p$ and $d_q$ by using CRT. Such a reconstructed $d$ will make $d \equiv d_p + d_q$ be the size of order $e$. Therefore it seems revealing no more available information on $d$ than that in the standard RSA. In the following we show that in fact, for the original Rebalanced RSA-CRT, there exists another secret exponent, $d' = -ed_p d_q + d_p + d_q$, which could reveal available information on $d'$ (mod $e$) as $d$ in our scheme.

Theorem 4.2 In the original Rebalanced RSA-CRT, the public key is $(N, e)$ and the secret key is $(d_p, d_q, p, q)$. Let $d' = -ed_p d_q + d_p + d_q$, then $d'$ can be used as another secret exponent for decryption.

Proof: Based on the key generation in the original Rebalanced RSA-CRT, we know that the reconstructed secret exponent $d$ satisfies $d_p \equiv d \pmod{p-1}$, $d_q \equiv d \pmod{q-1}$, and $ed \equiv 1 \pmod{\phi(N)}$. Thus there exists two numbers $k_p$ and $k_q$ such that $ed_p = k_p (p-1) + 1$ and $ed_q = k_q (q-1) + 1$. Similar to our scheme, we know $e (-ed_p d_q + d_p + d_q) = -k_p k_q (p-1) (q-1) + 1$. Let $d' = -ed_p d_q + d_p + d_q$. Therefore $ed' \equiv 1 \pmod{\phi(N)}$.

Note that here $e$ is of 1024 bits, $d_p$ and $d_q$ are of 160 bits, and hence $d'$ is about of 1344 bits and $d'$ (mod $e$) $\equiv d_p + d_q$ is about of 161 bits. Now we want to measure the amount of information on $d$ and $d'$ in our scheme and the original Rebalanced RSA-CRT respectively. For the proposed Scheme A, we can write $d = Ae + B$, where $d$ and $e$ are of 888 bits and 568 bits respectively. Hence $A (= -d_p d_q)$ is about of 320 bits and $B (= d_p + d_q)$ is about of 161 bits. This means the uncertainty for $d$ in our scheme is 481 bits. For the original Rebalanced RSA-CRT, similarly we can write $d' = Ae + B$, where $A$ is about of 320 bits and $B$ is about of 161 bits. Therefore, the original Rebalanced RSA-CRT has the same uncertainty for $d'$ as that for the secret exponent $d$ in our scheme. Note that this does not imply that our variant and the original Rebalanced RSA-CRT are security-equivalent, but gives us great confidence for the security of our variant. Our variant and the original Rebalanced RSA-CRT seem have the same property. So far the original Rebalanced RSA-CRT with 160-bit CRT-exponents is still secure.

Now we further consider if the above property in the proposed scheme will lead to insecurity. More precisely, with the help of $d = Ae + B$, where $e$ is of 568 bits, $d$ is about of 888 bits, $A$ is about of 320 bits and $B$ is about of 161 bits, whether could those existing attacks on RSA variants become workable for our scheme? And whether does there exist any new efficient attack? To the best of our knowledge, the answer for the first question is negative. For Wiener’s attack, this property can not further improve Wiener’s bound because the convergent condition for continued fraction remains unchanged. For the Boneh-Durfee Attack, no obvious information from this property is available to help solve the small inverse problem. As for the partial key exposure attacks for MSBs, we can not obtain a good approximation of $d$ by this property. So far, it is still an open problem if there exists any new efficient attack on the original Rebalanced RSA-CRT and/or our variant.
5 Another Variant of Rebalanced RSA-CRT

Here we consider another variant of Rebalanced RSA-CRT, called Scheme B. In Scheme B, we produces a 512-bit public exponent, e.g., \( e = 2^{511} + 1 \), and two 198-bit CRT-exponents \( d_p, d_q \). The encryption is about 3 times faster than that of Rebalanced RSA-CRT, but the decryption is a little slower than that. The key generation of the proposed scheme is as follows:

5.1 The Proposed Scheme (Scheme B)

Key Generation in Scheme B

Step 1. Randomly select an odd number \( e \) of 512 bits.

Step 2. Randomly select an odd number \( k_p \) of 198 bits, such that \( \gcd(k_p, e) = 1 \).

Step 3. Based on Theorem 4.1, we can uniquely determine two numbers \( d_p, k_p < d_p < 2k_p \), and \( p', e < p' < 2e \), satisfying \( ed_p - k_pp' = 1 \).

Step 4. If \( p = p' + 1 \) is not a prime number, then go to Step 2.

Step 5. Randomly select an odd number \( k_q \) of 198 bits, such that \( \gcd(k_q, e) = 1 \).

Step 6. Based on Theorem 4.1, we can uniquely determine two numbers \( d_q, k_q < d_q < 2k_q \), and \( q', e < q' < 2e \), satisfying \( ed_q - k_qq' = 1 \).

Step 7. If \( q = q' + 1 \) is not a prime number, then go to Step 5.

Step 8. The public key is \((N, e)\); the secret key is \((d_p, d_q, p, q)\).

Special Public Exponent and Security Considerations

In Scheme B, we can also select the special public exponent \( e = 2^{511} + 1 \). Thus only 512 modular multiplications are required for encryption. Consequently, the encryption in Scheme B is about 3 times faster than that in Rebalanced RSA-CRT. Compared with Scheme A, the encryption is still faster than that, but the decryption is a little slower. We generate an instance for Scheme B in Appendix A.2. Basically, the security analysis of Scheme B is very similar to that of Scheme A. Due the limit of space, we omit describing it and leave the details in the full version.

6 Implementations and Comparisons

In order to demonstrate our key generation algorithms in the proposed schemes are actually feasible, we implemented our algorithms and measured the average running time. The machine used for our implementations is a personal computer (PC) with 1.72 GHz CPU and 256 MB DRAM. The programming language we used is C under NTL with GMP on Windows system using Cygwin tools. We have experimented 100 samples for Scheme A. The average key-generation-time is 289 seconds. The average number of iterations for loops running from Step 2 to Step 4 is 35134. Besides, we use the well-known factoring algorithm, ECM algorithm[12], in the key generation. Note that in Step 4 and Step 7 of Scheme A we did not exactly find all prime factors of \( P \) (and \( Q \)), whose bit-size are less than or equal to 56, in order to compose a 56-bit number. Instead, we only try to find some smaller factors of \( P \) (and \( Q \)), e.g., those prime factors whose bit-size are less than or equal to 25. Of course, one can exactly find all prime factors whose bit-size are less than or equal to 56. But this will incur more time-consuming because the running time for the key generation in the proposed scheme is dominated by factorization. In Scheme B, the key generation time is only 86
Table 1: Comparisons of various RSA variants in terms of encryption and decryption.

<table>
<thead>
<tr>
<th></th>
<th>RSA-Basic</th>
<th>RSA-Short-D</th>
<th>RSA-CRT</th>
<th>Rebalanced RSA-CRT</th>
<th>Scheme A</th>
<th>Scheme B</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Public Exponent</strong></td>
<td>$2^{16}+1$</td>
<td>$1024$ bits</td>
<td>$2^{16}+1$</td>
<td>$1024$ bits</td>
<td>$2^{567}+1$</td>
<td>$2^{511}+1$</td>
</tr>
<tr>
<td><strong>Num of Multiplication</strong></td>
<td>$16+1$</td>
<td>$1024 \times 1.5$</td>
<td>$16+1$</td>
<td>$1024 \times 1.5$</td>
<td>$567+1$</td>
<td>$511+1$</td>
</tr>
<tr>
<td>in Encryption</td>
<td>$=17$</td>
<td>$=1536$</td>
<td>$=17$</td>
<td>$=1536$</td>
<td>$=568$</td>
<td>$=512$</td>
</tr>
<tr>
<td><strong>Unit Time for</strong></td>
<td>$0.011$</td>
<td>$1$</td>
<td>$0.011$</td>
<td>$1$</td>
<td>$0.37$</td>
<td>$0.33$</td>
</tr>
<tr>
<td>Encryption</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Secret Exponent</strong></td>
<td>$1024$ bits</td>
<td>$512$ bits</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>CRT-Exponent</strong></td>
<td>-</td>
<td>-</td>
<td>$512$ bits</td>
<td>$160$ bits</td>
<td>$160$ bits</td>
<td>$198$ bits</td>
</tr>
<tr>
<td>(Modular Size)</td>
<td>$1024 \times 1.5$</td>
<td>$512 \times 1.5$</td>
<td>$2 \times 512 \times 1.5$</td>
<td>$2 \times 160 \times 1.5$</td>
<td>$2 \times 198 \times 1.5$</td>
<td></td>
</tr>
<tr>
<td>in Decryption</td>
<td>=1536</td>
<td>=768</td>
<td>+2=1538</td>
<td>+2=482</td>
<td>+2=596</td>
<td>+2=596</td>
</tr>
<tr>
<td><strong>Num of Operations</strong></td>
<td>$2 \times \log_2 d \times (\log_2 N)^2$</td>
<td>$2 \times \frac{3}{2} \times \log_2 d_p \times (\log_2 N) \times (\log_2 N)^2$</td>
<td>$2 = \frac{1}{2} \log_2 d_p \times (\log_2 N)^2$</td>
<td>$2 = \frac{1}{2} \log_2 d_q \times (\log_2 N)^2$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>in Decryption</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Unit Time for</strong></td>
<td>$1$</td>
<td>$0.5$</td>
<td>$0.25$</td>
<td>$0.078125$</td>
<td>$0.09966$</td>
<td></td>
</tr>
<tr>
<td>Decryption</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1 summarizes the parameters used in various RSA variants and gives comparisons of these RSA variants in terms of encryption and decryption. We recall that the number of binary operations to compute $Z^a \pmod{b}$ is $1.5 \log_2 a \cdot (\log_2 b)^2$. If $a$ is the special form of $2^m + 1$, then the number of binary operations will be reduced to $(m+1) \cdot (\log_2 b)^2[9]$. In addition, we assume that a full modular exponentiation, $Z^a \pmod{b}$, where both $a$ and $b$ are of 1024 bits, takes one unit time to compute. It is clear that the encryption in Scheme A and Scheme B are about 2.7 times and 3 times faster than that in Rebalanced RSA-CRT respectively.

7 Conclusions

This paper presents two variants of Rebalanced RSA-CRT, Scheme A and Scheme B, to further reduce the encryption cost in the original Rebalanced RSA-CRT. In Scheme A, we not only shorten the public exponent in Rebalanced RSA-CRT from 1024 bits to 568 bits, but also make the public exponent to be of the special form: $e = 2^{567} + 1$. The encryption time is therefore reduced to about $\frac{1}{2.7}$ of the time required by the original Rebalanced RSA-CRT. Indeed, the key generation in our scheme is slow (289 seconds in average) due to factorization. This can be easily improved by algorithm optimization and parallel techniques. In Scheme B, the public exponent is even shorter than that in Scheme A, e.g. $e = 2^{511} + 1$, but the decryption is a little slower. We use 198-bit CRT-exponents in Scheme B, which is rather larger than the smallest 160 bits. We also show that in fact there exists another secret exponent, $d' = -ed_p d_q + d_p + d_q$, which may bring us more useful information over the regular secret exponent $d$ in Rebalanced RSA-CRT. More precisely, $d' \pmod{e} \equiv d_p + d_q$ is only about of 160 bits, but $d \pmod{e}$ is the size of order $e$. However it is still an open problem whether this property is actually helpful to break Rebalanced RSA-CRT.
References


Appendix A: Examples of Scheme A and Scheme B

Appendix A.1: An example of Scheme A with $e = 2^{559} + 1$, $d_p$ and $d_q$ of 160 bits, $p$ of 513 bits and $q$ of 512 bits.

\[
p = 1 049FC5D7 247DCF38 D46F45D5 4C5CB74C 289924A3 21BBAC38 4CC06AC3 7BF81A6B 1BC5596F AB047D99 2A112C09 78767657 AB1F4F9D FAF071BE 4AE5BD89
q = B3872669 694F86B0 0ECAAA80 F57C7259 761FF048 A9DDB83A 71588C23 3F39A82A 3547350A 9C34D87E 8E22E27E 3F590793 B1C3924C D21A834D 1985BACB A6998E07
d_p = C8D2B8F3 3AEE12F7 CB570958 F7DC88D3 1D11AAD1
d_q = C43BE36C C17B8BC8 4071661C FA465A67 DDCD2385
e = 2^{567} + 1
\]

Appendix A.2: An example of Scheme B with $e = 2^{511} + 1$, $d_p$ and $d_q$ of 198 bits, $p$ and $q$ of 512 bits.

\[
p = 8AF105A8 85F84D30 16ED6D69 E1DA359F 09BDA979 6CDA651E DBCC4F52 994AEB4 A70B64B 3E3E0383 D73AF9B5 444919D1 7F00C90D F0765B4C A1884148 368E8F1B
q = 83CCE530 49C698ED 36032BBF B0F21A34 70B9608C 50BCB458 CE53F6C0 3E50BE33 16D59EA5 A7FA305 DDB2D6DE E99D5A7E 11BA28D1 7E49C682 B6BBD086 34F4668F
d_p = 3B 511D01FC 82437878 BB60BF77 47B30EC5 0CF65340 573A7503
d_q = 27 0345E74F DC319B67 4DC6FD9A 7EDF33EA 0F6BB29B 434801FF
e = 2^{511} + 1
\]