More Efficient Cryptosystems From k^{th} Power Residues*

Zhenfu Cao[1], Xiaolei Dong[1], Licheng Wang[2], and Jun Shao[3]

[1] Department of Computer Science and Engineering, Shanghai Jiaotong University
[2] State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications
[3] School of Computer and Information Engineering, Zhejiang Gongshang University
{zfcao,dong-xl}@cs.sjtu.edu.cn, wanglc@bupt.edu.cn, chn.junshao@gmail.com

Abstract. At Eurocrypt 2013, Joye and Libert proposed a method for constructing public key cryptosystems (PKCs) and lossy trapdoor functions (LTDFs) from $(2^\alpha)^{th}$-power residue symbols. Their work can be viewed as non-trivial extensions of the well-known PKC scheme due to Goldwasser and Micali, and the LTDF scheme due to Freeman et al., respectively. In this paper, we will demonstrate that this kind of work can be extended *more generally*: all related constructions can work for any k^{th} residues if k only contains small prime factors, instead of $(2^\alpha)^{th}$-power residues only. The resultant PKCs and LTDFs are more efficient than that from Joye-Libert method in terms of decryption speed with the same message length.

Keywords: Goldwasser-Micali cryptosystem, k^{th} power residuosity, k-residue discrete logarithm, additive homomorphism, lossy trapdoor function

1 Introduction

Public key cryptosystem (PKC) is one of fundamental building blocks for securing digital communications. Many public key encryption schemes have been proposed so far [31,18,4,15,8,22,16,13]; however, it is fair to say that they can be classified into several main categories, such as the ElGamal-type [18,15,8,16,13], the RSA-type [31,4,8,22], the GM-type [20,23], and others [2,26,17,25,28]. Most of the first two types of cryptosystems focus on how to enhance the security or enrich the properties of original cryptosystem, while most of the third type of cryptosystems focus on how to improve the effectiveness and efficiency of the original cryptosystem.

At Eurocrypt 2013, Joye and Libert [23] proposed a very natural generalization of the GM cryptosystem, while it results in the most efficient GM-type cryptosystem in terms of bandwidth (i.e., the ciphertext length) and decryption speed. Specially, the GM cryptosystem uses quadratic residue symbols w.r.t. modulus n, while the Joye-Libert cryptosystem makes use of $(2^\alpha)^{th}$ power residue symbols w.r.t. modulus n. The bandwidth and the time complexity of decryption of the Joye-Libert cryptosystem are $\log n$ and $\mathcal{O}(\alpha(\log n)^2(\log\log n))$, respectively.

In this paper, we propose two new GM-type cryptosystems that take advantages of k^{th}-power residue symbols, where $k = p_1^{\alpha_1}\cdots p_s^{\alpha_s}$ $(2 \le p_1 < \cdots < p_s)$ only contains small prime factors. Surprisingly, our proposal is more efficient than the Joye-Libert cryptosystem in terms of decryption speed. In particular, the time complexity of decryption in our proposal can be approximately reduced to $\mathcal{O}(\alpha'(\log n)^2(\log\log n))$, where $\alpha' = \sum_i^s \alpha_i$ in general is observably less than α even with the similar scales of message space and ciphertext space. Our proposal inherits the additive homomorphism property of the GM cryptosystem, which allows the additively homomorphic operation on larger messages while keeping the same efficiency. Hence, our proposal also yields efficient lossy trapdoor functions, even with better lossiness and faster inversion algorithm compared to the Joye-Libert constructions.

* This work was supported by the National Natural Science Foundation of China (NSFC) (nos. 61033014, 61371083, 61373154, 61370194) and the NSFC A3 Foresight Program (no. 61161140320).

1.1 Related work

At STOC 1982, Goldwasser and Micali proposed the first probabilistic key encryption scheme [20]. It is quite simple and elegant. The public key and private key are $\{n, a\}$ and p, respectively. Here, n is an RSA modulus $n = p \cdot q$, and a is a non-square in $\mathbb{J}_{n,2}$ (i.e., the set of elements in $\mathbb{Z}_n^*$ whose Jacobi symbol is 1). For an ℓ-bit message m, it is encrypted bit by bit. That is, the ciphertext is $(c_1, \cdots, c_\ell)$ where $c_i = \langle a^{m_i} x_i^2 \rangle_n$ for some random $x_i \in \mathbb{Z}_n$. Accordingly, the message is recovered bit by bit: If c_i is a quadratic residue modulo p, then $m_i = 0$; otherwise, $m_i = 1$. The ciphertext is $\ell \log n$ bits in length, and the time complexities of encryption and decryption are both $\mathcal{O}(\ell(\log n)^2)$. These results suggest that the GM cryptosystem is efficient in encryption and decryption, while it is inefficient in the bandwidth utilization. Several proposals were made to address this issue.

Short afterwards (at CRYPTO 1984), by using the Blum-Blum-Shub pseudorandom generator [5,6], Blum and Goldwasser [7] proposed another efficient probabilistic encryption scheme that achieves a smaller bandwidth in $\ell + \log n$ bits, and a smaller time complexity of encryption $\mathcal{O}(\frac{\ell(\log n)^2}{\log \log n})$. As mentioned in [7], the decryption time complexity of the Blum-Goldwasser cryptosystem is $\mathcal{O}((\log n)^3) + \mathcal{O}(\frac{\ell(\log n)^2}{\log \log n})$, and it is slightly faster than schemes RSA [31] or Rabin [30] when $\ell > \log n$.

During 1988 to 1990, Cao [9,10,11] proposed two types of extensions of the GM cryptosystem. One is based on the cubic residue in ring $\mathbb{Z}[\omega]$ that allows 3-adic encoding. It results in an even fast decryption with the same length message. The other extension is based on the k^{th}-power residues that enables segment encryption instead of bit encryption in the GM cryptosystem. The concept of indistinguishability due to Goldwasser and Micali is also extended to k-indistinguishability in these work (it is also summarized in [12]).

Four years later (at STOC 1994), Benaloh and Tuinstra [3] makes another extension on the GM cryptosystem. Their cryptosystem sets the public key as a triple $\{n, k, a\}$ such that n is still an RSA modulus $n = p \cdot q$, k is an ℓ-bit length prime, $k | \phi(n)$, $k^2 \nmid p - 1$, $a \in \mathbb{Z}_n$ and $a^{\phi(n)/k} \not\equiv 1 \pmod n$, where $\phi(n)$ is Euler's totient function. The encryption of a message m with ℓ-bit length but smaller than k is given by $c = \langle a^m x^k \rangle_n$ for a random $x \in \mathbb{Z}_n$. Clearly, the encryption time complexity is $\mathcal{O}((\log n)^2 (\log \log n))$ considering that the encryption cost is mainly occupied by two exponentiations. The most promising feature of the Benaloh-Tuinstra cryptosystem is that the bandwidth is reduced to $\log n$ bits. However, the decryption requires searching over the entire message space $[0, k) \subseteq \{0,1\}^\ell$ for locating m such that $a^{m \cdot \phi(n)/k} \equiv c^{\phi(n)/k} \pmod n$ holds. Thus, the scheme is in practice limited to small values of ℓ, say $\ell < 40$ [23].

Another four years later (at CCS 1998), Naccache and Stern [27] further improved the GM cryptosystem along the line of the Benaloh-Tuinstra cryptosystem. The public key is still $\{n, k, a\}$, and only k has different properties. That is, k is a product of small (odd) primes $k = \prod p_i$ such that $p_i | \phi(n)$ and $p_i^2 \nmid \phi(n)$. With the new properties of k, the message m is recovered from $m \equiv m_i \pmod{p_i}$ through Chinese Remainder Theorem (CRT). It results in that the size of message space could be as large as $\prod p_i$, while the size of searching space is only $\sum p_i$. However, the condition $p_i^2 \nmid \phi(n)$ makes its message space cannot be further enlarged.

Most recently (at EUROCRYPT 2013), Joye and Libert [23] made a further improvement on the Naccache-Stern cryptosystem by setting $k = 2^\alpha$. But the newly conceived decryption algorithm can be efficiently finished by a bit-by-bit manner. The time complexities of encryption and decryption are $\mathcal{O}((\log n)^2 (\log \log n))$ and $\mathcal{O}(\alpha (\log n)^2 (\log \log n))$, respectively. The fact allows the message to be very long, say 128 bits as they suggested.

1.2 Our Contribution

In this paper, we make a further extension on Goldwasser and Micali's work and obtain two cryptosystems, denoted by V_0 and V_1 respectively, that are more efficient than other GM-type cryptosystems. The main idea is to set k as a product of *powers* of small primes, and coupling with a fast decryption algorithm for recovering k-residue discrete logarithm (See Definition 4). To prove the security of our proposal, we introduce two assumptions. One is the k^{th}-power residuosity (k^{th}-PR) assumption: given an element $x \in \mathbb{Z}_n^*$ such that the generalized Jacobi symbol (see Definition 3) is 1, no probabilistic polynomial time (PPT) adversary can decide whether x is a k^{th}-power residue or not. The other is the strong k^{th}-power residuosity (k^{th}-SPR) assumption: given an element $x \in \mathbb{Z}_n^*$, no PPT adversary can decide whether x is a k^{th}-power residue or not. Our proposal is a generalized framework in the sense that when $k = 2$, or k is a prime such that $k|\phi(n), k^2 \nmid \phi(n)$, or k is a product of small (odd) primes $k = \prod p_i$ such that $p_i|\phi(n)$ and $p_i^2 \nmid \phi(n)$, or $k = 2^\alpha$ with $\alpha \geq 1$, our first scheme V_0 is instantiated to the GM cryptosystem, the Benaloh-Tuinstra cryptosystem, the Naccache-Stern cryptosystem and the Joye-Libert cryptosystem, respectively. In brief, the highlights of our proposal are summarized as follows.

- Efficient in encryption and bandwidth. The encryption time complexity and the bandwidth of our proposal are $\mathcal{O}((\log n)^2(\log\log n))$ and $\log n$ bits, respectively. These features are as good as that in the Benaloh-Tuinstra cryptosystem, the Naccache-Stern cryptosystem and the Joye-Libert cryptosystem.
- Lower ciphertext expansion factor. With the same length of modulus n, the ciphertext expansion factor of our second scheme V_1 is about 2 under some reasonable setting on k. This feature is better than all aforementioned GM-type cryptosystems.
- Large message space. It would be good if the message space could be enlarged with the same ciphertext space while keeping the efficiency in encryption and decryption. When $k = \prod p_i^{\alpha_i}$ with that p_i's are small primes, and α_i's are positive numbers, the message m in our proposal is reconstructed from $m_i = m \pmod{p_i^{\alpha_i}}$ through CRT like the Naccache-Stern cryptosystem does. However, the space of m_i is enlarged from $[0, p_i - 1)$ to $[0, p_i^{\alpha_i} - 1)$ without increasing the searching space.
- Faster in decryption. The decryption time complexity of our proposal is $\mathcal{O}(\alpha'(\log n)^2(\log\log n))$, where $\alpha' = \sum \alpha_i$ under the setting $k = \prod p_i^{\alpha_i}$ with small distinct primes p_i and positive α_i $(i = 1, \cdots, s)$. Compared with the Joye-Libert cryptosystem, our schemes can decrypt even faster in practice. The reason is that the time complexity of decryption in our proposal is mainly occupied by α', which is in general observably smaller than α in the Joye-Libert cryptosystem, under the condition $2^\alpha \approx \prod p_i^{\alpha_i}$ (i.e., the roughly equal size of message space). This advantage is manifested by intensive tests (See Section 5.2).
- Additively Homomorphic over large message space. Our proposal admits additive homomorphism over large message space. In particular, our second scheme V_1 can also yields an efficient lossy trapdoor function with even better lossiness and faster inversion algorithm compared to the Joye-Libert constructions.

2 Background

In this section, we review some definitions related to our proposal, and introduce the notations in our paper. In particular, we review the definitions and theorems related to k^{th}-power residuosity, generalized Legendre symbol, and generalized Jacobi symbol. We also define the k-residue discrete logarithm problem.

For simplicity, we would like to firstly introduce the notations used in the rest of this paper in Table 1.

Table 1. Notations used in this paper

Notation	Description
$\mathbb{Z}_n$	the set of non-negative minimal residues w.r.t. modulus n, i.e., $\{0, 1, \cdots, n-1\}$
$\langle x\rangle_n$	$x \bmod n$, result in a non-negative minimal residue
$(x)_n$	$x \bmod n$, result in a absolute minimal residue
(a, b)	the greatest common divisor of a and b
$[a, b]$	the least common multiple of a and b
$\mathbb{Z}_n^*$	the multiplication group w.r.t. modulus n, i.e., $\{a \in \mathbb{Z}_n \mid (a, n) = 1\}$
$\mathrm{ord}_n(a)$	a's order w.r.t. modulus n
$\left(\frac{a}{p}\right)_k$	a's generalized Legendre symbol w.r.t modulus p (see Definition 2)
$\mathbb{J}_{p,k}^{(i)}$	the set of elements $a \in \mathbb{Z}_p^*$ such that $\left(\frac{a}{p}\right)_k = \left(\frac{\omega_{p,i}}{p}\right)_k$ (see Theorem 2)
$\left(\frac{a}{n}\right)_k$	a's generalized Jacobi symbol w.r.t modulus n (see Definition 3)
$\mathbb{J}_{n,k}$	the set of elements $a \in \mathbb{Z}_n^*$ such that $\left(\frac{a}{n}\right)_k = 1$, $\mathbb{J}_{n,k} = \mathbb{R}_{n,k} \cup \mathbb{NR}_{n,k}$
$\mathbb{R}_{n,k}$	the set of k^{th}-power residues w.r.t. modulus n, i.e., $\{(x^k)_n \mid x \in \mathbb{Z}_n^*\}$
$\mathbb{NR}_{n,k}$	the set of k^{th}-power non-residues in $\mathbb{J}_{n,k}$, i.e., $\mathbb{J}_{n,k} \setminus \mathbb{R}_{n,k}$
$\log x$	the logarithm of x w.r.t. the base 2, i.e., $\log_2 x$

Definition 1 **(k^{th}-Power Residuosity).** *For any positive integers k and n, we say that an integer $b \in \mathbb{Z}_n^*$ is a k^{th}-power residue w.r.t. modulus n if and only if the following congruent equation has solution(s)*

$$x^k \equiv b \pmod{n}. \tag{1}$$

Accordingly, if Equation (1) has no solution, we say b is a k^{th}-power non-residue w.r.t. modulus n. □

Theorem 1 **(Existence of k^{th}-Power Residuosity).** *[12, Theorem 4.1, page 55] For any prime p and integer $n = p^\alpha$ or $n = 2p^\alpha$, an integer $b \in \mathbb{Z}_n^*$ is a k^{th}-power residue w.r.t. modulus n if and only if $b^{\frac{\phi(n)}{(k,\phi(n))}} \equiv 1 \pmod{n}$, where $\phi(n)$ is Euler quotient function.* □

Definition 2 **(Generalized Legendre Symbol).** *For any integers a, k and prime p such that $k|p-1$ and $(a, p) = 1$, the generalized Legendre symbol (GLS) is defined by*

$$\left(\frac{a}{p}\right)_k = \left(a^{\frac{p-1}{k}}\right)_p. \tag{2}$$

□

From Definition 2, it is easy to deduce that if $a \equiv b \pmod{p}$, then $\left(\frac{a}{p}\right)_k = \left(\frac{b}{p}\right)_k$, and that GLS is multiplicative for any integers a, b such that $(a, p) = (b, p) = 1$, i.e.,

$$\left(\frac{ab}{p}\right)_k \equiv \left(\frac{a}{p}\right)_k \left(\frac{b}{p}\right)_k \pmod{p}. \tag{3}$$

Theorem 2 **(Number of GLS).** *[12,24] There are exactly k distinct Legendre symbols, including 1, and there are k distinct elements $\omega_{p,0}, \cdots, \omega_{p,k-1} \in \mathbb{Z}_p^*$ such that $\left(\frac{\omega_{p,0}}{p}\right)_k = 1$ and $\left(\frac{\omega_{p,i}}{p}\right)_k \neq \left(\frac{\omega_{p,j}}{p}\right)_k \quad (i \neq j)$.* □

Definition 3 **(Generalized Jacobi Symbol).** *For integers a, k, n, where $n = pq$ with two primes p and q, $k|p-1$, $k|q-1$ and $(a,n) = 1$, the generalized Jacobi symbol (GJS) is defined by*

$$\left(\frac{a}{n}\right)_k = \left(\frac{a}{p}\right)_k \left(\frac{a}{q}\right)_k. \tag{4}$$

□

Lemma 1 **(Properties of $\mathbb{R}$, $\mathbb{J}$ and $\mathbb{NR}$).** *For arbitrary integers a, k and two primes p, q so that $k|p-1, k|q-1$, we have*

(1) $\mathbb{R}_{p,k} = \mathbb{J}_{p,k}$.
(2) $|\mathbb{R}_{p,k}| = \frac{p-1}{k}$.
(3) $a \in \mathbb{R}_{pq,k} \Leftrightarrow (a)_p \in \mathbb{R}_{p,k}$ *and* $(a)_q \in \mathbb{R}_{q,k}$.
(4) $a \in \mathbb{NR}_{pq,k} \Leftrightarrow (a)_p \in \mathbb{NR}_{p,k}$ *and* $(a)_q \in \mathbb{NR}_{q,k}$.

Proof. Properties (1), (3) and (4) are apparently according to definitions on the related notions. So, let us prove property (2). The basic idea can be find in [24]. Suppose that g is one of p's primitive roots, then when $p \nmid a$, we have that $x^k \equiv a \pmod p$ has a solution if and only if $k = (k, p-1)|\mathrm{ind}_g(a)$, where $\mathrm{ind}_g(a)$ denotes a's index (i.e., discrete logarithm) w.r.t. base g and modulus p. Thus, for $\mathrm{ind}_g(a) = k, 2k, \cdots, \frac{p-1}{k} \cdot k$,

$$g^k,\ g^{2k},\ \cdots,\ g^{\frac{p-1}{k}k} \tag{5}$$

are exactly all k^{th}-power residues w.r.t. the modulus p. From expression (5), we can see that each number is distinct in the sense of taking modulo p. Thus, property (2) holds. □

Definition 4 **(k-Residue Discrete Logarithm, k-RDL).** *For prime p and two positive integers b, k such that $k|p-1$ and $\mathrm{ord}_p(b) = k$, the k-discrete logarithm problem is to find x ($0 \le x < k$) satisfying $b^x \equiv y \pmod p$ for a given integer $y \in \mathbb{Z}_p^*$. We call x as y's k-discrete logarithm w.r.t. base b and modulus p. When k contains only small prime factors, we call x as y's k-residue discrete logarithm (k-RDL) w.r.t. base b and modulus p, denoted as $x = RDL_{b,p}^k(y)$.*

We will show that the k-RDL problem can be solved effectively and efficiently in Section 3.3. This fact is the base of our subsequent construction.

3 New PKC Scheme From k^{th} Power Residue

In this section, we further generalize the Joye-Libert cryptosystem that is a generalized of the GM cryptosystem.

3.1 Basic Scheme

Our basic construction, denoted by V_0, consists of the following three algorithms.

`KeyGen`**:** On inputting the security parameter κ, `KeyGen` outputs the public key $pk = \{n, k, a\}$ and private key $sk = \{p, q\}$, where
- $n = p \cdot q$, $p = k \cdot p' + 1$, $q = k \cdot q' + 1$, and $(k, p'q') = 1$;
- p, q are big primes, while k is a product of small primes;

– p', q' both contain big prime factors.
– choose a from $\mathbb{Z}_n^*$ such that $\mathrm{ord}_p(a) = \mathrm{ord}_q(a) = k$. See the details on how to choose a in Section 3.2.

`Enc`: On inputting a message $m \in \mathbb{Z}_k$, the encryptor selects $x \in \mathbb{Z}_n^*$ at random and outputs the ciphertext

$$c = \langle a^m x^k \rangle_n. \tag{6}$$

`Dec`: On inputting a ciphertext $c \in \mathbb{Z}_n^*$, the decryptor knowing p can obtain the message as follows.

1. Compute $b = \left(\frac{a}{p}\right)_k$, which actually can be pre-computed.
2. Compute $y = \left(\frac{c}{p}\right)_k$.
3. Recover the message $m = RDL_{b,p}^k(y)$ by using the method in Section 3.3.

Correctness From Equation (7) and Lemma 2, we can easily obtain the correctness of our proposal.

$$y \equiv \left(\frac{c}{p}\right)_k \equiv \left(\frac{a^m}{p}\right)_k \left(\frac{x^k}{p}\right)_k \equiv \left(\frac{a}{p}\right)_k^m \equiv b^m \pmod p \tag{7}$$

Lemma 2 (The Order of b). *Assume that b is computed as that in our proposal, then we have* $\mathrm{ord}_p(b) = k$.

Proof. From

$$1 \equiv b^{\mathrm{ord}_p(b)} \equiv a^{\frac{p-1}{k}\mathrm{ord}_p(b)} \pmod p,$$

we have that $k | p'\mathrm{ord}_p(b)$. With $(k, p') = 1$, we can obtain $k | \mathrm{ord}_p(b)$. On the other hand, from

$$b^k \equiv \left(a^{\frac{p-1}{k}}\right)^k \equiv 1 \pmod p.$$

we have that $\mathrm{ord}_p(b) | k$. This finishes the lemma ☐

Remark 1. We notice that the cases $k = 2$ and $k = 2^\alpha$ ($\alpha \geq 1$) are roughly corresponding to the GM cryptosystem and the Joye-Libert cryptosystem, respectively. The only subtle difference is the choose of a. In particular, a is not explicitly selected from $\mathbb{NR}_{n,k}$ in our proposal. In fact, Joye and Libert's idea for proving that $\mathrm{ord}_p\left(\left(\frac{a}{p}\right)_{2^\alpha}\right) = 2^\alpha$ (when $a \in \mathbb{NR}_{n,2}$) cannot be easily extended to the case of $k > 2$ or the case that k only contains small prime factors. Fortunately, we evade this obstacle by selecting a to be a k-order element, and Lemma 2 enables our scheme work well.

Remark 2. Recall that Shor's quantum algorithm [32] for factoring n consists of the following classical step: If for some random element $a \in \mathbb{Z}_n^*$, $\mathrm{ord}_n(a) = k$ is even, then $(a^{k/2} \pm 1, n)$ might be a non-trivial factor of n. However, in our scheme V_0, even if k is even, this will not lead to the factorization of n, since in this case we have $n | a^{k/2} + 1$, $p \nmid a^{k/2} - 1$ and $q \nmid a^{k/2} - 1$. That is, $(a^{k/2} \pm 1, n)$ must be 1 or n.

3.2 Generation of a

Before giving the method of generating a, we would like to introduce Lemma 3 and Lemma 4, which support the feasibility of our method.

Lemma 3. *For $u, v \in \mathbb{Z}_p^*$ such that p is a big prime, if* $(\mathrm{ord}_p(u), \mathrm{ord}_p(v)) = 1$, *then* $\mathrm{ord}_p(uv) = \mathrm{ord}_p(u)\mathrm{ord}_p(v)$.

Proof. The proof follows the idea in [12, page 123]. From $(uv)^{\mathrm{ord}_p(uv)} \equiv 1 \pmod p$, we have that

$$((uv)^{\mathrm{ord}_p(uv)})^{\mathrm{ord}_p(u)} \equiv (u^{\mathrm{ord}_p(u)} v^{\mathrm{ord}_p(u)})^{\mathrm{ord}_p(uv)} \equiv v^{\mathrm{ord}_p(u)\mathrm{ord}_p(uv)} \equiv 1 \pmod p.$$

The above reduction shows that $\mathrm{ord}_p(v) | \mathrm{ord}_p(u)\mathrm{ord}_p(uv)$, which leads to $\mathrm{ord}_p(v)|\mathrm{ord}_p(uv)$. Similarly, $\mathrm{ord}_p(u)|\mathrm{ord}_p(uv)$ holds. Hence, we have that $\mathrm{ord}_p(u)\mathrm{ord}_p(v)|\mathrm{ord}_p(uv)$.

On the other hand, $(uv)^{\mathrm{ord}_p(u)\mathrm{ord}_p(v)} \equiv 1 \pmod p$ leads to $\mathrm{ord}_p(uv)|\mathrm{ord}_p(u)\mathrm{ord}_p(v)$. Hence, we have that $\mathrm{ord}_p(uv) = \mathrm{ord}_p(u)\mathrm{ord}_p(v)$.

□

Lemma 4. *If p, p' are primes, $p'^{\alpha}|p-1$, $p'^{\alpha+1} \nmid p-1$, $\alpha \geq 1$, $b^{\frac{p-1}{p'}} \not\equiv 1 \pmod p$, and $a = \langle b^{\frac{p-1}{p'^{\alpha}}} \rangle_p$, then we have that* $\mathrm{ord}_p(a) = p'^{\alpha}$.

Proof. The proof also follows the idea in [12, page 124]. From $a^{p'^{\alpha}} \equiv (b^{(p-1)/p'^{\alpha}})^{p'^{\alpha}} \equiv 1 \pmod p$, we have that $\mathrm{ord}_p(a)|p'^{\alpha}$. On the other hand, we also have $\mathrm{ord}_p(a) = p'^{\alpha}$. If not, we have that $\mathrm{ord}_p(a) = p'^{\alpha'}$ for some $\alpha' < \alpha$, which leads to

$$1 \equiv a^{\mathrm{ord}_p(a)} \equiv (b^{(p-1)/p'^{\alpha}})^{\mathrm{ord}_p(a)} \equiv b^{(p-1)/p'^{\alpha-\alpha'}} \pmod p$$

Raise the both sides of the above equation to the power of $p'^{\alpha-\alpha'-1}$, we can obtain that $b^{(p-1)/p'} \equiv 1 \pmod p$, which is a contradiction for the condition of b. This finishes the lemma. □

Now, we can give the method to choose a such that $\mathrm{ord}_p(a) = k$. Assume that $k = \prod_{i=1}^{s} p_i^{\alpha_i}$, where p_i $(i = 1, \cdots, s,\ p_1 < \cdots < p_s)$ are small primes, and $\alpha_i \geq 1$. Our main idea is to generate $a_i \in \mathbb{Z}_p^*$ $(i = 1, \cdots, s)$ such that $\mathrm{ord}_p(a_i) = p_i^{\alpha_i}$, and then compute $a = a_1 \cdot \ldots \cdot a_s \bmod p$. According to Lemma 3, we have that $\mathrm{ord}_p a = \prod_{i=1}^{s} \mathrm{ord}_p(a_i) = \prod_{i=1}^{s} p_i^{\alpha} = k$. Thus, we only need to show how to choose a_i such that $\mathrm{ord}_p(a_i) = p_i^{\alpha_i}$. In fact, we can choose randomly b_i from $\mathbb{Z}_p^*$ such that

$$b_i^{\frac{p-1}{p_i}} \not\equiv 1 \pmod p. \tag{8}$$

It is easy to see that Inequality (8) holds with probability $1 - 1/p_i$ at least. After that, compute $a_i = \langle b_i^{(p-1)/p_i^{\alpha_i}} \rangle_p$. According to Lemma 4, we have that $\mathrm{ord}_p(a_i) = p_i^{\alpha_i}$.

Piecing all above together, the algorithmic description of generation of a can be found in Algorithm 1.

Now, let us show how to generate a such that $\mathrm{ord}_p(a) = \mathrm{ord}_q(a) = k$. By using the above method, we can find $a_1 \in \mathbb{Z}_p^*$ and $a_2 \in \mathbb{Z}_q^*$ such that $\mathrm{ord}_p(a_1) = k_1$ and $\mathrm{ord}_q(a_2) = k_2$, respectively. Then, according to CRT, from $a \equiv a_1 \pmod p$ and $a \equiv a_2 \pmod q$, we can obtain

$$a = \langle M_1 \cdot q \cdot a_1 + M_2 \cdot p \cdot a_2 \rangle_{pq},$$

where M_1 and M_2 are positive integers such that

$$M_1 \cdot q \equiv 1 \pmod p, \quad \text{and} \quad M_2 \cdot p \equiv 1 \pmod q.$$

Therefore, $\mathrm{ord}_p(a) = \mathrm{ord}_p(a_1) = k_1$ and $\mathrm{ord}_q(a) = \mathrm{ord}_q(a_2) = k_2$. In particular, when $k_1 = k_2 = k$, we obtain a generator a for Scheme V_0, while when $(k_1, k_2) = 1$, we obtain another generator a for Scheme V_1 that will be introduced later.

Algorithm 1 Generation of a

Input:

Prime p and integer k, where $k|p-1$, $k=\prod_{i=1}^{s} p_i^{\alpha_i}$, and p_i are small distinct primes.

Output:

$a \in \mathbb{Z}_p^*$ such that $\mathrm{ord}_p(a)=k$.

```
a ← 1;
for i from 1 to s do
    Loop: b_i ←$ Z_p;                          ▷ Choose a_i such that ord_p(a_i) = p_i^{α_i}
    if b_i^{(p−1)/p_i} ≡ 1 (mod p) then
        goto Loop;
    end if
    a_i ← ⟨b_i^{(p−1)/p_i^{α_i}}⟩_p;
    a ← ⟨a a_i⟩_p;
end for
Output a.
```

3.3 Solve the k-RDL Problem

Suppose that p is a prime and k, b are two positive integers such that $k|p-1$ and $\mathrm{ord}_p(b) = k$. When k contains only small prime factors, we can solve the following k-RDL problem efficiently as long as there exists a solution:

$$y \equiv b^m \pmod{p}. \tag{9}$$

The basic idea of computing $m \in \mathbb{Z}_k$ follows that in [12, pages 126–128].

Assume that $k = \prod_{i=1}^{s} p_i^{\alpha_i}$, where p_i are small primes such that $p_1 < \cdots < p_s$ and $\alpha_i \geq 1$ $(i = 1, \cdots, s)$. Our main idea is to generate $m_i \in \mathbb{Z}_{p_i^{\alpha_i}}$ $(i = 1, \cdots, s)$ such that

$$m_i \equiv m \pmod{p_i^{\alpha_i}}, \tag{10}$$

and then compute $m \in \mathbb{Z}_k$ satisfying Equation (9) by using CRT. Therefore, the main task of computing m is to compute m_i $(i = 1, \cdots, s)$ satisfying Equation (10).

Firstly, suppose that $y_0 = y$ and m_i can be represented as

$$m_i = m_{i,0} + m_{i,1} \cdot p_i + \cdots + m_{i,\alpha_i - 1} \cdot p_i^{\alpha_i - 1}. \tag{11}$$

Then from Equation (10), we have that

$$\left(\frac{y_0}{p}\right)_{p_i^{\alpha_i}} \equiv \left(\frac{b^{m_i}}{p}\right)_{p_i^{\alpha_i}} \equiv \left(\frac{b}{p}\right)_{p_i^{\alpha_i}}^{m_i} \pmod{p}. \tag{12}$$

Raise the both sides of Equation (12) to the power of $p_i^{\alpha_i - 1}$, we have that

$$y_0^{\frac{p-1}{p_i}} \equiv b^{\frac{p-1}{p_i} \cdot m_i} \equiv b^{\frac{p-1}{p_i} \cdot (m_{i,0} + m_{i,1} \cdot p_i + \cdots + m_{i,\alpha_i - 1} \cdot p_i^{\alpha_i - 1})} \equiv b^{\frac{p-1}{p_i} \cdot m_{i,0}} \pmod{p}. \tag{13}$$

We can determine $m_{i,0}$ by searching it in $\{0, 1, \cdots, p_i - 1\}$. This is quite efficient since p_i is small.

Next, set $y_1 = \langle y_0 \cdot b^{-m_{i,0}} \rangle_p$. Similar with Equation (13), we have that

$$y_1^{\frac{p-1}{p_i^2}} \equiv b^{\frac{p-1}{p_i^2} (m_{i,1} p_i + \cdots + m_{i,\alpha_i - 1} p_i^{\alpha_i - 1})} \equiv b^{\frac{p-1}{p_i} \cdot m_{i,1}} \pmod{p}. \tag{14}$$

Again, we can determine $m_{i,1}$ by searching it in $\{0, 1, \cdots, p_i - 1\}$. Similarly, we can determine $m_{i,j}$ for $(j = 2, \cdots, \alpha_i - 1)$ iteratively, and finally recover m_i according to Equation (11).

The above process of computing m_i can be improved. Notice that $\langle b^{\frac{p-1}{p_i} \cdot \ell_i} \rangle_p$ $(\ell_i = 0, \cdots, p_i - 1)$ is independent with m_i, we can pre-compute and store them in a table T_i. Then, after obtaining $\langle y_j^{(p-1)/p_i^{j+1}} \rangle_p$, we can look up it in the table T_i to determine $m_{i,j}$ $(j = 0, 1, \cdots, \alpha_i - 1)$ directly, without further exponentiation calculation. If necessary, we can further employ the hash table technique suggested in [27] to reduce the cost for storing and searching in T_i.

The algorithmic description of solving k-RDL problem can be found in Algorithm 2.

Algorithm 2 Fast Solution of k-RDL Problem

Input:

Prime modulus p, base b, and integer k, where $\mathrm{ord}_p(b) = k$, $k | p - 1$, $k = \prod_{i=1}^{s} p_i^{\alpha_i}$, and p_i are small distinct primes.
Integer $y \in \mathbb{Z}_p$

Output:

$m \in \mathbb{Z}_k$ such that $b^m \equiv y \pmod{p}$

for i from 1 to s **do**
  $\beta_{i,0} \leftarrow 1;$ ▷ Pre-computing, construct T_i
  $\beta_{i,1}, \leftarrow \langle b^{\frac{p-1}{p_i}} \rangle_p;$
  $T_i \leftarrow \{\beta_{i,0}, \beta_{i,1}\};$
  for ℓ_i from 2 to $p_i - 1$ **do**
    $\beta_{i,\ell_i} = \langle \beta_{i,1} \cdot \beta_{i,\ell_i - 1} \rangle_p;$
    $T_i \leftarrow T_i \cup \{\beta_{i,\ell_i}\};$
  end for
end for
for i from 1 to s **do**
  $m_i \leftarrow 0$; $y_0 \leftarrow y$; $e_0 \leftarrow p_i$;
  $P_0 \leftarrow 1$; $P_1 \leftarrow p_i$;
  for j from 0 to $\alpha_i - 1$ **do**
    $t \leftarrow \langle y_j^{(p-1)/P_1} \rangle_p;$
    locate $t = \beta_{i,\ell_i}$ in Table T_i; ▷ Searching in T_i
    $m_{i,j} \leftarrow \ell_i;$
    $P_0 \leftarrow m_{i,j} P_0;$
    $m_i \leftarrow m_i + P_0;$ ▷ Reconstruct m_i
    $y_{j+1} \leftarrow \langle y_j b^{-P_0} \rangle_p;$
    $P_0 \leftarrow P_1;$
    $P_1 \leftarrow P_1 \cdot p_i;$
  end for
end for
$m \leftarrow \mathbf{CRT}(m_1, p_1^{\alpha_1}; \cdots; m_s, p_s^{\alpha_s});$ ▷ Reconstruct m
Output m.

---

4 Security Proofs

In this section, we will give the security analysis of our proposal in two parts. In the first part, we obtain the semantic security under a weak assumption but with restrictions on the prime factors of k. In particular, we prove that our proposal is semantically secure under the k^{th}-PR assumption, while k should satisfy that

$k = 2^\alpha \prod_{i=1}^{s} p_i^{\alpha_i}$, where $\alpha = \max\{\alpha_1, \cdots, \alpha_s\}$, and $p_1 < \cdots < p_s$ are small odd primes. In the second part, we obtain the semantic security with less restriction on k, but under a strong assumption. In particular, we prove that our proposal is semantically secure under the k^{th}-SPR assumption.

4.1 Security under the k^{th}-PR Assumption

Before giving the security analysis, we would like to introduce the k^{th}-power residuosity (k^{th}-PR) assumption and some basic results related to the k^{th}-PR assumption.

Definition 5 **(k^{th}-Power Residuosity (k^{th}-PR) Assumption).** *Let $n = pq$ be the product of two large primes p and q with $k|p-1$ and $k|q-1$. The k^{th}-power residuosity problem in $\mathbb{Z}_n^*$ is to distinguish the following two distributions*

$$\mathcal{D}_0 = \{x : x \xleftarrow{\$} \mathbb{NR}_{n,k}\} \quad \textit{and} \quad \mathcal{D}_1 = \{x : x \xleftarrow{\$} \mathbb{R}_{n,k}\}. \tag{15}$$

The k^{th}-power residuosity assumption posits that, without knowing the factorization of n, the advantage $\mathbf{Adv}_{\mathcal{A}}^{k^{th}-PR}(1^\kappa)$ of any PPT k^{th}-power residuosity distinguisher $\mathcal{A}$ defined as follows is negligible with respect to the system security parameter κ,

$$\mathbf{Adv}_{\mathcal{A}}^{k^{th}-PR}(1^\kappa) = |\Pr[\mathcal{A}(x,k,n) = 1 | x \leftarrow \mathbb{NR}_{n,k}] - \Pr[\mathcal{A}(x,k,n) = 1 | x \leftarrow \mathbb{R}_{n,k}]|$$

where probabilities are taken over all coin tosses.

Theorem 3. *Let $k = 2^\alpha \prod_{i=1}^{s} p_i^{\alpha_i}$ and $k' = 2\prod_{i=1}^{s} p_i$, where $\alpha = \max\{\alpha_1, \cdots, \alpha_s\}$, and p_i's are distinct small odd primes. Then, we have that the k^{th}-PR assumption implies the k'^{th}-PR assumption. More precisely, for any PPT k'^{th}-power residuosity distinguisher $\mathcal{A}$ with advantage ϵ', there exists a PPT k^{th}-power residuosity distinguisher $\mathcal{B}$ with advantage ϵ such that $\epsilon' \le \frac{4p_s^2 \cdot \alpha}{s}\epsilon$.*

The proof of this theorem is not as easy as it may seem. We shall postpone the proof until we have established a few preliminary lemmas. In this subsection, we implicitly contains n, p, q, k in all the parts related to k^{th}-PR assumption, and they remain the same.

Lemma 5. *Let $t = (2 \cdot p_i)^{\alpha_i}$ and $t' = 2 \cdot p_i$ for some small odd prime p_i, then we have that the t^{th}-PR assumption implies the t'^{th}-PR assumption. More precisely, for any PPT t'^{th}-power residuosity distinguisher $\mathcal{A}$ with advantage ϵ', there exists a PPT t^{th}-power residuosity distinguisher $\mathcal{B}$ with advantage ϵ such that $\epsilon' \le \frac{4p_s^2 \cdot \alpha}{s}\epsilon$. In other words, define a series of sets as*

$$D_j^{(i)} = \{(x^{(2p_i)^j})_n : x \in \mathbb{NR}_{n,2p_i}\} \quad (j = 0, \cdots, \alpha_i - 1). \tag{16}$$

We have that if $(2p_i)^{th}$-PR assumption holds, then elements sampled randomly and uniformly from the set $D_j^{(i)}$ are computationally indistinguishable from elements sampled randomly and uniformly from $\mathbb{R}_{n,(2p_i)^{\alpha_i}}$ for all $j \in \{0, 1, \cdots, \alpha_i - 1\}$. That is,

$$D_0^{(i)} \stackrel{c}{\approx} D_1^{(i)} \stackrel{c}{\approx} \cdots \stackrel{c}{\approx} D_{\alpha_i-1}^{(i)} \stackrel{c}{\approx} \mathbb{R}_{n,(2p_i)^{\alpha_i}}. \tag{17}$$

Proof. The proof is given in two steps.

CLAIM 1. If $(2p_i)^{th}$-PR assumption holds, for $j = 1, \cdots, \alpha_i - 1$, no PPT adversary can distinguish elements in $D_j^{(i)}$ from elements in $D_{j-1}^{(i)}$.

Suppose that $\mathcal{A}$ is a PPT distinguisher that can tell apart from the uniform distribution of elements in $D_j^{(i)}$ from that in $D_{j-1}^{(i)}$ with non-negligible advantage ϵ'. Let us construct a PPT $(2p_i)^{th}$-power residue distinguisher $\mathcal{B}$ that can solve the $(2p_i)^{th}$-power residuosity problem with non-negligible advantage ϵ such that $\epsilon' \leq 4p_i^2 \cdot \epsilon$.

$\mathcal{B}$ takes as input $w \in \mathbb{J}_{n,2p_i}$ and aims to tell apart $w \in \mathbb{R}_{n,2p_i}$ or $w \in \mathbb{NR}_{n,2p_i}$. To this end, $\mathcal{B}$ chooses a random element $z \xleftarrow{\$} \mathbb{Z}_n^*$, sets $x = (z^{(2p_i)^j} w^{(2p_i)^{j-1}})_n$ and feeds $\mathcal{A}$ with $(x, (2p_i)^j, n)$. When $\mathcal{A}$ halts, $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs.

Now, let us show that the above reduction is correct.

- Case I: Suppose $w \in \mathbb{R}_{n,2p_i}$, then we have $x \notin D_{j-1}^{(i)}$ and

$$w \equiv w'^{2p_i} \pmod{n} \tag{18}$$

for some $w' \in \mathbb{Z}_n^*$. Consider that the generalized Legendre symbol $\left(\frac{w'}{p}\right)_{2p_i}$ (resp. $\left(\frac{w'}{q}\right)_{2p_i}$) takes at most $2p_i$ different values, and $z \in_R \mathbb{Z}_n^*$ is chosen randomly and uniformly, then the probability to have

$$\left(\frac{zw'}{n}\right)_{2p_i} = 1 \quad \text{and} \quad zw' \notin \mathbb{R}_{n,2p_i} \tag{19}$$

is at least $\frac{1}{4p_i^2}$ according to property (2) of Lemma 1. If the condition (19) holds, the resultant elements $zw' \bmod n$ are statistically indistinguishable from random elements of $\mathbb{NR}_{n,2p_i}$. With Equation (18), we can further have $x \equiv (zw')^{(2p_i)^j} \pmod{n}$. Thus, $x \in D_j^{(i)}$. Recall that the condition (19) is equivalent to

$$\left(\frac{zw'}{p}\right)_{2p_i} = \left(\frac{zw'}{q}\right)_{2p_i} = -1. \tag{20}$$

We can deduce that $x \notin D_{j+1}^{(i)}$. If not, we have a representation $x \equiv y^{(2p_i)^{j+1}} \pmod{n}$ for some $y \in \mathbb{NR}_{n,2p_i}$, which implies $\left(\frac{x}{p}\right)_{(2p_i)^{j+1}} = 1$. But this is impossible, since from Equation (20) we have that

$$\left(\frac{x}{p}\right)_{(2p_i)^{j+1}} = \left(\frac{(zw')^{(2p_i)^j}}{p}\right)_{(2p_i)^{j+1}} = \left((zw')^{(2p_i)^j}\right)^{\frac{p-1}{(2p_i)^{j+1}}} = \left(\frac{zw'}{p}\right)_{2p_i} = -1. \tag{21}$$

Consequently, $x \equiv (zw')^{(2p_i)^j} \pmod{n}$ is uniformly distributed in $D_j^{(i)}$ with probability at least $\frac{1}{4p_i^2}$.

- Case II: Suppose $w \in \mathbb{NR}_{n,2p_i}$, then we clearly have $x \in D_{j-1}^{(i)}$ because $x \equiv (z^{2p_i} w)^{(2p_i)^{j-1}} \pmod{n}$ and $z^{2p_i} w \in \mathbb{NR}_{n,2p_i}$. □

CLAIM 2. If $(2p_i)^{th}$-PR assumption holds, no PPT adversary can distinguish the uniform distribution of element in $D_{\alpha_i-1}^{(i)}$ from that in $\mathbb{R}_{n,(2p_i)^{\alpha_i}}$.

Let $\mathcal{A}$ be an algorithm that can distinguish random elements in $D_{\alpha_i-1}^{(i)}$ from random elements in $\mathbb{R}_{n,(2p_i)^{\alpha_i}}$ with non-negligible advantage ϵ'. Let us build a $(2p_i)^{th}$-power residuosity distinguisher $\mathcal{B}$ with the same non-negligible advantage $\epsilon = \epsilon'$.

$\mathcal{B}$ takes as input an element $w \in \mathbb{J}_{n,2p_i}$ with the goal of deciding whether $w \in \mathbb{R}_{n,2p_i}$ or $w \in \mathbb{NR}_{n,2p_i}$. To do this, $\mathcal{B}$ simply defines $x = (w^{(2p_i)^{\alpha_i-1}})_n$ and runs $\mathcal{A}$ on input of $(x, (2p_i)^{\alpha_i}, n)$. When $\mathcal{A}$ halts, $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs.

It is easy to see that this reduction is correct: If $w \in \mathbb{R}_{n,2p_i}$, then $x \in \mathbb{R}_{n,(2p_i)^{\alpha_i}}$. If $w \in \mathbb{NR}_{n,2p_i}$, we immediately have $x \in D^{(i)}_{\alpha_i-1}$. □

Combining CLAIM 1 and CLAIM 2, we can get the expected reduction chain (17). Note that the loss factor of the advantages in each reduction step is at most $\frac{1}{4p_i^2}$, and all reductions are directly based on the $(2p_i)^{th}$-PR assumption. For achieving the indistinguishability result of $D^{(i)}_j \overset{c}{\approx} R_{n,(2p_i)^{\alpha_i}}$ for $j = 0, \cdots, \alpha_i - 1$, we need at most α_i steps. Thus, the total loss factor of the advantages is at most $\frac{1}{4p_i^2 \cdot \alpha_i}$.

This concludes the Lemma. ■

Lemma 6. *For a prime p and two positive integers k_1, k_2 with $k_1|p-1, k_2|p-1$. If $x \in \mathbb{R}_{p,k_1}$ and $x \in \mathbb{R}_{p,k_2}$, then $x \in \mathbb{R}_{p,[k_1,k_2]}$, where $[k_1, k_2]$ denotes the least common multiple of k_1 and k_2.*

Proof. $x \in \mathbb{R}_{p,k_1}$ and $x \in \mathbb{R}_{p,k_2}$ imply that $x \equiv r_i^{k_1} \equiv r_2^{k_2} \pmod p$ for some k_1, k_2. Then, for p's any primitive root g, we have

$$r_1 \equiv g^a \pmod p \quad \text{and} \quad r_2 \equiv g^b \pmod p$$

for some a, b. Thus,

$$x \equiv g^{ak_1} \equiv g^{bk_2} \pmod p.$$

This suggests $ak_1 \equiv bk_2 \pmod{p-1}$. Then, we have

$$a\frac{k_1}{(k_1,k_2)} \equiv \frac{k_2}{(k_1,k_2)} b \ \left(\text{mod } \frac{p-1}{(k_1,k_2)}\right).$$

and $\frac{k_1}{(k_1,k_2)}|b$. Thus, we get

$$r_2 \equiv \left(g^{b/\frac{k_1}{(k_1,k_2)}}\right)^{\frac{k_1}{(k_1,k_2)}} \pmod p$$

and

$$x \equiv \left(g^{b/\frac{k_1}{(k_1,k_2)}}\right)^{\frac{k_1 k_2}{(k_1,k_2)}} \equiv \left(g^{b/\frac{k_1}{(k_1,k_2)}}\right)^{[k_1,k_2]} \pmod p.$$

That is, $x \in \mathbb{R}_{p,[k_1,k_2]}$. □

We are now in a position to prove Theorem 3.

Proof. Firstly, we set $t = \prod_{i=1}^{s}(2p_i)^{\alpha_i}$ for some distinct primes p_i's and positive integers α_i's ($i = 1, \cdots, s$), and define $\mathbb{I}$ as the following Cartesian product

$$\mathbb{I} = \{0, 1, \cdots, \alpha_1\} \times \cdots \times \{0, 1, \cdots, \alpha_s\}.$$

Then, for any given vector $\boldsymbol{a} = (a_1, \cdots, a_s) \in \mathbb{I}$, let us define a series of Cartesian product sets

$$\mathbb{D}_{\boldsymbol{a}} = D^{(1)}_{a_1} \times \cdots \times D^{(s)}_{a_s}, \tag{22}$$

where $D^{(i)}_{\alpha_i} = \mathbb{R}_{n,(2p_i)^{\alpha_i}}$ $(i = 1, \cdots, s)$, while $D^{(i)}_{a_i}$ for $a_i < \alpha_i$ $(i = 1, \cdots, s)$ is defined in (16).

The basic idea to prove Theorem 3 is that if k'^{th}-PR assumption holds, then for any given two vectors $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{I}$, the elements sampled randomly and uniformly from $\mathbb{D}_{\boldsymbol{a}}$ are computationally indistinguishable from that from $\mathbb{D}_{\boldsymbol{b}}$. That is,

$$\mathbb{D}_{\boldsymbol{a}} \stackrel{c}{\approx} \mathbb{D}_{\boldsymbol{b}}. \tag{23}$$

More precisely, if there exists a PPT distinguisher $\mathcal{A}$ that can tell apart elements randomly chosen from $\mathbb{D}_{\boldsymbol{a}}$ from that from $\mathbb{D}_{\boldsymbol{b}}$ with advantage ϵ', then there exists a k^{th}-power residue distinguisher $\mathcal{B}$ that can decide whether a given random element is in $\mathbb{NR}_{n,k}$ or not with advantage $\epsilon \geq \frac{s}{4p_s^2\alpha}\epsilon'$, where $\alpha = \max\{\alpha_1, \cdots, \alpha_s\}$.

At first, let us consider the case of $s = 2$, i.e., $\boldsymbol{a} = (a_1, a_2)$ and $\boldsymbol{b} = (b_1, b_2)$. We only need to discuss the situation when $a_1 \neq b_1$ and $a_2 \neq b_2$ (otherwise the conclusion is hold apparently according to Lemma 5.

A successful distinguisher that can tell apart uniform distributions over $D_{a_1}^{(1)} \times D_{a_2}^{(2)}$ and $D_{b_1}^{(1)} \times D_{b_2}^{(2)}$ means that it cannot only tell apart uniform distributions over $D_{a_1}^{(1)}$ and $D_{b_1}^{(1)}$, but also tell apart uniform distributions over $D_{a_2}^{(2)}$ and $D_{b_2}^{(2)}$. More precisely, if there exists a PPT distinguisher $\mathcal{A}$ that can tell apart uniform distributions over $D_{a_1}^{(1)} \times D_{a_2}^{(2)}$ and $D_{b_1}^{(1)} \times D_{b_2}^{(2)}$ with non-negligible advantage ϵ', then both $\mathcal{A}$'s advantages for telling apart uniform distributions over $D_{a_1}^{(1)}$ and $D_{b_1}^{(1)}$, and $D_{a_2}^{(2)}$ and $D_{b_2}^{(2)}$ are no less than ϵ'. However, according to the proof of Lemma 5, such $\mathcal{A}$ means a $2p_1$-th residue distinguisher $\mathcal{B}_1$ with advantage

$$\epsilon_1 \geq \frac{1}{4p_1^2 \cdot |a_1 - b_1|}\epsilon',$$

and a $2p_2$-th residue distinguisher $\mathcal{B}_2$ with advantage

$$\epsilon_2 \geq \frac{1}{4p_2^2 \cdot |a_2 - b_2|}\epsilon'.$$

With access to either $\mathcal{B}_1$ or $\mathcal{B}_2$, we can easily decide whether a given element $x \in \mathbb{Z}_n^*$ is belong to $\mathbb{NR}_{n,2p_1p_2}$ or not. Thus, we obtain a $(2p_1p_2)$-th residue distinguisher $\mathcal{B}$ with advantage

$$\epsilon = \epsilon_1 + \epsilon_2 \geq \left(\frac{1}{4p_1^2|a_1 - b_1|} + \frac{1}{4p_2^2|a_2 - b_2|}\right)\epsilon'. \tag{24}$$

By a very similar reduction, we have that for $s > 2$, if there exists a PPT distinguisher $\mathcal{A}$ that can tell apart uniform distributions over $D_{a_1}^{(1)} \times \cdots \times D_{a_s}^{(s)}$ and $D_{b_1}^{(1)} \times \cdots \times D_{b_s}^{(s)}$ with non-negligible advantage ϵ, then we can obtain a k'^{th} residue distinguisher $\mathcal{B}$, which can be used to decide whether a given element $x \in \mathbb{Z}_n^*$ is belong to $\mathbb{NR}_{n,k}$ or not, with advantage

$$\epsilon = \epsilon_1 + \cdots + \epsilon_s \geq \left(\frac{1}{4p_1^2|a_1 - b_1|} + \cdots + \frac{1}{4p_s^2|a_s - b_s|}\right)\epsilon'. \tag{25}$$

In particular, when $\boldsymbol{a} = (0, \cdots, 0)$ and $\boldsymbol{b} = (\alpha_1, \cdots, \alpha_s)$, we have that

$$\mathbb{D}_{\boldsymbol{a}} \stackrel{c}{\approx} \mathbb{D}_{\boldsymbol{b}} = \mathbb{R}_{n,(2p_1)^{\alpha_1}} \times \cdots \times \mathbb{R}_{n,(2p_s)^{\alpha_s}}$$

and Equation (25) becomes

$$\epsilon = \epsilon_1 + \cdots + \epsilon_s \geq \left(\frac{1}{4p_1^2\alpha_1} + \cdots + \frac{1}{4p_s^2\alpha_s}\right)\epsilon' \geq \frac{s}{4p_s^2\alpha}\epsilon'. \tag{26}$$

Further, according to Lemma 5, Lemma 6 and

$$[(2p_1)^{\alpha_1}, \cdots, (2p_s)^{\alpha_s}] = 2^{\alpha} p_1^{\alpha_1} \cdots p_s^{\alpha_s} = k$$

with $\alpha = \max\{\alpha_1, \cdots, \alpha_s\}$, if there is a PPT distinguisher that can tell apart $\mathbb{D}_{\boldsymbol{a}}$ and $\mathbb{D}_{\boldsymbol{b}}$, we can easily decide whether a given element $x \in \mathbb{Z}_n^*$ is belong to $\mathbb{NR}_{n,k}$ or not.

This finishes the proof. □

Now, we can give the security proof of our proposal based on the k^{th}-PR assumption.

Theorem 4. *Let $k = 2^{\alpha} \prod_{i=1}^{s} p_i^{\alpha_i}$ and $k' = 2 \prod_{i=1}^{s} p_i$, where $\alpha = \max\{\alpha_1, \cdots, \alpha_s\}$, and p_i's are distinct small odd primes. If $a \in \mathbb{NR}_{n,k'}$, then our proposal is semantically secure under the k^{th}-PR assumption. More precisely, for any PPT IND-CPA adversary $\mathcal{A}$, we have a distinguisher $\mathcal{B}$ such that*

$$Adv_{\mathcal{A}}^{ind-cpa}(1^{\kappa}) \leq \frac{4p_s^2 \cdot \alpha}{s} \cdot Adv_{\mathcal{B}}^{k^{th}-PR}(1^{\kappa}).$$

Proof. The basic idea is very similar to the proof of Theorem 2 of [23] in which the so-called Gap-2^{α}-Res assumption was used. Let us define two games G_0 and G_1 as follows: In G_0, the challenger $\mathcal{C}$ sends the real public key (n, k, a) to the adversary $\mathcal{A}$, while in G_1, $\mathcal{C}$ at first picks $z \in \mathbb{Z}_n^*$ at random and sets $a' = \langle x^{k'} \rangle_n$, then sends $\mathcal{A}$ the forged public key (n, k, a'). Apparently, in G_1, $a' \in \mathbb{R}_{n,k'}$. Thus, the ciphertext carries no information about the message and $\mathcal{A}$ has no advantage in Game G_1. The left thing is to prove that these two games are indistinguishable in $\mathcal{A}$'s view. The only difference between G_0 and G_1 is that $a \in \mathbb{NR}_{n,k'}$ in G_0, while $a \in \mathbb{R}_{n,k'}$ in G_1. Therefore, under the k^{th} residuosity assumption, $\mathcal{A}$ has no non-negligible advantage to distinguish them. □

4.2 Security under the k^{th}-SPR Assumption

Definition 6 (Strong k^{th} Power Residuosity (k^{th}-SPR) Assumption). *Let $n = pq$ be the product of two large primes p and q with $k|p-1$ and $k|q-1$. The strong k^{th}-power residuosity problem in $\mathbb{Z}_n^*$ is to decide whether a given random element $a \in \mathbb{Z}_n^*$ is a k^{th}-power residue or not. The strong k^{th}-power residuosity assumption posits that, without knowing the factorization of n, the advantage $\mathbf{Adv}_{\mathcal{A}}^{k^{th}-SPR}(1^{\kappa})$ of any PPT strong k^{th}-power residuosity distinguisher $\mathcal{A}$, defined as the follows, is negligible with respect to the system security parameter κ,*

$$\mathbf{Adv}_{\mathcal{A}}^{k^{th}-SPR}(1^{\kappa}) = |\Pr[\mathcal{A}(x, k, n) = 1 | x \leftarrow \mathbb{Z}_N^*] - \Pr[\mathcal{A}(x, k, n) = 1 | x \leftarrow \mathbb{R}_{n,k}]|$$

where probabilities are taken over all coin tosses.

Theorem 5. *Our proposal is semantically secure under the strong k^{th}-power residuosity assumption.*

Proof. Under the strong k^{th}-power residuosity assumption, our scheme is still provably secure even without the condition $a \in \mathbb{NR}_{n,k'}$. Consider that in Game G_0, $\mathrm{ord}_p(a) = k$ and $(k, (p-1)/k) = 1$, then $(a)_p \notin R_{p,k}$. Thus, $a \notin \mathbb{R}_{n,k}$. The remained reduction is identical to what was given in the proof of Theorem 4, except that in Game G_1, the assignment $a' = \langle x^{k'} \rangle_n$ should be replaced with $a' = \langle x^k \rangle_n$. □

Remark 3. The confidence of using the strong k^{th}-power residue assumption lies in the result given by Adleman and McDonnell [1]. It shows that if there exists an oracle that can determine whether or not a random $z < n$ is a k^{th}-power residue (modulo n), then we can build an efficient (although not quite polynomial time) algorithm to factor n [14].

5 Implementation Issues, Performance and Applications

5.1 Provable Security under the k^{th}-PR Assumption

Careful reader may notice that there exists a small gap between the proof of Theorem 4 and real setting of our proposal. That is, the part of public key a does not always satisfy $a \in \mathbb{NR}_{n,k'}$. In this section, we fill the gap by introducing a probabilistic generation of such a.

Recall that $k = 2^\alpha p_1^{\alpha_1} \cdots p_s^{\alpha_s}$, $k' = 2p_1 \cdots p_s$, and $p - 1 = kp'$. We select $h_p \in \mathbb{Z}_p^*$ at random, and set $r = \langle h_p^{k' \cdot p'} \rangle_p$. Clearly, we have that $\left(\frac{r}{p}\right)_{k'} = 1$. If $p \equiv 3 \pmod 4$, we then set $r_p \leftarrow \langle -r \rangle_p$. In this case, $\left(\frac{r_p}{p}\right)_{k'} = \left(\frac{-r}{p}\right)_{k'} = -1$. If $p \equiv 1 \pmod 4$, then $\alpha > 1$. We need further elaboration. Let $\zeta = e^{2i\pi/2^\alpha}$ be the primitive 2^α-th root of unity, where $i = \sqrt{-1}$, we then set $r_p = \langle \zeta \cdot r \rangle_p$. Now, we have that

$$\left(\frac{r_p}{p}\right)_{k'} \equiv \left(\frac{\zeta}{p}\right)_{k'} \equiv \zeta^{\frac{p-1}{k'}} \equiv e^{\frac{2i\pi}{2^\alpha} \cdot \frac{p-1}{2p_1 \cdots p_s}} \equiv e^{i\pi \cdot \frac{p-1}{2^\alpha p_1 \cdots p_s}} \equiv -1 \pmod p.$$

Similarly, we can select $h_q \in \mathbb{Z}_q^*$ at random and get r_q such that $\left(\frac{r_q}{q}\right)_{k'} = -1$. By using CRT, we have that

$$a = r_p + p\langle p^{-1}(r_q - r_p)\rangle_q$$

as what we need. In fact, since $a \equiv r_p \pmod p$ and $a \equiv r_q \pmod q$, we have that

$$\left(\frac{a}{n}\right)_{k'} = \left(\frac{r_p}{p}\right)_{k'} \cdot \left(\frac{r_q}{q}\right)_{k'} = (-1) \cdot (-1) = 1.$$

That is, $a \in \mathbb{NR}_{n,k'}$.

Next, let us analyze the order of $b = \left(\frac{a}{p}\right)_k$. It is easy to check that $a^{2k/k'} \equiv 1 \pmod p$ holds in both cases. Thus, we have that $l = \mathrm{ord}_p(b) | \mathrm{ord}_p(a) | 2k/k'$. Based on our test, we find that when h_p and h_q are selected at random, l reaches the up bound $2k/k'$ with high probability. Note that due to the decrease of l, the message space is reduced from $[0, k)$ to $[0, l)$.

5.2 Performance analysis and comparisons

In this section, we only count the time cost in algorithms `Enc` and `Dec`, but not that in algorithm `KeyGen`, since it is run only once.

The workload of algorithm `Enc` in our proposal is the same as the Joye-Libert cryptosystem, i.e., it is mainly occupied by two modular exponentiations. However, the time cost in algorithm `Dec` has a subtle difference. `Dec` in the both cryptosystems is quite efficient due to efficient solutions of k-residue discrete logarithm that is performed by a "bit-by-bit" manner. The "bit-by-bit" manner is actually performed in a binary manner in the Joye-Libert cryptosystem, while it is performed in a p_i-adic manner in our proposal, where p_i could be larger than 2. Theoretically, for message space $\mathbb{Z}_{2^\alpha}$, the decryption time in the Joye-Libert cryptosystem is proportional to α. That is, its time complexity in algorithm `Dec` is $\mathcal{O}(\alpha \cdot T_e(n))$, where $T_e(n) = \mathcal{O}((\log n)^2(\log\log n))$ is the time complexity for performing modular exponentiations. While it is proportional to $\alpha' = \sum \alpha_i$ in our proposal. Note that the time cost on searching in Table T_i in our proposal can be made to constant, say by using the hash table technique as used in [27]. Therefore, the time complexity in algorithm Dec in our proposal can be approximately reduced to

$$T_{dec}(n) = \mathcal{O}\left(T_e(n) \cdot \sum_{i=1}^{s} \alpha_i\right).$$

The above analysis also says that for the same size message space, as long as $\alpha' < \alpha$, our scheme could outperform the Joye-Libert cryptosystem in algorithm Dec. This is manifested by our testing on different choice on k. Our testing environment and common settings are given below.

- CPU: Intel(R) Core(TM)2 Duo E6550@2.33GHz.
- RAM: 3GB.
- OS: Windows 7, Service Pack 1.
- Supporting Library: MIRCAL Version 5.4, produced by Shamus Software Corp.
- p', q': Large numbers contain a big prime no less than 600 bits.
- Size of message space: Roughly equal to 2^{128}.
- n's in all tests are of similar bit-length.

When $k = 2^{128}$ (i.e., $\alpha = 128$), our proposal is instantiated to the Joye-Libert cryptosystem, the average decryption time, over 100 trials, is about 400ms. When $k = 3^{81}$, the average decryption time is reduced to about 267ms. When $k = 5^{56}$ and $k = 7^{46}$, the decryption time is further reduced to about 188ms and 155ms, respectively. We performed similar tests on all small primes less than 1000, while keeping the message space size similar with but not smaller than 2^{128}. The results are illustrated in Figure 1, and some results with typical settings on k are given in Table 2. From Figure 1, we can see that when small primes, say $p_i < 100$, are used, the average decryption time decreases dramatically along the increase of used primes. It follows the fact that once k is fixed, when p_i increases, the exponent $\alpha_i = \lceil \log_{p_i} k \rceil$ decreases apparently, and the decryption cost is mainly determined by α_i, instead of the value of p_i. This conclusion is further manifested by Figure 2, where the average decryption time is plotted w.r.t. each setting on exponent value. Our test also suggests that this kind of speed-up ratio vanishes when p_i becomes too large.

Table 2. Decryption Speed (s)

k	Aver. Dec. Time (s)
2^{128}	0.404940
3^{81}	0.266570
5^{56}	0.187670
7^{46}	0.154760
11^{38}	0.129200
13^{35}	0.118080
17^{32}	0.109210
19^{31}	0.105460
97^{20}	0.068800
257^{16}	0.055850
571^{14}	0.050070
929^{13}	0.048670

5.3 Enlarge Message Space by Using Chinese Remainder Theorem

In our original scheme V_0, the message space is $\mathbb{Z}_k$, while the ciphertext space is $\mathbb{Z}_n^*$. Thus, if we assume that the bit-lengths of k, p' and q' are roughly equal, i.e. $\log k \approx \log p' \approx \log q'$, then the ciphertext expansion factor is

$$\rho_{V_0} = \frac{\log n}{\log k} \approx \frac{\log \phi(n)}{\log k} = \frac{2\log k + \log p' + \log q'}{\log k} \approx 4.$$

Now, if we set $p = k_1 p' + 1, q = k_2 q' + 1$ such that

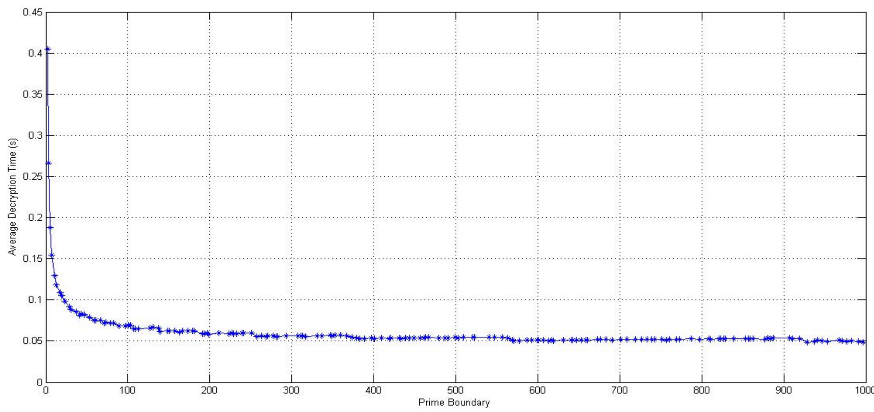

Fig. 1. Decryption Speed

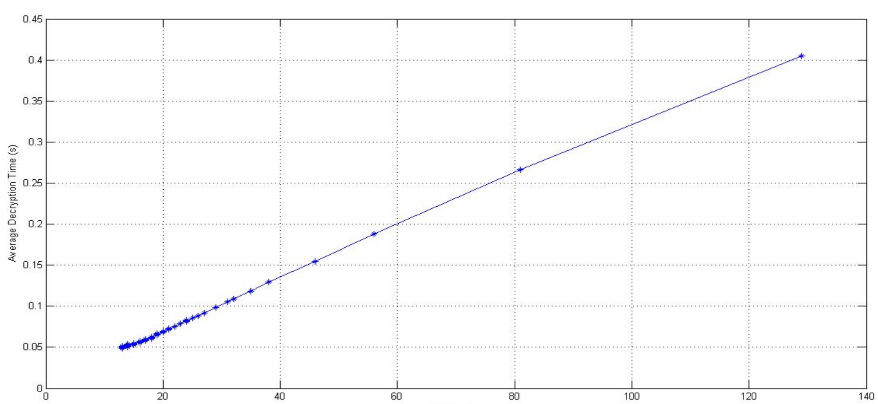

Fig. 2. Decryption Time vs. Exponent Value

- $(k_1, k_2) = (k_1, p') = (k_2, q') = 1$.
- p', q' both contain large prime factors.
- k_1, k_2 both only contain small odd prime factors.
- $\mathrm{ord}_p(a) = k_1$ and $\mathrm{ord}_q(a) = k_2$ (See Section 3.2 for details on how to choose proper a).

Then, for a given L such that $L < k$, $\log L \approx \log k$, and $k = k_1 k_2$, the public key is (n, a, L) and the private key is (p, q, k_1, k_2). Note that without publishing k, the encryption process has to be changed accordingly. Given message $m \in \mathbb{Z}_L$, the ciphertext is $c = \langle a^m \rangle_n$. Upon receiving a ciphertext $c \in \mathbb{Z}_n^*$, the decryptor knowing (p, q, k_1, k_2) can recover the message $m \in \mathbb{Z}_L$ as follows.

1. Let $a_1 = \langle a \rangle_p$, $a_2 = \langle a \rangle_q$, $y_p = \langle c \rangle_p$ and $y_q = \langle c \rangle_q$.
2. Compute $m_1 = RDL_{a_1,p}^{k_1}(y_p)$ and $m_2 = RDL_{a_2,q}^{k_2}(y_q)$ by using the method in Section 3.3.
3. Reconstruct the message $m \in \mathbb{Z}_L$ from
$$m \equiv m_1 \pmod{k_1} \quad \text{and} \quad m \equiv m_2 \pmod{k_2},$$
by using CRT.

By doing so, we obtain an improved scheme, denoted by V_1, that achieves better ciphertext expansion factor

$$\rho_{V_1} = \frac{\log n}{\log L} \approx \frac{\log \phi(n)}{\log k} = \frac{\log k_1 + \log k_2 + \log p' + \log q'}{\log k_1 + \log k_2} \approx 2$$

under the reasonable assumption that the lengthes of k_1, k_2, p' and q' are roughly equal.

Note that in V_1, it is necessary to keep k_1, k_2 and their product k secret; otherwise, n can be factorized. On the other hand, V_1 becomes a deterministic encryption, and it can be viewed as an extension of the basic scheme in [27]. The advantage of V_1 lies in that the message space is enlarged from $[0, \prod_{i=1}^{s} p_i)$ to $[0, \prod_{i=1}^{s} p_i^{\alpha_i})$ without increasing the searching space.

Further, it is easy to extend V_1 to be a probabilistic version, denoted by V_1', which consists of the following three algorithms:

`KeyGen`: On inputting the security parameter κ, `KeyGen` outputs the public key $pk = \{n, a, L, g\}$ and private key $sk = \{p, q, k_1, k_2, k_3\}$, where
- $n = p \cdot q$, $p = k_1 p' + 1$, $q = k_2 q' + 1$.
- p, q both are big primes.
- p', q' both contain large prime factors.
- k_1, k_2 both only contain small odd prime factors.
- $k_3 > 1$ is a positive integer such that $(k_3, k_1 k_2) = 1$.
- $(k_1, k_2) = (k_1, p') = (k_2, q') = 1$.
- $L < k$ and $\log L \approx \log k$ (where $k = k_1 k_2$).
- $\mathrm{ord}_p(a) = k_1$, $\mathrm{ord}_q(a) = k_2$ and $\mathrm{ord}_p(g) = \mathrm{ord}_q(g) = k_3$ (See Section 3.2 for details on how to choose proper a and g).

`Enc`: On inputting a message $m \in \mathbb{Z}_L$, the encryptor selects $x \in \mathbb{Z}_n^*$ at random and outputs the ciphertext
$$c = \langle a^m g^x \rangle_n. \tag{27}$$

`Dec`: On inputting a ciphertext $c \in \mathbb{Z}_n^*$, the decryptor the decryptor knowing (p, q, k_1, k_2, k_3) can recover the message $m \in \mathbb{Z}_L$ as follows.

1. Let $a_1 = \langle a \rangle_p$, $a_2 = \langle a \rangle_q$, $c_p = \langle c^{k_3} \rangle_p$ and $c_q = \langle c^{k_3} \rangle_q$.
2. Let $y_p = \langle c_p^{1/k_3 \bmod k_1} \rangle_p$ and $y_q = \langle c_q^{1/k_3 \bmod k_2} \rangle_q$.
3. Compute $m_1 = RDL_{a_1,p}^{k_1}(y_p)$ and $m_2 = RDL_{a_2,q}^{k_2}(y_q)$ by using the method in Section 3.3.
4. Reconstruct the message $m \in \mathbb{Z}_L$ from

$$m \equiv m_1 \pmod{k_1} \quad \text{and} \quad m \equiv m_2 \pmod{k_2},$$

by using CRT.

The security analysis of scheme V_1' can be obtained similar with that of the probabilistic version in [27]. Apparently, the ciphertext expansion factor of V_1' is the same as that of scheme V_1. In sequel, when scheme V_1 is mentioned, it includes both V_1 and V_1' if without further specifications.

5.4 Lossy Trapdoor Functions From k^{th} Residues

Lossy Trapdoor Functions (LTDFs), introduced by Peikert and Waters at STOC'08 [29], have been proven to be an extremely useful and versatile cryptographic primitive [21]. Many research efforts have recently been dedicated to design efficient LTDFs based on different cryptographic assumptions [21,19]. Recently, Joye and Libert [23] proposed a quite efficient LTDF with short outputs and keys, and large lossiness based on their GM-type cryptosystem. By using the same framework of Joye-Libert, our proposal can also yield an efficient LTDF with short outputs and keys, and large lossiness.

Before giving our construction on LTDFs, We briefly recall the definition of lossy trapdoor functions given in [29,21]. In formally, a collection of lossy trapdoor functions consists of two families of functions. Functions in one family are injective and can be efficiently inverted using a trapdoor. Functions in the other family are "lossy," which means that the size of their image is significantly smaller than the size of their domain. The only computational requirement is that a description of a randomly chosen function from the family of injective functions is computationally indistinguishable from a description of a randomly chosen function from the family of lossy functions [19].

Formally, for given security parameter κ, a collection of (μ, ν)-lossy trapdoor functions is a 4-tuple of PPT algorithms `InjGen`, `LossyGen`, `Evaluation` and `Inversion` such that [23,21]:

- `InjGen` outputs a pair $(\sigma, t) \in \{0,1\}^* \times \{0,1\}^*$ where σ is an index of an injective function f, while t is f's trapdoor.
- `LossyGen` outputs $\sigma \in \{0,1\}^*$ as an index of an lossy function f.
- `Evaluation` takes as input an function index $\sigma \in \{0,1\}^*$ and an input $x \in \{0,1\}^\mu$, outputs the value $f_\sigma(x)$ such that
  - if σ is an output of `InjGen`, then $f_\sigma(\cdot)$ is an injective function.
  - if σ is an output of `LossyGen`, then $f_\sigma(\cdot)$ has image size $2^{\mu-\nu}$.
- `Inversion` takes as input an image $f_\sigma(x)$ and the corresponding trapdoor t, outputs x.
- The two ensembles $\{\sigma | (\sigma, t) \leftarrow \texttt{InjGen}(1^\kappa)\}$ and $\{\sigma | \sigma \leftarrow \texttt{LossyGen}(1^\kappa)\}$ are computationally indistinguishable.

Now, our construction goes as follows. Without loss of generality, let us assume that the desired input using k-adic encoding and the length is no more than ℓ (i.e., $\mu = \ell \log k$).

- `InjGen`(1^κ): To sample an injective function, algorithm `InjGen` performs the following steps:
  1. Generate a modulus $n = pq$ so that $p = kp' + 1$ and $q = kq' + 1$, where both p' and q' contain large prime factors, while k is a product of small primes. Choose a such that $\text{ord}_p(a) = \text{ord}_q(a) = k$.

2. For each $i \in \{1, \cdots, \ell\}$, pick h_i in the subgroup of order $p'q'$, by setting $h_i = g_i^k \bmod n$ for a randomly chosen $g_i \in \mathbb{Z}_n^*$.
3. Choose $r_1, \cdots, r_\ell \xleftarrow{R} \mathbb{Z}_{p'q'}$ and compute a matrix $Z = (z_{i,j})_{i,j \in \{1,\cdots,\ell\}}$ with

$$z_{i,j} = \begin{cases} a h_j^{r_i} \bmod n, \text{ if } i = j \\ h_j^{r_i} \bmod n, \quad \text{otherwise.} \end{cases}$$

4. Output the function index $\sigma = (n, Z)$ and the trapdoor p.

– `LossyGen`(1^κ): The process of `LossyGen` is identical to the process of `InjGen`, except that
  1. the matrix entry $z_{i,j} = h_j^{r_i} \bmod n$ for $\forall i, j \in \{1, \cdots, \ell\}$.
  2. without outputting p.
– `Evaluation`: Given $\sigma = (n, Z = (z_{i,j})_{i,j \in \{1,\cdots,\ell\}})$, the algorithm `Evaluation` encodes the input x as k-adic string $\tilde{x} = x_1 \cdots x_\ell$ with $x_i \in \mathbb{Z}_k$ for each i. Then, it computes and returns $\tilde{y} = (y_1, \cdots, y_\ell)$ with $y_j = \prod_{i=1}^{\ell} z_{i,j}^{x_i} \bmod n \in \mathbb{Z}_n^*$.
– `Inversion`: Given trapdoor p and $\tilde{y} = (y_1, \cdots, y_\ell) \in \mathbb{Z}_n^\ell$, one can apply the decryption algorithm for each y_j to recover $x_j \in \mathbb{Z}_k$ $(j = 1, \cdots, \ell)$ and then reconstruct $x = \sum_{j=1}^{\ell} x_j k^{j-1}$, since when Z is given by algorithm `InjGen`, we have that $\left(\frac{y_j}{p}\right)_k \equiv \left(\frac{a}{p}\right)_k^{x_j} \pmod p$ holds.

It is easy to see that the above LTDF construction is quite similar with that in [23], which leads to the similar security analysis. Hence, we omit the security analysis, but just give analysis on the lossiness.

In our LTDF, the input space is $\mathbb{Z}_k^\ell$, but the output is entirely determined by $\sum_{i=1}^{\ell} r_i x_i \bmod p'q'$, where $r_1, \cdots, r_\ell \in \mathbb{Z}_{p'q'}$ are selected at random, while $x_1, \cdots, x_\ell \in \mathbb{Z}_k$ are specified by inputs. Thus, the image size is at most $p'q'$, and the residual leakage is at most $\log(p'q')$ bits. That is, we lose $\nu_{(2)} = \ell \log k - \log(p'q') = \log \frac{k^{\ell+2}}{\phi(n)}$ bits. This suggests that the lossiness would increase if k or ℓ increases. In the k-adic context, our LTDF loses exactly $\nu_{(k)} = \ell - \log_k(p'q')$ k-adic "bits". Usually, we set $\log k \approx \log p' \approx \log q'$ for a large message space. Then, we have that $\nu_{(k)} = \ell - 2$. This means that our LTDF keeps *only two* k-adic "bits", and loses all other k-adic "bits". We can also have that $\nu_{(2)} = (\ell - 2)\alpha$ for $\alpha = \log k$. Further, when $\ell \geq 4$, we get $\nu_{(2)} \geq 2\alpha$. This lossiness is very parallel to what was obtained in [23].

One interesting question is that the above LTDF is based on scheme V_0, how about the lossiness when the LTDF is based on Scheme V_1? As usual, we assume that $\log k_1 \approx \log p' \approx \log k_2 \approx \log q'$, then the lossiness could be improved to $\nu_{(2)} = (\ell - 1)\alpha$ for $\alpha = \log k$ and $\nu_{(k)} = \ell - 1$. This result says that by using Scheme V_1, we can obtain an LTDF that keeps *only one* k-adic "bit" and loses all other k-adic "bits". This shows that by using the same building framework, the smaller the ciphertext expansion factor of the GM-type cryptosystem, the better the lossiness of the resulting LTDF. Recall the ciphertext expansion factor of the Joye-Libert cryptosystem, it is

$$\rho_{JL} = \frac{\log n}{\alpha} > \frac{\log n}{\frac{1}{4}\log n - \kappa} > 4 \approx \rho_{V_0} \approx 2\rho_{V_1}.$$

As a result, the LTDF based on our scheme V_1 is better than that based on the Joye-Libert cryptosystem in terms of lossiness.

6 Conclusions

It never overestimates the importance of the GM cryptosystem. It enriches modern cryptography on many aspects. On one hand, it plays a major role in the development of modern cryptography by introducing the concept of "semantic security" and "indistinguishability". On the other hand, it is the first additive homomorphic encryption scheme. Many subsequent improvements focus on how to reduce the bandwidth, ciphertext expansion factor, and encryption/decryption cost, while keeping the property of additive homomorphism. By using $(2^{\alpha})^{th}$-power residues, the Joye-Libert cryptosystem, a non-trivial extension of the GM cryptosystem, is not only efficient in bandwidth and encryption/decryption speed, but also supports additive homomorphism over an exponential-scale message space. In this paper, we have made further extension on the GM cryptosystem by using k^{th}-power residues where k is merely required to be a product of powers of small primes. The proposed schemes (V_0 and V_1) do not only inherit all advantages from the Joye-Libert cryptosystem, but also enhance the decryption speed observably. Moreover, scheme V_1 achieves lower ciphertext expansion factor and results in a lossy trapdoor functions with better lossiness compared to the Joye-Libert constructions.

References

1. Leonard M. Adleman and Robert McDonnell. An application of higher reciprocity to computational number theory (abstract). In *FOCS 1982*, pages 100–106. IEEE Computer Society Press, 1982.
2. Miklos Ajtai. Generating hard instances of lattice problems (extended abstract). In *STOC 1996*, pages 99–108. ACM Press, 1996.
3. Josh Benaloh and Dwight Tuinstra. Receipt-free secret-ballot elections. In *STOC 1994*, pages 544–553. ACM Press, 1994.
4. Daniel Bleichenbacher. Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1. In *CRYPTO 1998*, volume 1462 of *Lecture Notes in Computer Science*, pages 1–12. Springer-Verlag, 1998.
5. Lenore Blum, Manuel Blum, and Michael Shub. Comparison of two pseudo-random number generators. In *CRYPTO 1982*, pages 61–78. Plenum Press, 1983.
6. Lenore Blum, Manuel Blum, and Michael Shub. A simple unpredictable pseudo-random number generator. *SIAM Journal on Computing*, 15(2):364–383, May 1986.
7. Manuel Blum and Shafi Goldwasser. An efficient probabilistic public-key encryption scheme which hides all partial information. In *CRYPTO 1984*, volume 196 of *Lecture Notes in Computer Science*, pages 289–299. Springer-Verlag, 1985.
8. Dan Boneh, Antoine Joux, and Phong Q. Nguyen. Why textbook ElGamal and RSA encryption are insecure. In *ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 30–43. Springer-Verlag, 2000.
9. Zhenfu Cao. A type of public key cryptosystem based on Eisenstein ring $\mathbb{Z}[\omega]$. In *Proceedings of the 3rd Chinese Conference of Source Coding, Channel Coding and Cryptography*, pages 178–186, 1988.
10. Zhenfu Cao. A type of puclic key cryptosystem based on k^{th} power residues (extended abstract). *Chinese Journal of Natural*, 12(11):877, 1989.
11. Zhenfu Cao. A type of puclic key cryptosystem based on k^{th} power residues (full version). *Journal of Chinese Institute of Communications*, 11(2):80–83, 1990.
12. Zhenfu Cao. *Public-key Cryptography (in Chinese)*. Heilongjiang Education Press, 1993.
13. David Cash, Eike Kiltz, and Victor Shoup. The twin Diffie-Hellman problem and applications. In *EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 127–145. Springer-Verlag, 2008.
14. Josh D. Cohen and Michael J. Fischer. A robust and verifiable cryptographically secure election scheme. In *FOCS 1985*, pages 372–382. IEEE Computer Society Press, 1985.
15. Ronald Cramer and Victor Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *CRYPTO 1998*, volume 1462 of *Lecture Notes in Computer Science*, pages 13–25. Springer-Verlag, 1998.
16. Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In *EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 45–64. Springer-Verlag, 2002.
17. Jintai Ding and Bo-Yin Yang. Multivariate public key cryptography. In *Post-quantum cryptography*, pages 193–241. Springer-Verlag, 2009.
18. Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.
19. David Mandell Freeman, Oded Goldreich, Eike Kiltz, Alon Rosen, and Gil Segev. More constructions of lossy and correlation-secure trapdoor functions. In *PKC 2010*, volume 6056 of *Lecture Notes in Computer Science*, pages 279–295. Springer-Verlag, 2010.
20. Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences (preliminary version published at STOC 1982)*, 28(2):270–299, 1984.

21. Brett Hemenway and Rafail Ostrovsky. Lossy trapdoor functions from smooth homomorphic hash proof systems. *Electronic Colloquium on Computational Complexity (ECCC), Revision 2 of Report No. 127 (2009)*, 2009.
22. Jakob Jonsson and Burton S. Kaliski Jr. On the security of RSA encryption in TLS. In *CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 127–142. Springer-Verlag, 2002.
23. Marc Joye and Benoît Libert. Efficient cryptosystems from 2^k-th power residue symbols. In *EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 76–92. Springer-Verlag, 2013.
24. Zhao Ke and Qi Sun. *Elementary Number Theory (in Chinese)*. Advanced Education Press, 2001.
25. R. J. McEliece. *A Public-Key System Based on Algebraic Coding Theory*, pages 114–116. Jet Propulsion Lab, 1978. DSN Progress Report 44.
26. Daniele Micciancio and Oded Regev. Lattice-based cryptography. In *Post-quantum cryptography*, pages 147–192. Springer-Verlag, 2009.
27. David Naccache and Jacques Stern. A new public key cryptosystem based on higher residues. In *CCS 1998*, pages 59–66. ACM Press, 1998.
28. Raphael Overbeck and Nicolas Sendrier. Code-based cryptography. In *Post-quantum cryptography*, pages 95–145. Springer-Verlag, 2009.
29. Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In *STOC 2008*, pages 187–196. ACM Press, 2008.
30. Michael Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical Report MIT/LCS/TR-212, Laboratory for Computer Science, Massachusetts Institute of Technology, January 1979.
31. Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communication of ACM*, 21(2):120–126, 1978.
32. P. W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *FOCS 1994*, pages 124–134. IEEE Computer Society Press, 1994.