New Assumptions and Efficient Cryptosystems from Power Residue Symbols

Xiaopeng Zhao[1], Zhenfu Cao[1] **, Xiaolei Dong[1], Jun Shao[2], Licheng Wang[3], and Zhusen Liu[1]

[1] School of Software Engineering, East China Normal University, Shanghai, China
52164500025@stu.ecnu.edu.cn,52184501023@stu.ecnu.edu.cn
zfcao@sei.ecnu.edu.cn,dongxiaolei@sei.ecnu.edu.cn
[2] School of Computer and Information Engineering, Zhejiang Gongshang University, Hangzhou, China
jshao@zjgsu.edu.cn
[3] State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China
wanglc@bupt.edu.cn

Abstract. The e-th power residue symbol $\left(\frac{\alpha}{\mathfrak{p}}\right)_e$, where α is a prime and $\mathfrak{p}$ is a prime ideal in the prime factorization of $p\mathbb{Z}[\zeta_e]$ with a large prime p, is a useful mathematical tool in cryptography. One famous example is the first semantic secure public-key cryptosystem due to Goldwasser and Micali (at STOC 1982). In this paper, we revisit the e-th power residue symbol and its applications. In particular, we prove that computing the e-th power residue symbol is equivalent to solving the discrete logarithm problem. By this result, we give a natural extension of the Goldwasser-Micali scheme, where e is a number only containing small prime factors. Compared to another extension of the Goldwasser-Micali scheme due to Joye and Libert (at EUROCRYPT 2013), our proposal is more efficient in terms of bandwidth utilization and decryption cost. With a new complexity assumption naturally extended from the one used in the Goldwasser-Micali scheme, our proposal is provable IND-CPA secure. Furthermore, we show that our result on the e-th power residue symbol can also be used to construct lossy trapdoor functions and circular and leakage resilient public key encryptions with more efficiency and better bandwidth utilization.

Keywords: power residue symbols · Goldwasser-Micali scheme · Joye-Libert scheme · lossy trapdoor functions · leakage resilient public-key encryption.

1 Introduction

We have witnessed the critical role of the power residue symbol in the history of public key encryption. Based on the quadratic residuosity assumption,

** Corresponding author

Goldwasser and Micali [17] proposed the first public key encryption (named GM) scheme with semantic security and additive homomorphism. This scheme is revolutionary but inefficient in terms of bandwidth, which hinders its use in practice. Following the light of the GM scheme, many attempts [5,13,4,9,2,10,23,25,26,3,11] have been made to address this issue.

Recall the encryption in the GM scheme. A message $m \in \{0,1\}$ in the GM scheme is encrypted by $c = y^m r^2 \bmod N$, where $\left(\frac{y}{N}\right) = \left(\frac{y}{p}\right) \times \left(\frac{y}{q}\right) = -1 \times -1 = 1$, r is a random element from $\mathbb{Z}_N$, $N = p \cdot q$, and p and q are big primes. It is easy to see that the value of $\log_r(r^2 \bmod N)$ determines the message space. Hence, one intuitive approach to improve the bandwidth utilization in the GM scheme is to enlarge $\log_r(r^e \bmod N)$. At STOC 1994, Benaloh and Tuinstra [2,13] set e as a special prime instead of 2. In particular, e is a prime, $e|p-1$, $e^2 \nmid p-1$, and $e \nmid q-1$. The corresponding decryption requires to locate m in $[0,e)$ by a brute-force method. Hence, e is limited to 40 bits. At ACM CCS 1998, Naccache and Stern [23] improved Benaloh and Tuinstra's method by setting e as a smooth and square-free integer $e = \prod p_i$ such that $p_i|\varphi(N)$ but $p_i^2 \nmid \varphi(N)$ for each prime p_i. The message m in this scheme is recovered from $m \equiv m_i \pmod{p_i}$ using the Chinese Remainder Theorem, and each m_i is computed by a brute-force method. Nevertheless, the constraint $p_i^2 \nmid \varphi(N)$ limits the possibility for enlarging the message space dramatically. At EUROCRYPT 2013, based on the 2^k-th power residue symbol, Joye and Libert [3] enlarged e to 2^k to obtain a nice and natural extension (named JL) of the GM scheme with better bandwidth utilization than previous schemes. Later on, Cao *et al.* [11] demonstrated that the JL scheme could be further improved by setting e as a product of small primes. As shown in [11], the resulting scheme (named CDWS) is more efficient than the JL scheme in terms of bandwidth utilization and decryption cost. Nonetheless, the corresponding security proof is complicated and hard to follow.

Due to the use in cryptography, algorithms for computing the e-th power residue symbol have also been attracted many researchers [12,30,15,19,20,7]. Several efficient algorithms for the cases of $e \in \{2,3,4,5,7,8,11,13\}$ have been proposed. However, as we know, these algorithms cannot be used for improving the GM-type schemes in [3,11] due to the small value of e. The general case of computing the e-th power residue symbol was tackled by Squirrel [30] and Boer [15], but the resulting algorithms are probabilistic and inefficient. Hence, their results cannot be applied in improving the GM scheme either. Although Freeman *et al.* [16] conducted that a "compatibility" identity can be used to compute the e-th power residue symbol, this identity could be useless in the case of prime e. As a result, we cannot use Freeman *et al.*'s algorithm to improve the GM scheme.

In order to solve the above problems, in this paper, we revisit the problem of computing the e-th power residue symbol, and obtain an efficient algorithm that can be applied in the GM-type scheme and other cryptographic primitives. Our contributions in this paper can be summarized as follows.

– **New algorithm for computing e-th power residue symbol:** We prove that computing the e-th power residue symbol is equivalent to solving the

discrete logarithm problem, if the parameters in the e-th power residue symbol $\left(\frac{\alpha}{\mathfrak{p}}\right)_e$ satisfy the following properties.

- α is an integer.
- $\mathfrak{p}$ is a prime ideal in the prime factorization of $p\mathbb{Z}[\zeta_e]$.
- p is a large prime number.

As we know, there exist several efficient algorithms for solving the discrete logarithm problem when the corresponding order is a product of small primes. Hence, we obtain an efficient algorithm for computing e-th power residue symbol when the above conditions are satisfied.

– **New extension of the GM scheme:** We demonstrate that we can obtain a natural extension of the GM scheme based on the e-th power residue symbol. Compared to the JL scheme, our extension enjoys better bandwidth utilization and higher decryption speed. While compared to the CDWS scheme, our extension has a simpler security proof.
– **New lossy trapdoor function:** As in [3,11], our GM extension can also be used to construct an efficient lossy trapdoor function, which inherits the advantages of our GM extension.
– **New circular and leakage resilient encryption:** We also give an instantiation of the subgroup indistinguishability (SG) assumption by using the e-th power residue symbol. At CRYPTO 2010, Brakerski and Goldwasser [6] gave a generic construction of circular and leakage resilient public key encryption based the SG assumption. Hence, we obtain a new circular and leakage resilient encryption scheme. Compared to the scheme in [6], our scheme is more efficient in terms of bandwidth utilization and decryption speed, due to the use of the e-th residue symbol instead of the Jacobi symbol.

The rest of this paper is organized as follows. In Section 2, we introduce some definitions and preliminaries about the e-th power residue symbol. In what follows, we show how to compute the e-th power residue symbol defined in Section 2 efficiently. Some properties and a complexity assumption related to the e-th power residue symbol are also analyzed and discussed in this section. After that, we give our extension of the GM scheme and its security and performance analysis in Section 5. In Section 6, we give two applications of our results on the e-th power residue symbol following the methods used in [3,6].

2 Notations and Basic Definitions

2.1 Notations

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

Table 1. Notations used in this paper.

Notation	Description
K	a number field.
$\mathcal{O}_K$	the ring of integers in a number field K.
$\#X$	the cardinality of X
$\langle X \rangle$	the group generated by a set X.
$a \equiv b(\mathfrak{D})$	the relation $a - b \in \mathfrak{D}$, where elements $a, b \in \mathcal{O}_K$ and the ideal $\mathfrak{D} \subset \mathcal{O}_K$
$\otimes$	the direct product of two algebraic structures
log	the binary logarithm
φ	the Euler's totient function
$\mathbb{N}$	set of natural numbers.
$\mathbb{Z}_n$	$\mathbb{Z}/n\mathbb{Z}$
$\mathbb{Z}_n^*$	the multiplicative group of $\mathbb{Z}/n\mathbb{Z}$
p, q	prime numbers.
e_p, e_q	$e_p \vert p-1$ and $e_q \vert q-1$.

2.2 Power Residue Symbols

Let K be a number field, and $\mathcal{O}_K$ be the ring of integers in K, and $e \geq 1$ be an integer. We say a prime ideal $\mathfrak{A}$ in $\mathcal{O}_K$ is prime to e if $\mathfrak{A} \nmid e\mathcal{O}_K$. It is easy to see that $\mathfrak{A}$ is relatively prime to e if and only if $\gcd(\mathtt{Norm}(\mathfrak{A}), e) = 1$, where $\mathtt{Norm}(\mathfrak{A}) = \#\left(\mathcal{O}_K/\mathfrak{A}\right)$. Notably, for every $\alpha \in \mathcal{O}_K$, $\alpha \notin \mathfrak{A}$, we have

$$\alpha^{\mathtt{Norm}(\mathfrak{A})-1} \equiv 1 \pmod{\mathfrak{A}}.$$

Let $\zeta_e = \exp(2\pi i/e)$ be an e-th primitive root of unity. If $\zeta_e \in K$ and $\mathfrak{A}$ is relatively prime to e, the order of the subgroup of $\left(\mathcal{O}_K/\mathfrak{A}\right)^{\times}$ generated by ζ_e mod $\mathfrak{A}$ is e. This indicates that e divides $\mathtt{Norm}(\mathfrak{A}) - 1$, hence we can define the *e-th power residue symbol* $\left(\frac{\alpha}{\mathfrak{A}}\right)_e$ as follows: if $\alpha \in \mathfrak{A}$, then $\left(\frac{\alpha}{\mathfrak{A}}\right)_e = 0$; otherwise, $\left(\frac{\alpha}{\mathfrak{A}}\right)_e$ is the unique e-th root of unity such that

$$\alpha^{\frac{\mathtt{Norm}(\mathfrak{A})-1}{e}} \equiv \left(\frac{\alpha}{\mathfrak{A}}\right)_e \pmod{\mathfrak{A}}.$$

Next, we extend the symbol multiplicatively to all ideals. Suppose the prime decomposition of an ideal $\mathfrak{A} \subset \mathcal{O}_K$ is $\mathfrak{A} = \prod_i \mathfrak{B}_i$ and every prime ideal $\mathfrak{B}_i$ is prime to e. For $\alpha \in \mathcal{O}_K$ define $\left(\frac{\alpha}{\mathfrak{A}}\right)_e = \prod_i \left(\frac{\alpha}{\mathfrak{B}_i}\right)_e$. From here on, We simply consider the case $K = \mathbb{Q}(\zeta_e)$ since it is well-known that $\mathcal{O}_K = \mathbb{Z}[\zeta_e]$ in this case. We suggest interested readers to refer to [18,22,24] for more details about the *e-th power residue symbols*.

An integer μ is said to be a *non-degenerate primitive (e_p, e_q)-th root of unity* modulo N if the orders of μ modulo p and q are e_p and e_q respectively. For example, if μ_p and μ_q are primitive roots modulo p and q respectively, then an

element $\mu \in \mathbb{Z}_N^*$ satisfies

$$\mu \equiv \mu_p^{\frac{p-1}{e_p}} \pmod{p}, \quad \mu \equiv \mu_q^{\frac{q-1}{e_q}} \pmod{q}$$

is a *non-degenerate primitive* (e_p, e_q)*-th root of unity* modulo N. With this definition we can easily determine the factorization of the principal ideal $p\mathbb{Z}[\zeta_{e_p}]$ in $\mathbb{Z}[\zeta_{e_p}]$ from Proposition I.8.3 in [24]. Namely, we have $p\mathbb{Z}[\zeta_{e_p}] = \prod_{i \in \mathbb{Z}_{e_p}^*} \mathfrak{p}_i$ and $\mathtt{Norm}(\mathfrak{p}_i) = p$ where

$$\mathfrak{p}_i = p\mathbb{Z}[\zeta_{e_p}] + (\zeta_{e_p} - \mu^i)\mathbb{Z}[\zeta_{e_p}]. \tag{1}$$

Similarly, we also have $q\mathbb{Z}[\zeta_{e_q}] = \prod_{i \in \mathbb{Z}_{e_q}^*} \mathfrak{q}_i$ and $\mathtt{Norm}(\mathfrak{q}_i) = q$ in $\mathbb{Z}[\zeta_{e_q}]$ where

$$\mathfrak{q}_i = q\mathbb{Z}[\zeta_{e_q}] + (\zeta_{e_q} - \mu^i)\mathbb{Z}[\zeta_{e_q}]. \tag{2}$$

Freeman *et al.* [16] investigated further the structure of the ideals $\mathfrak{a}_i = \mathfrak{p}_i\mathfrak{q}_i$ in the case of $e_p = e_q = e$.

Lemma 1 (Freeman *et al.* [16]). *Let e be a positive integer, $K = \mathbb{Q}(\zeta_e)$ and $\mathcal{O}_K = \mathbb{Z}[\zeta_e]$. Let N be a product of two distinct primes p, q with $p \equiv 1 \pmod{e}$ and $q \equiv 1 \pmod{e}$ and let $\mu \in \mathbb{Z}_N^*$ be a* non-degenerate primitive (e, e)-th root of unity *modulo N. Define $\mathfrak{p}_i = p\mathcal{O}_K + (\zeta_e - \mu^i)\mathcal{O}_K$, $\mathfrak{q}_i = q\mathcal{O}_K + (\zeta_e - \mu^i)\mathcal{O}_K$ and $\mathfrak{a}_i = N\mathcal{O}_K + (\zeta_e - \mu^i)\mathcal{O}_K$ for each $i \in \mathbb{Z}_e^*$. We have $\mathfrak{a}_i = \mathfrak{p}_i\mathfrak{q}_i$ and $\mathtt{Norm}(\mathfrak{a}_i) = N$ for each $i \in \mathbb{Z}_e^*$ and $N\mathcal{O}_K = \prod_{i \in \mathbb{Z}_e^*} \mathfrak{a}_i$.*

The ideals $\mathfrak{p}_1$ in (1) and $\mathfrak{q}_1$ in (2) are mainly considered in the following discussion. As a matter of convenience, we shall denote $\mathfrak{p}_1$, $\mathfrak{q}_1$ and $\mathfrak{a}_1$ by $\mathfrak{p}$, $\mathfrak{q}$ and $\mathfrak{a}$ respectively.

3 Computation and Properties of e-th Power Residue Symbols

In this section, we show how to compute the *e-th power residue symbols* with respect to $\mathfrak{p}$ and $\mathfrak{q}$ efficiently. We also investigate more properties of *e-th power residue symbols*.

3.1 Computing e-th Power Residue Symbols

For a general integer e and an ideal in $\mathbb{Z}[\zeta_e]$, it is really tough to design an efficient and deterministic algorithm to compute *e-th power residue symbols* since we can hardly find a deterministic way to decrease the norm of ideals. In fact, efficient and deterministic algorithms are only known in the case of $e \in \{2, 3, 4, 5, 7[12], 8[19], 11[20], 13[7]\}$. The general case is tackled probabilistically by Squirrel [30] and Boer [15]. However, their algorithms are not quite efficient and no rigorous proof has been found to prove that they run in polynomial time. For a composite e, Freeman *et al.* constructed a "compatibility"

identity [Appendix B, [16]] to decrease the size of the power e so as to reduce the amount of computation. However, this identity can not compute the *e-th power residue symbols* in the case of prime powers.

Quadratic and power residues are useful building-blocks of many cryptographic applications. In most applications, the factorization of N is transparent to participants who want to learn the values of *e-th power residue symbols*. Therefore, we don't necessarily consider the computation in the general case. We can actually do better with the factorization of N. The following simple theorem demonstrates that computing $\left(\frac{\alpha}{\mathfrak{p}}\right)_{e_p}$ for an integer α (and hence also $\left(\frac{\alpha}{\mathfrak{q}}\right)_{e_q}$) is closely related to solving the *discrete logarithm problem* (DLP) in a specific cyclic group. Recall that the DLP is defined as: given a finite cyclic group $\mathbb{G}$ of order n with a generator α and an element $\beta \in \mathbb{G}$, find the integer $x \in \mathbb{Z}_n$ such that $\alpha^x = \beta$.

Theorem 1. *With notations as in Lemma 1, we deduce that* $\left(\frac{\alpha}{\mathfrak{p}}\right)_{e_p} = \zeta_{e_p}^x$ *if and only if* $\mu^x = \alpha^{\frac{p-1}{e_p}}$ *in* $\mathbb{Z}_p^*$. *Therefore, the solution to the DLP in the finite cyclic subgroup* $\langle \mu \rangle$ *of order* e_p *allows one to compute the symbols* $\left(\frac{\alpha}{\mathfrak{p}}\right)_{e_p}$.

Proof. If $\mu^x = \alpha^{\frac{p-1}{e_p}}$ in $\mathbb{Z}_p^*$, then $\alpha^{\frac{p-1}{e_p}} - \zeta_e^x = \mu^x - \zeta_{e_p}^x \in \mathfrak{p}$. It follows that $\left(\frac{\alpha}{\mathfrak{p}}\right)_{e_p} = \zeta_{e_p}^x$. Conversely, If $\left(\frac{\alpha}{\mathfrak{p}}\right)_{e_p} = \zeta_{e_p}^x$ for some $x \in \mathbb{Z}_{e_p}$, that is $\alpha^{\frac{p-1}{e_p}} - \zeta_{e_p}^x \in \mathfrak{p}$. Since the order of $\alpha^{\frac{p-1}{e_p}}$ divides e_p, $\alpha^{\frac{p-1}{e_p}}$ can be expressed as μ^y for some $y \in \mathbb{Z}_{e_p}$, which implies $\mu^x - \mu^y \in \mathfrak{p}$ and hence $\mu^x = \mu^y$. The fact that the order of μ is e_p forces $x = y$. ■

Although the DLP is considered in general to be intractable, it can be easily solved in a few particular situations, e.g., if the order of $\mathbb{G}$ is *smooth* (it only contains small prime factors), the Pohlig-Hellman algorithm [28] turns out to be quite efficient. In other words, if e_p is chosen with appropriate prime factors and given the factorization of N, we can compute the above symbols by using only the Pohlig-Hellman algorithm. We remark that e_p is better to be chosen as a power of a small prime number.

Algorithm 1 Pohlig-Hellman algorithm for prime powers

Input: an integer $\mu \in \mathbb{Z}_p$ **of order** e^k **where** e **and** p **are primes with** $e^k \mid p-1$ **and** $x \in \langle \mu \rangle$

Output: $\boldsymbol{y} = (y_{k-1}, \ldots, y_0)_e$ **such that** $\mu^y \equiv x \pmod{p}$

1: Compute the values $\mu^{e^{k-1}i}$ for each $0 \leq i < e$ and store them in a lookup table
2: Compute the values μ^{-e^i} for each $0 \leq i < k$ and store them in a lookup table
3: $x_0 \longleftarrow x$
4: Call the hash algorithm to find $y_0 \in \mathbb{Z}_e$ such that $\mu^{e^{k-1}y_0} \equiv x_0^{e^{k-1}} \pmod{p}$
5: **for** $1 \leq i \leq k-1$ **do**
6: $\quad x_i \longleftarrow x_{i-1}\mu^{-y_{i-1}e^{i-1}} \bmod p$
7: $\quad$ Call the hash algorithm to find $y_i \in \mathbb{Z}_e$ such that $\mu^{e^{k-1}y_i} \equiv x_i^{e^{k-i-1}} \pmod{p}$
8: **end for**
9: **return** $\boldsymbol{y} = (y_{k-1}, \ldots, y_0)_e$

Remark 1. The above algorithm can be made a little faster. According to the step 6 above, we have

$$x_i^{e^{k-i-1}} \equiv x_{i-1}^{e^{k-i-1}} \mu^{-y_{i-1}e^{k-2}} \pmod{p}$$

We can cut down the number of operations of computing $x_i^{e^{k-i-1}}$ by evaluating

$$x_{i-1}^{e^{k-i-1}} \quad \text{and} \quad x_{i-1}^{e^{k-(i-1)-1}} \equiv \left(x_{i-1}^{e^{k-i-1}}\right)^e \pmod{p}$$

successively in the previous step. So the way to optimize the algorithm is: we record the values of $x_{i-1}^{e^{k-i-1}}$ for odd indices i, then the number of operations of computing each even part is highly reduced.

3.2 Some Properties of Power Residue Symbols

In this section, we assume $e_p = e_q = e$. For an arbitrary $k \geq 2$, we say an integer $x \in \mathbb{Z}_N^*$ is a *k-th residue* modulo N if there exists an integer $y \in \mathbb{Z}_N^*$ such that $y^k \equiv x \pmod{N}$. Note that if x is an *e-th residue* modulo N, then we have $\left(\frac{x}{\mathfrak{p}_i}\right)_e = \left(\frac{x}{\mathfrak{q}_i}\right)_e = 1$ for each $i \in \mathbb{Z}_e^*$. We denote the set of all *e-th residues* in $\mathbb{Z}_N^*$ by $\mathcal{ER}_N^e$. Correspondingly, the set $\left\{x \in \mathbb{Z}_N^* \;\middle|\; \left(\frac{x}{\mathfrak{a}}\right)_e = 1\right\}$ is denoted by $\mathbb{J}_N^e$. The following theorem is crucial for the instantiation of *subgroup indistinguishability assumption* defined in Section 6.1.

Theorem 2. *With notations as in Lemma 1, assume that* $e_p = e_q = e$ *and let* $\mathcal{ER}_m^e = \left\{x^e \bmod m \;\middle|\; x \in \mathbb{Z}_m^*\right\}$ *denote the set of all* e-th residues *in* $\mathbb{Z}_m^*$ *and let* $\mathscr{U} = \left\{1, \zeta_e, \ldots, \zeta_e^{e-1}\right\}$ *denote the multiplicative subgroup of roots of unity in* $\mathbb{Z}[\zeta_e]$*, then*

(1) $\mathbb{Z}_p^*/\mathcal{ER}_p^e \cong \mathscr{U}$ *(and hence also* $\mathbb{Z}_q^*/\mathcal{ER}_q^e \cong \mathscr{U}$*)*

(2) If we require further that $\gcd(\frac{p-1}{e}, e) = \gcd(\frac{q-1}{e}, e) = 1$*, then there is a* $\nu \in \mathbb{Z}$ *such that*

- ν *is a* non-degenerate primitive (e,e)-th root of unity *modulo* N.
- $\left(\frac{\nu}{\mathfrak{a}_i}\right)_e = 1$ *for every ideal* $\mathfrak{a}_i \subset \mathbb{Z}[\zeta_e]$.

Furthermore, we have

$$\mathbb{J}_N^e = \langle \nu \rangle \otimes \mathcal{ER}_N^e$$

Proof.

(1) Consider the homomorphism $\theta : \mathbb{Z}_p^* \to \mathscr{U}$ defined by $x \mapsto \left(\frac{x}{\mathfrak{p}}\right)_e$. Since the number of roots of the polynomial $f(x) = x^{\frac{p-1}{e}} - 1$ over the field $\mathbb{Z}[\zeta_e]/\mathfrak{p}$ is at most $\frac{p-1}{e}$ and the cardinality of $\mathcal{ER}_p^e$ is exactly $\frac{p-1}{e}$, an integer $z \in \mathbb{Z}_p^*$ satisfying $\left(\frac{z}{\mathfrak{p}}\right)_e = 1$ must be in $\mathcal{ER}_p^e$. Hence the kernel of θ is $\mathcal{ER}_p^e$ and we have the desired isomorphism due to the fact that the cardinality of left hand side is equal to the cardinality of right hand side. Of course, elements in different cosets of $\mathcal{ER}_p^e$ in $\mathbb{Z}_p^*$ have different *e-th power residue symbols*, and there is a one to one correspondence between the cosets of $\mathcal{ER}_p^e$ in $\mathbb{Z}_p^*$ and the e-th roots of unity via the *e-th power residue symbols*.

(2) The condition $\gcd(\frac{p-1}{e}, e) = \gcd(\frac{q-1}{e}, e) = 1$ implies that there exist integers $s_p \in \mathbb{Z}_e^*$, $t_p, s_q \in \mathbb{Z}_e^*$, t_q such that $s_p\frac{p-1}{e} + t_p e = s_q\frac{q-1}{e} + t_q e = 1$. Let $\mu_p \equiv \mu \bmod p$ and $\mu_q \equiv \mu \pmod q$. Observe that every primitive e-th root of unity in $\mathbb{Z}_p^*$ has the form μ_p^i for some $i \in \mathbb{Z}_e^*$. It follows that

$$\left(\frac{\mu_p^{s_p}}{\mathfrak{p}}\right)_e = \left(\frac{\zeta_e^{s_p}}{\mathfrak{p}}\right)_e = \zeta_e^{\frac{p-1}{e}s_p}$$

Similarly,

$$\left(\frac{\mu_q^{-s_q}}{\mathfrak{q}}\right)_e = \left(\frac{\zeta_e^{-s_q}}{\mathfrak{q}}\right)_e = \zeta_e^{-\frac{q-1}{e}s_q}$$

Hence, letting ν be the integer congruent to $\mu_p^{s_p}$ modulo p and $\mu_q^{-s_q}$ modulo q. Then,

$$\left(\frac{\nu}{\mathfrak{a}}\right)_e = \left(\frac{\nu}{\mathfrak{p}}\right)_e \left(\frac{\nu}{\mathfrak{q}}\right)_e = \zeta_e^{\left(s_p\frac{p-1}{e} - s_q\frac{q-1}{e}\right)} = 1$$

Since $\nu \in \mathbb{Z}$, the result $\left(\frac{\nu}{\mathfrak{a}_i}\right)_e = 1$ follows from the Galois equivalence. To prove the last statement we only need to prove that every element of $\mathbb{J}_N^e$ can be written as a product of two elements in $\langle \nu \rangle$ and $\mathcal{ER}_N^e$ respectively as $\langle \nu \rangle \cap \mathcal{ER}_N^e = \varnothing$. For any $x \in \mathbb{J}_N^e$, since there exists $j \in \mathbb{Z}_e$ such that $\left(\frac{\nu^j}{\mathfrak{p}}\right) = \left(\frac{x}{\mathfrak{p}}\right)$ and $\left(\frac{\nu^j}{\mathfrak{q}}\right) = \left(\frac{x}{\mathfrak{q}}\right)$, we have $x \equiv \nu^j y^e \pmod p$ and $x \equiv \nu^j z^e \pmod q$ for some $x \in \mathbb{Z}_p^*$ and $y \in \mathbb{Z}_q^*$ from (1). Take $w \equiv y \pmod p$ and $w \equiv z \pmod q$, then we have $x \equiv \nu^j w^e \pmod N$, as desired. ∎

Remark 2. In particular, if $e = 2$, then we deduce from Theorem 2 the well-known theorem which says that $\mathbb{J}_N \cong \{\pm 1\} \otimes \mathbb{QR}_N$ where N is a *Blum integer* and $\mathbb{QR}_N = \left\{x^2 \mid x \in \mathbb{Z}_N^*\right\}$ and $\mathbb{J}_N = \left\{\left(\frac{x}{N}\right)_2 = 1 \mid x \in \mathbb{Z}_N^*\right\}$.

4 A New Assumption from Power Residue Symbols

In this section, we shall give a formal definition of the assumption our cryptosystem relies on. We start with the following definition of a set which is contrary to $\mathcal{ER}_N^{\mathrm{lcm}(e_p,e_q)}$. Let e denote $\gcd(p-1,q-1)$ we define

$$\mathcal{J}_N^{(e_p,e_q)} = \left\{ x \in \mathbb{Z}_N^* \;\middle|\; \left(\frac{x}{\mathfrak{a}}\right)_e = 1,\ \left(\frac{x}{\mathfrak{p}}\right)_{e_p} \text{ and } \left(\frac{x}{\mathfrak{q}}\right)_{e_q} \text{ are primitive} \right\}.$$

Note that the condition $\left(\frac{x}{\mathfrak{a}}\right)_e = 1$ ensures that $\left(\frac{x}{\mathfrak{a}_i}\right)_e = 1$ for each $i \in \mathbb{Z}_e^*$ by the Galois equivalence, hence $\left(\frac{x}{N\mathbb{Z}[\zeta_e]}\right)_e = 1$[4].

Definition 1 ((e_p,e_q)-th Power Residue ((e_p,e_q)-PR) Assumption). *Given a security parameter κ. A* PPT *algorithm* $\mathsf{RSAgen}(\kappa)$ *generates two integers e_p and e_q and a random RSA modulus $N = pq$ such that $p \equiv 1 \bmod e_p$ and $q \equiv 1 \bmod e_q$, and chooses at random $\mu \in \mathbb{Z}_N^*$ a* non-degenerate primitive (e_p,e_q)-th root of unity *modulo N. The (e_p,e_q)-*PR *assumption with respect to* $\mathsf{RSAgen}(\kappa)$ *asserts that the advantage* $\mathsf{Adv}^{(e_p,e_q)\text{-PR}}_{\mathcal{A},\mathsf{RSAgen}}(\kappa)$ *defined as*

$$\left|\mathsf{Pr}\left(\mathcal{A}(N,x,\mathrm{lcm}(e_p,e_q)) = 1 \;\middle|\; x \xleftarrow{\$} \mathcal{ER}_N^{\mathrm{lcm}(e_p,e_q)}\right) - \mathsf{Pr}\left(\mathcal{A}(N,x,\mathrm{lcm}(e_p,e_q)) = 1 \;\middle|\; x \xleftarrow{\$} \mathcal{J}_N^{(e_p,e_q)}\right)\right|$$

is negligible for any PPT *adversary $\mathcal{A}$; the probabilities are taken over the experiment of running $(N,(e_p,e_q),\mu) \leftarrow \mathsf{RSAgen}(\kappa)$ and choosing at random $x \in \mathcal{ER}_N^{\mathrm{lcm}(e_p,e_q)}$ and $x \in \mathcal{J}_N^{(e_p,e_q)}$.*

Since the standard QR assumption only requires that the Jacobi symbol is 1, the hardness of the $(2,1)$-PR assumption can only potentially be a weaker assumption than hardness of the standard QR assumption. Also, the hardness of the $(2^k,1)$-PR assumption can only potentially be a weaker assumption than hardness of the Gap-2^k-Res assumption with $q \equiv 3 \pmod 4$ defined in [Definition 4, [3]] since $x \in \mathbb{J}_N \setminus \mathbb{QR}_N$ if and only if $x \in \mathbb{J}_N$ and $\left(\frac{x}{\mathfrak{p}}\right)_{2^k}$ is primitive (for an arbitrary μ).

[4] The IND-CPA secure of the JL scheme is equivalent to the Gap-2^k-Res assumption [Section 4.1, [3]], which was considered in [1] by Abdalla *et al.* However, the hardness of this assumption depends on the choice of q in fact (recall that $p \equiv 1 \bmod 2^k$). In detail, if 2^ℓ $(2 \le \ell \le k)$ is a common divisor of $p-1$ and $q-1$, the symbol $\left(\frac{x}{\mathfrak{a}_i}\right)_{2^\ell}$ must be equal to 1 for each $x \in \mathcal{ER}_N^{2^\ell}$, but it does not necessarily hold true for elements chosen from $\mathbb{J}_N \setminus \mathbb{QR}_N$. In this case, the generic algorithms introduced in the first paragraph of Section 3.1 may break this assumption if given some (e_p,e_q)-th primitive root of unity modulo N, which we believe can be obtained by using Coppersmith's method to solve the modular equation $x^{2^\ell} - 1 \equiv 0 \pmod N$. Even if $\ell = 2$, the JL scheme may leak 1-bit information of a plaintext.

5 A New Homomorphic Public-Key Cryptosystem

We generalize the GM scheme as well as the JL scheme. Our new homomorphic cryptosystem can efficiently encrypt larger messages than both of them and the decryption is much faster than that of the JL scheme.

5.1 Description

The setting of our new cryptosystem (denoted by Π) is essentially the same as the GM scheme and the JL scheme. More precisely, the setting $e_p = 2$, $e_q = 1$ corresponds to the GM scheme and the setting $e_p = 2^k$, $e_q = 1$ corresponds to the JL scheme.

$\mathsf{KeyGen}\,(1^\kappa)$ Given a security parameter κ. KeyGen selects *smooth* integers e_p and e_q, then generates an RSA modulus $N = pq$ a product of two large and equally sized primes p and q such that $e_p \mid p-1, e_q \mid q-1$ and picks at random $\mu \in \mathbb{Z}_N^*$ a *non-degenerate primitive* (e_p, e_q)*-th root of unity* modulo N and $y \xleftarrow{\$} \mathcal{J}_N^{(e_p,e_q)}$. The public and secret keys are $\mathsf{pk} = \{N, \mathrm{lcm}(e_p, e_q), y\}$ and $\mathsf{sk} = \{p, q, e_p, e_q, \mu\}$.

$\mathsf{Enc}\,(\mathsf{pk}, m)$ To encrypt a message $m \in \mathbb{Z}_{\mathrm{lcm}(e_p,e_q)}$, Enc picks a random $r \in \mathbb{Z}_N^*$ and returns the ciphertext

$$c = y^m r^{\mathrm{lcm}(e_p,e_q)} \bmod N.$$

$\mathsf{Dec}\,(\mathsf{sk}, c)$ Given the ciphertext c and the secret key $\mathsf{sk} = \{p, q, e_p, e_q, \mu\}$, Dec first computes $\left(\frac{c}{\mathfrak{p}}\right)_{e_p} = \zeta_{e_p}^{z_p}$ and $\left(\frac{c}{\mathfrak{q}}\right)_{e_q} = \zeta_{e_q}^{z_q}$ by means of Theorem 1. Then, it recovers the message $m \in \mathbb{Z}_{\mathrm{lcm}(e_p,e_q)}$ from

$$m \equiv z_p k_p^{-1} \bmod e_p \quad \text{and} \quad m \equiv z_q k_q^{-1} \bmod e_q$$

by using the Chinese Remainder Theorem with non-pairwise coprime moduli, where $\left(\frac{y}{\mathfrak{p}}\right)_{e_p} = \zeta_{e_p}^{k_p}$ and $\left(\frac{y}{\mathfrak{q}}\right)_{e_q} = \zeta_{e_q}^{k_q}$ are pre-computed.

5.2 Security analysis

The cryptosystem Π also has the similar security analysis as for the GM scheme.

Theorem 3. *The cryptosystem* Π *is* IND-CPA *secure under the* (e_p, e_q)-PR *assumption.*

Proof. Consider changing the distribution of the public key. Under the (e_p, e_q)-PR assumption, we may choose y uniformly in $\mathcal{ER}_N^{\mathrm{lcm}(e_p,e_q)}$ instead of choosing it from $\mathcal{J}_N^{(e_p,e_q)}$, while this is done without noticing the adversary. In this case, the ciphertext carries no information about the message and hence Π is IND-CPA secure.

5.3 Parameter Selection

The key generation requires two primes p and q such that $e_p \mid p-1$ and $e_q \mid q-1$, where e_p and e_q are better to be chosen so that they are powers of small primes and $0 \leq \log(e_p) < \frac{\log(p)}{2}, 0 \leq \log(e_q) < \frac{\log(q)}{2}$ in practice. The algorithm to produce p and q is similar in spirit to the algorithm described in [Section 5.1, [3]]. The major difference is that the size of $\log e_p + \log e_q$ is bounded by $\frac{1}{2}\log N$. The reason is provided by the following proposition [Lemma 8, [31]] related to Coppersmith's method for finding small roots of bivariate modular equations.

Proposition 1. *Let p and q be equally sized primes and $N = pq$. Let e be a divisor of $\varphi(N) = (p-1)(q-1)$. If there exists a positive constant c such that $e > N^{\frac{1}{2}+c}$ holds, then there exists a* PPT *algorithm that given N and e, it factorizes N.*

Note that taking $\frac{1}{4}\log N < \log e_p + \log e_q < \frac{1}{2}\log N$ does not contradict the setting of *Φ-Hiding Assumption* [8] as the prime factors of $\varphi(N)$ are very small. However, $\log e_p + \log e_q$ shall not be close to $\frac{1}{2}\log N$ because we don't know whether there exists an attack of mixing together Coppersmith's attack and exhaustive searches. In particular, if we take $e_p = 2^k$, $e_q = 2$ and $k > \frac{1}{4}\log N$, the low-order $\frac{1}{4}\log N$ bits of p is revealed to an adversary, and hence it can factorize N by implementing Coppersmith's attack [14]. Therefore, if we choose e_p and e_q not to be a power of 2 and to be coprime, we may handle messages at least twice as long as the JL scheme does. The key generation also requires a random integer $y \in \mathbb{Z}_N^*$ in $\mathcal{J}_N^{(e_p,e_q)}$. We can use (2) in Theorem 2 for uniformly sampling integers in $\mathbb{J}_N^{\gcd(p-1,q-1)}$. A random integer modulo N has a probability of exactly $\frac{\varphi(e_p)\varphi(e_q)}{e_p e_q}$ of being in the set

$$\left\{ x \in \mathbb{Z}_N^* \;\middle|\; \left(\frac{x}{\mathfrak{p}}\right)_{e_p} \text{ and } \left(\frac{x}{\mathfrak{q}}\right)_{e_q} \text{ are primitive} \right\}.$$

If we take $e_p = e_1^{f_1}$ and $e_q = e_2^{f_2}$ where e_1 and e_2 are distinct primes, the above probability is equal to $\frac{(e_1-1)(e_2-1)}{e_1 e_2}$. Therefore, a suitable $y \in \mathcal{J}_N^{(e_p,e_q)}$ is likely to be obtained after several trials.

5.4 Performance and Comparisons

Now, we investigate the performance of our cryptosystem and make comparisons with the Paillier cryptosystem [26] and the JL scheme [3], two famous schemes in the literature on homomorphic encryption.

All the three cryptosystems require the generation of two large suitable primes. Though both the cryptosystem *Π* and JL scheme need to select other elements, it altogether takes a negligible amount of time compared with the selection of the primes.

It is easy to see that the Paillier cryptosystem takes about four times as long as the Π or the JL scheme to encrypt messages or perform homomorphic operations because the modular multiplications are computed over $\mathbb{Z}_{N^2}^*$.

One major drawback of the JL scheme is that its decryption [Algorithm 1, [3]] is slow. When decrypting a 128-bit message, it needs roughly

$$\log p - 128 + \frac{128(128-1)}{4} + \frac{128}{2} = \log p + 4000$$

modular multiplications over $\mathbb{Z}_p^*$ on average according to the remark following [Algorithm 1, [3]]. However, if we take $e_p = 929^{13} > 2^{128}$ and $e_q = 1$, the major time consuming part of Π's decryption is performing the Pohlig-Hellman algorithm to compute $\left(\frac{\cdot}{\mathfrak{p}}\right)_{e_p}$. If the storage is enough, in order to speed up, we may pre-evaluate the quantities $\mu^{929^{13}k} \bmod p$ for $k = 0, 1, \ldots, 928$ and $\mu^{-929^j} \bmod p$ for $j = 0, 1, \ldots, 12$ in a lookup table. If we ignore the constant time which it spends on the hash algorithm, then the decryption only requires

$$\log p - 128 + \sum_{\substack{k=0 \\ k \text{ is even}}}^{12} \log(929^k) + 128 \approx \log p + 414$$

modular multiplications over $\mathbb{Z}_p^*$ on average according to the remark following Algorithm 3.1. If N is taken as 2048 bits, the decryption of Π is approximately 3.5 times faster than that of the JL scheme. Also, it is easy to see that the larger the e_p is, the faster Π's encryption is and the larger the storage space Π will require. Even though we do not use the lookup table, Π's decryption still runs faster than that of JL scheme.

Comparatively, the advantage of the Paillier cryptosystem is that the ciphertext expansion is small and it supports homomorphic operations over larger messages. The Π and the JL scheme have better performance of performing messages of small or specific size. For example, as mentioned in [3], they can be used to encrypt a 128- or 256-bit symmetric key in a KEM/DEM construction [29]. Note that the ciphertext expansion of the JL scheme is at least twice as large as that of the Π according to the analysis in Section 5.3.

6 Applications

6.1 Circular and Leakage Resilient Public-Key Encryption

Brakerski and Goldwasser introduced the notion of *subgroup indistinguishability* (SG) *assumption* in [Section 3.1, [6]]. They instantiated the SG assumption based on the QR and the DCR assumptions and proposed a generic construction of schemes which achieved *key-dependent security* and *auxiliary-input security* based on the SG assumption. However, the scheme based on the QR assumption can only encrypt a 1-bit message at a time. In this section, we will show how to instantiate the SG assumption under the new hardness assumption called power

residue assumption. In this way, the scheme becomes much more efficient in bandwidth exploitation.

Definition 2 (Subgroup Indistinguishability Assumption [6]). *Given a security parameter κ, and three commutative multiplicative groups (indexed by κ) $\mathbb{G}_U$, $\mathbb{G}_M$ and $\mathbb{G}_L$ such that $\mathbb{G}_U$ is a direct product of $\mathbb{G}_M$ (of order M) and $\mathbb{G}_L$ (of order L) where $\mathbb{G}_M$ is cyclic and $\gcd(M, L) = 1$. We require that the generator h for $\mathbb{G}_M$ is efficiently computable from the description of $\mathbb{G}_U$. We further requires that there exists a* PPT *algorithm that outputs $I_{\mathbb{G}_U} = (OP_{\mathbb{G}_U}, S_{\mathbb{G}_M}, S_{\mathbb{G}_L}, h, T)$ an instance of $\mathbb{G}_U$, where $OP_{\mathbb{G}_U}$ is an efficient algorithm performs group operations in $\mathbb{G}_U$, $S_{\mathbb{G}_M}, S_{\mathbb{G}_L}$ are efficient algorithms sample a random element from $\mathbb{G}_M, \mathbb{G}_L$ respectively and T is a known upper bound such that $T \geq M \cdot L$. For any adversary $\mathcal{A}$ we denote the* subgroup distinguishing advantage *of $\mathcal{A}$ by*

$$SGAdv[\mathcal{A}] = \left|\mathsf{Pr}\left(\mathcal{A}(1^\kappa, x) \;\middle|\; x \xleftarrow{\$} \mathbb{G}_U\right) - \mathsf{Pr}\left(\mathcal{A}(1^\kappa, x) \;\middle|\; x \xleftarrow{\$} \mathbb{G}_L\right)\right|$$

The subgroup indistinguishability assumption *is that for any* PPT *$\mathcal{A}$ it holds that for a properly sampled instance $I_{\mathbb{G}_U}$, we have that $SGAdv[\mathcal{A}]$ is negligible.*

Now, we instantiate the SG assumption from the *e-th power residue symbols*. Let e be a *smooth* integer. We sample a random RSA modulus $N = pq$ such that $e = \gcd(p-1, q-1)$ and $\gcd(\frac{p-1}{e}, e) = \gcd(\frac{q-1}{e}, e) = 1$. Let $\mathcal{ER}_N^e$ and $\mathbb{J}_N^e$ be as in Section 3.2, then we have shown there exists a $\nu \in \mathbb{J}_N^e \setminus \mathcal{ER}_N^e$ such that $\mathbb{J}_N^e = \langle\nu\rangle \otimes \mathcal{ER}_N^e$ from (2) in Theorem 2. The groups $\mathbb{J}_N^e$, $\langle\nu\rangle$ and $\mathcal{ER}_N^e$ have orders $\frac{\varphi(N)}{e}$, e and $\frac{\varphi(N)}{e^2}$ respectively and we denote $\frac{\varphi(N)}{e}$ by N'. The condition $\gcd(\frac{p-1}{e}, e) = \gcd(\frac{q-1}{e}, e) = 1$ implicates that $\gcd(e, \frac{\varphi(N)}{e^2}) = 1$. We define the following assumption which is similar to the (e_p, e_q)-PR assumption defined previously.

Definition 3 (e-th Power Residue (e-PR) Assumption). *Given a security parameter κ. A* PPT *algorithm $\mathsf{RSAgen}(\kappa)$ generates an integer e with small prime factors and a random RSA modulus $N = pq$ such that $e = \gcd(p-1, q-1)$ and $\gcd(\frac{p-1}{e}, e) = \gcd(\frac{q-1}{e}, e) = 1$, and chooses at random $\mu \in \mathbb{Z}_N^*$ a* non-degenerate primitive (e, e)-th root of unity *modulo N. The* e-PR *assumption with respect to $\mathsf{RSAgen}(\kappa)$ asserts that the advantage $\mathsf{Adv}^{e\text{-PR}}_{\mathcal{A},\mathsf{RSAgen}}(\kappa)$ defined as*

$$\left|\mathsf{Pr}\left(\mathcal{A}(N, x, e) = 1 \;\middle|\; x \xleftarrow{\$} \mathcal{ER}_N^e\right) - \mathsf{Pr}\left(\mathcal{A}(N, x, e) = 1 \;\middle|\; x \xleftarrow{\$} \mathbb{J}_N^e\right)\right|$$

is negligible for any PPT *adversary $\mathcal{A}$; the probabilities are taken over the experiment of running $(N, e, \mu) \leftarrow \mathsf{RSAgen}(\kappa)$ and choosing at random $x \in \mathcal{ER}_N^e$ and $x \in \mathbb{J}_N^e$.*

Since there exist efficient sampling algorithms that sample a random element from $\mathcal{ER}_N^e$ and $\mathbb{J}_N^e$ according to Theorem 2, the e-PR assumption leads immediately to the instantiation of the SG assumption by setting $\mathbb{G}_U = \mathbb{J}_N^e$, $\mathbb{G}_M = \langle\nu\rangle$, $\mathbb{G}_L = \mathcal{ER}_N^e$, $h = \nu$, and $T = N \geq eN'$. The corresponding encryption scheme is presented in Appendix A.

6.2 Constructing Lossy Trapdoor Functions from the (e_p, e_q)-th Residue Assumption

Lossy Trapdoor Functions *Lossy trapdoor functions* (LTDF) were introduced by Peikert and Waters [27] and since then numerous applications emerge in cryptography. Informally speaking, LTDF consist of two families of functions. The functions in one family are injective trapdoor functions, while functions in the other family are lossy, that is, the image size is smaller than the domain size. It also requires that the functions sampled from the first and the second family are computationally indistinguishable. Using the constructions in [27], one can obtain CCA-secure public-key encryptions. So far, LTDF are mainly constructed from assumptions such as DDH [27], LWE [27], QR [16], DCR [16], *Φ-Hiding* [21], etc.

Joye and Libert constructed a LTDF with short outputs and keys based on the k-QR, k-SJS and DDH assumptions in [3]. Of course, it is an easy matter to generalize their constructions, using our techniques based on the power residue symbols. Hence, we only propose a new generic construction of the LTDF and the corresponding conclusions. We follow the definition of the LTDF in [3] and omit the security analysis since it proceeds in exactly the same way in [3].

$\mathsf{InjGen}(1^\kappa)$ Given a security parameter κ, let ℓ_N, k and n (n is a multiple of k) be parameters determined by κ. InjGen defines $m = \frac{n}{k}$ and performs the following steps.

1. Select *smooth* integers e_p and e_q such that $k < \log(e_p) + \log(e_q) < \frac{\ell_N}{2}$. Generate an ℓ_N-bit RSA modulus $N = pq$ such that $p-1 = e_p p', q-1 = e_q q'$ and $0 \leq \log(e_p) < \frac{\log(p)}{2}, 0 \leq \log(e_q) < \frac{\log(q)}{2}$ for large primes p, q, p', q'. Pick at random $\mu \in \mathbb{Z}_N^*$ a *non-degenerate primitive* (e_p, e_q)*-th root of unity* modulo N and $y \xleftarrow{\$} \mathcal{J}_N^{(e_p, e_q)}$.
2. For each $i \in \{1, \ldots, m\}$, pick h_i in $\mathcal{ER}_N^{\mathrm{lcm}(e_p, e_q)}$ at random.
3. Choose $r_1, \ldots, r_m \xleftarrow{\$} \mathbb{Z}_{p'q'}$ and compute a $m \times m$ matrix $Z = (Z_{i,j})$ with

$$Z_{i,j} = \begin{cases} y \cdot h_j^{r_i} \bmod N, & \text{if } i = j; \\ h_j^{r_i} \bmod N, & \text{otherwise.} \end{cases}$$

Output the evaluation key $\mathsf{ek} = \{N, Z\}$ and the secret key $\mathsf{sk} = \{p, q, e_p, e_q, \mu, y\}$.

$\mathsf{LossyGen}(1^\kappa)$ The process of LossyGen is identical to the process of InjGen, except that

- Set $Z_{i,j} = h_j^{r_i} \bmod N$ for each $1 \leq i, j \leq m$.
- LossyGen does not output the secret key sk.

$\mathsf{Evaluation}(\mathsf{ek}, x)$ Given $\mathsf{ek} = \left\{N, Z = (Z_{i,j})_{i,j \in \{1,\ldots,m\}}\right\}$ and a message $x \in \{0,1\}^n$, Evaluation parses x as a k-adic string $\boldsymbol{x} = (x_1, \ldots, x_m)$ with $x_i \in \mathbb{Z}_{2^k}$ for each i. Then, it computes and returns $\boldsymbol{y} = (y_1, \ldots, y_m) \in \mathbb{Z}_N^m$ with $y_j = \prod_{i=1}^m Z_{i,j}^{x_i} \bmod N$.

$\mathsf{Inversion}(\mathsf{sk}, \boldsymbol{y})$ Given $\mathsf{sk} = \{p, q, e_p, e_q, \mu, y\}$ and $\boldsymbol{y} = (y_1, \ldots, y_m) \in \mathbb{Z}_N^m$, $\mathsf{Inversion}$ applies the decryption algorithm $\mathsf{Dec}(\mathsf{sk}, y_j)$ of the Π for each y_j to recover x_j for $j = 1$ to m. It recovers and outputs the input $x \in \{0,1\}^n$ from the resulting vector $\boldsymbol{x} = (x_1, \ldots, x_m) \in \mathbb{Z}_{2^k}^m$.

Proposition 2. *Let $\ell = n - \log(p'q')$. The above construction is a (n, ℓ)-LTDF if the (e_p, e_q)-*th power residue assumption *holds and the* DDH *assumption holds in the subgroup $\mathcal{ER}_N^{\mathrm{lcm}(e_p, e_q)}$.*

Clearly, our new proposed LTDF outperforms the Joye-Libert LTDF in terms of its fast decryption and small ciphertext expansion. Our LTDF has $\ell = n - \log(p'q') > (n - \ell_N) + \log(e_p) + \log(e_q)$ bits of lossiness. Therefore, the lossiness may also be improved as there are no known attacks against the factorization of N when $\frac{\ell_N}{4} < \log(e_p) + \log(e_q) < \frac{\ell_N}{2}$ and $0 \leq \log(e_p) < \frac{\log(p)}{2}, 0 \leq \log(e_q) < \frac{\log(q)}{2}$.

References

1. Michel Abdalla, Fabrice Ben Hamouda, and David Pointcheval. Tighter reductions for forward-secure signature schemes. In *International Workshop on Public Key Cryptography*, pages 292–311. Springer, 2013.
2. Josh Daniel Cohen Benaloh. *Verifiable secret-ballot elections*. PhD thesis, Yale University, New Haven, CT, USA, 1987.
3. Fabrice Benhamouda, Javier Herranz, Marc Joye, and Benoît Libert. Efficient cryptosystems from 2^k-th power residue symbols. *J. Cryptology*, 30(2):519–549, 2017.
4. Lenore Blum, Manuel Blum, and Mike Shub. A simple unpredictable pseudo-random number generator. *SIAM Journal on computing*, 15(2):364–383, 1986.
5. Manuel Blum and Shafi Goldwasser. An efficient probabilistic public-key encryption scheme which hides all partial information. In *Workshop on the Theory and Application of Cryptographic Techniques*, pages 289–299. Springer, 1984.
6. Zvika Brakerski and Shafi Goldwasser. Circular and leakage resilient public-key encryption under subgroup indistinguishability - (or: Quadratic residuosity strikes back). In *Advances in Cryptology - CRYPTO 2010*, volume 6223 of *LNCS*, pages 1–20. Springer, 2010.
7. Eric Brier and David Naccache. The thirteenth power residue symbol. *IACR Cryptology ePrint Archive*, 2019:1176, 2019.
8. Christian Cachin, Silvio Micali, and Markus Stadler. Computationally private information retrieval with polylogarithmic communication. In *Advances in Cryptology - EUROCRYPT 1999*, volume 1592 of *LNCS*, pages 402–414. Springer, 1999.
9. Zhenfu Cao. A type of public key cryptosystem based on Eisenstein ring $\mathbb{Z}[\omega]$. *Proceedings of the 3rd Chinese Conference of Source Coding, Channel Coding and Cryptography*, pages 178–186, 1988.
10. Zhenfu Cao. A new public-key cryptosystem based on k^{th}-power residues (full version). *Journal of the China Institute of Communications*, 11(2):80–83, 1990.
11. Zhenfu Cao, Xiaolei Dong, Licheng Wang, and Jun Shao. More efficient cryptosystems from k-th power residues. *IACR Cryptology ePrint Archive*, 2013:569, 2013.

12. Perlas C. Caranay and Renate Scheidler. An efficient seventh power residue symbol algorithm. *International Journal of Number Theory*, 6(08):1831–1853, 2010.
13. Josh D. Cohen and Michael J. Fischer. A robust and verifiable cryptographically secure election scheme (extended abstract). In *26th Annual Symposium on Foundations of Computer Science*, pages 372–382. IEEE Computer Society, 1985.
14. Don Coppersmith. Small solutions to polynomial equations, and low exponent RSA vulnerabilities. *Journal of cryptology*, 10(4):233–260, 1997.
15. Koen de Boer. Computing the power residue symbol. *Master's thesis. Nijmegen, Radboud University. www.koendeboer.com*, 2016.
16. David Mandell Freeman, Oded Goldreich, Eike Kiltz, Alon Rosen, and Gil Segev. More constructions of lossy and correlation-secure trapdoor functions. *Journal of cryptology*, 26(1):39–74, 2013.
17. Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of computer and system sciences*, 28(2):270–299, 1984.
18. Kenneth Ireland and Michael Rosen. *A classical introduction to modern number theory*, volume 84. Springer Science & Business Media, 2013.
19. Marc Joye. Evaluating octic residue symbols. *IACR Cryptology ePrint Archive*, 2019:1196, 2019.
20. Marc Joye, Oleksandra Lapiha, Ky Nguyen, and David Naccache. The eleventh power residue symbol. *IACR Cryptology ePrint Archive*, 2019:870, 2019.
21. Eike Kiltz, Adam O'Neill, and Adam Smith. Instantiability of RSA-OAEP under chosen-plaintext attack. *Journal of Cryptology*, 30(3):889–919, 2017.
22. Franz Lemmermeyer. *Reciprocity laws: from Euler to Eisenstein*. Springer Science & Business Media, 2013.
23. David Naccache and Jacques Stern. A new public key cryptosystem based on higher residues. In *CCS 1998*, pages 59–66. ACM, 1998.
24. Jürgen Neukirch. *Algebraic number theory*, volume 322. Springer Science & Business Media, 2013.
25. Tatsuaki Okamoto and Shigenori Uchiyama. A new public-key cryptosystem as secure as factoring. In *Advances in Cryptology - EUROCRYPT 1998*, volume 1403 of *LNCS*, pages 308–318. Springer, 1998.
26. Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in Cryptology - EUROCRYPT 1999*, volume 1592 of *LNCS*, pages 223–238. Springer, 1999.
27. Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. *SIAM Journal on Computing*, 40(6):1803–1844, 2011.
28. Stephen C. Pohlig and Martin E. Hellman. An improved algorithm for computing logarithms over GF(p) and its cryptographic significance. *IEEE Trans. Information Theory*, 24(1):106–110, 1978.
29. Victor Shoup. Using hash functions as a hedge against chosen ciphertext attack. In *Advances in Cryptology - EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 275–288. Springer, 2000.
30. Douglas Squirrel. Computing reciprocity symbols in number fields, 1997. *Undergraduate thesis, Reed College*.
31. Takashi Yamakawa, Shota Yamada, Goichiro Hanaoka, and Noboru Kunihiro. Adversary-dependent lossy trapdoor function from hardness of factoring semi-smooth RSA subgroup moduli. In *Advances in Cryptology - CRYPTO 2016*, volume 9815 of *LNCS*, pages 3–32. Springer, 2016.

A Description of the Encryption Scheme in Section 6.1

$\mathsf{KeyGen}\,(1^\kappa)$ Given a security parameter κ, KeyGen selects a *smooth* integer e and samples a random RSA modulus $N = pq$ such that $e = \gcd(p-1, q-1)$ and $\gcd(\frac{p-1}{e}, e) = \gcd(\frac{q-1}{e}, e) = 1$. KeyGen selects ν as in Theorem 2 and $\ell \in \mathbb{N}$ which is polynomial in κ. It also samples $\boldsymbol{s} \xleftarrow{\$} (\mathbb{Z}_e)^\ell$ and sets the secret key $\mathsf{sk} = \boldsymbol{s}$. It then samples $\boldsymbol{g} \xleftarrow{\$} (\mathcal{ER}_N^e)^\ell$ and sets

$$g_0 = \left(\prod_{1 \le i \le \ell} {g_i}^{s_i} \right)^{-1} \mod N.$$

The public key is set to be $\mathsf{pk} = \{N, g_0, \boldsymbol{g}\}$.

$\mathsf{Enc}\,(\mathsf{pk}, m)$ On inputs a public key $\mathsf{pk} = \{N, g_0, \boldsymbol{g}\}$ and a message $m \in \langle \nu \rangle$, Enc samples r from the set $\left\{1, 2, \ldots, N^2\right\}$ and computes $\boldsymbol{c} = \boldsymbol{g}^r$ and $c_0 = m \cdot g_0^r$. It returns the ciphertext $(c_0, \boldsymbol{c})$.

$\mathsf{Dec}\,(\mathsf{sk}, c)$ On inputs the secret key $\mathsf{sk} = \boldsymbol{s}$ and a ciphertext $\{c_0, \boldsymbol{c}\}$, Dec computes and returns $m = c_0 \cdot \prod_{1 \le i \le \ell} c_i^{s_i}$.