New Assumptions and Efficient Cryptosystems from the e-th Power Residue Symbol

Xiaopeng Zhao[1], Zhenfu Cao[1,2,3](✉), Xiaolei Dong[1](✉), Jun Shao[4], Licheng Wang[5], and Zhusen Liu[1]

[1] Shanghai Key Laboratory of Trustworthy Computing, East China Normal University, Shanghai, China
`52164500025@stu.ecnu.edu.cn, zfcao@sei.ecnu.edu.cn`
`dongxiaolei@sei.ecnu.edu.cn, 52184501023@stu.ecnu.edu.cn`

[2] Cyberspace Security Research Center, Peng Cheng Laboratory, Shenzhen, Shanghai

[3] Institute of Intelligent Science and Technology, Tongji University, Shanghai, China

[4] School of Computer and Information Engineering, Zhejiang Gongshang University, Hangzhou, China
`chn.junshao@gmail.com`

[5] 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 $\left(\frac{\alpha}{\mathfrak{p}}\right)_e$ is a useful mathematical tool in cryptography, where α is an integer, $\mathfrak{p}$ is a prime ideal in the prime factorization of $p\mathbb{Z}[\zeta_e]$ with a large prime p satisfying $e \mid p-1$, and ζ_e is an e-th primitive root of unity. One famous case of the e-th power symbol 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 cryptosystem, where e is an integer only containing small prime factors. Compared to another extension of the Goldwasser-Micali cryptosystem 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 cryptosystem, our proposal is provable IND-CPA secure. Furthermore, we show that our results 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 symbol · Goldwasser-Micali cryptosystem · Joye-Libert cryptosystem · lossy trapdoor function · 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, Goldwasser and Micali [18] 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 [2–5, 10–12, 14, 24, 26, 27] 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 $N = p \cdot q$, p and q are large primes, $\left(\frac{y}{N}\right) = \left(\frac{y}{p}\right) \times \left(\frac{y}{q}\right) = -1 \times -1 = 1$ and r is an element picked at random from $\mathbb{Z}_N$. 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, 14] set e as a special prime instead of 2. In particular, e is a prime, $e \mid p-1$, $e^2 \nmid p-1$, 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 [24] improved Benaloh and Tuinstra's method by setting e as a smooth and square-free integer $e = \prod p_i$ such that $p_i \mid \varphi(N)$ 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 where 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.* [12] demonstrated that the JL scheme could be further improved by setting e as a product of small primes. As shown in [12], 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.

By virtue of the fruitful use in cryptography, algorithms for computing the e-th power residue symbol have also attracted many researchers [7, 8, 13, 16, 20, 21, 31]. 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, 12] owing to the small value of e. The general case of computing the e-th power residue symbol was tackled by Squirrel [31] and Boer [16], but the resulting algorithms are probabilistic and inefficient. Hence, their results may not be applied in improving the GM scheme either. Although Freeman *et al.* [17] 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 a prime power 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.
  - p is a prime number satisfying $e \mid p - 1$.
  - $\mathfrak{p}$ is a prime ideal in the prime factorization of $p\mathbb{Z}[\zeta_e]$, and ζ_e is an e-th primitive root of unity.

  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, 12], 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 on 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, due to the use of the e-th power 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 4. In Section 5, we give two applications of our results on the e-th power residue symbol following the methods described 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
letters in $\mathfrak{mathfrak}$	ideals in $\mathcal{O}_K$
$a = b \pmod{\mathfrak{D}}$	the relation $a - b \in \mathfrak{D}$, where elements $a, b \in \mathcal{O}_K$
$\#X$	the cardinality of a set X
X^n	the Cartesian product $\prod_{i=1}^n X$
$\langle X \rangle$	the group generated by a set X
$x \xleftarrow{R} X$	x is sampled from the uniform distribution over a set X
$\otimes$	the direct product of two algebraic structures
φ	the Euler's totient function
$\gcd(x, y)$	the greatest common divisor of x and y
$\mathrm{lcm}(x, y)$	the least common multiple of x and y
log	the binary logarithm
ζ_e	an e-th primitive root of unity, i.e., $\zeta_e = \exp(2\pi i/e)$
$\mathbb{Z}_n$	the ring $\{0, 1, \ldots, n-1\}$
$\mathbb{Z}_n^*$	the multiplicative group $\{x \in \mathbb{Z}_n \mid \gcd(x, n) = 1\}$
p, q	large prime numbers
N	$N = p \cdot q$
e_p, e_q	$e_p \mid p - 1$ and $e_q \mid q - 1$

2.2 Power Residue Symbols

We say a prime ideal $\mathfrak{A}$ in $\mathcal{O}_K$ is prime to an integer $e(\geq 1)$ if $\mathfrak{A} \nmid e\mathcal{O}_K$. It is easy to deduce that the corresponding necessary and sufficient condition is $\gcd(\texttt{Norm}(\mathfrak{A}), e) = 1$, where $\texttt{Norm}(\mathfrak{A}) = \#\left(\mathcal{O}_K/\mathfrak{A}\right)$. Then, we have

$$\alpha^{\texttt{Norm}(\mathfrak{A})-1} = 1 \pmod{\mathfrak{A}} \quad (\text{for } \alpha \in \mathcal{O}_K, \alpha \notin \mathfrak{A}).$$

Furthermore, if we have an additional condition that $\zeta_e \in K$, then we have that the order of group $\langle \zeta_e/\mathfrak{A} \rangle$ generated in $\left(\mathcal{O}_K/\mathfrak{A}\right)^{\times}$ is e, and hence $e \mid \texttt{Norm}(\mathfrak{A}) - 1$. Now, 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

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

The definition can be naturally extended to the case that $\mathfrak{A}$ is not a prime ideal, such that $\mathfrak{A} = \prod_i \mathfrak{B}_i$ and $\gcd(\texttt{Norm}(\mathfrak{B}_i), e) = 1$. In particular, we define

$$\left(\frac{\alpha}{\mathfrak{A}}\right)_e = \prod_i \left(\frac{\alpha}{\mathfrak{B}_i}\right)_e .$$

In the rest of this paper, we simply consider the case of $K = \mathbb{Q}(\zeta_e)$, since we have $\mathcal{O}_K = \mathbb{Z}[\zeta_e]$ in this case. We suggest interested readers to refer to [19, 23, 25] for more details about the *power residue symbol*.

2.3 Security Definition

A public key encryption is composed of three algorithms: the key generation algorithm `KeyGen`, the encryption algorithm `Enc`, and the decryption algorithm `Dec`. The `IND-CPA` security for a public key encryption is defined as follows:

Definition 1 (`IND-CPA` Security). *The public key encryption scheme PKE* $= (\texttt{KeyGen}, \texttt{Enc}, \texttt{Dec})$ *is said to be* `IND-CPA` *secure if for any probabilistic polynomial time (*`PPT`*) distinguisher, given the public key* `pk` *generated by* `KeyGen`*, and any pair of messages* m_0*,* m_1 *of equal length, the non-negative advantage function* $\epsilon(\kappa)$ *in the security parameter* κ *for distinguishing* $c_0 = \texttt{Enc}(\texttt{pk}, m_0)$ *and* $c_1 = \texttt{Enc}(\texttt{pk}, m_1)$ *is negligible, i.e., we have* $\lim_{\kappa\to\infty} P(\kappa)\cdot\epsilon(\kappa) = 0$ *for every polynomial* P*.*

3 Computation and Properties of the Power Residue Symbol

In this section, we show how to compute the *power residue symbol* in some circumstance and investigate some relative properties that we will employ in this paper later.

3.1 Computing Power Residue Symbols

In this subsection, we show that computing the power residue symbol is equivalent to solving the discrete logarithm problem if some specific conditions are satisfied.

Before giving the proof, we would like to introduce the concept of the non-degenerate primitive (e_p, e_q)-th root of unity modulo N. Let μ_p and μ_q be primitive roots modulo p and q respectively. We say an integer μ is a *non-degenerate primitive* (e_p, e_q)*-th root of unity* modulo N if both the following two congruences hold.

$$\mu = \mu_p^{\frac{p-1}{e_p}\alpha} \pmod{p} \quad \text{for some } \alpha \in \mathbb{Z}_{e_p}^*, \text{ and}$$
$$\mu = \mu_q^{\frac{q-1}{e_q}\beta} \pmod{q} \quad \text{for some } \beta \in \mathbb{Z}_{e_q}^*$$

According to the result in [25, Proposition I.8.3], we have

$$p\mathbb{Z}[\zeta_{e_p}] = \prod_{i\in\mathbb{Z}^*_{e_p}} \mathfrak{p}_i,\ \mathtt{Norm}(\mathfrak{p}_i) = p\ (i \in \mathbb{Z}^*_{e_p}),\ \text{and}$$
$$q\mathbb{Z}[\zeta_{e_q}] = \prod_{j\in\mathbb{Z}^*_{e_q}} \mathfrak{q}_j,\ \mathtt{Norm}(\mathfrak{q}_j) = q\ (j \in \mathbb{Z}^*_{e_q}),$$

where $\mathfrak{p}_i = p\mathbb{Z}[\zeta_{e_p}] + (\zeta_{e_p} - \mu^i)\mathbb{Z}[\zeta_{e_p}]$ and $\mathfrak{q}_j = q\mathbb{Z}[\zeta_{e_q}] + (\zeta_{e_q} - \mu^j)\mathbb{Z}[\zeta_{e_q}]$.

With the notation μ and some integer α, we can give Theorem 1 which shows that computing $\left(\frac{\alpha}{\mathfrak{p}_1}\right)_{e_p}$ is equivalent to solving the discrete logarithm in the cyclic subgroup $\langle\mu\rangle$ of $\mathbb{Z}^*_p$ with order e_p. We can obtain a similar result for the case of $\left(\frac{\alpha}{\mathfrak{q}_1}\right)_{e_q}$ by analogy with Theorem 1.

Theorem 1. $\left(\frac{\alpha}{\mathfrak{p}_1}\right)_{e_p} = \zeta^x_{e_p} \Longleftrightarrow \mu^x = \alpha^{\frac{p-1}{e_p}} \pmod{p}$.

Proof. We give the proof in two parts as follows.

$\Longrightarrow$ From the definition of the *power residue symbol* and $\mathtt{Norm}(\mathfrak{p}_1) = p$, we have that $\left(\frac{\alpha}{\mathfrak{p}_1}\right)_{e_p} = \alpha^{\frac{\mathtt{Norm}(\mathfrak{p}_1)-1}{e_p}} = \alpha^{\frac{p-1}{e_p}} \pmod{\mathfrak{p}_1}$. Together with $\left(\frac{\alpha}{\mathfrak{p}_1}\right)_{e_p} = \zeta^x_{e_p}$, we obtain that $\zeta^x_{e_p} = \alpha^{\frac{p-1}{e_p}} \pmod{\mathfrak{p}_1}$. Furthermore, from the definition of $\mathfrak{p}_1$, we have $\mu = \zeta_{e_p} \pmod{\mathfrak{p}_1}$. Then, $\mu^x = \zeta^x_{e_p} = \alpha^{\frac{p-1}{e_p}} \pmod{\mathfrak{p}_1}$ is deduced. At last, due to $\mu^x = \alpha^{\frac{p-1}{e_p}} \pmod{\mathfrak{p}_1}$ and $(\mu^x, \alpha^{\frac{p-1}{e_p}}) \in \mathbb{Z}^2$, we can finally get $\mu^x = \alpha^{\frac{p-1}{e_p}} \pmod{p}$.

$\Longleftarrow$ From $\mu^x = \alpha^{\frac{p-1}{e_p}} (\bmod\ p)$, we have that $\mu^x = \alpha^{\frac{p-1}{e_p}} \pmod{\mathfrak{p}_1}$. Furthermore, we have that $\left(\frac{\alpha}{\mathfrak{p}_1}\right)_{e_p} = \alpha^{\frac{p-1}{e_p}} \pmod{\mathfrak{p}_1}$ and $\zeta_{e_p} = \mu \pmod{\mathfrak{p}_1}$ as in the previous case. Hence, we have that $\left(\frac{\alpha}{\mathfrak{p}_1}\right)_{e_p} = \alpha^{\frac{p-1}{e_p}} = \mu^x = \zeta^x_{e_p} \pmod{\mathfrak{p}_1}$ and $\left(\frac{\alpha}{\mathfrak{p}_1}\right)_{e_p} = \zeta^x_{e_p}$.

This completes the proof. □

It is well-known that the discrete logarithm problem is intractable in general but quite easy in some special cases. For instance, when the order of the underlying finite cyclic group is *smooth*, i.e., it only contains small prime factors, the discrete logarithm problem can be easily solved by virtue of the Pohlig-Hellman algorithm [29]. In our case, if e_p is chosen with appropriate prime factors, the e_p-th *power residue symbol* can be efficiently computed by virtue of the Pohlig-Hellman algorithm. For the completeness, we describe the Pohlig-Hellman algorithm for prime powers in Algorithm 1.

Algorithm 1 Pohlig-Hellman algorithm for prime powers

```
Input: $(g, y, p, s^k)$, where $p$ and $s$ are primes, $s^k \mid p-1$, and the order of $g$ in $\mathbb{Z}_p^*$ is $s^k$.
Output: $x = (x_{k-1}, \ldots, x_0)_s$, where $g^x = y \pmod p$, $x = \sum_{i=0}^{k-1} x_i \cdot s^i$, and $x_i \in [0, s-1]$ for $i \in [0, k-1]$.
1: $y_0 \leftarrow y$
2: Find $x_0 \in \mathbb{Z}_s$ such that $\left(g^{s^{k-1}}\right)^{x_0} = y_0^{s^{k-1}} \pmod p$.
3: for $1 \leq i \leq k-1$ do
4:     $y_i \longleftarrow y_{i-1}\left(g^{-s^{i-1}}\right)^{x_{i-1}} \pmod p$
5:     Find $x_i \in \mathbb{Z}_s$ such that $\left(g^{s^{k-1}}\right)^{x_i} = y_i^{s^{k-i-1}} \pmod p$.
6: end for
7: return $\mathbf{x} = (x_{k-1}, \ldots, x_0)_s$
```

Remark 1 (Hints for Optimization). From line 2 and line 5 in Algorithm 1, we can see that values of $\left(g^{s^{k-1}}\right)^i \pmod p$ for each $i \in [0, s-1]$ are used repeatedly. Hence, we can save the computational cost by pre-computing and storing these values. Similar method can be also applied to $g^{-s^i} \pmod p$ for each $i \in [0, k-1]$ to save more computational cost.

Furthermore, according to line 4 in Algorithm 1, we have that

$$y_i^{s^{k-i-1}} = \left(y_{i-1}\left(g^{-s^{i-1}}\right)^{x_{i-1}}\right)^{s^{k-i-1}} = y_{i-1}^{s^{k-i-1}}\left(g^{-s^{k-2}}\right)^{x_{i-1}} \pmod p.$$

We can save the cost of computing $y_i^{s^{k-i-1}}$ if we have known the value of $y_{i-1}^{s^{k-i-1}}$, which can be recorded during the computing process of $y_{i-1}^{s^{k-(i-1)-1}}$. However, this optimization cannot be applied for every y_i ($i \in [0, k-1]$). It is because that once the computation of $y_i^{s^{k-i-1}}$ is based on the value of $y_{i-1}^{s^{k-(i-1)-1}}$, there is no $y_i^{s^{k-i-2}}$ for computing $y_{i+1}^{s^{k-i-2}}$. As a result, this optimization can only be applied on the odd indices.

3.2 A New Assumption from Power Residue Symbols

In this subsection, we would like to give a new assumption named (e_p, e_q)-th power residue (denoted as (e_p, e_q)-$\mathtt{PR}$) assumption which will be used in our proposed public key encryption in Section 4 and lossy trapdoor functions in Section 5.1.

We set that $\mathbb{ER}_N^e = \{x \mid \exists y \in \mathbb{Z}_N^*, y^e = x \pmod N\}$ and

$$\mathbb{NR}_N^{(e_p,e_q)} = \left\{ x \mid x \in \mathbb{Z}_N^*, \left(\frac{x}{\mathfrak{a}_1}\right)_t = 1, \left(\frac{x}{\mathfrak{p}_1}\right)_{e_p} \text{ and } \left(\frac{x}{\mathfrak{q}_1}\right)_{e_q} \text{ are primitive} \right\},$$

where N, e_p, e_q, $\mathfrak{p}_1$, and $\mathfrak{q}_1$ are the same as those in Section 3.1, $\mathfrak{a}_1 = \mathfrak{p}_1\mathfrak{q}_1$, and $t = \gcd(p-1, q-1)$. We define the (e_p, e_q)-$\mathtt{PR}$ assumption as follows.

Definition 2 (**(e_p, e_q)-th Power Residue Assumption**)**.** *Given a security parameter κ and $N, (e_p, e_q), \mu, x$, it is intractable to decide whether x is in $\mathbb{ER}_N^{\mathrm{lcm}(e_p,e_q)}$ or $\mathbb{NR}_N^{(e_p,e_q)}$ if x is chosen at random from $\mathbb{ER}_N^{\mathrm{lcm}(e_p,e_q)}$ and $\mathbb{NR}_N^{(e_p,e_q)}$. Formally, the advantage $\mathtt{Adv}_{\mathcal{A}}^{(e_p,e_q)\text{-}\mathtt{PR}}(\kappa)$ defined as*

$$\Big|\mathtt{Prob}\left[\mathcal{A}\left(N, \mathrm{lcm}(e_p,e_q), \mu, x\right) = 1 \mid x \overset{R}{\leftarrow} \mathbb{ER}_N^{\mathrm{lcm}(e_p,e_q)}\right] - \mathtt{Prob}\left[\mathcal{A}\left(N, \mathrm{lcm}(e_p,e_q), \mu, x\right) = 1 \mid x \overset{R}{\leftarrow} \mathbb{NR}_N^{(e_p,e_q)}\right]\Big|$$

is negligible for any $\mathtt{PPT}$ *adversary $\mathcal{A}$; the probabilities are taken over the experiment of generating $(N, (e_p, e_q), \mu)$ with respect to the security parameter κ and choosing at random x from $\mathbb{ER}_N^{\mathrm{lcm}(e_p,e_q)}$ and $\mathbb{NR}_N^{(e_p,e_q)}$.*

Remark 2. It is easy to see that if we set $t = 2$, $e_p = 2$ and $e_q = 1$, the (e_p, e_q)-$\mathtt{PR}$ assumption becomes the standard quadratic residuosity ($\mathtt{QR}$) assumption with $\gcd(p-1, q-1) = 2$. Furthermore, if we set $t = 2$, $e_p = 2^k$ and $e_q = 1$, the (e_p, e_q)-$\mathtt{PR}$ assumption becomes the $\mathtt{Gap}$-2^k-$\mathtt{Res}$ assumption with $q = 3 \pmod 4$ which has been used in [1] and [3]. From [3, Theorem 2], we note that the $\mathtt{Gap}$-2^k-$\mathtt{Res}$ assumption with $q = 3 \pmod 4$ solely relies on a $\mathtt{QR}$-based assumption, namely, the k-$\mathtt{QR}$ assumption.

3.3 Some Properties of Power Residue Symbols

In this subsection, we present some properties of *power residue symbols* that will be used in the design of circular and leakage resilient public key encryption (especially for the instantiation of *subgroup indistinguishability assumption*) in Section 5.2. Note that only in this subsection and Section 5.1, we require that $e_q = e_p = e$.

If $e_q = e_p = e$, according to the result in [17], we have

$$\mathfrak{a}_i = \mathfrak{p}_i \mathfrak{q}_i, \ \mathtt{Norm}(\mathfrak{a}_i) = N, \ \text{and } N\mathbb{Z}[\zeta_e] = \prod_{i \in \mathbb{Z}_e^*} \mathfrak{a}_i,$$

where $p\mathbb{Z}[\zeta_e] = \prod_{i \in \mathbb{Z}_e^*} \mathfrak{p}_i$, $\mathtt{Norm}(\mathfrak{p}_i) = p$, $q\mathbb{Z}[\zeta_e] = \prod_{i \in \mathbb{Z}_e^*} \mathfrak{q}_i$, $\mathtt{Norm}(\mathfrak{q}_i) = q$, and $\mathfrak{a}_i = N\mathbb{Z}[\zeta_e] + (\zeta_e - \mu^i)\mathbb{Z}[\zeta_e]$ for each $i \in \mathbb{Z}_e^*$.

Let $\mathbb{ER}_\Delta^e = \{x \in \mathbb{Z}_N^* \mid \exists y, y^e = x \pmod{\Delta}\}$, $\mathbb{J}_N^e = \{x \in \mathbb{Z}_N^* \mid \left(\frac{x}{\mathfrak{a}_1}\right)_e = 1\}$, and $\mathscr{U} = \left\{\zeta_e^i \mid i \in [0, e-1]\right\}$, where $\Delta \in \{p, q, N\}$. We have the following theorems.

Theorem 2. $\mathbb{Z}_p^*/\mathbb{ER}_p^e \cong \mathscr{U} \cong \mathbb{Z}_q^*/\mathbb{ER}_q^e$.

Proof. We would like to prove $\mathbb{Z}_p^*/\mathbb{ER}_p^e \cong \mathscr{U}$ at first. Consider the homomorphism $\theta : \mathbb{Z}_p^* \to \mathscr{U}$ defined by $x \mapsto \left(\frac{x}{\mathfrak{p}_1}\right)_e$. Since the number of the distinct roots of the polynomial $f(x) = x^{\frac{p-1}{e}} - 1$ over the field $\mathbb{Z}[\zeta_e]/\mathfrak{p}_1$ is at most $\frac{p-1}{e}$ and the cardinality of $\mathbb{ER}_p^e$ is exactly $\frac{p-1}{e}$, we have that an element $z \in \mathbb{Z}_p^*$ satisfying $\left(\frac{z}{\mathfrak{p}_1}\right)_e = 1$ must lie in $\mathbb{ER}_p^e$. Hence, we have that the kernel of θ is $\mathbb{ER}_p^e$, i.e., the homomorphism $\tau : \mathbb{Z}_p^*/\mathbb{ER}_p^e \to \mathscr{U}$ induced by θ is a monomorphism. Furthermore, we know the cardinality of $\mathbb{Z}_p^*/\mathbb{ER}_p^e$ equals to $\frac{p-1}{\frac{p-1}{e}} = e$, which is also the value of the cardinality of $\mathscr{U}$. As a result, $\mathbb{Z}_p^*/\mathbb{ER}_p^e \cong \mathscr{U}$.

Similarly, we can get $\mathbb{Z}_q^*/\mathbb{ER}_q^e \cong \mathscr{U}$. Hence, we accomplish the proof. □

Theorem 3. *If the condition* $\gcd(\frac{p-1}{e}, e) = \gcd(\frac{q-1}{e}, e) = 1$ *holds, then there exists an integer* ν *satisfying the following properties.*

- ν *is a* non-degenerate primitive (e, e)-th root of unity *modulo* N*.*
- $\left(\frac{\nu}{\mathfrak{a}_i}\right)_e = 1$ *for all* $i \in \mathbb{Z}_e^*$*.*
- $\mathbb{J}_N^e = \langle \nu \rangle \otimes \mathbb{ER}_N^e$.

Proof. We give the proof one by one.

- 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 $s_p \cdot \frac{p-1}{e} + t_p \cdot e = s_q \cdot \frac{q-1}{e} + t_q \cdot e = 1$. Let $\mu_p = \mu \pmod{p}$ and $\mu_q = \mu \pmod{q}$. We can get a *non-degenerate primitive* (e, e)*-th root of unity* modulo N by the following congruences.

$$\begin{cases} \nu = \mu_p^{s_p} \pmod{p} \\ \nu = \mu_q^{-s_q} \pmod{q} \end{cases}$$

- When ν is generated as above, we have

$$\left(\frac{\nu}{\mathfrak{p}_1}\right)_e = \left(\frac{\mu_p^{s_p}}{\mathfrak{p}_1}\right)_e = \left(\frac{\zeta_e^{s_p}}{\mathfrak{p}_1}\right)_e = \zeta_e^{\frac{p-1}{e} s_p}$$

and

$$\left(\frac{\nu}{\mathfrak{q}_1}\right)_e = \left(\frac{\mu_q^{-s_q}}{\mathfrak{q}_1}\right)_e = \left(\frac{\zeta_e^{-s_q}}{\mathfrak{q}_1}\right)_e = \zeta_e^{-\frac{q-1}{e} s_q}.$$

Hence, we have

$$\left(\frac{\nu}{\mathfrak{a}_1}\right)_e = \left(\frac{\nu}{\mathfrak{p}_1}\right)_e \left(\frac{\nu}{\mathfrak{q}_1}\right)_e = \zeta_e^{\left(s_p \frac{p-1}{e} - s_q \frac{q-1}{e}\right)} = 1.$$

Since $\nu \in \mathbb{Z}$, the result $\left(\frac{\nu}{\mathfrak{a}_i}\right)_e = 1$ $(i \in \mathbb{Z}_e^*)$ follows from the Galois equivalence.

- To prove the last property 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 $\mathbb{ER}_N^e$ respectively since $\langle \nu \rangle \cap \mathbb{ER}_N^e = \{1\}$. For any $x \in \mathbb{J}_N^e$, since there exists $j \in \mathbb{Z}_e$ such that $\left(\frac{\nu^j}{\mathfrak{p}_1}\right)_e = \left(\frac{x}{\mathfrak{p}_1}\right)_e$ and $\left(\frac{\nu^j}{\mathfrak{q}_1}\right)_e = \left(\frac{x}{\mathfrak{q}_1}\right)_e$, we have $x = \nu^j y^e \pmod{p}$ and $x = \nu^j z^e \pmod{q}$ for some $x \in \mathbb{Z}_p^*$ and $y \in \mathbb{Z}_q^*$ from Theorem 2. Take $w = y \pmod{p}$ and $w = z \pmod{q}$, then we have $x = \nu^j w^e \pmod{N}$, as desired.

As a result, we obtain this theorem. □

Remark 3. According to Theorem 3, when $e = 2$, we have the well-known result: $\mathbb{J}_N \cong \{+1, -1\} \otimes \mathbb{QR}_N$, where N is a *Blum integer*, $\mathbb{J}_N = \left\{x \in \mathbb{Z}_N^* \mid \left(\frac{x}{N}\right)_2 = 1\right\}$, and $\mathbb{QR}_N = \left\{x \mid \exists y \in \mathbb{Z}_N^*, x = y^2 \pmod{N}\right\}$.

4 A New Homomorphic Public Key Cryptosystem

In this section, we present a natural extension of the `GM` scheme [18] by virtue of the *power residue symbol*.

4.1 Description

`KeyGen` (1^κ)**:** Given a security parameter κ, `KeyGen` outputs the public and private key pair as follows:

$$\mathtt{pk} = \{N, \mathrm{lcm}(e_p, e_q), y\}, \quad \mathtt{sk} = \{p, q, e_p, e_q, \mu\},$$

where $N = pq$, $e_p \mid p - 1$, $e_q \mid q - 1$, p and q are large primes, e_p and e_q are *smooth* integers, y is chosen randomly from $\mathbb{NR}_N^{(e_p, e_q)}$, and μ is a *non-degenerate primitive* (e_p, e_q)*-th root of unity* modulo N. Note that μ is generated by definition.

`Enc` $(\mathtt{pk}, m)$**:** 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

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

`Dec` $(\mathtt{sk}, c)$**:** Given the ciphertext c and the private key $\mathtt{sk} = \{p, q, e_p, e_q, \mu\}$, `Dec` first computes z_p and z_q satisfying $\left(\frac{c}{\mathfrak{p}_1}\right)_{e_p} = \zeta_{e_p}^{z_p}$ and $\left(\frac{c}{\mathfrak{q}_1}\right)_{e_q} = \zeta_{e_q}^{z_q}$ by means of Theorem 1. Then, `Dec` recovers the message $m \in \mathbb{Z}_{\mathrm{lcm}(e_p, e_q)}$ from

$$m = z_p k_p^{-1} \pmod{e_p} \quad \text{and} \quad m = z_q k_q^{-1} \pmod{e_q} \tag{1}$$

via the Chinese Remainder Theorem with non-pairwise coprime moduli, where k_p, k_q satisfying $\left(\frac{y}{\mathfrak{p}_1}\right)_{e_p} = \zeta_{e_p}^{k_p}$ and $\left(\frac{y}{\mathfrak{q}_1}\right)_{e_q} = \zeta_{e_q}^{k_q}$ respectively, can be pre-computed.

Correctness. The correctness and the additive homomorphism property of the above public key encryption can be easily obtained by the following arguments:

$$\zeta_{e_p}^{z_p} = \left(\frac{c}{\mathfrak{p}_1}\right)_{e_p} = \left(\frac{y^m r^{\mathrm{lcm}(e_p, e_q)}}{\mathfrak{p}_1}\right)_{e_p} = \left(\frac{y}{\mathfrak{p}_1}\right)_{e_p}^m = \zeta_{e_p}^{m k_p}, \text{ and}$$

$$\zeta_{e_q}^{z_q} = \left(\frac{c}{\mathfrak{q}_1}\right)_{e_q} = \left(\frac{y^m r^{\mathrm{lcm}(e_p, e_q)}}{\mathfrak{q}_1}\right)_{e_q} = \left(\frac{y}{\mathfrak{q}_1}\right)_{e_q}^m = \zeta_{e_q}^{m k_q}$$

Thus, we derive the formula (1). Since every message $m \in \mathbb{Z}_{\mathrm{lcm}(e_p,e_q)}$ corresponds to the unique pair $(\alpha, \beta) \in \mathbb{Z}_{e_p} \times \mathbb{Z}_{e_q}$ such that $m = \alpha \pmod{e_p}$ and $m = \beta \pmod{e_q}$, the decryption algorithm recovers the unique $m \in \mathbb{Z}_{\mathrm{lcm}(e_p,e_q)}$ from the formula (1). Furthermore, the scheme is homomorphic for the addition modulo $\ell = \mathrm{lcm}(e_p, e_q)$: if $c_0 = y^{m_0} r_0^\ell \pmod N$ and $c_1 = y^{m_1} r_1^\ell \pmod N$ are the ciphertexts of two messages m_0 and m_1 respectively, then $c_0 \cdot c_1 = y^{m_0+m_1}(r_0 r_1)^\ell \pmod N$ is a ciphertext of $m_0 + m_1 \pmod \ell$.

4.2 Security Analysis

The security of the above public key encryption scheme can be obtained by the similar security analysis as for the GM scheme.

Theorem 4. *Our proposed public key encryption 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 $\mathbb{ER}_N^{\mathrm{lcm}(e_p,e_q)}$ instead of choosing it from $\mathbb{NR}_N^{(e_p,e_q)}$, while this is done without noticing the adversary. In this case, the ciphertext carries no information about the message and hence our proposed public key encrypiton is IND-CPA secure. □

4.3 Parameter Selection

As described in the algorithm KeyGen, p and q are large primes, $p = 1 \pmod{e_p}$, $q = 1 \pmod{e_q}$, and both e_p and e_q only contain small prime factors. In practice, it would be preferable to choose p, q, e_p, e_q such that $0 \le \log e_p < \frac{\log p}{2}$, $0 \le \log e_q < \frac{\log q}{2}$, where p and q are generated in a similar way as in [3, Section 5.1]. The major difference is that the size of $\log e_p + \log e_q$ is bounded by $\frac{\log N}{2}$. The reason is provided by the following proposition related to Coppersmith's method for finding small roots of bivariate modular equations.

Proposition 1. *[32, Lemma 8] Let p and q be equally sized primes and $N = pq$. Let $\mathtt{d}$ be a divisor of $\varphi(N) = (p-1)(q-1)$. If there exists a positive constant $\mathtt{c}$ such that $\mathtt{d} > N^{\frac{1}{2}+\mathtt{c}}$ holds, then there exists a* PPT *algorithm that given N and $\mathtt{d}$, it factorizes N.*

Note that taking $\frac{\log N}{4} < \log e_p + \log e_q < \frac{\log N}{2}$ does not contradict the setting of *Φ-Hiding Assumption* [9] as the prime factors of $\varphi(N)$ known to the public are very small. However, $\log e_p + \log e_q$ shall not be close to $\frac{\log N}{2}$ because we don't know whether there exists an attack of mixing together Coppersmith's attack and exhaustive searches. In particular, if we take $e_p = 2^k$, $e_q = 1$ and $k > \frac{\log N}{4}$, the low-order $\frac{\log N}{4}$ bits of p is revealed to an adversary, and hence it can factorize N by implementing Coppersmith's attack [15]. 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 algorithm also requires a random integer $y \in \mathbb{Z}_N^*$ sampled from $\mathbb{NR}_N^{(e_p,e_q)}$. We can use Theorem 2 and the following fact for uniformly sampling integers in $\mathbb{NR}_N^{(e_p,e_q)}$. Note that a random integer in $\mathbb{Z}_N^*$ has a probability of exactly $\frac{\varphi(e_p)\varphi(e_q)}{e_p e_q}$ of being in the set containing all $x \in \mathbb{Z}_N^*$ whose symbols $\left(\frac{x}{\mathfrak{p}_1}\right)_{e_p}$ and $\left(\frac{x}{\mathfrak{q}_1}\right)_{e_q}$ are primitive. Let $t = \gcd(p-1, q-1)$. We first randomly choose an element $x \in \mathbb{Z}_N^*$ such that $\left(\frac{x}{\mathfrak{p}_1}\right)_t = \zeta_t^\alpha$ and $\left(\frac{x}{\mathfrak{q}_1}\right)_t = \zeta_t^\beta$ are primitive after several trials. Then, we can obtain a suitable element $y \in \{\gamma \in \mathbb{Z}_N^* \mid \left(\frac{\gamma}{\mathfrak{a}_1}\right)_t = 1\}$ from the relations $y = x^{-\left(\alpha^{-1} \bmod t\right)\beta} z^t \pmod{p}$ and $y = x \pmod{q}$, where z is a random value chosen from $\mathbb{Z}_p^*$. If $y \in \mathbb{NR}_N^{(e_p,e_q)}$, we have done; otherwise, we repeat the above steps until y is in $\mathbb{NR}_N^{(e_p,e_q)}$.

4.4 Performance and Comparisons

The prominent operation in the `JL` scheme and our proposal is the modular multiplications over $\mathbb{Z}_p^*$, if the time for searching an item in a table is negligible. For decrypting a 128-bit message, the `JL` scheme, according to the remark following [3, Algorithm 1], roughly needs

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

modular multiplications on average. On the contrary, our proposal (specially Algorithm 1 with optimization) only needs about

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

modular multiplications on average, when we set $e_p = 929^{13} > 2^{128}$ and $e_q = 1$. If N is taken as 2048 bits, the decryption of our proposal is approximately 3.5 times faster than that of the `JL` scheme. We note that both `JL` scheme and our proposal can be used to encrypt a 128- or 256-bit symmetric key in a `KEM`/`DEM` construction [30].

On the other hand, our proposal has the similar computational cost with the `CDWS` scheme in algorithms `Enc` and `Dec`. The main difference between these two schemes is the choice of y. In particular, in the `CDWS` scheme, y is from $\{y \in \mathbb{Z}_N^* \mid \exists(x, x'), y^{\frac{p-1}{e_p}} = x \pmod{p}, y^{\frac{q-1}{e_q}} = x' \pmod{q}\}$, which is contained by $\mathbb{NR}_N^{(e_p,e_q)}$. This means that we can obtain y more efficiently than the `CDWS` scheme does. Furthermore, our security proof is much easier to follow due to the choice of y.

5 More Cryptographic Designs Based on the Power Residue Symbols

5.1 Lossy Trapdoor Functions

Lossy trapdoor functions (`LTDFs`) were introduced by Peikert and Waters [28] and since then numerous applications emerge in cryptography. Informally speaking, the `LTDFs` 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 [28], one can obtain `IND-CCA` secure public key encryptions. So far, the `LTDFs` are mainly constructed from assumptions such as `DDH` [28], `LWE` [28], `QR` [17], `DCR` [17], and *Φ-Hiding* [22].

Joye and Libert constructed `LTDFs` 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 `LTDFs` and the corresponding conclusions. We follow the definition of the `LTDFs` in [3] and omit the security analysis since it proceeds in exactly the same way in [3].

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

1. Select *smooth* integers e_p and e_q such that $k < \log(\mathrm{lcm}(e_p, e_q)) < \ell_N/2$. Generate an ℓ_N-bit `RSA` modulus $N = pq$ such that $p - 1 = e_p p'$ and $q-1 = e_q q'$ for large primes p, q, p', q'. Pick at random μ a *non-degenerate primitive* (e_p, e_q)*-th root of unity* modulo N and $y \xleftarrow{R} \mathbb{NR}_N^{(e_p,e_q)}$.
2. For each $i \in \{1, \dots, m\}$, pick h_i in $\mathbb{ER}_N^{\mathrm{lcm}(e_p,e_q)}$ at random.
3. Choose $r_1, \dots, r_m \xleftarrow{R} \mathbb{Z}_{p'q'}$ and compute a $m \times m$ matrix $(Z_{i,j})_{i,j \in \{1,\dots,m\}}$ 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 and the secret key as follows:

$$\mathtt{ek} = \{N, Z\}, \quad \mathtt{sk} = \{p, q, e_p, e_q, \mu, y\}.$$

$\mathtt{LossyGen}(1^\kappa)$: 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 $1 \le i, j \le m$.
- `LossyGen` does not output the secret key `sk`.

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

$\mathtt{Inversion}(\mathtt{sk}, \boldsymbol{y})$**:** Given $\mathtt{sk} = \{p, q, e_p, e_q, \mu, y\}$ and $\boldsymbol{y} = (y_1, \ldots, y_m) \in (\mathbb{Z}_N^*)^m$, `Inversion` applies the decryption algorithm $\mathtt{Dec}(\mathtt{sk}, y_j)$ of our cryptosystem in Section 4 for each y_j to recover x_j for $j = 1$ to m. `Inversion` 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 $\mathbb{ER}_N^{\mathrm{lcm}(e_p,e_q)}$.*

Clearly, our new proposed `LTDFs` outperform that in [3] in terms of the decryption cost and ciphertext expansion. Our `LTDFs` have $\ell = n - \log(p'q') > (n - \ell_N) + \log e_p + \log e_q$ bits of lossiness. Therefore, the lossiness may also be improved as there are no known attacks against the factorization of N when $\ell_N/4 < \log e_p + \log e_q < \ell_N/2$ and $0 \leq \log e_p < \frac{\log p}{2}, 0 \leq \log e_q < \frac{\log q}{2}$.

5.2 Circular and Leakage Resilient Public Key Encryption

Brakerski and Goldwasser introduced the notion of *subgroup indistinguishability (*`SG`*) assumption* in [6, Section 3.1]. They instantiated the `SG` assumption based on the `QR` and the `DCR` assumptions and proposed a generic construction of schemes which achieve *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 another new hardness assumption named *e-th power residue assumption*. In this way, the scheme becomes much more efficient in bandwidth exploitation.

Definition 3 (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*

$$SGAdv[\mathcal{A}] = \left| \mathtt{Prob}\left[\mathcal{A}(1^\kappa, x) \mid x \xleftarrow{R} \mathbb{G}_U\right] - \mathtt{Prob}\left[\mathcal{A}(1^\kappa, x) \mid x \xleftarrow{R} \mathbb{G}_L\right]\right|$$

The subgroup indistinguishability assumption *is that for any* PPT *adversary $\mathcal{A}$ 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 symbol.* 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 $\mathbb{ER}_N^e$ and $\mathbb{J}_N^e$ be described as in Section 3.3. Then, there exists a $\nu \in \mathbb{J}_N^e \backslash \mathbb{ER}_N^e$ such that $\mathbb{J}_N^e = \langle \nu \rangle \otimes \mathbb{ER}_N^e$ from Theorem 3. The groups $\mathbb{J}_N^e$, $\langle \nu \rangle$ and $\mathbb{ER}_N^e$ have orders $\frac{\varphi(N)}{e}$, e and $\frac{\varphi(N)}{e^2}$ respectively. 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 as follows the e-th power residue (e-PR) assumption which is similar to the (e_p, e_q)-PR assumption defined previously.

Definition 4 (e-th Power Residue Assumption). *Given a security parameter κ. A* PPT *algorithm* $\mathtt{RSAgen}(\kappa)$ *generates a* smooth *integer e 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 μ a* non-degenerate primitive (e,e)-th root of unity *modulo N. The e-*PR *assumption with respect to* $\mathtt{RSAgen}(\kappa)$ *asserts that the advantage* $\mathtt{Adv}^{e\text{-PR}}_{\mathcal{A},\mathtt{RSAgen}}(\kappa)$ *defined as*

$$\left|\mathtt{Prob}\left[\mathcal{A}(N, x, e) = 1 \mid x \xleftarrow{R} \mathbb{ER}_N^e\right] - \mathtt{Prob}\left[\mathcal{A}(N, x, e) = 1 \mid x \xleftarrow{R} \mathbb{J}_N^e\right]\right|$$

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

Since there exist efficient sampling algorithms that sample a random element from $\mathbb{ER}_N^e$ and $\mathbb{J}_N^e$ according to Theorem 2 and Theorem 3, the e-PR assumption leads immediately to the instantiation of the SG assumption by setting $\mathbb{G}_U = \mathbb{J}_N^e$, $\mathbb{G}_M = \langle \nu \rangle$, $\mathbb{G}_L = \mathbb{ER}_N^e$, $h = \nu$, and $T = N \geq eN'$. The corresponding encryption scheme is presented as follows:

$\mathtt{KeyGen}(1^\kappa)$**:** 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 an integer ν as in Theorem 3, and an $\ell \in \mathbb{N}$ which is polynomial in κ. KeyGen also samples $\boldsymbol{s} \xleftarrow{R} (\mathbb{Z}_e)^\ell$ and sets the secret key $\mathtt{sk} = \boldsymbol{s}$. KeyGen then samples $\boldsymbol{g} \xleftarrow{R} (\mathbb{ER}_N^e)^\ell$ and sets

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

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

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

$\mathtt{Dec}(\mathtt{sk}, c)$**:** On inputting the secret key $\mathtt{sk} = \boldsymbol{s}$ and a ciphertext $\{c_0, \boldsymbol{c}\}$, Dec computes and returns $m = c_0 \cdot \prod_{1 \leq i \leq \ell} c_i^{s_i} \pmod{N}$.

6 Conclusion

In this paper, we have made natural extension on the GM cryptosystem by using the e-th *power residue symbol*, where e is merely required to be *smooth* in practice. Our proposals are proved to be secure under new well-defined assumptions. Furthermore, they inherit all advantages from the JL cryptosystem and LTDFs, also enhance the decryption speed as well as the efficiency of the bandwidth utilization.

When applied to the Brakerski-Goldwasser framework for building circular and leakage resilient public key encryptions, our scheme takes advantages of the e-th *power residue symbol* rather than the Jacobi symbol, thereby is more efficient in bandwidth utilization.

Acknowledgements. This work is supported in part by the National Natural Science Foundation of China (Grant No. 61632012, 61672239, 61602180), in part by Natural Science Foundation of Shanghai (Grant No. 16ZR1409200) and in part by China Postdoctoral Science Foundation (Grant No. 2017M611502).

References

1. Michel Abdalla, Fabrice Ben Hamouda, and David Pointcheval. Tighter reductions for forward-secure signature schemes. In Public-Key Cryptography - PKC 2013, volume 7778 of LNCS, pages 292–311. Springer, 2013. https://doi.org/10.1007/978-3-642-36362-7_19
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. https://doi.org/10.1007/s00145-016-9229-5
4. Lenore Blum, Manuel Blum, and Mike Shub. A simple unpredictable pseudorandom number generator. SIAM Journal on computing, 15(2):364–383, 1986. https://doi.org/10.1137/0215025
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. https://doi.org/10.1007/3-540-39568-7_23
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. https://doi.org/10.1007/978-3-642-14623-7_1
7. Eric Brier, Houda Ferradi, Marc Joye, and David Naccache. New number-theoretic cryptographic primitives. IACR Cryptology ePrint Archive, 2019:484, 2019.
8. Eric Brier and David Naccache. The thirteenth power residue symbol. IACR Cryptology ePrint Archive, 2019:1176, 2019.

9. 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. https://doi.org/10.1007/3-540-48910-x_28
10. 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.
11. 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.
12. Zhenfu Cao, Xiaolei Dong, Licheng Wang, and Jun Shao. More efficient cryptosystems from k-th power residues. IACR Cryptology ePrint Archive, 2013:569, 2013.
13. Perlas C. Caranay and Renate Scheidler. An efficient seventh power residue symbol algorithm. International Journal of Number Theory, 6(08):1831–1853, 2010. https://doi.org/10.1142/s1793042110003770
14. 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. https://doi.org/10.1109/sfcs.1985.2
15. Don Coppersmith. Small solutions to polynomial equations, and low exponent RSA vulnerabilities. Journal of cryptology, 10(4):233–260, 1997. https://doi.org/10.1007/s001459900030
16. Koen de Boer. Computing the power residue symbol. Master's thesis. Nijmegen, Radboud University. `www.koendeboer.com`, 2016.
17. 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. https://doi.org/10.1007/s00145-011-9112-3
18. Shafi Goldwasser and Silvio Micali. Probabilistic encryption. Journal of computer and system sciences, 28(2):270–299, 1984.
19. Kenneth Ireland and Michael Rosen. A classical introduction to modern number theory, volume 84. Springer Science & Business Media, 2013.
20. Marc Joye. Evaluating octic residue symbols. IACR Cryptology ePrint Archive, 2019:1196, 2019.
21. Marc Joye, Oleksandra Lapiha, Ky Nguyen, and David Naccache. The eleventh power residue symbol. IACR Cryptology ePrint Archive, 2019:870, 2019.
22. Eike Kiltz, Adam O'Neill, and Adam Smith. Instantiability of RSA-OAEP under chosen-plaintext attack. Journal of Cryptology, 30(3):889–919, 2017. https://doi.org/10.1007/s00145-016-9238-4
23. Franz Lemmermeyer. Reciprocity laws: from Euler to Eisenstein. Springer Science & Business Media, 2013.
24. David Naccache and Jacques Stern. A new public key cryptosystem based on higher residues. In CCS 1998, pages 59–66. ACM, 1998. https://doi.org/10.1145/288090.288106

25. Jürgen Neukirch. Algebraic number theory, volume 322. Springer Science & Business Media, 2013.
26. 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. https://doi.org/10.1007/bfb0054135
27. 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. https://doi.org/10.1007/bfb0054135
28. Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. SIAM Journal on Computing, 40(6):1803–1844, 2011. https://doi.org/10.1145/1374376.1374406
29. 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. https://doi.org/10.1109/tit.1978.1055817
30. 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. https://doi.org/10.1007/3-540-45539-6_19
31. Douglas Squirrel. Computing reciprocity symbols in number fields, 1997. Undergraduate thesis, Reed College.
32. 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. https://doi.org/10.1007/978-3-662-53008-5_1