New Public Key Encryption with Equality Test Based on non-Abelian Factorization Problems

Huijun Zhu, Licheng Wang, Shuming Qiu and Xinxin Niu

State Key Laboratory of Networking and Switching Technology

Beijing University of Posts and Telecommunications, 100876 Beijing, P.R. China

[e-mail: wanglc2012@126.com]

*Corresponding author: Licheng Wang

Received June 11, 2017; revised August 31, 2017; accepted September 15, 2017; published February 28, 2018

Abstract

In this paper, we present a new public key encryption scheme with equality test (PKEwET). Compared to other PKEwET schemes, we find that its security can be improved since the proposed scheme is based on non-Abelian factorization problems. To our knowledge, it is the first scheme regarding equality test that can resist quantum algorithm attacks. We show that our scheme is one-way against chosen-ciphertext attacks in the case that the computational Diffie-Hellman problem is hard for a Type-I adversary. It is indistinguishable against chosen-ciphertext attacks in the case that the Decisional Diffie-Hellman problem is hard in the random oracle model for a Type-II adversary. To conclude the paper, we demonstrate that our scheme is more efficient.

Keywords: non-Abelian, public key encryption, equality test, flexible authorization; quantum algorithm

http://doi.org/10.3837/tiis.2018.02.013 **ISSN : 1976-7277**

1. Introduction

In the current cloud computing era, cloud service providers offer near-infinite storage space and computing power for users. Many users outsource their data to cloud servers. The problems of computing on large datasets that are stored on an untrusted server and the techniques for securely outsourcing such data face great challenges. Benabbas et al. presented the first practical verifiable computation scheme and proposed the notion of verifiable database (VDB) [29]. Subsequently, Chen et al. proposed a new secure outsourcing algorithm for (variable-exponent, variable-base) exponentiation modulo a prime in the two untrusted program model [30]. Moreover, Chen et al. proposed a new VDB framework from vector commitment based on the idea of commitment binding [28].

As cloud computing technology improves, traditional cryptographic systems encounter new challenges. Some new cryptographic primitives have been proposed. Public Key Encryption with keyword Search (PKEwKS) was proposed by Boneh et al. [7] which has a good prospects in cloud computing and many cryptographers have carried out relevant research on the topic [8-14]. Later, another related cryptographic primitive called Deterministic encryption (DE) was proposed by Bellare et al. in 2007 [4]. Bellare et al. [5] and Boldyreva et al. [6] made further efforts to research this cryptosystem.

Recently, a new cryptographic primitive called Public Key Encryption with Equality Test (PKEwET) was proposed by Yang et al. at CT-RSA 2010 [15]. Yang used pairing to design his scheme and its security was based on the Computational Diffie-Hellman (CDH) assumption in bilinear groups. Tang et al. presented an enhanced PKEwET scheme with a fine-grained authorization [16]. His scheme is one-way against chosen-ciphertext attacks (OW-CCA) secure against Type-I adversary under the CDH assumption, and it is indistinguishable against chosen-ciphertext attacks (IND-CCA) secure against Type-II adversary under an extended Decisional bilinear Diffie-Hellman (DBDH) assumption. Later, Tang made some further improvements in [17] and [18]. In [17], Tang proposed a new primitive that supports equality test with user-specified authorization. A proxy can be authorized to do the equality test by two users who hold the public and private key pairs. But this scheme's security is based on the CDH assumption. In [18], Tang extended [16] to a two-proxy setting whose security is based on the CDH assumption and the decisional Diffie-Hellman (DDH) assumption. In 2014, Ma et al. presented a special public key encryption scheme with a delegated equality test (PKE-DET) [19]. This cryptosystem enabled only a delegated party to perform the equality test in a practical multi-user setting. Ma also continued his work by adding four types of authorization policies into the PKEwET scheme [1]. Moreover, Ma proposed an identity-based encryption scheme with equality test (IBEwET) scheme by combining PKEwET with identity-based encryption [25]. The scheme

is one-way chosen-ciphertext security against a chosen identity attack (OW-ID-CCA). Its security is based on the Bilinear Diffie–Hellman (BDH) assumption in the random oracle model.

As far as we know, the securities of the above PKEwET schemes are based on the assumption of the discrete logarithm problem (DLP). However, Shor's quantum algorithm reveals that the DLP can be solved in polynomial time [2]. As a result, these existing schemes will be insecure in the quantum computing era. Therefore, it is necessary to study the new PKEwET scheme to bolster it against quantum algorithm attack. Cryptography based on the non-commutative algebra structure was proposed by Magyarik and Wagner in 1984 [21]. After more than ten years of development, many cryptographers concentrated their efforts on non-commutative cryptosystems and proposed a number of cryptosystems based on non-commutative algebraic structures (Braid group [22], Thompson group [23], Artin group [24], matrices of group ring [27] and so on.). By observing this research, we know that their cryptographic assumptions are essentially word (decision) problem (WP), conjugacy search problem (CSP), group factorization problem (GFP), factorization search problem (FSP), membership problem [20] and so on. Recently, Baba et al. proposed the non-Abelian factorization problem in 2011 [3]. Later, Gu et al. presented two novel public key encryption schemes based on the non-Abelian factorization problems [26]. Moreover, Gu et al. proved that it was no use to employ Shor's quantum algorithm to solve the group factorization problem when the underlying group G is non-Abelian. So far, the non-Abelian factorization problem has no efficient quantum algorithm.

1.1 Our Contribution

For all the previously stated reason, our purpose is to design a new PKEwET scheme that is secure against quantum algorithms. We concentrate on non-commutative cryptography and find that we can use the non-Abelian Group to design the new scheme.

In our proposed scheme, we mainly use the non-Abelian group to design the scheme. Moreover, we have improved the scheme in [1]. We provide an analysis to illustrate that our scheme is more efficient than the scheme in [1]. Moreover, our proposed cryptosystem is proved to be OW-CCA secure in the case that the computational Diffie-Hellman ($CDH_{g,f}^{G}$) problem is hard in the random oracle model for Type-I adversary and is IND-CCA secure in the case that the Decisional Diffie-Hellman ($DDH_{g,f}^{G}$) problem is hard in the random oracle model for Type-II adversary.

1.2 Organization

The remainder of this paper is organized as follows. In Section 2, basic knowledge, cryptographic assumptions, and the related security model are presented. In Section 3, we propose our scheme and four types of authorization. In Section 4, we provide a security proof of our scheme. The performances analysis is given in Section 5. Finally, concluding remarks are presented in Section 6.

2. Preliminaries

In this section, we introduce some basic knowledge including the cryptographic assumption and security model that are used in this paper.

2.1 Cryptographic assumptions

Before illustrating our scheme, we introduce some assumptions used in this paper, on which the security of our scheme is based.

Definition 2.1 Factorization problem ($FP_{g,f}^{G}$): Given a finite group G and choosing two random elements $g, f \in G$ that satisfy $\langle g \rangle \bigcap \langle f \rangle = \{e\}$ and $gf \neq fg$, the factorization problem is to split a given element $g_0 \in G$ into a pair $(g_1, f_1) \in G \times G$, such that $g_1 \in \langle g \rangle$ and $f_1 \in \langle f \rangle$.

Definition 2.2 Computational Diffie-Hellman problem ($CDH_{g,f}^{G}$) : Given a finite group G and a quintuple $\left(g, f, g^{a_1} f^{b_1}, g^{a_2} f^{b_2}\right) \in G$ where $a_1, b_1, a_2, b_2 \in Z_q$, the objective is to compute $g^{a_1+a_2} f^{b_1+b_2}$. Here, $g, f \in G$, satisfies $\langle g \rangle \bigcap \langle f \rangle = \{e\}$ and $gf \neq fg$.

Definition 2.3 Decisional Diffie-Hellman problem($DDH_{g,f}^{G}$) : Given a finite group G and a quintuple $\left(g, f, g^{a_1} f^{b_1}, g^{a_2} f^{b_2}, g^{a_3} f^{b3}\right) \in G$ where $a_1, b_1, a_2, b_2, a_3, b_3 \in Z_q$, the

objective is to decide whether $g^{a_3} f^{b_3} = g^{a_1+a_2} f^{b_1+b_2}$ or not. Here, $g, f \in G$, satisfies $\langle g \rangle \cap \langle f \rangle = \{e\}$ and $gf \neq fg$.

2.2 Security Models

In this subsection, we review the security model of PKEwET-FA , which was defined in [1].

The security model includes six algorithms: **Setup, KeyGen, Encrypt, Decrypt, Authorization-α ,** and **Test-α** (α =1,2,3,4)}. Suppose that the system has a unique index i for user u_i. The **Setup** algorithm establishes system parameters sp. The **KeyGen** algorithm generates the public key and private key for user u_i. The **Encrypt** algorithm outputs a ciphertext C for a message M and the public key. The **Decrypt** algorithm outputs a message M or $\perp$ using a private key. The **Authorization** algorithm generates the trapdoor with the private key sk. The **Test** algorithm takes two ciphertexts, the trapdoors as inputs and outputs 1 when they are the same message or 0 otherwise. Because the Type-4 authorization is a combination of Type-1 and Type-2 authorization, we leave out Type-4 authorization queries for simplicity and allow only Type-α (α = 1, 2, 3) authorization queries to the adversary in the security games.

We describe two types of adversaries as follows:

(1) Type-I adversary: For Type-I adversary, he/she can not obtain any information except for the test information with Type-α trapdoor information.

(2) Type-II adversary: For Type-II adversary, he/she can not judge whether the challenge ciphertext CT^* is encrypted by which message without Type-α trapdoor information.

First we define OW-CCA security for Type-α (α = 1, 2, 3) authorization against Type-I adversary as shown in **Table 1**.

In the security models of the Type-I adversary, the adversary A_1 submits a target t which he/she wants to challenge before the game.

Here, $O_1(i) = KeyGen(i, sp)$, $O_2(i, CT_i) = dec(sk_i, CT_i)$, $O_3(i, \cdot) = ET - Auth(sk_i, \cdot)$, $O_4(i) = O_1(i)$ or $O_4(i) = \perp$, and the condition is that $i \neq t$,

$O_5(i, CT_i) = O_2(i, CT_i)$ or $O_5(i, CT_i) = \perp$, and the condition is that $CT_i \neq CT^*$, $O_6 = O_3$.

Table 1. The security models for the Type-I adversary

Experiment Exp_{S,A_1}^{OW-CCA}
$(pk, sk) \leftarrow KeyGen(1^k)$
$M \leftarrow A_1^{O_1, O_2, O_3}(i, pk)$
$CT^* \leftarrow Enc(pk, M)$
$M^* \leftarrow A_2^{O_4, O_5, O_6}(i, pk)$
If $M = M^*$, then return 1;
Else return 0.

Second we define IND-CCA security for Type-α (α = 1, 2, 3) authorization against Type-II adversary as in **Table 2**.

In the security models of Type-II adversary, the adversary A_2 submits a target t which he/she wants to challenge before the game.

Here, $O_1(i) = KeyGen(i, sp)$, $O_2(i, CT_i) = dec(sk_i, CT_i)$, $O_3(i, \cdot) = ET - Auth(sk_i, \cdot)$, $O_4(i) = O_1(i)$ or $O_4(i) = \perp$, and the condition is that $i \neq t$; $O_5(i, CT_i) = O_2(i, CT_i)$ or $O_5(i, CT_i) = \perp$, the condition is that $CT_i \neq CT^*$, $O_6 = O_3$. For O_6, when $\alpha = 1$, then $i \neq t$; when $\alpha = 2$ or $\alpha = 3$, then $CT_i \neq CT^*$.

Table 2. The security models for the Type-II adversary

Experiment $Exp_{S,A_2}^{IND-CCA}$
$(pk, sk) \leftarrow KeyGen(1^k)$
$M \leftarrow A_1^{O_1,\ O_2,\ O_3}(i, pk)$
$b^* \xleftarrow{R} \{0,1\}$
$CT^* \leftarrow Enc(pk, M_b)$
$b^* \leftarrow A_2^{O_4, O_5, O_6}(i, pk)$
If $b^* = b$, then return 1;
Else return 0.

3. The Proposed Scheme

In this section, we provide the description of our proposed scheme.

Setup: It takes a system security parameter λ as input and outputs the system parameters sp as follows. Let G be any finite group with identity e and $G = \Theta(2^{2\lambda})$. Let $g, f \in G$ be two non-commuting elements with orders $l, \kappa = \Theta(2^{2\lambda})$ respectively. We demand that $\langle g \rangle \cap \langle f \rangle = \{e\}$. Choose hash functions $H_1 : G \to \{0,1\}^{\lambda+2k}$, $H_2 : G^3 \times \{0,1\}^{\lambda+2k} \to \{0,1\}^{4k}$ and H_3, H_4, H_5, $H_6 : \{0,1\}^{\lambda} \to Z_q$, where k is the length of elements in Z_q. Then, output $sp = (G, g, f, H_1, H_2, H_3, H_4, H_5, H_6)$.

KeyGen (sp): It takes system parameters sp as input and outputs:

$(pk, sk) = ((U = g^{\alpha_1} f^{\beta_1}, V = g^{\alpha_2} f^{\beta_2}), (g^{\alpha_1}, f^{\beta_1}, g^{\alpha_2}, f^{\beta_2}))$

where α_1, α_2, β_1, $\beta_2 \in Z_q$ are selected randomly.

Encrypt (M, pk): This function encrypts a message $M \in \{0,1\}^{\lambda}$ with the public key pk as follows.

(1) Constructs a straight line $f(x)$ and generates two points as follows:

1) Creates two points $p_1 = (H_3(M), H_4(M))$ and $p_2 = (H_5(M), H_6(M))$;

2) Uses the two points, p_1 and p_2, to construct a straight line $f(x) = ax + b$;

3) Chooses $x_1, x_2 \in \{0,1\}^k$ randomly and obtain two points (x_1, y_1), (x_2, y_2) on $f(x)$, where $x_1 \neq 0$ or $x_2 \neq 0$.

(2) Chooses $r_1, r_2, r_3, r_4 \in Z_q^*$ randomly and computes:

$C_1 = g^{r_1} f^{r_2}$, $C_2 = g^{r_3} f^{r_4}$, $C_3 = M \| r_3 \| r_4 \oplus H_1(g^{r_1} U f^{r_2})$,

$C_4 = x_1 \| x_2 \| y_1 \| y_2 \oplus H_2(g^{r_3} V f^{r_4}, C_1, C_2, C_3)$

The ciphertext is $CT = (C_1, C_2, C_3, C_4)$.

Decrypt (CT, sk): To decrypt the ciphertext $CT = (C_1, C_2, C_3, C_4)$ with the private key sk, this algorithm performs the following calculations:

$M \| r_3 \| r_4 = C_3 \oplus H_1(g^{\alpha_1} C_1 f^{\beta_1})$, $x_1 \| x_2 \| y_1 \| y_2 = C_4 \oplus H_2(g^{\alpha_2} C_2 f^{\beta_2}, C_1, C_2, C_3)$

Constructs $f(x)$ as in **Encrypt** (1), and then checks the following:

$f(x_1) = y_1$, $f(x_2) = y_2$ and $C_2 = g^{r_3} f^{r_4}$

When all equations hold, then M is output. Otherwise, an error message $\perp$ is output.

4 types of authorization algorithm and test algorithm:

Assume that there are two users A and B with the ciphertext $CT_i = (C_{i,1}, C_{i,2}, C_{i,3}, C_{i,4})$ (resp. $CT_j = (C_{j,1}, C_{j,2}, C_{j,3}, C_{j,4})$). Let $r_{i,1}, r_{i,2}, r_{i,3}, r_{i,4}$ (resp. $r_{j,1}, r_{j,2}, r_{j,3}, r_{j,4}$) be random numbers that are used in CT_i (resp. CT_j).

(1) **Type-1 Authorization and Test**:

· **Auth-1**: For user A, it takes a part of private key $(g^{\alpha_{i,2}}, f^{\beta_{i,2}})$ as input and outputs a trapdoor $td_{1,i} = (g^{\alpha_{i,2}}, f^{\beta_{i,2}})$.

For user B, it takes a part of private key $(g^{\alpha_{j,2}}, f^{\beta_{j,2}})$ as input and outputs a trapdoor $td_{1,j} = (g^{\alpha_{j,2}}, f^{\beta_{j,2}})$.

· **Test-1**($CT_i, td_{1,i}, CT_j, td_{1,j}$): To test CT_i and CT_j with $td_{1,i}$ and $td_{1,j}$, the function performs the following operations:

$$x_{i,1} \| x_{i,2} \| y_{i,1} \| y_{i,2} = C_{i,4} \oplus H_2(g^{\alpha_{i,2}} C_{i,2} f^{\beta_{i,2}}, C_{i,1}, C_{i,2}, C_{i,3})$$

$$x_{j,1} \| x_{j,2} \| y_{j,1} \| y_{j,2} = C_{j,4} \oplus H_2(g^{\alpha_{j,2}} C_{j,2} f^{\beta_{j,2}}, C_{j,1}, C_{j,2}, C_{j,3})$$

and recovers

$$f_i(x) = (q, (x_{i,1}, y_{i,1}), (x_{i,2}, y_{i,2})), \quad f_j(x) = (q, (x_{j,1}, y_{j,1}), (x_{j,2}, y_{j,2}))$$

Finally, it outputs 1 when $f_i(x) = f_j(x)$ holds; otherwise, it outputs 0.

(2) **Type-2 Authorization and Test**:

· **Auth-2**: For user A, it takes a part of the private key $(g^{\alpha_{i,2}}, f^{\beta_{i,2}})$ as input and outputs a trapdoor $td_{2,i,CT_i} = H_2(g^{\alpha_{i,2}} C_{i,2} f^{\beta_{i,2}}, C_{i,1}, C_{i,2}, C_{i,3})$.

For user B, it takes a part of the private key $(g^{\alpha_{j,2}}, f^{\beta_{j,2}})$ as input and outputs a trapdoor $td_{2,\mathrm{j},CT_j} = H_2(g^{\alpha_{j,2}} C_{j,2} f^{\beta_{j,2}}, C_{j,1}, C_{j,2}, C_{j,3})$.

· **Test-2**($CT_i, td_{2,i,CT_i}, CT_j, td_{2,j,CT_j}$): To test CT_i and CT_j with td_{2,i,CT_i} and td_{2,i,CT_j}, the function performs the following operation:

$x_{i,1} \| x_{i,2} \| y_{i,1} \| y_{i,2} = C_{i,4} \oplus td_{2,i,CT_i}$, $x_{j,1} \| x_{j,2} \| y_{j,1} \| y_{j,2} = C_{j,4} \oplus td_{2,j,CT_j}$

and recovers

$f_i(x) = (q, (x_{i,1}, y_{i,1}), (x_{i,2}, y_{i,2}))$, $f_j(x) = (q, (x_{j,1}, y_{j,1}), (x_{j,2}, y_{j,2}))$

Finally, it outputs 1 when $f_i(x) = f_j(x)$ holds; otherwise, it outputs 0.

(3) **Type-3 Authorization and Test**:

Notice: Let x be a bit-string. Then, $[x]_a^b$ defines a substring of x, from a to b.

· **Auth-3**: For user A, it takes the straight line $f_i(x) = a_i x_i + b_i$, $x_{i,1}, x_{i,2}, r_{i,3}, r_{i,4}$, which were used in the encryption algorithm and a part of the private key $(g^{\alpha_{i,2}}, f^{\beta_{i,2}})$, then it outputs a trapdoor

$td_{3,i,CT_i,\ j,CT_j} = (W_i, Z_i) = ([H_2(g^{\alpha_{i,2}} C_{i,2} f^{\beta_{i,2}}, C_{i,1}, C_{i,2}, C_{i,3})]_{2l}^{4l-1}, g^{a_i x_{i,1}} w_i f^{a_i x_{i,2}}$, where

$w_i = g^{r_{i,3}} C_{j,2} f^{r_{i,4}}$

For user B, it takes the straight line $f_j(x) = a_j x_j + b_j$, $x_{j,1}, x_{j,2}, r_{j,3}, r_{j,4}$, which were used in encryption algorithm and a part of the private key $(g^{\alpha_{j,2}}, f^{\beta_{j,2}})$, then it outputs a trapdoor

$td_{3,j,CT_j,\ i,CT_i} = (W_j, Z_j) = ([H_2(g^{\alpha_{j,2}} C_{j,2} f^{\beta_{j,2}}, C_{j,1}, C_{j,2}, C_{j,3})]_{2l}^{4l-1}, g^{a_j x_{j,1}} w_j f^{a_j x_{j,2}}$, where

$w_j = g^{r_{j,3}} C_{i,2} f^{r_{j,4}}$.

· **Test-3**($CT_i, td_{3,i,CT_i,j,CT_j}, CT_j, td_{3,j,CT_j,i,CT_i}$): This algorithm performs as follows:

$y_{i,1} \| y_{i,2} = [C_{i,4}]_{2l}^{4l-1} \oplus W_i$, $y_{j,1} \| y_{j,2} = [C_{j,4}]_{2l}^{4l-1} \oplus W_j$

Then, it computes $\delta_i = g^{-y_{i,1}} Z_i f^{-y_{i,2}}, \delta_j = g^{-y_{j,1}} Z_j f^{-y_{j,2}}$.

Finally, it outputs 1 when $\delta_i = \delta_j$ holds; otherwise, it outputs 0.

(4) **Type-4 Authorization and Test**:

· **Auth-4**: For user A, this function takes a part of the private key $(g^{\alpha_{i,2}}, f^{\beta_{i,2}})$ as input and outputs a trapdoor $td_{4,i,CT_i} = td_{2,i,CT_i} = H_2(g^{\alpha_{i,2}} C_{i,2} f^{\beta_{i,2}}, C_{i,1}, C_{i,2}, C_{i,3})$.

For user B, it takes a part of the private key $(g^{\alpha_{j,2}}, f^{\beta_{j,2}})$ as input and outputs a trapdoor $td_{4,j} = td_{1,j} = (g^{\alpha_{j,2}}, f^{\beta_{j,2}})$.

· **Test-4**$(CT_i, td_{4,i,CT_i}, CT_j, td_{4,j})$: To test CT_i and CT_j with td_{2,i,CT_i} and td_{2,i,CT_j}, it runs as follows:

$$x_{i,1} \| x_{i,2} \| y_{i,1} \| y_{i,2} = C_{i,4} \oplus td_{2,i,CT_i}$$

$$x_{j,1} \| x_{j,2} \| y_{j,1} \| y_{j,2} = C_{j,4} \oplus H_2(g^{\alpha_{j,2}} C_{j,2} f^{\beta_{j,2}}, C_{j,1}, C_{j,2}, C_{j,3})$$

and recovers

$$f_i(x) = (q, (x_{i,1}, y_{i,1}), (x_{i,2}, y_{i,2})), \quad f_j(x) = (q, (x_{j,1}, y_{j,1}), (x_{j,2}, y_{j,2}))$$

Finally, it outputs 1 when $f_i(x) = f_j(x)$ holds; otherwise, it outputs 0.

4. Security analysis

In this section, we provide proofs of the following theorems.

Theorem 4.1 The proposed cryptosystem is OW-CCA secure in the case that the $CDH_{g,f}^{G}$ problem is hard in the random oracle model for the Type-I adversary.

Proof. Suppose that there is a Type-I adversary A_1 that can break the OW-CCA security of the above cryptosystem in polynomial-time. Now, we construct an algorithm γ to solve the $CDH_{g,f}^{G}$ problem in G with non-negligible probability. Given a 4-tuple $\left(g, f, g^{a_1} f^{b_1}, g^{a_2} f^{b_2}\right) \in G$, the object of algorithm γ is to compute $g^{a_3} f^{b_3} = g^{a_1+a_2} f^{b_1+b_2}$. The game between γ and A_1 runs as follows. During the

interaction, γ will store its responses to all queries and hold the H_1 watch list.

Init. The adversary A_1 submits a target t that he/she wants to challenge before the game.

Setup. The simulator generates the system parameters sp with a security parameter λ as in the algorithm Setup. Then it generates n-pairs of public/private keys as follows:

For each $1 \le i \le n\,(i \ne t)$, it chooses a random $\alpha_{i,1}, \alpha_{i,2}, \beta_{i,1}, \beta_{i,2} \in Z_q^*$, and computes $U_i = g^{\alpha_{i,1}} f^{\beta_{i,1}}, V_i = g^{\alpha_{i,2}} f^{\beta_{i,2}}$. The public key is (U_i, V_i), and the private key is $(g^{\alpha_{i,1}}, f^{\beta_{i,1}}, g^{\alpha_{i,2}}, f^{\beta_{i,2}})$. If $i = t$, let $U_t = g^{a_1} f^{b_1}, V_t = g^{a_{t,2}} f^{b_{t,2}}$, while γ uses only the private key $V_t = g^{a_{t,2}} f^{b_{t,2}}$. Note that γ has no information about the private key of U_t. Then, the simulator gives the public keys to A_1.

Phase 1. After receiving the public key, A_1 can perform the decryption key queries, decryption queries, authorization queries, and random oracle H_1 queries. The game between A_1 and γ runs as follows:

* O_{H_1}-queries: A_1 can perform the O_{H_1}-queries adaptively. To respond to the queries, γ holds a list of tuples- $H_1 - (\sigma_i, \tau_i)$ and responds as follows:

· When σ_i is already in H_1 in the tuple (σ_i, τ_i), then γ outputs $H_1(\sigma_i) = \tau_i$ as the answer.

· Otherwise, γ chooses $\tau_i \in \{0,1\}^{\lambda+2k}$ randomly, stores a new tuple (σ_i, τ_i) into the H_1-list and outputs $H_1(\sigma_i) = \tau_i$ as the answer.

∗Decryption key queries: A_1 can perform the decryption key queries adaptively and γ responds to A_1 with sk_i, which is generated during the setup ($i \ne t$).

∗Decryption queries: Let $CT_i = (C_{i,1}, C_{i,2}, C_{i,3}, C_{i,4})$.

· If $i \neq t$, γ calls the decryption key queries for sk_i. Then, it runs decryption algorithm and outputs the answer to A_1.

· Otherwise, γ answers as follows:

If each tuple (σ_i, τ_i) is in the H_1-list, γ calculates:

1) $M_i \| r_{i,3} \| r_{i,4} = C_{i,3} \oplus H_1(\sigma_i)$

$x_{i,1} \| x_{i,2} \| y_{i,1} \| y_{i,2} = C_{i,4} \oplus H_2(g^{\alpha_{i,2}} C_{i,2} f^{\beta_{i,2}}, C_{i,1}, C_{i,2}, C_{i,3})$

2) It generates $p_{i,1}, p_{i,2}$ as in the Encrypt algorithm by using M_i;

3) It constructs a straight line $f_i(x)$, by using the two points $p_{i,1}, p_{i,2}$ generated during step 2);

4) γ returns M_i, if $f_i(x_{i,1}) = y_{i,1}$, $f_i(x_{i,2}) = y_{i,2}$ and $C_{i,2} = g^{r_{i,3}} f^{r_{i,4}}$ all hold.

Otherwise, it outputs $\perp$ to A_1.

∗Authorization queries: For the Type-α ($\alpha = 1,2,3$) authorization, it runs as follows:

1) When $\alpha = 1$, with input i, γ runs the **Auth-1** algorithm and answers A_1 with $td_{1,i} = (g^{\alpha_{i,2}}, f^{\beta_{i,2}})$;

2) When $\alpha = 2$, with input (i, CT_i), γ runs the **Auth-2** algorithm and answers A_1 with $td_{2,i,CT_i} = H_2(g^{\alpha_{i,2}} C_{i,2} f^{\beta_{i,2}}, C_{i,1}, C_{i,2}, C_{i,3})$;

3) When $\alpha = 3$, with input (i, CT_i, j, CT_j), γ runs the **Auth-3** algorithm and answers A_1 with $td_{3,i,CT_i,\ j,CT_j} = (W_i, Z_i) = ([H_2(g^{\alpha_{i,2}} C_{i,2} f^{\beta_{i,2}}, C_{i,1}, C_{i,2}, C_{i,3})]_{2l}^{4l-1}, g^{a_i x_{i,1}} w_i f^{a_i x_{i,2}}$, where $w_i = g^{r_{i,3}} C_{j,2} f^{r_{i,4}}$.

Challenge. γ selects a message M_t randomly which is challenged and chooses $r_{t,1}, r_{t,2} \in Z_q^*$. Then, it outputs $CT_t = (C_{t,1}, C_{t,2}, C_{t,3}, C_{t,4})$ as follows:

$$C_{t,1} = g^{a_2} f^{b_2},\ C_{t,2} = g^{r_{t,1}} f^{r_{t,2}},\ C_{t,3} = M \| r_{t,1} \| r_{t,2} \oplus H_1(g^{a_3} f^{b_3}),$$

$$C_{t,4} = x_1 \| x_2 \| y_1 \| y_2 \oplus H_2(g^{r_{t,1}} V_t f^{r_{t,2}}, C_{t,1}, C_{t,2}, C_{t,3})$$

Finally, it gives CT_t to A_1 as the challenge ciphertext.

Phase 2. A_1 continues performing queries as in Phase 1. The restrictions are as follows:

· In the decryption key queries, $i \neq t$;

· In the decryption queries, CT_t is not allowed.

Guess. Finally A_1 submits a guess M^*. If $M^* = M_t$ holds, γ outputs $g^{a_3} f^{b_3} = g^{a_1+a_2} f^{b_1+b_2}$ as the answer.

Theorem 4.2 The proposed cryptosystem is IND-CCA secure in the case when the $DDH_{g,f}^{G}$ problem is hard in the random oracle model for Type-II adversary.

proof. Suppose that there is a Type−II adversary A_2 that can break the IND-CCA security of the above cryptosystem in polynomial-time. Now, we construct an algorithm γ to solve the $DDH_{g,f}^{G}$ problem in G with non-negligible probability. Given a 5-tuple $\left(g, f, g^{a_1} f^{b_1}, g^{a_2} f^{b_2}, g^{a_3} f^{b3}\right) \in G$, the object of algorithm γ is to test whether $g^{a_3} f^{b_3} = g^{a_1+a_2} f^{b_1+b_2}$ holds. The game between γ and A_2 runs as follows. During the interaction, γ will store its responses to all queries and hold the H_1 watch list.

Init. The adversary A_2 submits a target t that he/she wants to challenge before the game.

Setup. The simulator generates the system parameters sp with a security parameter λ as in the algorithm setup. Then, it generates n-pairs of public/private keys as follows:

For each $1 \le i \le n\,(i \ne t)$, it chooses a random $\alpha_{i,1}, \alpha_{i,2}, \beta_{i,1}, \beta_{i,2} \in Z_q^*$, and computes $U_i = g^{\alpha_{i,1}} f^{\beta_{i,1}}, V_i = g^{\alpha_{i,2}} f^{\beta_{i,2}}$. The public key is (U_i, V_i), and the private key is $(g^{\alpha_{i,1}}, f^{\beta_{i,1}}, g^{\alpha_{i,2}}, f^{\beta_{i,2}})$. If $i = t$, let $U_t = g^{a_1} f^{b_1}, V_t = g^{a_{t,2}} f^{b_{t,2}}$, while γ only has the private key $V_t = g^{a_{t,2}} f^{b_{t,2}}$. Notice that γ has no information about the private key U_t. Then, it gives the public keys to A_2.

Phase 1. After receiving the public key, A_2 can perform decryption key queries, decryption queries, authorization queries, and random oracle H_1 queries. The game between A_2 and γ runs as follows:

* O_{H_1}-queries: A_2 can perform the O_{H_1}-queries adaptively. To respond to the queries, γ holds a list of tuples- $H_1 - (\sigma_i, \tau_i)$ and responds as follows:

· If there is σ_i already in H_1 in the tuple (σ_i, τ_i), then, γ outputs $H_1(\sigma_i) = \tau_i$ as the answer.

· Otherwise, γ chooses $\tau_i \in \{0,1\}^{\lambda+2k}$ randomly, stores a new tuple (σ_i, τ_i) into H_1-list and outputs $H_1(\sigma_i) = \tau_i$ as the answer.

∗Decryption key queries: A_2 can perform decryption key queries adaptively and γ responds to A_2 with sk_i which is generated in the Setup ($i \ne t$).

∗Decryption queries: Let $CT_i = (C_{i,1}, C_{i,2}, C_{i,3}, C_{i,4})$.

· If $i \ne t$, γ calls the decryption key queries for sk_i. Then it runs decryption algorithm and outputs the answer to A_2.

· Otherwise, γ answers as follows:

If each tuple (σ_i, τ_i) is in H_1-list, γ calculates:

1) $M_i \| r_{i,3} \| r_{i,4} = C_{i,3} \oplus H_1(\sigma_i)$

$x_{i,1} \| x_{i,2} \| y_{i,1} \| y_{i,2} = C_{i,4} \oplus H_2(g^{\alpha_{i,2}} C_{i,2} f^{\beta_{i,2}}, C_{i,1}, C_{i,2}, C_{i,3})$

2) It generates $p_{i,1}, p_{i,2}$ as in the Encrypt algorithm by using M_i;

3) It constructs a straight line $f_i(x)$ by using the two points $p_{i,1}, p_{i,2}$ generated in the step 2;

4) γ returns M_i, if $f_i(x_{i,1}) = y_{i,1}$, $f_i(x_{i,2}) = y_{i,2}$ and $C_{i,2} = g^{r_{i,3}} f^{r_{i,4}}$ all hold.

Otherwise, it outputs $\perp$ to A_2.

∗Authorization queries: For Type-α ($\alpha = 1,2,3$) authorization, it runs as follows:

1) When $\alpha = 1$, with input i, γ runs the **Auth-1** algorithm and answers A_2 with $td_{1,i} = (g^{\alpha_{i,2}}, f^{\beta_{i,2}})$;

2) When $\alpha = 2$, with input (i, CT_i), γ runs the **Auth-2** algorithm and answers A_2 with $td_{2,i,CT_i} = H_2(g^{\alpha_{i,2}} C_{i,2} f^{\beta_{i,2}}, C_{i,1}, C_{i,2}, C_{i,3})$;

3) When $\alpha = 3$, with input (i, CT_i, j, CT_j), γ runs the **Auth-3** algorithm and answers A_2 with $td_{3,i,CT_i,\ j,CT_j} = (W_i, Z_i) = ([H_2(g^{\alpha_{i,2}} C_{i,2} f^{\beta_{i,2}}, C_{i,1}, C_{i,2}, C_{i,3})]_{2l}^{4l-1}, g^{a_i x_{i,1}} w_i f^{a_i x_{i,2}})$, where $w_i = g^{r_{i,3}} C_{j,2} f^{r_{i,4}}$.

Challenge. A_2 takes two messages M_0, M_1 randomly to γ. Then, γ selects a random bit $b \in \{0,1\}$ and chooses $r_{t,1}, r_{t,2} \in Z_q^*$. Then, it outputs $CT_t = (C_{t,1}, C_{t,2}, C_{t,3}, C_{t,4})$ as follows:

$C_{t,1} = g^{a_2} f^{b_2}$, $C_{t,2} = g^{r_{t,1}} f^{r_{t,2}}$, $C_{t,3} = M_b \| r_{t,1} \| r_{t,2} \oplus H_1(g^{a_3} f^{b_3})$,

$C_{t,4} = x_1 \| x_2 \| y_1 \| y_2 \oplus H_2(g^{r_{t,1}} V_t f^{r_{t,2}}, C_{t,1}, C_{t,2}, C_{t,3})$

Finally, it gives CT_t to A_2 as the challenge ciphertext.

Phase 2. A_2 continues performing queries as in Phase 1. The restrictions are as follows:

· In the decryption key queries, $i \neq t$;

· In the decryption queries, CT_t is not allowed.

· For Type-α ($\alpha = 1,2,3$) authorization queries:

1) When $\alpha = 1$, $i \neq t$;

2) When $\alpha = 2$, (t, CT_t) is not allowed;

3) When $\alpha = 3$, $(t, CT_t, \cdot)$ is not allowed.

Guess. Finally, A_2 submits a guess b^*. If $b^* = b$ holds, γ outputs 1 meaning $g^{a_3} f^{b_3} = g^{a_1+a_2} f^{b_1+b_2}$; otherwise, it outputs 0.

5. Performances Analysis

Here, we analyze the efficiency of our scheme. In **Table 3**, we provide comparisons to other PKEwET schemes without flexible authorization. From the second row to the fifth row, we show complexity comparisons of the encryption algorithm (C_{Enc}), decryption algorithm (C_{Dec}), authorization algorithm (Auth) and test algorithm (Test). The last row shows the ability to resist quantum attacks (RQA). **Table 4** shows comparisons to the schemes in [1] with flexible authorization. The second to the fifth column show complexity comparisons of 4 types of authorization algorithms, while the sixth to the ninth column show complexity comparisons of 4 types of test algorithms. Here, e means modular exponentiation, i means modular inversion, m means modular multiplication and p means pairing evaluation. **Table 5** shows comparisons to the scheme in [26]. The second to the fifth row shows the complexity comparisons of C_{Enc}, C_{Dec}, Auth and Test. The last line shows the ability to resist RQA.

From the comparisons in **Table 3**, we can see that there is no authorization in [15] , [16], [18] and in [17], because these algorithms do not use different types of authorizations according to different situations. In [1], Ma added four types of authorization policies. **Table 3** shows that our scheme is more efficient in C_{Enc} and C_{Dec}.

From **Table 4**, we observe that our scheme is more efficient than the scheme in [1], in Type-2, Type-4 authorization algorithms and in Type-1, Type-2, Type-4 test algorithms. However, our proposed scheme has no advantage in Type-3 authorization and Type-3 test algorithm. The reason is that the Diffie-Hellman-like key agreement based on the non-Abelian factorization problem requires 2 exponentiations [3].

From **Table 5**, we can see that our proposed scheme is less efficient than the scheme in [26]. However, our scheme supports four types authorization and four types equality test. The scheme in [26] is a general public-key encryption scheme that does not support equality test. Both of them can resist quantum attacks.

As we all know, Shor's quantum algorithm can solve integer factorization problem and DLP [2,31]. Then, many cryptographers have paid efforts to some non-commutative cryptosystems. Baba et al. analyse non-commutative properties of groups and propose the non-Abelian group factorization problem. Gu et al. have shown that there is no use to employ Shor's algorithm to solve the group factorization problem $FP_{g,f}^{G}$ problem when the underlying group G is non-Abelian. The core idea of Shor's quantum algorithm is a quantum order-finding process. But in [26], even Shor's ordering finding processes is assumed available as an oracle, one cannot solve the $FP_{g,f}^{G}$ problem since the order-lifting sub-process is blocked by the noncommutativity of G. In this paper, the proposed scheme is based on the intractability assumption of the $FP_{g,f}^{G}$ problem when G is non-Abelian. Therefore, we claim that our proposal can resist Shor's quantum algorithm attack according to Gu's arguments.

Table 3. The comparison of computational complexity (time)

	[15]	[16]	[17]	[18]	[25]	[1]	Ours
C_{Enc}	3e	4e+1m	5e	4e+1m	6e+2p	6e+1m	4e+1i+6m
C_{Dec}	3e	2e	2e	2e	2e+2p+1i	5e+1m	2e+1i+5m
Auth	✖	3e	0	3e	0	Type-4	Type-4
Test	2p	4p+2i	2e+2m+2i	4p+2i	4p+2i		
RQA	No	No	No	No	No	No	Yes

Table 4. The comparison of computational complexity (time)

	Auth				Test			
[1]	Type-1	Type-2	Type-3	Type-4	Type-1	Type-2	Type-3	Type-4
	0	2e	2e+2p	1e	2e+2p+1m	2e+2p+1m	2e+2p+2i+1m	2e+2p+1m
ours	0	4m	4e+2i+8m	2m	2i+4m	2i	4e+4i+4m	2i+2m

Table 5. The comparison of the scheme in [26] (time)

	C_{Enc}	C_{Dec}	Auth	Test	RQA
[26]	4e	2e	✖	✖	yes
ours	4e+1i+6m	2e+1i+5m	Type-4	Type-4	yes

6. Conclusion

The public key encryption with equality test provides the ability to determine whether two different ciphertexts contain the same message without decryption. This type of encryption has wide applications in cloud computing. Recently, many cryptographers have spent much effort researching this topic. However, because Shor’s quantum algorithm can solve integer factorization problem and DLP, many existing PKEwET schemes will be in danger in the quantum computing era. In this paper, we analyze the PKEwET scheme in Ma and present a new scheme based on the non-Abelian factorization problems. We prove that our scheme is secure and achieves different levels of security for attacker with different authorizations. Compared to previous schemes, it can not only resist quantum algorithm attack but is also more efficient than them.

References

[1] S. Ma, Q. Huang, M. Zhang, and B. Yang, "Efficient Public Key Encryption With Equality Test Supporting Flexible Authorization," *IEEE Trans. on Information Forensics and Security*, vol. 10, no. 3, pp.458-470, 2015. Article (CrossRef Link).

[2] Shor PW. "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM review*, vol. 41, no. 2, 303-332, 1999. Article (CrossRef Link).

[3] S. Baba, S. Kotyad and R. Teja, "A non-Abelian factorization problem and an associated cryptosystem[J]," *IACR Cryptology ePrint Archive*, 2011. Article (CrossRef Link).

[4] M. Bellare, A. Boldyreva and A. ONeill, "Deterministic and efficiently searchable encryption,"*Annual International Cryptology Conference. Springer Berlin Heidelberg*, pp. 535-552, 2007. Article (CrossRef Link).

[5] M. Bellare, M. Fischlin, A. ONeill, and T. Ristenpart, "Deterministic encryption: Definitional equivalences and constructions without random oracles," in *Proc. of Advances in Cryptology (Lecture Notes in Computer Science)*, vol. 5157. Berlin, Germany: Springer-Verlag, pp. 360-378, Aug. 2008. Article (CrossRef Link).

[6] A. Boldyreva, S. Fehr, and A. ONeill, "On notions of security for deterministic encryption, and efficient constructions without random oracles," in *Proc. of Annual International Cryptology Conference. Springer Berlin Heidelberg*, 335-359, 2008. Article (CrossRef Link).

[7] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Proc. of International Conference on the Theory and Applications of Cryptographic Techniques. Springer Berlin Heidelberg*, 506-522, 2004. Article (CrossRef Link).

[8] N. Cao, C. Wang, M. Li, K. Ren and W. Lou, "Privacy-preserving multi-keyword ranked search over encrypted cloud data," *IEEE Trans. Parallel Distrib. Syst*, vol. 25, no. 1, 222C233, 2014. Article (CrossRef Link).

[9] L. Fang, W. Susilo, C. Ge et al. "Public key encryption with keyword search secure against keyword guessing attacks without random oracle," *Information Sciences*, vol. 238, 221-241, 2013. Article (CrossRef Link).

[10] M. Abdalla, M. Bellare, D. Catalano, et al. "Searchable encryption revisited: Consistency properties, relation to anonymous IBE, and extensions," *Advances in CryptologyCCRYPTO 2005. Springer Berlin Heidelberg*, 2005, 205-222. Article (CrossRef Link).

[11] J. W. Byun, H. S. HRhee, H. A. Park, et al. "Off-line keyword guessing attacks on recent keyword search schemes over encrypted data," *Secure Data Management. Springer Berlin Heidelberg*, 2006, 75-83. Article (CrossRef Link).

[12] P. Xu, H. Jin, Q. Wu, et al. "Public-key encryption with fuzzy keyword search: A provably secure scheme under keyword guessing attack," *IEEE Transactions on computers*, vol. 62, no. 11, 2266-2277, 2013. Article (CrossRef Link).

[13] Y. Yu, J. Ni, H. Yang, et al. "Efficient public key encryption with revocable keyword search,"*Security and Communication Networks*, vol. 7, no. 2, 466-472, 2014. Article (CrossRef Link).

[14] M. Nishioka, "Perfect keyword privacy in PEKS systems," *Provable Security. Springer Berlin Heidelberg*, 175-192, 2012. Article (CrossRef Link).

[15] G. Yang, C. H. Tan, Q. Huang, et al. "Probabilistic public key encryption with equality test," in *Proc. of Cryptographers Track at the RSA Conference. Springer Berlin Heidelberg*, 119-131, 2010. Article (CrossRef Link).

[16] Q. Tang, "Towards public key encryption scheme supporting equality test with fine-grained authorization," in *Proc. of Australasian Conference on Information Security and Privacy. Springer Berlin Heidelberg*, 389-406, 2011. Article (CrossRef Link).

[17] Q. Tang, "Public key encryption supporting plaintext equality test and user-specified authorization," *Security and Communication Networks*, vol. 5, no. 12, 1351-1362, 2012. Article (CrossRef Link).

[18] Q. Tang, "Public key encryption schemes supporting equality test with authorization of different granularity," *International journal of applied cryptography*, vol. 2, no. 4, 304-321, 2012. Article (CrossRef Link).
[19] S. Ma, M. Zhang, Q. Huang, et al. "Public key encryption with delegated equality test in a multi-user setting," *The Computer Journal*, bxu026, 2014. Article (CrossRef Link).
[20] A. G. Myasnikov, V. Shpilrain and A. Ushakov, "Non-commutative Cryptography and Complexity of Group-theoretic Problems," *Providence, RI, USA: American Mathematical Society*, 2011. Article (CrossRef Link).
[21] N. R. Wagner, M. R. Magyarik. "A public-key cryptosystem based on the word problem," in *Proc. of Workshop on the Theory and Application of Cryptographic Techniques*. *Springer Berlin Heidelberg*, 1984, 19-36, 1984. Article (CrossRef Link).
[22] K. H. Ko, S. J. Lee, J. H. Cheon, J. W.Han, J. Kang and C. Park. "New public-key cryptosystem using braid groups," *CRYPTO 2000, LNCS 1880*, pp. 166-183. Springer, 2000. Article (CrossRef Link).
[23] V. Shpilrain and A. Ushakov. "Thompson's group and public key cryptography," *ACNS 2005*, LNCS 3531, pp. 151-164. Springer, 2005. Article (CrossRef Link).
[24] V. Shpilrain, G. Zapata. "Combinatorial group theory and public key cryptography," *Applicable Algebra in Engineering, Communication and Computing*, vol. 17, no. 3-4, 291-302, 2006. Article (CrossRef Link).
[25] S. Ma, "Identity-based encryption with outsourced equality test in cloud computing," *Information Sciences*, 328, 389-402, 2016. Article (CrossRef Link).
[26] L. Gu, L. Wang, K. Ota, M. Dong, Z. Cao, Y. Yang. "New public key cryptosystems based on non-abelian factorization problems," *Security and Communication Networks*, vol. 6, no. 7, pp. 912-922, 2013. Article (CrossRef Link).
[27] D. Kahrobaei, C. Koupparis, V. Shpilrain. "Public key exchange using matrices over group rings," *Groups-Complexity-Cryptology*, 5(1), 2013. Article (CrossRef Link).
[28] X. Chen, J. Li, X. Huang, J. Ma, and W. Lou, "New Publicly Verifiable Databases with Efficient Updates," *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 5, 546-556, 2015. Article (CrossRef Link).
[29] S. Benabbas, R. Gennaro, and Y. Vahlis, "Verifiable delegation of computation over large datasets," *Advances in Cryptology-CRYPTO 2011*, LNCS 6841, Springer, pp.111-131, 2011. Article (CrossRef Link).
[30] X. Chen, J. Li, J. Ma, Q. Tang, and W. Lou, "New algorithms for secure outsourcing of modular exponentiations," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 9, 2386-2396, 2014. Article (CrossRef Link).
[31] Shor PW. "Algorithms for quantum computation: discrete logarithms and factoring," in *Proc. of FOCS 1994. IEEE Computer Society: Washington, D.C.*, 124–134, 1994. Article (CrossRef Link).

Huijun Zhu received the B.S. degree from Luoyang Normal University in 2007 and the M.S. degree from Henan Polytechnic University in 2010. Her research interests include modern cryptography, network security, cloud computing, etc. She is currently a Ph.D candidate in Beijing University of Posts and Telecommunications.

Licheng Wang received the B.S. degree from Northwest Normal University in 1995, the M.S. degree from Nanjing University in 2001, and the PhD degree from Shanghai Jiao Tong University in 2007. His current research interests include modern cryptography, network security, trust management, etc. He is an associate professor in Beijing University of Posts and Telecommunications.

Shuming Qiu received the B.S. degree from Nanchang Hangkong University in 2006, the M.S. degree from Jiangxi Normal University in 2009. He is currently a Ph.D. student in Beijing University of Posts and Telecommunications, Beijing, China and a lecturer at Jiangxi Normal University, Nanchang, China. His research interests include algebra, information security and cryptography (in particular, cryptographic protocols), etc.

Xinxin Niu is an professor of Computer Science and Technology at Beijing University of Posts and Telecommunications. She received the MS degree from Beijing University of Posts and Telecommunications in 1988, the PhD degree from Chinese University of Hong Kong in 1997. Her current research interests include network security, digital watermarking and digital rights management, etc.