Certificateless Public-key Authenticate Searchable Encryption with Probabilistic Trapdoor Generation

Leixiao Cheng [1] and Fei Meng [1]*

[1]School of Mathematics, Shandong University, Jinan 250100, China
mengfei_sdu@163.com

Abstract. Boneh et al. proposed the cryptographic primitive public key encryption with keyword search (PEKS) to search on encrypted data without exposing the privacy of the keyword. Most standard PEKS schemes are vulnerable to inside keyword guessing attacks (KGA), i.e., a malicious server may generate a ciphertext by its own and then to guess the keyword of the trapdoor by testing. Huang et al. solved this problem by proposing the public-key authenticated encryption with keyword search (PAEKS) achieving single trapdoor indistinguishability (TI).
Certificateless public-key authenticated encryption with keyword search (CLPAEKS) is first formally proposed by He et al. as a combination of the Huang's PAEKS and the certificateless public key cryptography (CLPKC). Lin et al. revised He's work and re-formalize the security requirements for CLPAEKS in terms of both trapdoor indistinguishability and ciphertext indistinguishability. However, trapdoor generation algorithms of all above works are deterministic. In this case, given two trapdoors, it's obviously to check whether the target keywords are identical embedded in them. This feature conflicts with trapdoor indistinguishability security.
In this paper, we initially propose a CLPAEKS scheme with probabilistic trapdoor generation. Formal proof shows that our scheme is provable secure in the random oracle model.

Keywords: Certificateless Public key encryption · Keyword search · Keyword guessing attacks · Ciphertext indistinguishability · Trapdoor indistinguishability

1 Introduction

With the widespread application of cloud storage and computing, searching for encrypted data in the cloud server has become a hot research topic. In 2004, Boneh et al. [4] proposed the first public key encryption with keyword search (PEKS). In a PEKS scheme, data sender generates and submits a ciphertext embedded with a ciphertext keyword to the cloud. Searching for a ciphertext

* Corresponding author

with specific keyword, the receiver generates the trapdoor with a target keyword. Obtaining a trapdoor, the cloud tests whether the target keyword of the trapdoor is identical to the ciphertext keyword of each ciphertext. If so, the cloud returns the successfully matched ciphertexts to the receiver. During this procedure, the keywords information of both the ciphertext and the trapdoor cannot be exposed to any third party i.e., ciphertext indistinguishability (CI) and trapdoor indistinguishability (TI). Ciphertext indistinguishability (CI) is a basic requirement for a PEKS scheme, and trapdoor indistinguishability (TI) is also important but cannot be captured by a lot of PEKS schemes. Since, the keywords are often chosen from a low-entropy keyword space in the real applications, it's feasible for the adversary to guess each keyword containing in a given trapdoor by the keyword guessing attacks (KGA) [6].

Recently, Huang et al. [12] introduced the notion of Public-key Authenticated Encryption with Keyword Search (PAEKS) to resist inside KGA (attacks from a malicious cloud tester), in which the sender encrypts a keyword under his public key and the senders secret key and then authenticates it, such that the cloud server cannot launch KGA successfully by encrypting a keyword itself. However, generation algorithms in PAEKS schemes [12, 14, 16, 18] are all deterministic. This feature conflicts with trapdoor indistinguishability security.

In the real application scenarios of cloud storage, a fully trusted third party is difficult to constructed. Since any entities involved in the cloud storage system may be dishonest or malicious, schemes based on either public key infrastructure or identity cryptography deeply relying on a completely honest key generation center (KGC) will be broken down by a dishonest KGC. In order to get rid of fully trusted in KGC, by combining the certificateless public key cryptography (CLPKC) [2] and PAEKS [12], He et al. [11] proposed the certificateless public key encryption with keyword search (CLPAEKS) to avoid the certificate management and the key escrow problem, and to achieve trapdoor privacy in PEKS scenario. However, [11] didn't capture the security requirements for trapdoor indistinguishability. Liu et al. [15] pointed out that [11] is vulnerable to KGA by a malicious receiver, which is not considered in their security model, re-formalized the security model to meet for trapdoor indistinguishability.

1.1 Motivation and contributions

Trapdoor generation algorithms of PAEKS schemes like [4, 12, 14, 16, 18] and CLPAEKS schemes like [11, 15] are all deterministic, so it's available to check whether the target keywords in two given trapdoors are identical or not. This feature obviously conflicts with trapdoor indistinguishability security in the real application, even though these works are provable secure.

In this paper, we initially propose certificateless public-key authenticate searchable encryption with probabilistic trapdoor generation. The security of our scheme is proved under static assumptions (i.e., DDH assumption) in the random oracle model.

In Table 1, we compare our schemes with some other PEKS schemes.

Table 1. Comparison with other PEKS schemes

	Ciphertext indistinguishability	Trapdoor indistinguishability	Probabilistic trapdoor	Certificateless
[4]	√			
[12]	√	√		
[16]	√	√		
[18]	√	√		
[14]	√	√		
[11]	√			√
[15]	√	√		√
Ours	√	√	√	√

1.2 Related works

In 2000 [1], Song et al. [23] introduced the first searchable symmetric encryption (SSE), which allows keyword search over encrypted data in the outsourcing scenarios. In 2004, Boneh et al. [1] initialized proposed the first public key encryption with keyword search scheme (PEKS). In 2005, Abdalla et al. [1] revised Boneh's work and provided a transform from an anonymous identity-based encryption scheme to a secure PEKS scheme. In addition, they also extended the basic notions of anonymous hierarchical identity-based encryption, public-key encryption with temporary keyword search, and identity-based encryption with keyword search. Since then, various PEKS schemes have been proposed. Golle et al. [10] defined the security model for conjunctive keyword search over encrypted data and presented the first schemes for conducting such searches securely. Park et al. [17] proposed the public key encryption with conjunctive field keyword search enabling an email gateway to search keywords conjunctively. Boneh et al. [5] provided several public key system that support comparison queries, subset queries and arbitrary conjunctive queries on encrypted data. Shi et al. [22] designed an encryption scheme called multi-dimensional range query over encrypted data to deal with the privacy issues related to the sharing of network audit logs. Moreover, proxy re-encryption with keyword search [21], decryptable searchable encryption [9], keyword updatable PEKS [19] and attribute-based encryption with keyword search [8, 24] enjoyed some other interesting features compared with standard PEKS.

However, in actual practice, keywords are chosen from much smaller space than the space of passwords, thus all the above schemes are susceptible to the keyword guessing attacks (KGA). Byun et al. [6] analysed the security vulnerabilities on [1, 10] by performing off-line keyword guessing attacks. Rhee et al. [20] introduced the concept of trapdoor indistinguishability and show that trapdoor indistinguishability is sufficient for thwarting keyword-guessing attacks and proposed a provably secure PEKS scheme in the designated tester model (dPEKS). Unfortunately, dPEKS suffers from an inherent insecurity called inside keyword guessing attack (IKGA) launched by the malicious tester (i.e., cloud server). Chen et al. [7] proposed a new PEKS framework named dual-server

PEKS scheme (DS-PEKS) using two uncolluded semi-trusted servers. Huang et al. [12] proposed a public-key authenticated encryption with keyword search scheme (PAEKS), which is secure against IKGA. It was then extended to certificateless PAEKS [11], identity-based setting with designed tester [14]. Recently, Noroozi et al. [16] found that Huang's work is insecure in the multi-receiver model and Qin et al. [18] also analysed a drawback in Huang's work, i.e., it's very easy to check whether the keywords in two ciphertexts are identical or not, so they revised Huang's work to overcome this drawback.

The concept of certificateless public key cryptography (CLPKC) was initially introduced by Al-Riyami and Paterson [10], keeping users private key unrevealed to (KGC) by allowing users to set a secret value themselves. Then, Certificateless Public Key Encryption (CLPKE) [3] and Certificateless Signature (CLS) [13] schemes are proposed. He et al. [11] proposed the certificateless public key encryption with keyword search (CLPAEKS) to avoid the certificate management and the key escrow problem with trapdoor privacy. The security models of [11], which only considered the searchable ciphertext indistinguishability without accessing the ciphertext oracle, cannot accurately capture the security requirements against inside keyword guessing attacks. Liu et al. [15] re-formalized the security model to meet for trapdoor indistinguishability. Unfortunately , all above PAEKS and CLPAEKS schemes cannot achieve probabilistic trapdoor generation. So, given two trapdoors, it's obviously whether the keywords in the trapdoors are identical or not.

1.3 Organization

This paper is organized as follows. Section 2 introduces the necessary preliminaries. Section 3 presents the system and security model of CLPAEKS. We give a concrete construction and formal security analysis of CLPAEKS in section 4 and section 5 respectively. In the end, section 6 summarizes the paper and prospects for the future research.

2 Preliminaries

In this section, we introduce some background knowledge bilinear maps and Diffie-Hellman assumption.

2.1 Bilinear map

We briefly recall the definitions of the bilinear map. Let $\mathbb{G}_0$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of prime order q. Let P be a generator of $\mathbb{G}_0$ and e be a efficient computable bilinear map, $e : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_T$. The bilinear map e has a few properties: (1) Bilinearity: for all $M, N \in \mathbb{G}_0$ and $a, b \in \mathbb{Z}_p$, we have $e(aM, bN) = e(M, N)^{ab}$. (2) Non-degeneracy: for any generator $P \in \mathbb{G}_0$, $e(P, P) \in \mathbb{G}_T$ is the generator of $\mathbb{G}_T$. (3) Computability: For any $M, N \in \mathbb{G}_0$, there is an efficient algorithm to compute $e(M, N)$.

2.2 DDH assumption

The Diffie-Hellman (DDH) assumption is defined as: given group parameter $(\mathbb{G}_0, q, P)$ and three random elements $x, y \in_R \mathbb{Z}_p^*$. We say that the DDH assumption holds, if there is no probabilistic polynomial time (PPT) adversary $\mathcal{B}$ can distinguish between the tuple (P, xP, yP, xyP) and the tuple (P, xP, yP, ϑ), where ϑ is randomly selected from $\mathbb{G}_0$. More specifically, the advantage ϵ of $\mathcal{B}$ in solving the DDH problem is defined as

$$\Big|\Pr[\mathcal{A}(P, xP, yP, xyP) = 1] - \Pr[\mathcal{A}(P, xP, yP, \vartheta) = 1]\Big|. \tag{1}$$

Definition 1 (DDH). *We say that the DDH assumption holds if no PPT algorithm has a non-negligible advantage ϵ in solving DDH problem.*

3 Definition and Security Model of CLPAEKS

3.1 System Model

Our proposed CLPAEKS scheme is composed of four entities: key generation center (KGC), cloud server (CS), data sender (ID_S), data receiver (ID_R). Specifically,

- KGC: It is responsible for generating public parameters, master secret key, and partial private/public keys of both sender and receiver.
- ID_S: Data sender generates its own user private/public keys and encrypts the ciphertext with the public keys of ID_S and ID_R and its own secret key, then it submits the encrypted data to the CS.
- ID_R: Receiver generates its own user private/public keys and the trapdoor with the public keys of ID_S and ID_R and its own secret key, then it submits the trapdoor to the CS to search for an encrypted data.
- CS: It is responsible for data processing, storage and retrieval.

3.2 Algorithms of CLPAEKS

The CLPAEKS consists of the following algorithms:

- **Setup**: Given security parameter λ, KGC runs this algorithm to generate the public parameter PP and master secret key MSK.
- **Partial public key**: Given a user's identity ID_i, KGC runs this algorithm to generate the partial public key R_{ID_i} for the user.
- **Partial private key**: Given a user's identity ID_i, KGC runs this algorithm to generate the partial private key d_{ID_i} for the user.
- **User private key**: Given a user's identity ID_i, the user U_{ID_i} runs this algorithm to generate its private key x_{ID_i}.
- **User public key**: Given a user's identity ID_i, the user U_{ID_i} runs this algorithm to generate its public key P_{ID_i}.

- **CLPAEKS**: Given sender's identity ID_S, receiver's identity ID_R, sender's secret key $SK_{ID_S} = (d_{ID_S}, x_{ID_S})$, public keys $\{PK_{ID_S} = (R_{ID_S}, P_{ID_S}), PK_{ID_R} = (R_{ID_R}, P_{ID_R})\}$, and a ciphertext keyword KW, the sender runs this algorithm to generate the ciphertext Ct.
- **TrapGen**: On input public parameter PP, public key of receiver PK_S, the key pair of receiver (PK_R, SK_R) and a target keyword KW', this algorithm is run by receiver to generate a Tr.
- **Test**: On input public parameter PP, ciphertext CT and trapdoor Tr, this algorithm checks whether the ciphertext keyword KW is identical to the target keyword KW'. If so, it outputs 1; otherwise it outputs 0.

Correctness: For any honestly generated key pairs (PK_R,SK_R) and (PK_S,SK_S), any two keywords KW, KW', Ct is generated by algorithm CLPAEKS and Tr generated by algorithm TrapGen. If $KW = KW'$, then the text algorithm outputs 1 with probability 1 i.e., Pr[Test(PP, Ct, Tr) = 1] = 1; otherwise Pr[Test$(PP, Ct, Tr) = 0] = 1 - negl(\lambda)$.

3.3 Security Model

In certificateless cryptography, there are two types of adversaries, i.e., Type1 adversary and Type2 adversary. Type1 adversary cannot access the systems master key, but it can replace any users public key. Type2 adversary cannot replace users public key, but it can access the systems master key.

We describe two security model: ciphertext indistinguishability (CI) security and trapdoor indistinguishability (TI) security. The notion of TI security of CLPAEKS is defined as follows:

Game 1 (type 1 adversary): In this game, the adversary is allowed to lunch replace public key query.

- **Setup**: Given a security parameter λ, the challenge sender identity ID_S and receiver identity ID_R, new public key PK'_{ID_R} the challenger $\mathcal{C}$ generates the public parameter PP, master secret key s and public key PK_{ID_R} of ID_S, then responds the adversary $\mathcal{A}$ with PP.
- **Phase** 1: The adversary $\mathcal{A}$ is allowed to issue polynomial queries the following oracles:

  Hash query : $\mathcal{A}$ is allowed to issue queries all hash oracles.

  Extract partial private key query : Given an identity ID_i , the challenger $\mathcal{C}$ generates the partial private key d_{ID_i} and returns it to $\mathcal{A}$. $\mathcal{A}$ cannot enquire the partial private key of ID_S or ID_R.

  Extract user private key query : Given an identity ID_i , the challenger $\mathcal{C}$ generates the user private key x_{ID_i} and returns it to $\mathcal{A}$.

  Public key query : Given an identity ID_i, the challenger $\mathcal{C}$ responds $\mathcal{A}$ with the public key of PK_{ID_i}.

  Replace public key query : Given an identity ID_i, $\mathcal{A}$ is allowed to ask the challenger $\mathcal{C}$ to replace the public key PK_{ID_i} with a new one PK'_{ID_i}. Note that it's prohibited from replacing the public key of the challenge senders identity ID_S before the challenge phase.

Ciphertext query $\mathcal{O}_C$: Given a ciphertext query (ID_S, ID_R, KW), $\mathcal{C}$ responds $\mathcal{A}$ with the ciphertext CT of keyword KW.

Trapdoor query $\mathcal{O}_T$: Given a trapdoor query (ID_S, ID_R, KW'), $\mathcal{C}$ responds $\mathcal{A}$ with the trapdoor Tr of keyword KW'.

- **Challenge**: $\mathcal{A}$ chooses two keywords KW_0^* and KW_1^* with the restriction that no element of $\{ID_S, ID_R, KW_j^*\}_{j\in\{0,1\}}$ has been queried on $\mathcal{O}_C$ or $\mathcal{O}_T$. $\mathcal{C}$ randomly chooses a bit $\theta \in \{0,1\}$ and responds $\mathcal{A}$ with a set of challenge trapdoors Tr_θ^*.
- **Phase** 2: This phase is the same as Phase 1 with the restriction that no element of $\{ID_S, ID_R, KW_j^*\}_{j\in\{0,1\}}$ can be queried on $\mathcal{O}_C$ or $\mathcal{O}_T$.
- **Guess**: $\mathcal{A}$ outputs a guess bit θ' of θ and it wins the game if $\theta' = \theta$. The advantage of $\mathcal{A}$ to win the TI security game is defined as $Adv_{\mathcal{A}}^{TI}(\lambda) = \Big|\Pr[\theta' = \theta] - \frac{1}{2}\Big|$.

Game 2 (type 2 adversary): In this game, the adversary is allowed to request for the master secret key.

- **Setup**: Given a security parameter λ, the challenger $\mathcal{C}$ generates the public parameter PP, master secret key s and public keys PK_{ID_S},PK_{ID_S}) of challenge sender ID_S, receiver ID_R, respectively. Finally, it responds $\mathcal{A}$ with PP,s,PK_{ID_S},PK_{ID_S}).
- **Phase** 1: The adversary $\mathcal{A}$ is allowed to issue polynomial queries the following oracles:

Hash query : $\mathcal{A}$ is allowed to issue queries all hash oracles.

Extract partial private key query : Given an identity ID_i , the challenger $\mathcal{C}$ generates the partial private key d_{ID_i} and returns it to $\mathcal{A}$.

Extract user private key query : Given an identity ID_i , the challenger $\mathcal{C}$ generates the user private key x_{ID_i} and returns it to $\mathcal{A}$. $\mathcal{A}$ cannot enquire the user private key of ID_S or ID_R.

Public key query : Given an identity ID_i, the challenger $\mathcal{C}$ responds $\mathcal{A}$ with the public key of PK_{ID_i}.

Ciphertext query $\mathcal{O}_C$: Given a ciphertext query (ID_S, ID_R, KW), $\mathcal{C}$ responds $\mathcal{A}$ with the ciphertext CT of keyword KW.

Trapdoor query $\mathcal{O}_T$: Given a trapdoor query (ID_S, ID_R, KW'), $\mathcal{C}$ responds $\mathcal{A}$ with the trapdoor Tr of keyword KW'.

- **Challenge**: $\mathcal{A}$ chooses two keyword sets KW_0^* and KW_1^* with the restriction that no element of $\{ID_S, ID_R, KW_j^*\}_{j\in\{0,1\}}$ has been queried on $\mathcal{O}_C$ or $\mathcal{O}_T$. $\mathcal{C}$ randomly chooses a bit $\theta \in \{0,1\}$ and responds $\mathcal{A}$ with a set of challenge trapdoors Tr_θ^*.
- **Phase** 2: This phase is the same as Phase 1 with the restriction that no element of $\{ID_S, ID_R, KW_j^*\}_{j\in\{0,1\}}$ can be queried on $\mathcal{O}_C$ or $\mathcal{O}_T$.
- **Guess**: $\mathcal{A}$ outputs a guess bit θ' of θ and it wins the game if $\theta' = \theta$. The advantage of $\mathcal{A}$ to win the TI security game is defined as $Adv_{\mathcal{A}}^{TI}(\lambda) = \Big|\Pr[\theta' = \theta] - \frac{1}{2}\Big|$.

Definition 2. *The CLPAEKS achieves TI security against inside keyword guessing attacks, if there exist no PPT adversary winning the above games with a non-negligible advantage ϵ.*

New, we introduce the notion of CI security as follows:

Game 3 (type 1 adversary): In this game, the adversary is allowed to lunch replace public key query.

- **Setup**: Given a security parameter λ, the challenge sender identity ID_S and receiver identity ID_R, new public key PK'_{ID_S} the challenger $\mathcal{C}$ generates the public parameter PP, master secret key s and public key PK_{ID_S} of ID_S, then responds the adversary $\mathcal{A}$ with PP.
- **Phase** 1: The adversary $\mathcal{A}$ is allowed to issue the same queries as in Game 1.
- **Challenge**: $\mathcal{A}$ chooses two keyword sets KW_0^* and KW_1^* with the restriction that no element of $\{ID_S, ID_R, KW_j^*\}_{j\in\{0,1\}}$ has been queried on $\mathcal{O}_C$ or $\mathcal{O}_T$. $\mathcal{C}$ randomly chooses a bit $\theta \in \{0,1\}$ and responds $\mathcal{A}$ with a set of challenge trapdoors Ct_θ^*.
- **Phase** 2: This phase is the same as Phase 1 with the restriction that no element of $\{ID_S, ID_R, KW_j^*\}_{j\in\{0,1\}}$ can be queried on $\mathcal{O}_C$ or $\mathcal{O}_T$.
- **Guess**: $\mathcal{A}$ outputs a guess bit θ' of θ and it wins the game if $\theta' = \theta$. The advantage of $\mathcal{A}$ to win the CI security game is defined as $Adv_{\mathcal{A}}^{CI}(\lambda) = \Big| \Pr[\theta' = \theta] - \frac{1}{2}\Big|$.

Game 4 (type 2 adversary): In this game, the adversary is allowed to request for the master secret key.

- **Setup**: Given a security parameter λ, the challenger $\mathcal{C}$ generates the public parameter PP, master secret key s and public keys PK_{ID_S},PK_{ID_S}) of challenge sender ID_S, receiver ID_R, respectively. Finally, it responds $\mathcal{A}$ with PP,s,PK_{ID_S},PK_{ID_S}).
- **Phase** 1: The adversary $\mathcal{A}$ is allowed to issue the same queries as in Game 2.
- **Challenge**: $\mathcal{A}$ chooses two keyword sets KW_0^* and KW_1^* with the restriction that no element of $\{ID_S, ID_R, KW_j^*\}_{j\in\{0,1\}}$ has been queried on $\mathcal{O}_C$ or $\mathcal{O}_T$. $\mathcal{C}$ randomly chooses a bit $\theta \in \{0,1\}$ and responds $\mathcal{A}$ with a set of challenge trapdoors Ct_θ^*.
- **Phase** 2: This phase is the same as Phase 1 with the restriction that no element of $\{ID_S, ID_R, KW_j^*\}_{j\in\{0,1\}}$ can be queried on $\mathcal{O}_C$ or $\mathcal{O}_T$.
- **Guess**: $\mathcal{A}$ outputs a guess bit θ' of θ and it wins the game if $\theta' = \theta$. The advantage of $\mathcal{A}$ to win the CI security game is defined as $Adv_{\mathcal{A}}^{CI}(\lambda) = \Big| \Pr[\theta' = \theta] - \frac{1}{2}\Big|$.

Definition 3. *The CLPAEKS achieves CI security, if there exist no PPT adversary winning the above security games with a non-negligible advantage ϵ.*

4 CLPAEKS with probabilistic trapdoor generation

The construction detail of our CLPAEKS is shown as follows.

- **Setup**: Given a security parameter λ, the algorithm chooses a bilinear map $e : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_T$, where $\mathbb{G}_0$ and $\mathbb{G}_T$ are groups with prime order q and P is the generator of $\mathbb{G}_0$, a random $s \in_R \mathbb{Z}_q^*$ such that $P_{pub} = sP$ and hash functions $H : \{0,1\}^* \longrightarrow \mathbb{G}_0$; $h_1 : \{0,1\}^* \times \mathbb{G}_0 \longrightarrow \mathbb{Z}_q^*$; $h_2 : \{0,1\}^* \times \mathbb{G}_0 \times \mathbb{G}_0 \times \mathbb{G}_0 \longrightarrow \mathbb{Z}_q^*$. It outputs the public parameter $PP = (e, \mathbb{G}_0, \mathbb{G}_T, q, P, P_{pub}, H, h_1, h_2)$, and keeps $MSK = s$ secret.
- **Extract partial private key**: Given a user's identity $ID_i \in \{0,1\}^*$ and MSK, the algorithm picks a random $r_{ID_i} \in_R \mathbb{Z}_q^*$ and computes $R_{ID_i} = r_{ID_1} P$, $\alpha_{ID_i} = h_1 ID_i, R_{ID_i}$, then sets $d_{ID_i} = r_{ID_i} + s\alpha_{ID_i}(\mod q)$. It publishes R_{ID_i} and sends the partial private key d_{ID_i} to user ID_i secretly.
- **Set secret value**: Given a user's identity $ID_i \in \{0,1\}^*$, the algorithm picks a random $x_{ID_i} \in_R \mathbb{Z}_q^*$ as a secret value.
- **Set private key**: Given a user's identity $ID_i \in \{0,1\}^*$, it's secret value x_{ID_i} and partial private key d_{ID_i}, the algorithm sets the user's private key as $SK_{ID_i} = \{x_{ID_i}, d_{ID_i}\}$.
- **Set public key**: Given a user's identity $ID_i \in \{0,1\}^*$, it's secret value x_{ID_i} and partial public information R_{ID_i}, the algorithm computes $P_{ID_i} = x_{ID_i} P$. sets the user's public key as $PK_{ID_i} = \{P_{ID_i}, R_{ID_i}\}$.
- **CLPAKES**: Given sender's identity ID_S, receiver's identity ID_R, sender's secret key SK_{ID_S}, public keys (PK_{ID_S}, PK_{ID_R}), and a ciphertext keyword KW, the algorithm randomly chooses $r_1, r_2 \in_R \mathbb{Z}_p^*$, then computes $\beta_{ID_S} = h_2(ID_S, P_{pub}, P_{ID_S}, R_{ID_S})$, $\beta_{ID_R} = h_2(ID_R, P_{pub}, P_{ID_R}, R_{ID_R})$ and generates the ciphertext as

$$\begin{aligned}
C_1 &= r_1(d_{ID_S} + \beta_{ID_S} x_{ID_S}) H(ID_S, ID_R, KW) + r_2 P, \\
C_2 &= r_2(R_{ID_R} + \alpha_{ID_R} P_{pub} + \beta_{ID_R} P_{ID_R}), \\
C_3 &= r_1(d_{ID_S} + \beta_{ID_S} x_{ID_S}) P, \\
C_4 &= r_1 P.
\end{aligned} \tag{2}$$

  Finally, it outputs $CT = (C_1, C_2, C_3, C_4)$.
- **Trapdoor**: Given sender's identity ID_S, receiver's identity ID_R, receiver's secret key SK_{ID_R}, public keys (PK_{ID_S}, PK_{ID_R}), and a target keyword KW', the algorithm randomly chooses $r', r'' \in_R \mathbb{Z}_p^*$, then computes

$$\begin{aligned}
T_1 &= r'(d_{ID_R} + \beta_{ID_R} x_{ID_R}) H(ID_S, ID_R, KW') + r'' P, \\
T_2 &= r'(R_{ID_R} + \alpha_{ID_R} P_{pub} + \beta_{ID_R} P_{ID_R}), \\
T_3 &= r''(R_{ID_S} + \alpha_{ID_S} P_{pub} + \beta_{ID_S} P_{ID_S}), \\
T_4 &= r' P.
\end{aligned} \tag{3}$$

  Finally, it outputs $Tr = (T_1, T_2, T_3, T_4)$.
- **Test**(PP, Ct, Tr): The algorithm checks whether the following equation holds or not:

$$e(C_1, T_2) \cdot e(T_3, C_4) = e(T_1, C_3) \cdot e(T_4, C_2). \tag{4}$$

  If so, it outputs 1; otherwise it outputs 0.

4.1 Correctness

If the ciphertext keyword KW is identical to the target keyword KW', we have the following equation:

$$\begin{aligned}
&e(C_1,T_2)\cdot e(T_3,C_4)\\
=&e(r_1(d_{ID_S}+\beta_{ID_S}x_{ID_S})H(ID_S,ID_R,KW)+r_2P, r'(R_{ID_R}+\alpha_{ID_R}P_{pub}+\beta_{ID_R}P_{ID_R}))\\
&\cdot e(r''(R_{ID_S}+\alpha_{ID_S}P_{pub}+\beta_{ID_S}P_{ID_S}), r_1P)\\
=&e(r'(d_{ID_R}+\beta_{ID_R}x_{ID_R})H(ID_S,ID_R,KW), r_1(d_{ID_S}+\beta_{ID_S}x_{ID_S})P)\cdot e(r''P, r_1(d_{ID_S}+\beta_{ID_S}x_{ID_S})P)\\
&\cdot e(r'P, r_2(R_{ID_S}+\alpha_{ID_S}P_{pub}+\beta_{ID_S}P_{ID_S}))\\
=&e(r'(d_{ID_R}+\beta_{ID_R}x_{ID_R})H(ID_S,ID_R,KW)+r''P, r_1(d_{ID_S}+\beta_{ID_S}x_{ID_S})P)\\
&\cdot e(r'P, r_2(R_{ID_S}+\alpha_{ID_S}P_{pub}+\beta_{ID_S}P_{ID_S}))\\
=&e(T_1,C_3)\cdot e(T_4,C_2).
\end{aligned} \tag{5}$$

The above equation holds with probability 1, if $KW = KW'$; it doesn't hold with overwhelming probability when $KW \neq KW'$.

5 Security Analysis

In this section, we provide a formal security analysis of our scheme.

Theorem 1. *Our scheme is semantically trapdoor indistinguishability secure against inside keyword guessing attacks in the random oracle model under the DDH assumption.*

Proof. Supposed that a PPT adversary $\mathcal{A}$ can break the TI security of our scheme with a non-negligible advantage $\epsilon > 0$, then there exists a PPT simulator $\mathcal{B}$ that can distinguish a DDH tuple from a random one with a non-negligible probability. The DDH challenger $\mathcal{C}$ selects $x, y \in_R \mathbb{Z}_p^*$, $\theta \in \{0,1\}$, $\mathcal{R} \in_R \mathbb{G}_0$ at random. Let $\mathcal{Z} = xyP$, if $\theta = 0$, $\mathcal{R}$ else. Next, $\mathcal{C}$ sends $\mathcal{B}$ the DDH tuple $\langle P, xP, yP, \mathcal{Z}\rangle$. Then, $\mathcal{B}$ plays the role of simulator in the following security game.

Game 1 (type1 adversary):

- **Setup**: $\mathcal{A}$ chooses the challenge sender's identity ID_S and challenge receiver's identity ID_R. The new public key to replace P_{ID_R} is P'_{ID_R}. $\mathcal{B}$ randomly chooses $\alpha_{ID_S}, \beta_{ID_S}, \alpha_{ID_R}, \beta_{ID_R} \in_R \mathbb{Z}_q^*$, $R_{ID_S}, R_{ID_R} \in_R \mathbb{G}_0$, computes $P_{pub} = \frac{1}{\alpha_{ID_R}}(x\cdot P - \beta_{ID_R}P'_{ID_R} - R_{ID_R})$, and adds $(ID_S, R_{ID_S}, \alpha_{ID_S})$ and $(ID_R, R_{ID_R}, \alpha_{ID_R})$ to the list L_{h_1}, adds $(ID_S, R_{ID_S}, \perp, \beta_{ID_S})$ and $(ID_R, R_{ID_R}, \perp, \beta_{ID_R})$ to list L_{E_1}. Then, it sends $PP = (e, \mathbb{G}_0, \mathbb{G}_T, q, P, P_{pub}, H, h_1, h_2)$ to $\mathcal{A}$.
- **Phase1**: The adversary $\mathcal{A}$ may issue the following queries:

  $\mathbf{h_1}$ query: $\mathcal{B}$ maintains a list L_{h_1} of tuple $(ID_i, R_{ID_i}, \alpha_{ID_i})$. This list is initially empty. Given a query tuple (ID_i, R_{ID_i}), $\mathcal{B}$ responds as follows:

  1. If $(ID_i, R_{ID_i}, \alpha_{ID_i})$ has already existed in the L_{h_1}, $\mathcal{B}$ responds $\mathcal{A}$ with $h_1(ID_i, R_{ID_i}) = \alpha_{ID_i}$.

2. Otherwise, $\mathcal{B}$ picks a random $\alpha_{ID_i} \in_R \mathbb{Z}_p^*$ and adds the tuple $(ID_i, R_{ID_i}, \alpha_{ID_i})$ into L_{h_1} and responds $\mathcal{A}$ with α_{ID_i}.

H query $\mathcal{O}_H$: $\mathcal{B}$ maintains a list L_H of tuple $(KW_i, ID'_S, ID'_R, u_i, c_i, H_i)$. This list is initially empty. Given a query tuple (KW_i, ID'_S, ID'_R), $\mathcal{B}$ responds as follows:

1. If $(KW_i, ID'_S, ID'_R, u_i, c_i, H_i)$ is in the L_H, $\mathcal{B}$ responds $\mathcal{A}$ with $H(ID'_S, ID'_R, KW_i) = H_i$.
2. Otherwise, $\mathcal{B}$ flips a random coin $c_i \in \{0, 1\}$ with the probability $\Pr[c_i = 0] = \delta$.
3. $\mathcal{B}$ picks a random $u_i \in_R \mathbb{Z}_p^*$ and sets $H(ID'_S, ID'_R, KW_i) = H_i = (1 - c_i)yP + u_iP$.
4. $\mathcal{B}$ adds the tuple $(KW_i, ID'_S, ID'_R, u_i, c_i, H_i)$ into L_H and responds $\mathcal{A}$ with H_i.

$\mathbf{h_2}$ query: $\mathcal{B}$ maintains a list L_{h_2} of tuple $(ID_i, P_{ID_i}, R_{ID_i}, \beta_{ID_i})$. This list is initially empty. Given a query tuple $(ID_i, P_{ID_i}, R_{ID_i})$, $\mathcal{B}$ responds as follows:

1. If $(ID_i, P_{ID_i}, R_{ID_i}, \beta_{ID_i})$ has already existed in the L_{h_2}, $\mathcal{B}$ responds $\mathcal{A}$ with $h_2(ID_i, P_{pub}, P_{ID_i}, R_{ID_i}) = \beta_{ID_i}$.
2. Otherwise, $\mathcal{B}$ picks a random $\beta_{ID_i} \in_R \mathbb{Z}_p^*$ and adds the tuple $(ID_i, P_{ID_i}, R_{ID_i}, \beta_{ID_i})$ into L_{h_2} and responds $\mathcal{A}$ with β_{ID_i}.

Extract partial private key query: $\mathcal{B}$ maintains a list L_{E_1} of tuple $(ID_i, R_{ID_i}, d_{ID_i})$. This list is initially empty. Given a query on ID_i, $\mathcal{B}$ responds as follows:

1. If $(ID_i, R_{ID_i}, d_{ID_i})$ is existed in the L_{E_1}, $\mathcal{B}$ responds $\mathcal{A}$ with d_{ID_i}.
2. Otherwise, if $ID_i \neq ID_S$ and $ID_i \neq ID_S$, $\mathcal{B}$ randomly picks $d_{ID_i}, \alpha_{ID_i} \in_R \mathbb{Z}_p^*$ and computes $R_{ID_i} = d_{ID_i}P - \alpha_{ID_i}P_{pub}$, then adds the tuple $(ID_i, R_{ID_i}, \alpha_{ID_i})$ into L_{h_1} and $(ID_i, R_{ID_i}, d_{ID_i})$ into L_{E_1} and responds $\mathcal{A}$ with d_{ID_i}.
3. If $ID_i = ID_S$ or $ID_i = ID_S$, $\mathcal{B}$ aborts.

Secret value query: $\mathcal{B}$ maintains a list L_{E_2} of tuple $(ID_i, x_{ID_i}, P_{ID_i})$. This list is initially empty. Given a query on ID_i, $\mathcal{B}$ responds as follows:

1. If $(ID_i, x_{ID_i}, P_{ID_i})$ is already in L_{E_2}, $\mathcal{B}$ responds $\mathcal{A}$ with x_{ID_i}.
2. Otherwise, $\mathcal{B}$ picks a random $x_{ID_i} \in_R \mathbb{Z}_q^*$ as a secret value and computes $P_{ID_i} = x_{ID_i}P$, then adds the tuple $(ID_i, x_{ID_i}, P_{ID_i})$ into L_{E_2} and responds $\mathcal{A}$ with x_{ID_i}.

Public key query: Given a query on ID_i, $\mathcal{B}$ retrieves R_{ID_i} and P_{ID_i} from L_{E_1} and L_{E_2}, then returns the public key $PK_{ID_i} = (R_{ID_i}, P_{ID_i})$ to $\mathcal{A}$.

Replace public key query: Given a query tuple $(ID_i, R_{ID_i}, P'_{ID_i})$, $\mathcal{B}$ sets $P_{ID_i} = P'_{ID_i}$, $d_{ID_i} = \perp$, $x_{ID_i} = \perp$. Note that $\mathcal{A}$ is prohibited from replacing the public key of ID_S before the challenge phase.

Ciphertext Oracle $\mathcal{O}_C$: Given a ciphertext query (KW_i, ID'_S, ID'_R), $\mathcal{B}$ retrieves the tuple $(KW_i, ID'_S, ID'_R, u_i, c_i, H_i)$ from L_H. If $c_i = 0$, it aborts the game and outputs a random bit θ' as the guess of θ. Otherwise, it picks $r_1, r_2 \in_R \mathbb{Z}_p^*$, and responds $\mathcal{A}$ with the ciphertext $Ct = (C_1, C_2, C_3, C_4)$, where

$$\begin{aligned} C_1 &= r_1(d_{ID'_S} + \beta_{ID'_S} x_{ID'_S}) u_i P + r_2 P, \\ C_2 &= r_2(R_{ID'_R} + \alpha_{ID'_R} P_{pub} + \beta_{ID'_R} P_{ID'_R}), \\ C_3 &= r_1(d_{ID'_S} + \beta_{ID'_S} x_{ID'_S}) P, \\ C_4 &= r_1 P. \end{aligned} \tag{6}$$

Trapdoor Oracle $\mathcal{O}_T$: Given a trapdoor query (KW_i, ID'_S, ID'_R), $\mathcal{B}$ retrieves the tuple $(KW_i, ID'_S, ID'_R, u_i, c_i, H_i)$ from L_H. If $c_i = 0$, it aborts the game and outputs a random bit θ' as the guess of θ. Otherwise, it picks $r', r'' \in_R \mathbb{Z}_p^*$, and responds $\mathcal{A}$ with the trapdoor $Tr = (T_1, T_2, T_3, T_4)$, where

$$\begin{aligned} T_1 &= r'(d_{ID'_R} + \beta_{ID'_R} x_{ID'_R}) u_i P + r'' P, \\ T_2 &= r'(R_{ID'_R} + \alpha_{ID'_R} P_{pub} + \beta_{ID'_R} P_{ID'_R}), \\ T_3 &= r''(R_{ID'_S} + \alpha_{ID'_S} P_{pub} + \beta_{ID'_S} P_{ID'_S}), \\ T_4 &= r' P. \end{aligned} \tag{7}$$

- **Challenge**: $\mathcal{A}$ chooses two keyword sets KW_0^* and KW_1^* with the restriction that no element of $(KW_j^*, ID_S, ID_R)_{j \in [0,1]}$ has been queried on $\mathcal{O}_C$ or $\mathcal{O}_T$. $\mathcal{B}$ retrieves $(KW_j^*, ID'_S, ID'_R, u_j^*, c_j^*, H_j^*)_{j \in [0,1]}$ from L_H and generates the challenge trapdoors as follows:
  1. If, for each $i \in [1, I]$, $c_0^* = c_1^* = 1$, $\mathcal{B}$ aborts the game and outputs a random bit θ' as the guess of θ.
  2. Otherwise, $c_0^* = 0$ or $c_1^* = 0$. $\mathcal{B}$ generates the challenge trapdoor Tr^* as Phase 1; $\mathcal{B}$ picks a bit $\hat{\theta} \in \{0, 1\}$ such that $c_{\hat{\theta}}^* = 0$. Then, it picks $r', r'' \in_R \mathbb{Z}_p^*$, and generates the challenge trapdoor $Tr^* = (T_1^*, T_2^*, T_3^*, T_4^*)$ for $\mathcal{A}$ as:

$$\begin{aligned} T_1^* &= r'(d_{ID_R} + \beta_{ID_R} x_{ID_R}) y P + r'' P = r'(R_{ID_R} + \alpha_{ID_R} P_{pub} + \beta_{ID_R} P_{ID_R}) y P = r' \mathcal{Z} + r'' P, \\ T_2^* &= r'(R_{ID_R} + \alpha_{ID_R} P_{pub} + \beta_{ID_R} P_{ID_R}) = r' x P, \\ T_3^* &= r'(R_{ID_S} + \alpha_{ID_S} P_{pub} + \beta_{ID_S} P_{ID_S}) = r'' y P, \\ T_4^* &= r' P. \end{aligned} \tag{8}$$

  Tr^* is a valid trapdoor, if $\mathcal{Z} = xyP$; otherwise, $\mathcal{Z} = \mathcal{R} \in_R \mathbb{G}_0$ so that T_1^* is a random element in $\mathbb{G}_0$ and Tr^* is random in the view of $\mathcal{A}$.
- **Phase2**: This phase is the same as Phase 1.
- **Guess**: $\mathcal{A}$ outputs a guess bit θ'' of $\hat{\theta}$. If $\theta'' = \hat{\theta}$, $\mathcal{B}$ guesses $\theta = 0$ which indicates that $\mathcal{Z} = xyP$ in the above game. Otherwise, $\mathcal{B}$ guesses $\theta = 1$ i.e., $\mathcal{Z} = \mathcal{R}$.

If $\mathcal{Z} = \mathcal{R}$, then Tr^* is random from the view of $\mathcal{A}$. Hence, $\mathcal{B}$'s probability to guess θ correctly is

$$\Pr\left[\mathcal{B}\left(P, xP, yP, \mathcal{Z} = \mathcal{R}\right) = 1\right] = \frac{1}{2}. \tag{9}$$

Else $\mathcal{Z} = xyP$, then Tr^* is an available trapdoor and $\mathcal{A}'s$ advantage of guessing θ' is ϵ. Therefore, $\mathcal{B}$'s probability to guess θ correctly is

$$\Pr\left[\mathcal{B}\left(P, xP, yP, \mathcal{Z} = xyP\right) = 0\right] = \frac{1}{2} + \epsilon. \tag{10}$$

The adversary $\mathcal{A}$ may issue at most q_H, q_C, q_T queries to the hash oracle $\mathcal{O}_H$, the ciphertext oracle $\mathcal{O}_C$ and the trapdoor oracle $\mathcal{O}_T$. We make some necessary restriction: q_H, q_C, q_T are polynomial; $\mathcal{A}$ isn't allowed to repeat a query to the hash oracle $\mathcal{O}_H$ and may repeat r_C queries to $\mathcal{O}_C$ and r_T queries to $\mathcal{O}_T$; $\mathcal{A}$ couldn't issue a ciphertext query (ID_S, ID_R, KW) to $\mathcal{O}_C$ or a trapdoor query to $\mathcal{O}_T$ before issuing the hash query (ID_S, ID_R, KW) to $\mathcal{O}_H$. Now, we donate the event $\mathcal{B}$ aborts in the above game by $\mathbf{E}_0$ and compute the probability of $\overline{\mathbf{E}_0}$. Then, $\mathcal{B}$ aborts the game in the following two cases:

1. $c_i = 0$ in the simulation of $\mathcal{O}_C$ or $\mathcal{O}_T$ in phase 1 and phase 2. We denote this event by $\mathbf{E}_1$ and the probability $\mathbf{E}_1$ doesn't occur is that
$$\Pr\left[\overline{\mathbf{E}_1}\right] = (1-\delta)^{q_C + q_T - r_C - r_T}.$$
2. For $c_0^* = c_1^* = 1$ challenge phase. We denote this event by $\mathbf{E}_2$ and the probability $\mathbf{E}_2$ doesn't occur is that
$$\Pr\left[\overline{\mathbf{E}_2}\right] = 1 - (1-\delta)^2.$$

We sets $\delta = 1 - \sqrt[2]{\frac{q_C+q_T-r_C-r_T}{q_C+q_T-r_C-r_T+1}}$, then $\mathcal{B}$ doesn't abort in the above game is that

$$\begin{aligned}\Pr\left[\overline{\mathbf{E}_0}\right] &= \Pr\left[\overline{\mathbf{E}_1}\right]\cdot\Pr\left[\overline{\mathbf{E}_2}\right]\\ &= \left(\frac{q_C+q_T-r_C-r_T}{q_C+q_T-r_C-r_T+1}\right)^{\frac{q_C+q_T-r_C-r_T}{2}}\cdot\frac{1}{q_C+q_T-r_C-r_T+1}\\ &\approx \frac{1}{(q_C+q_T-r_C-r_T+1)\cdot e^{\frac{1}{2}}}.\end{aligned} \tag{11}$$

Hence, $\mathcal{B}$ doesn't abort with a non-negligible probability.

In conclusion, $\mathcal{B}$'s probability to win the above security game is

$$\begin{aligned}\Pr\left[\theta' = \theta\right] &= \Pr\left[\theta' = \theta \wedge \mathbf{E}_0\right] + \Pr\left[\theta' = \theta \wedge \overline{\mathbf{E}_0}\right]\\ &= \Pr\left[\theta' = \theta \| \mathbf{E}_0\right]\cdot\Pr\left[\mathbf{E}_0\right] + \Pr\left[\theta' = \theta \| \overline{\mathbf{E}_0}\right]\cdot\Pr\left[\overline{\mathbf{E}_0}\right]\\ &= \frac{1}{2}\cdot\Pr\left[\mathbf{E}_0\right] + \left(\frac{1}{2}+\epsilon\right)\cdot\Pr\left[\overline{\mathbf{E}_0}\right]\\ &= \frac{1}{2} + \epsilon\cdot\Pr\left[\overline{\mathbf{E}_0}\right].\end{aligned} \tag{12}$$

Game 2 (type2 adversary):

- **Setup**: $\mathcal{A}$ chooses the challenge sender's identity ID_S and challenge receiver's identity ID_R. $\mathcal{B}$ randomly chooses $s \in_R \mathbb{Z}_q^*$, sets $P_{pub} = sP$,$P_{ID_S} = yP$,$P_{ID_R} = xP$, and adds $(ID_S, \perp, P_{ID_S})$ and $(ID_R, \perp, P_{ID_R})$ to L_{E_2}. Then, it sends $PP = (e, \mathbb{G}_0, \mathbb{G}_T, q, P, P_{pub}, H, h_1, h_2)$, s, P_{ID_S}, P_{ID_R} to $\mathcal{A}$.

- **Phase1**: Most queries issued by the adversary $\mathcal{A}$ are defined as same as those in Game 1, except the following queries:

  Extract partial private key query: $\mathcal{B}$ maintains a list L_{E_1} of tuple $(ID_i, R_{ID_i}, d_{ID_i})$. This list is initially empty. Given a query on ID_i, $\mathcal{B}$ responds as follows:
  1. If $(ID_i, R_{ID_i}, d_{ID_i})$ is existed in the L_{E_1}, $\mathcal{B}$ responds $\mathcal{A}$ with d_{ID_i}.
  2. Otherwise, if $(ID_i, R_{ID_i}, \alpha_{ID_i})$ is already in L_{h_1}, $\mathcal{B}$ retrieves α_{ID_i} from L_{h_1}; otherwise $\mathcal{B}$ randomly chooses $r_{ID_i}, \alpha_{ID_i} \in_R \mathbb{Z}_q^*$, sets $R_{ID_i} = r_{ID_i}P$, and adds the tuple $(ID_i, R_{ID_i}, r_{ID_i}, \alpha_{ID_i})$ into L_{h_1}. Then, $\mathcal{B}$ computes $d_{ID_i} = r_{ID_i} + s\alpha_{ID_i} (\mod q)$, adds the tuple $(ID_i, R_{ID_i}, d_{ID_i})$ into L_{E_1} and responds $\mathcal{A}$ with d_{ID_i}.

  Secret value query: $\mathcal{B}$ maintains a list L_{E_2} of tuple $(ID_i, x_{ID_i}, P_{ID_i})$. This list is initially empty. Given a query on ID_i, $\mathcal{B}$ responds as follows:
  1. If $ID_i \neq ID_S$ and $ID_i \neq ID_R$. If $(ID_i, x_{ID_i}, P_{ID_i})$ is already in L_{E_2}, $\mathcal{B}$ responds $\mathcal{A}$ with x_{ID_i}; otherwise, $\mathcal{B}$ picks a random $x_{ID_i} \in_R \mathbb{Z}_q^*$ as a secret value and computes $P_{ID_i} = x_{ID_i}P$, then adds the tuple $(ID_i, x_{ID_i}, P_{ID_i})$ into L_{E_2} and responds $\mathcal{A}$ with x_{ID_i}.
  2. If $ID_i = ID_S$ or $ID_i = ID_R$, $\mathcal{B}$ aborts.
- **Challenge**: $\mathcal{A}$ chooses two keyword sets KW_0^* and KW_1^* with the restriction that no element of $(KW_j^*, ID_S, ID_R)_{j\in[0,1]}$ has been queried on $\mathcal{O}_C$ or $\mathcal{O}_T$. $\mathcal{B}$ retrieves $(KW_j^*, ID_S', ID_R', u_j^*, c_j^*, H_j^*)_{j\in[0,1]}$ from L_H and generates the challenge trapdoors as follows:
  1. If $c_0^* = c_1^* = 1$, $\mathcal{B}$ aborts the game and outputs a random bit θ' as the guess of θ.
  2. Otherwise, $c_0^{t*} = 0$ or $c_1^{t*} = 0$. $\mathcal{B}$ generates the challenge trapdoor Tr^* as Phase 1; $\mathcal{B}$ picks a bit $\hat{\theta} \in \{0,1\}$ such that $c_{\hat{\theta}}^* = 0$. Then, it picks $r', r'' \in_R \mathbb{Z}_p^*$, and generates the challenge trapdoor $Tr^* = (T_1^*, T_2^*, T_3^*, T_4^*)$ for $\mathcal{A}$ as:

$$\begin{aligned}
T_1^* &= r'(d_{ID_R} + \beta_{ID_R}x_{ID_R})yP + r''P = r'(d_{ID_R} + \beta_{ID_R}x)yP = r'\beta_{ID_R}\mathcal{Z} + r'd_{ID_R}\beta_{ID_R}xP + r''P,\\
T_2^* &= r'(R_{ID_R} + \alpha_{ID_R}P_{pub} + \beta_{ID_R}P_{ID_R}) = r'd_{ID_R}P + r'\beta_{ID_R}xP,\\
T_3^* &= r''(R_{ID_S} + \alpha_{ID_S}P_{pub} + \beta_{ID_S}P_{ID_S}) = r''d_{ID_S}P + r''\beta_{ID_S}xP,\\
T_4^* &= r'P.
\end{aligned} \tag{13}$$

  Tr^* is a valid trapdoor, if $\mathcal{Z} = xyP$; otherwise, $\mathcal{Z} = \mathcal{R} \in_R \mathbb{G}_0$ so that T_1^* is a random element in $\mathbb{G}_0$ and Tr_t^* is random in the view of $\mathcal{A}$.
- **Phase2**: This phase is the same as Phase 1.
- **Guess**: $\mathcal{A}$ outputs a guess bit θ'' of $\hat{\theta}$. If $\theta'' = \hat{\theta}$, $\mathcal{B}$ guesses $\theta = 0$ which indicates that $\mathcal{Z} = xyP$ in the above game. Otherwise, $\mathcal{B}$ guesses $\theta = 1$ i.e., $\mathcal{Z} = \mathcal{R}$.

If $\mathcal{Z} = \mathcal{R}$, then Tr^* is random from the view of $\mathcal{A}$. Hence, $\mathcal{B}$'s probability to guess θ correctly is

$$\Pr\left[\mathcal{B}\left(P, xP, yP, \mathcal{Z} = \mathcal{R}\right) = 1\right] = \frac{1}{2}. \tag{14}$$

Else $\mathcal{Z} = xyP$, then Tr^* is an available trapdoor and $\mathcal{A}'s$ advantage of guessing θ' is ϵ. Therefore, $\mathcal{B}$'s probability to guess θ correctly is

$$\Pr\left[\mathcal{B}\left(P, xP, yP, \mathcal{Z} = xyP\right) = 0\right] = \frac{1}{2} + \epsilon. \tag{15}$$

The adversary $\mathcal{A}$ may issue at most q_H, q_C, q_T queries to the hash oracle $\mathcal{O}_H$, the ciphertext oracle $\mathcal{O}_C$ and the trapdoor oracle $\mathcal{O}_T$. We make some necessary restriction: q_H, q_C, q_T are polynomial; $\mathcal{A}$ isn't allowed to repeat a query to the hash oracle $\mathcal{O}_H$ and may repeat r_C queries to $\mathcal{O}_C$ and r_T queries to $\mathcal{O}_T$; $\mathcal{A}$ couldn't issue a ciphertext query (ID_S, ID_R, KW) to $\mathcal{O}_C$ or a trapdoor query to $\mathcal{O}_T$ before issuing the hash query (ID_S, ID_R, KW) to $\mathcal{O}_H$. Now, we donate the event $\mathcal{B}$ aborts in the above game by $\mathbf{E}_0$ and compute the probability of $\overline{\mathbf{E}_0}$. Then, $\mathcal{B}$ aborts the game in the following two cases:

1. $c_i = 0$ in the simulation of $\mathcal{O}_C$ or $\mathcal{O}_T$ in phase 1 and phase 2. We denote this event by $\mathbf{E}_1$ and the probability $\mathbf{E}_1$ doesn't occur is that
$$\Pr\left[\overline{\mathbf{E}_1}\right] = (1-\delta)^{q_C + q_T - r_C - r_T}.$$
2. For e$c_0^* = c_1^* = 1$ challenge phase. We denote this event by $\mathbf{E}_2$ and the probability $\mathbf{E}_2$ doesn't occur is that
$$\Pr\left[\overline{\mathbf{E}_2}\right] = 1 - (1-\delta)^2.$$

We sets $\delta = 1 - \sqrt[2]{\frac{q_C + q_T - r_C - r_T}{q_C + q_T - r_C - r_T + 1}}$, then $\mathcal{B}$ doesn't abort in the above game is that

$$\begin{aligned} \Pr\left[\overline{\mathbf{E}_0}\right] &= \Pr\left[\overline{\mathbf{E}_1}\right] \cdot \Pr\left[\overline{\mathbf{E}_2}\right] \\ &= \left(\frac{q_C + q_T - r_C - r_T}{q_C + q_T - r_C - r_T + 1}\right)^{\frac{q_C + q_T - r_C - r_T}{2}} \cdot \frac{1}{q_C + q_T - r_C - r_T + 1} \\ &\approx \frac{1}{(q_C + q_T - r_C - r_T + 1) \cdot e^{\frac{1}{2}}}. \end{aligned} \tag{16}$$

Hence, $\mathcal{B}$ doesn't abort with a non-negligible probability.

In conclusion, $\mathcal{B}$'s probability to win the above security game is

$$\begin{aligned} \Pr\left[\theta' = \theta\right] &= \Pr\left[\theta' = \theta \wedge \mathbf{E}_0\right] + \Pr\left[\theta' = \theta \wedge \overline{\mathbf{E}_0}\right] \\ &= \Pr\left[\theta' = \theta \| \mathbf{E}_0\right] \cdot \Pr\left[\mathbf{E}_0\right] + \Pr\left[\theta' = \theta \| \overline{\mathbf{E}_0}\right] \cdot \Pr\left[\overline{\mathbf{E}_0}\right] \\ &= \frac{1}{2} \cdot \Pr\left[\mathbf{E}_0\right] + \left(\frac{1}{2} + \epsilon\right) \cdot \Pr\left[\overline{\mathbf{E}_0}\right] \\ &= \frac{1}{2} + \epsilon \cdot \Pr\left[\overline{\mathbf{E}_0}\right]. \end{aligned} \tag{17}$$

Theorem 2. *Our scheme is semantically ciphertext indistinguishability secure against inside keyword guessing attacks in the random oracle model under the DDH assumption.*

Proof. Supposed that a PPT adversary $\mathcal{A}$ can break the CI security of our scheme with a non-negligible advantage $\epsilon > 0$, then there exists a PPT simulator $\mathcal{B}$ that can distinguish a DDH tuple from a random one with a non-negligible probability. The DDH challenger $\mathcal{C}$ selects $x, y \in_R \mathbb{Z}_p^*$, $\theta \in \{0,1\}$, $\mathcal{R} \in_R \mathbb{G}_0$ at random. Let $\mathcal{Z} = xyP$, if $\theta = 0$, $\mathcal{R}$ else. Next, $\mathcal{C}$ sends $\mathcal{B}$ the DDH tuple $\langle P, xP, yP, \mathcal{Z}\rangle$. Then, $\mathcal{B}$ plays the role of simulator in the following security game.

Game 3 (type1 adversary):

- **Setup**: $\mathcal{A}$ chooses the challenge sender's identity ID_S and challenge receiver's identity ID_R. The new public key to replace P_{ID_R} is P'_{ID_R}. $\mathcal{B}$ randomly chooses $\alpha_{ID_S}, \beta_{ID_S}, \alpha_{ID_R}, \beta_{ID_R} \in_R \mathbb{Z}_q^*$, $R_{ID_S}, R_{ID_R} \in_R \mathbb{G}_0$, computes $P_{pub} = \frac{1}{\alpha_{ID_S}}(x \cdot P - \beta_{ID_S} P'_{ID_S} - R_{ID_S})$, and adds $(ID_S, R_{ID_S}, \alpha_{ID_S})$ and $(ID_R, R_{ID_R}, \alpha_{ID_R})$ to the list L_{h_1}, adds $(ID_S, R_{ID_S}, \perp, \beta_{ID_S})$ and $(ID_R, R_{ID_R}, \perp, \beta_{ID_R})$ to list L_{E_1}. Then, it sends $PP = (e, \mathbb{G}_0, \mathbb{G}_T, q, P, P_{pub}, H, h_1, h_2)$ to $\mathcal{A}$.
- **Phase1**: Most queries issued by the adversary $\mathcal{A}$ are the same as those of Game 1, except the **H** query:

  H query $\mathcal{O}_H$: $\mathcal{B}$ maintains a list L_H of tuple $(KW_i, ID'_S, ID'_R, u_i, c_i, H_i)$. This list is initially empty. Given a query tuple (KW_i, ID'_S, ID'_R), $\mathcal{B}$ responds as follows:
  1. If $(KW_i, ID'_S, ID'_R, u_i, c_i, H_i)$ is in the L_H, $\mathcal{B}$ responds $\mathcal{A}$ with $H(ID'_S, ID'_R, KW_i) = H_i$.
  2. Otherwise, $\mathcal{B}$ flips a random coin $c_i \in \{0,1\}$ with the probability $\Pr[c_i = 0] = \delta$.
  3. $\mathcal{B}$ picks a random $u_i \in_R \mathbb{Z}_p^*$ and sets $H(ID'_S, ID'_R, KW_i) = H_i = (1 - c_i)u_i yP + c_i u_i P$.
  4. $\mathcal{B}$ adds the tuple $(KW_i, ID'_S, ID'_R, u_i, c_i, H_i)$ into L_H and responds $\mathcal{A}$ with H_i.
- **Challenge**: $\mathcal{A}$ chooses two keyword sets KW_0^* and KW_1^* with the restriction that no element of $(KW_j^*, ID_S, ID_R)_{j\in[0,1]}$ has been queried on $\mathcal{O}_C$ or $\mathcal{O}_T$. $\mathcal{B}$ retrieves $(KW_j^*, ID'_S, ID'_R, u_j^*, c_j^*, H_j^*)_{j\in[0,1]}$ from L_H and generates the challenge ciphertexts as follows:
  1. If $c_0^{i*} = c_1^{i*} = 1$, $\mathcal{B}$ aborts the game and outputs a random bit θ' as the guess of θ.
  2. Otherwise, $c_0^* = 0$ or $c_1^* = 0$. $\mathcal{B}$ generates the challenge ciphertext Ct^* as Phase 1; $\mathcal{B}$ picks a bit $\hat{\theta} \in \{0,1\}$ such that $c_{\hat{\theta}}^* = 0$. Then, it picks $r_1, r_2 \in_R \mathbb{Z}_p^*$, and generates the challenge ciphertext $Ct^* = (C_1^*, C_2^*, C_3^*, C_4^*)$ for $\mathcal{A}$ as:

$$\begin{aligned} C_1^* &= r_1(d_{ID_S} + \beta_{ID_S} x_{ID_S})u_i yP + r_2 P = r_1 u_i \mathcal{Z} + r_2 P, \\ C_2^* &= r_2(R_{ID_R} + \alpha_{ID_R} P_{pub} + \beta_{ID_R} P_{ID_R}), \\ C_3^* &= r_1(d_{ID_S} + \beta_{ID_S} x_{ID_S})P = r_1 xP, \\ C_4^* &= r_1 P. \end{aligned} \tag{18}$$

  Ct^* is a valid trapdoor, if $\mathcal{Z} = xyP$; otherwise, $\mathcal{Z} = \mathcal{R} \in_R \mathbb{G}_0$ so that C_1^* is a random element in $\mathbb{G}_0$ and Ct^* is random in the view of $\mathcal{A}$.
- **Phase2**: This phase is the same as Phase 1.

- **Guess**: $\mathcal{A}$ outputs a guess bit θ'' of $\hat{\theta}$. If $\theta'' = \hat{\theta}$, $\mathcal{B}$ guesses $\theta = 0$ which indicates that $\mathcal{Z} = xyP$ in the above game. Otherwise, $\mathcal{B}$ guesses $\theta = 1$ i.e., $\mathcal{Z} = \mathcal{R}$.

If $\mathcal{Z} = \mathcal{R}$, then Ct^* is random from the view of $\mathcal{A}$. Hence, $\mathcal{B}$'s probability to guess θ correctly is

$$\Pr\left[\mathcal{B}\left(P, xP, yP, \mathcal{Z} = \mathcal{R}\right) = 1\right] = \frac{1}{2}. \tag{19}$$

Else $\mathcal{Z} = xyP$, then Ct^* is an available trapdoor and $\mathcal{A}'s$ advantage of guessing θ' is ϵ. Therefore, $\mathcal{B}$'s probability to guess θ correctly is

$$\Pr\left[\mathcal{B}\left(P, xP, yP, \mathcal{Z} = xyP\right) = 0\right] = \frac{1}{2} + \epsilon. \tag{20}$$

The adversary $\mathcal{A}$ may issue at most q_H, q_C, q_T queries to the hash oracle $\mathcal{O}_H$, the ciphertext oracle $\mathcal{O}_C$ and the trapdoor oracle $\mathcal{O}_T$. We make some necessary restriction: q_H, q_C, q_T are polynomial; $\mathcal{A}$ isn't allowed to repeat a query to the hash oracle $\mathcal{O}_H$ and may repeat r_C queries to $\mathcal{O}_C$ and r_T queries to $\mathcal{O}_T$; $\mathcal{A}$ couldn't issue a ciphertext query (ID_S, ID_R, KW) to $\mathcal{O}_C$ or a trapdoor query to $\mathcal{O}_T$ before issuing the hash query (ID_S, ID_R, KW) to $\mathcal{O}_H$. Now, we donate the event $\mathcal{B}$ aborts in the above game by $\mathbf{E}_0$ and compute the probability of $\overline{\mathbf{E}_0}$. Then, $\mathcal{B}$ aborts the game in the following two cases:

1. $c_i = 0$ in the simulation of $\mathcal{O}_C$ or $\mathcal{O}_T$ in phase 1 and phase 2. We denote this event by $\mathbf{E}_1$ and the probability $\mathbf{E}_1$ doesn't occur is that
$$\Pr\left[\overline{\mathbf{E}_1}\right] = (1-\delta)^{q_C + q_T - r_C - r_T}.$$
2. For $c_0^* = c_1^* = 1$ challenge phase. We denote this event by $\mathbf{E}_2$ and the probability $\mathbf{E}_2$ doesn't occur is that
$$\Pr\left[\overline{\mathbf{E}_2}\right] = 1 - (1-\delta)^2.$$

We sets $\delta = 1 - \sqrt[2]{\frac{q_C + q_T - r_C - r_T}{q_C + q_T - r_C - r_T + 1}}$, then $\mathcal{B}$ doesn't abort in the above game is that

$$\begin{aligned}
\Pr\left[\overline{\mathbf{E}_0}\right] &= \Pr\left[\overline{\mathbf{E}_1}\right] \cdot \Pr\left[\overline{\mathbf{E}_2}\right] \\
&= \left(\frac{q_C + q_T - r_C - r_T}{q_C + q_T - r_C - r_T + 1}\right)^{\frac{q_C + q_T - r_C - r_T}{2}} \cdot \frac{1}{q_C + q_T - r_C - r_T + 1} \\
&\approx \frac{1}{(q_C + q_T - r_C - r_T + 1) \cdot e^{\frac{1}{2}}}.
\end{aligned} \tag{21}$$

Hence, $\mathcal{B}$ doesn't abort with a non-negligible probability.

In conclusion, $\mathcal{B}$'s probability to win the above security game is

$$\begin{aligned}
\Pr\left[\theta' = \theta\right] &= \Pr\left[\theta' = \theta \wedge \mathbf{E}_0\right] + \Pr\left[\theta' = \theta \wedge \overline{\mathbf{E}_0}\right] \\
&= \Pr\left[\theta' = \theta \| \mathbf{E}_0\right] \cdot \Pr\left[\mathbf{E}_0\right] + \Pr\left[\theta' = \theta \| \overline{\mathbf{E}_0}\right] \cdot \Pr\left[\overline{\mathbf{E}_0}\right] \\
&= \frac{1}{2} \cdot \Pr\left[\mathbf{E}_0\right] + \left(\frac{1}{2} + \epsilon\right) \cdot \Pr\left[\overline{\mathbf{E}_0}\right] \\
&= \frac{1}{2} + \epsilon \cdot \Pr\left[\overline{\mathbf{E}_0}\right].
\end{aligned} \tag{22}$$

Game 4 (type2 adversary):

- **Setup**: $\mathcal{A}$ chooses the challenge sender's identity ID_S and challenge receiver's identity ID_R. $\mathcal{B}$ randomly chooses $s \in_R \mathbb{Z}_q^*$, sets $P_{pub} = sP$,$P_{ID_S} = yP$,$P_{ID_R} = xP$, and adds $(ID_S, \perp, P_{ID_S})$ and $(ID_R, \perp, P_{ID_R})$ to L_{E_2}. Then, it sends $PP = (e, \mathbb{G}_0, \mathbb{G}_T, q, P, P_{pub}, H, h_1, h_2)$, s, P_{ID_S}, P_{ID_R} to $\mathcal{A}$.
- **Phase1**: Most queries issued by the adversary $\mathcal{A}$ are the same as those of Game 2, except the **H** query. $\mathcal{B}$ simulates the **H** query by the same way as in Game 3.
- **Challenge**: $\mathcal{A}$ chooses two keyword sets KW_0^* and KW_1^* with the restriction that no element of $(KW_j^*, ID_S, ID_R)_{j\in[0,1]}$ has been queried on $\mathcal{O}_C$ or $\mathcal{O}_T$. $\mathcal{B}$ retrieves $(KW_j^*, ID'_S, ID'_R, u_j^*, c_j^*, H_j^*)_{j\in[0,1]}$ from L_H and generates the challenge ciphertexts as follows:
  1. If $c_0^* = c_1^* = 1$, $\mathcal{B}$ aborts the game and outputs a random bit θ' as the guess of θ.
  2. Otherwise, $c_0^* = 0$ or $c_1^* = 0$. $\mathcal{B}$ generates the challenge ciphertext Ct^* as Phase 1; $\mathcal{B}$ picks a bit $\hat{\theta} \in \{0,1\}$ such that $c_{\hat{\theta}}^* = 0$. Then, it picks $r_1, r_2 \in_R \mathbb{Z}_p^*$, and generates the challenge ciphertext $Ct^* = (C_1^*, C_2^*, C_3^*, C_4^*)$ for $\mathcal{A}$ as:

$$\begin{aligned}
C_1^* &= r_1(d_{ID_S} + \beta_{ID_S} x_{ID_S}) u_i yP + r_2 P = r_1 d_{ID_S} u_i yP + r_1 \beta_{ID_S} u_i \mathcal{Z} + r_2 P,\\
C_2^* &= r_2(R_{ID_R} + \alpha_{ID_R} P_{pub} + \beta_{ID_R} P_{ID_R}) = r_2 d_{ID_S} P + r_2 \beta_{ID_R} xP,\\
C_3^* &= r_1(d_{ID_S} + \beta_{ID_S} x_{ID_S})P = r_1 d_{ID_S} P + r_1 \beta_{ID_S} yP,\\
C_4^* &= r_1 P.
\end{aligned} \tag{23}$$

  Ct^* is a valid trapdoor, if $\mathcal{Z} = xyP$; otherwise, $\mathcal{Z} = \mathcal{R} \in_R \mathbb{G}_0$ so that C_1^* is a random element in $\mathbb{G}_0$ and Ct^* is random in the view of $\mathcal{A}$.
- **Phase2**: This phase is the same as Phase 1.
- **Guess**: $\mathcal{A}$ outputs a guess bit θ'' of $\hat{\theta}$. If $\theta'' = \hat{\theta}$, $\mathcal{B}$ guesses $\theta = 0$ which indicates that $\mathcal{Z} = xyP$ in the above game. Otherwise, $\mathcal{B}$ guesses $\theta = 1$ i.e., $\mathcal{Z} = \mathcal{R}$.

If $\mathcal{Z} = \mathcal{R}$, then Ct^* is random from the view of $\mathcal{A}$. Hence, $\mathcal{B}$'s probability to guess θ correctly is

$$\Pr\left[\mathcal{B}\left(P, xP, yP, \mathcal{Z} = \mathcal{R}\right) = 1\right] = \frac{1}{2}. \tag{24}$$

Else $\mathcal{Z} = xyP$, then Ct^* is an available trapdoor and $\mathcal{A}'s$ advantage of guessing θ' is ϵ. Therefore, $\mathcal{B}$'s probability to guess θ correctly is

$$\Pr\left[\mathcal{B}\left(P, xP, yP, \mathcal{Z} = xyP\right) = 0\right] = \frac{1}{2} + \epsilon. \tag{25}$$

The adversary $\mathcal{A}$ may issue at most q_H, q_C, q_T queries to the hash oracle $\mathcal{O}_H$, the ciphertext oracle $\mathcal{O}_C$ and the trapdoor oracle $\mathcal{O}_T$. We make some necessary restriction: q_H, q_C, q_T are polynomial; $\mathcal{A}$ isn't allowed to repeat a query to the hash oracle $\mathcal{O}_H$ and may repeat r_C queries to $\mathcal{O}_C$ and r_T queries to $\mathcal{O}_T$; $\mathcal{A}$ couldn't issue a ciphertext query (ID_S, ID_R, KW) to $\mathcal{O}_C$ or a trapdoor query to $\mathcal{O}_T$ before issuing the hash query (ID_S, ID_R, KW) to $\mathcal{O}_H$. Now, we donate the event $\mathcal{B}$ aborts in the above game by $\mathbf{E}_0$ and compute the probability of $\overline{\mathbf{E}_0}$. Then, $\mathcal{B}$ aborts the game in the following two cases:

1. $c_i = 0$ in the simulation of $\mathcal{O}_C$ or $\mathcal{O}_T$ in phase 1 and phase 2. We denote this event by $\mathbf{E}_1$ and the probability $\mathbf{E}_1$ doesn't occur is that
$$\Pr\left[\overline{\mathbf{E}_1}\right] = (1-\delta)^{q_C+q_T-r_C-r_T}.$$
2. For $c_0^* = c_1^* = 1$ challenge phase. We denote this event by $\mathbf{E}_2$ and the probability $\mathbf{E}_2$ doesn't occur is that
$$\Pr\left[\overline{\mathbf{E}_2}\right] = 1-(1-\delta)^2.$$

We sets $\delta = 1 - \sqrt[2]{\frac{q_C+q_T-r_C-r_T}{q_C+q_T-r_C-r_T+1}}$, then $\mathcal{B}$ doesn't abort in the above game is that

$$\begin{aligned}\Pr\left[\overline{\mathbf{E}_0}\right] &= \Pr\left[\overline{\mathbf{E}_1}\right]\cdot\Pr\left[\overline{\mathbf{E}_2}\right] \\ &= \left(\frac{q_C+q_T-r_C-r_T}{q_C+q_T-r_C-r_T+1}\right)^{\frac{q_C+q_T-r_C-r_T}{2}}\cdot\frac{1}{q_C+q_T-r_C-r_T+1} \\ &\approx \frac{1}{(q_C+q_T-r_C-r_T+1)\cdot e^{\frac{1}{2}}}.\end{aligned} \tag{26}$$

Hence, $\mathcal{B}$ doesn't abort with a non-negligible probability.

In conclusion, $\mathcal{B}$'s probability to win the above security game is

$$\begin{aligned}\Pr[\theta'=\theta] &= \Pr[\theta'=\theta\wedge\mathbf{E}_0] + \Pr\left[\theta'=\theta\wedge\overline{\mathbf{E}_0}\right] \\ &= \Pr[\theta'=\theta\|\mathbf{E}_0]\cdot\Pr[\mathbf{E}_0] + \Pr\left[\theta'=\theta\|\overline{\mathbf{E}_0}\right]\cdot\Pr\left[\overline{\mathbf{E}_0}\right] \\ &= \frac{1}{2}\cdot\Pr[\mathbf{E}_0] + \left(\frac{1}{2}+\epsilon\right)\cdot\Pr\left[\overline{\mathbf{E}_0}\right] \\ &= \frac{1}{2}+\epsilon\cdot\Pr\left[\overline{\mathbf{E}_0}\right].\end{aligned} \tag{27}$$

6 Conclusion

In this paper, we initially propose a CLPAEKS scheme with probabilistic trapdoor generation. We provide formal proof of our scheme in the random oracle model against inside keyword guessing attacks. By combining our scheme and [14], it would be able to design a secure designated CLPAEKS scheme with probabilistic trapdoor generation. For the compact of this paper, we leave it as a further work.

References

1. Abdalla, M., Bellare, M., Catalano, D., Kiltz, E., Kohno, T., Lange, T., Malone-Lee, J., Neven, G., Paillier, P., Shi, H.: Searchable encryption revisited: Consistency

properties, relation to anonymous ibe, and extensions. In: Shoup, V. (ed.) Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings. Lecture Notes in Computer Science, vol. 3621, pp. 205–222. Springer (2005)

2. Al-Riyami, S.S., Paterson, K.G.: Certificateless public key cryptography. In: Laih, C. (ed.) Advances in Cryptology - ASIACRYPT 2003, 9th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, November 30 - December 4, 2003, Proceedings. Lecture Notes in Computer Science, vol. 2894, pp. 452–473. Springer (2003)
3. Al-Riyami, S.S., Paterson, K.G.: CBE from CL-PKE: A generic construction and efficient schemes. In: Vaudenay, S. (ed.) Public Key Cryptography - PKC 2005, 8th International Workshop on Theory and Practice in Public Key Cryptography, Les Diablerets, Switzerland, January 23-26, 2005, Proceedings. Lecture Notes in Computer Science, vol. 3386, pp. 398–415. Springer (2005)
4. Boneh, D., Crescenzo, G.D., Ostrovsky, R., Persiano, G.: Public key encryption with keyword search. In: Cachin, C., Camenisch, J. (eds.) Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings. Lecture Notes in Computer Science, vol. 3027, pp. 506–522. Springer (2004)
5. Boneh, D., Waters, B.: Conjunctive, subset, and range queries on encrypted data. In: TCC 2007. pp. 535–554 (2007)
6. Byun, J.W., Rhee, H.S., Park, H., Lee, D.H.: Off-line keyword guessing attacks on recent keyword search schemes over encrypted data. In: SDM 2006. pp. 75–83 (2006)
7. Chen, R., Mu, Y., Yang, G., Guo, F., Wang, X.: Dual-server public-key encryption with keyword search for secure cloud storage. IEEE Trans. Inf. Forensics Secur. **11**(4), 789–798 (2016)
8. Fei MENG, Leixiao CHENG, M.W.: Abdks: Attribute-based encryption with dynamic keyword search in fog computing. Frontiers of Computer Science
9. Fuhr, T., Paillier, P.: Decryptable searchable encryption. In: Susilo, W., Liu, J.K., Mu, Y. (eds.) Provable Security, First International Conference, ProvSec 2007, Wollongong, Australia, November 1-2, 2007, Proceedings. Lecture Notes in Computer Science, vol. 4784, pp. 228–236. Springer (2007)
10. Golle, P., Staddon, J., Waters, B.R.: Secure conjunctive keyword search over encrypted data. In: Jakobsson, M., Yung, M., Zhou, J. (eds.) Applied Cryptography and Network Security, Second International Conference, ACNS 2004, Yellow Mountain, China, June 8-11, 2004, Proceedings. Lecture Notes in Computer Science, vol. 3089, pp. 31–45. Springer (2004)
11. He, D., Ma, M., Zeadally, S., Kumar, N., Liang, K.: Certificateless public key authenticated encryption with keyword search for industrial internet of things. IEEE Trans. Ind. Informatics **14**(8), 3618–3627 (2018)
12. Huang, Q., Li, H.: An efficient public-key searchable encryption scheme secure against inside keyword guessing attacks. Inf. Sci. **403**, 1–14 (2017)
13. Huang, X., Susilo, W., Mu, Y., Zhang, F.: On the security of certificateless signature schemes from asiacrypt 2003. In: Desmedt, Y., Wang, H., Mu, Y., Li, Y. (eds.) Cryptology and Network Security, 4th International Conference, CANS 2005, Xiamen, China, December 14-16, 2005, Proceedings. Lecture Notes in Computer Science, vol. 3810, pp. 13–25. Springer (2005)
14. Li, H., Huang, Q., Shen, J., Yang, G., Susilo, W.: Designated-server identity-based authenticated encryption with keyword search for encrypted emails. Inf. Sci. **481**, 330–343 (2019)

15. Liu, X., Li, H., Yang, G., Susilo, W., Tonien, J., Huang, Q.: Towards enhanced security for certificateless public-key authenticated encryption with keyword search. In: Steinfeld, R., Yuen, T.H. (eds.) Provable Security - 13th International Conference, ProvSec 2019, Cairns, QLD, Australia, October 1-4, 2019, Proceedings. Lecture Notes in Computer Science, vol. 11821, pp. 113–129. Springer (2019)
16. Noroozi, M., Eslami, Z.: Public key authenticated encryption with keyword search: revisited. IET Inf. Secur. **13**(4), 336–342 (2019)
17. Park, D.J., Kim, K., Lee, P.J.: Public key encryption with conjunctive field keyword search. In: WISA 2004. pp. 73–86 (2004)
18. Qin, B., Chen, Y., Huang, Q., Liu, X., Zheng, D.: Public-key authenticated encryption with keyword search revisited: Security model and constructions. Inf. Sci. **516**, 515–528 (2020)
19. Rhee, H.S., Lee, D.H.: Keyword updatable PEKS. In: Kim, H., Choi, D. (eds.) Information Security Applications - 16th International Workshop, WISA 2015, Jeju Island, Korea, August 20-22, 2015, Revised Selected Papers. Lecture Notes in Computer Science, vol. 9503, pp. 96–109. Springer (2015)
20. Rhee, H.S., Park, J.H., Susilo, W., Lee, D.H.: Trapdoor security in a searchable public-key encryption scheme with a designated tester. J. Syst. Softw. **83**(5), 763–771 (2010)
21. Shao, J., Cao, Z., Liang, X., Lin, H.: Proxy re-encryption with keyword search. Inf. Sci. **180**(13), 2576–2587 (2010)
22. Shi, E., Bethencourt, J., Chan, T.H., Song, D.X., Perrig, A.: Multi-dimensional range query over encrypted data. In: S&P 2007. pp. 350–364 (2007)
23. Song, D.X., Wagner, D.A., Perrig, A.: Practical techniques for searches on encrypted data. In: 2000 IEEE Symposium on Security and Privacy, Berkeley, California, USA, May 14-17, 2000. pp. 44–55. IEEE Computer Society (2000)
24. Zheng, Q., Xu, S., Ateniese, G.: VABKS: verifiable attribute-based keyword search over outsourced encrypted data. In: INFOCOM 2014. pp. 522–530 (2014)